Eigenfaces

Alessandro di Tocco

October 2020

Indice

Introduzione

Il progetto tratta il riconoscimento facciale tramite la tecnica delle eigenfaces (o autofacce), seguendo come traccia l'articolo "*Eigenfaces for Recognition*" di *Turk* e *Pentland*, pubblicato nel 1991.

1 Il metodo delle eigenfaces per il riconoscimento facciale

1.1 DataBase

Per prima cosa bisogna acquisire un DataBase di immagini. Per questo progetto è stato usato l'ORL_FaceDataSet, che è costituito da 400 immagini di dimensione 112x92. Ogni persona è rappresentata in 10 immagini con espressioni e illuminazioni diverse. Utilizziamo come esempio solo una porzione del DataBase fornito, le prime $N = 40$ immagini.

1.2 Calcolo delle eigenfaces

Rappresentiamo l'immagine di una faccia $I(x, y)$ come una matrice bidimensionale di dimensione $R \times C$, dove R è il numero di righe e C il numero di colonne. Nel nostro caso $R = 112$ e $C = 92$, perciò un'immagine diventa un vettore di dimensione 10304 o equivalentemente un punto in uno spazio 10304-dimensionale. Possiamo quindi immaginare un insieme di immagini come una collezione di punti in questo enorme spazio. Però, poichè le immagini sono simili fra di loro, non sono distribuite in modo casuale in tutto lo spazio, quindi si possono descrivere attraverso un sottospazio più piccolo usando l'Analisi delle componenti principali (PCA). La PCA consiste nel selezionare le variabili che meglio descrivono la dispersione tra i vari dati, eliminando quelle con una minor varianza. Questo processo viene svolto calcolando gli autovettori significativi (cioè non nulli) della matrice di Covarianza.
Prendiamo un insieme di immagini $\Gamma_1, \Gamma_2, ..., \Gamma_N$. La media delle facce è definita come $\Psi = \frac{1}{N} \sum_{i=1}^{N} \Gamma_i$. Ogni faccia differisce dalla faccia media del vettore $\Phi_i = \Gamma_i - \Psi$. Usiamo quindi su questo insieme di vettori molto grandi l'analisi delle componenti principali cercando gli autovettori della matrice di Covarianza

$$Cov = \frac{1}{N} \sum_{i=1}^{N} \Phi_n \Phi_n^T = AA^T$$

dove $A = [\Phi_1, \Phi_2, ..., \Phi_N]$.Vista la grandezza della matrice di Covarianza sarebbe molto complicato calcolarne direttamente gli autovettori, però notiamo che se il numero di foto nel DataBase è più piccolo rispetto alla dimensione dello spazio ($R \times C < N$) abbiamo solo N autovettori significativi. Possiamo quindi risolvere questo problema trovando gli autovettori di una matrice $N \times N$.

Consideriamo gli autovettori v_i della matrice $A^T A$ tali che

$$A^T A v_i = \mu_i v_i$$

Moltiplicando a sinistra da entrambe le parti per A si ottiene

$$A A^T A v_i = \mu_i A v_i$$

da cui si deduce che gli N autovettori di C sono Av_i, perciò gli $u_i = Av_i$ sono le autofacce (eigenfaces) cercate.

1.3 Scelta della migliore approssimazione

Nel passaggio precedente abbiamo calcolato le autofacce, ora dobbiamo scegliere quante di queste sono essenziali per avere una buona approssimazione del DataBase. In questo progetto abbiamo deciso un'approssimazione del 95%, ma per scopi differenti potrebbero servire approssimazioni differenti, per cui nel codice sarà facilmente modificabile questo valore.
Vogliamo "salvare" solo le autofacce più rappresentative dello spazio cioè quelle con gli autovalori più grandi e la cui somma pesata sia maggiore di 0.95, cioè dati λ_i gli autovalori, ordinati decrescentemente, e Λ la somma di tutti questi, cerchiamo il minimo M tale per cui

$$\frac{\lambda_1 + \lambda_2 + ... + \lambda_M}{\lambda} \geq 0.95$$

Quindi teniamo solo gli autovettori rispettivi ai primi M autovalori.

1.4 Spazio delle facce

Adesso possiamo costruire lo spazio delle facce, questo non è altro che lo spazio generato dalle M autofacce che meglio approssimano l'insieme delle facce iniziale. Perciò possiamo riscrivere ogni faccia del DataBase come combinazione lineare delle M autofacce, ottenendo N vettori M-dimensionali, invece che N vettori $R \times C$ dimensionali; supponendo che il numero delle immagini nel nostro DataBase sia molto più piccolo rispetto alla dimensione dello spazio delle immagini questo comporta un enorme risparmio di memoria.
Sia Γ una faccia del DataBase, questa viene proiettata nello spazio delle autofacce tramite una semplice operazione

$$\omega_k = u_k^T (\Gamma - \Psi)$$

I pesi appena calcolati formano un vettore $\Omega^T = [\omega_1, \omega_2, ..., \omega_M]$ che descrive il contributo di ogni faccia nel rappresentare l'immagine inserita. Tale vettore può essere utilizzato per calcolare la distanza dallo spazio delle facce.
Il modo più semplice per determinare quale elemento del DataBase meglio descive la faccia inserita è calcolare la k-sima faccia che minimizza la distanza Euclidea

$$\epsilon_k = ||(\Omega - \Omega_k)||$$

dove Ω_k è la k-sima immagine del DataBase. Una faccia appartiene al DataBase se $\epsilon_k < \theta_\epsilon$ dove θ_ϵ è un valore scelto arbitrariamente. Per studiare se l'immagine inserita è una faccia calcoliamo la distanza fra l'immagine mediata e l'immagine proiettata nello spazio delle facce, cioè definiti $\Phi = \Gamma - \Psi$ e $\Phi_f = \sum_{i=1}^{M} \omega_i u_i$ calcoliamo

$$\epsilon = ||\Phi - \Phi_f||$$

1.5 Riconoscimento facciale

Ora diamo un esempio di implementazione di riconoscimento facciale, su cui è basato il codice matlab.

- Per prima cosa l'utente inserisce un immagine e il sistema calcola la proiezione dell'immagine nello spazio delle facce.
- Si determina se l'immagine inserita è una faccia calcolando la distanza dell'immagine, proiettata nello spazio delle facce, dallo spazio delle facce stesso. Tale distanza deve essere minore di una soglia θ_1. Se l'immagine non è una faccia il programma termina.
- L'immagine inserita è una faccia nota se $\min(||(\Omega - \Omega_k)||) < \theta_2$ dove in questo caso per Ω_k si intende la media dei vettori che corrispondono alla k-sima persona; chiamiamo tali Ω_k classi di persona. Se la faccia inserita appartiene ad una persona nota ma non fa parte del DataBase, ricalcoliamo la classe che rappresenta tale persona come la media fra la classe vecchia e la nuova immagine.
- Se la faccia inserita appartiene ad una persona nota ma non fa parte del DataBase, ricalcoliamo la classe che rappresenta tale persona come la media fra la classe vecchia e la nuova immagine. Altrimenti vuol dire che il nostro DataBase è già aggiornato.