Rilevamento di volti tramite reti neurali

Samuel Rota Bulò

Università Ca' Foscari di Venezia
Dipartimento di informatica

26 maggio 2006

Contenuti

Descrizione del problema

Con rilevamento di volti (o face detection) intendiamo un processo che a partire da un'immagine ritorna la posizione dei volti in essa presenti.

- Importante perchè pone le basi per eventuali altri processi di più alto livello come per esempio il riconoscimento di volti.
- La ricerca ha posto la sua attenzione su questo problema in particolare dagli anni '90 in avanti.

Stato dell'arte

- Anni '70: prime semplici tecniche di face detection su volti frontali, con background uniforme (fototessere) [12]
- Anni '70-'90: scarso interesse
- Anni '90: crescente interesse verso face recognition e di conseguenza detection [1]
- Distinguiamo due categorie principali
  1. feature-based: estrazione di feature visuali di basso livello (colore, geometria, etc...) per poi organizzarli in feature facciali (bocca, labbra, naso, occhi, etc...) e individuarne la locazione.
  2. image-based: face detection riformulato come problema di pattern recognition; approccio di apprendimento per esempi.

Feature-based face detection

- Ha una fase iniziale di estrazione di feature visuali di basso livello (edges, luminosità, colore, movimento, misure generalizzate) sfruttando le proprietà dei pixel (posizione, colore, etc . . .)
- Sfruttando la conoscenza della geometria del volto, le feature di basso livello vengono astratte in feature di alto livello (bocca, naso, occhi, etc. . .)
- Utilizzabili per sistemi real-time
- Le principali tencniche sono
  1. Feature searching: cercano caratteristiche facciali prominenti (bocca, naso, occhi, etc . . .) e le loro posizioni relative per individuare la posizione di un volto. [3, 5]
  2. Constellation analysis: raggruppano le caratteristiche facciali in una "face-like constellation" utilizzando modelli di volto più robusti basati su analisi statistiche. [8, 14]
  3. Active Shape Models: tecniche che rilasciano una "forma attiva" vicino ad una feature e questa forma, interagendo con l'immagine locale si modifica per prendere la forma della feature.[6, 15, 2]

Image-based face detection

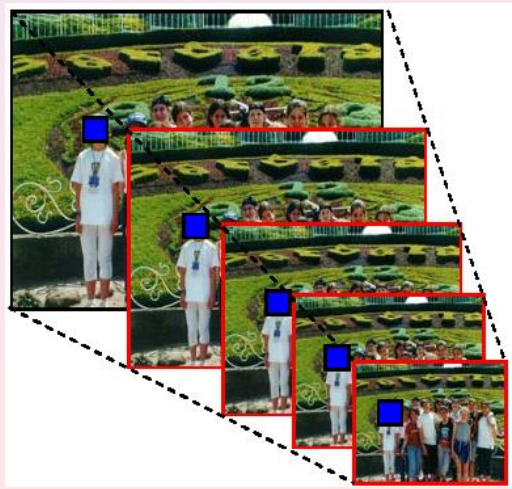
- Face detection=pattern recognition
- Non si conferisce conoscenza su cosa è un volto (face knowledge), ma viene inferita da degli esempi.
  - $\uparrow$ eliminiamo errori dovuti a modelli di volto incompleti o inesatti
  - $\downarrow$ critica la fase di addestramento
- Più robusto a variazioni ambientali e fisiche del volto (illuminazione, sfondo, occhiali, baffi, etc. . .)
- Spesso associate a tecniche di scansione e scalatura dell'immagine.
- Le principali tecniche sono
  1. Linear Subspace Methods: *Principal Component Analysis* (PCA) , *Linear Discriminant Analysis* (LDA) e *Factor Analisys* (FA)[9, 13] .
  2. Statistical approaches: sistemi basati sulla teoria dell'informazione, support vector machines e regole di decisione Bayesiane.
  3. Neural networks: le reti neurali sono una tecnica popolare per il pattern recognition, di cui vedremo in dettaglio un particolare tipo. [4, 7, 10, 11]

Soluzione proposta

- Articolo "Neural Network-Based Face Detection", H.A. Rowley, S. Baluja, T. Kanade [10]
- Concentriamoci su un sotto problema più "semplice" da risolvere.

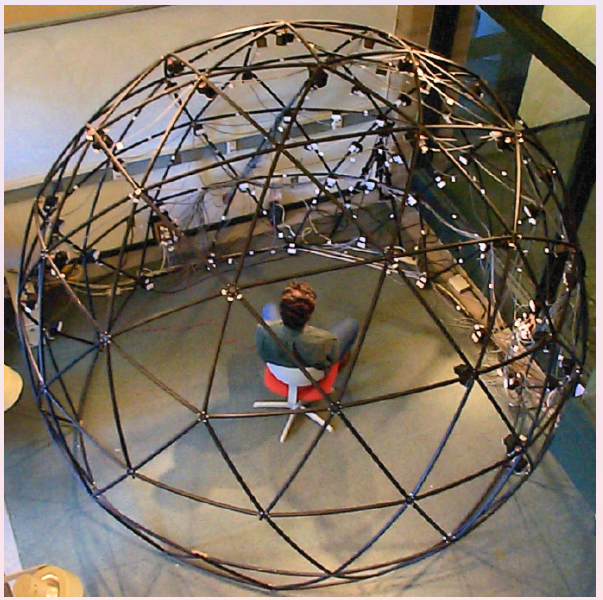
Data una finestra $n \times m$ sull'immagine, classifichiamo il suo contenuto in "volto" o "non volto".

- Per risolvere il problema originario facciamo ripetute classificazioni volto/non volto sull'immagine originale a scale diverse, spostando la finestra d'azione



Soluzione proposta

- Il filtro per la classificazione viene creato attraverso una rete neurale di tipo Multilayer Perceptron formata da 2 layer nascosti
- I training sets e i validation sets vengono costruiti a partire da un database di volti frontali, di soggetti diversi con vari orientamenti, scale, condizioni di illuminazione.
- Come test set abbiamo preso un insieme di immagini reali con e senza volti all'interno, ed abbiamo verificato la qualità della rete ottenuta.



Ulteriori problemi da risolvere

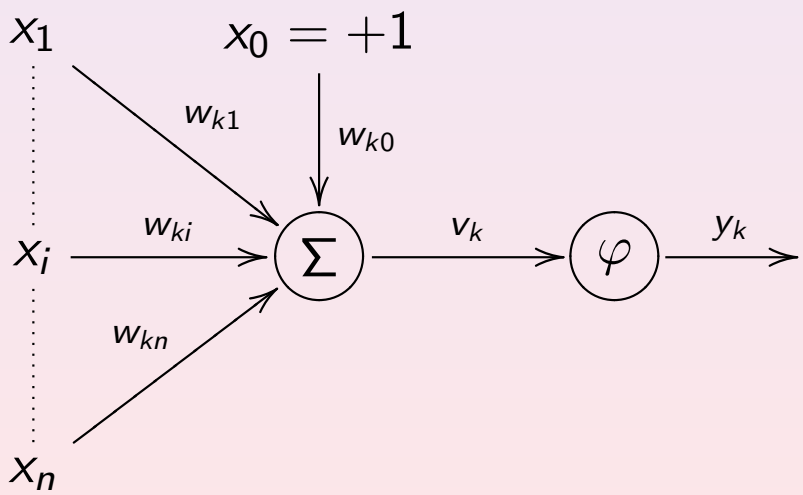
- Fare in modo che le condizioni di illuminazione cui sono sottoposti i volti non incidano sul training e sul rilevamento.
- Effettuare un opportuno tuning dei parametri impostabili della rete neurale, ivi compresa la politica di apprendimento (scelta training set, test set, etc. . .)
- Trovare un sistema per eliminare il rumore dai rilevamenti (falsi positivi[1] e falsi negativi[2]).
- Dal momento che uno stesso volto può dare origine a diversi rilevamenti in un intorno del suo centro (anche a scale diverse), bisogna raggruppare in qualche modo i rilevamenti di uno stesso volto.

[1] Non-volti classificati come volti
[2] Volti classificati come non-volti

Neurone di McCulloch-Pitts

- Unità di calcolo fondamentale della rete neurale
- Caratterizzata da 4 componenti:

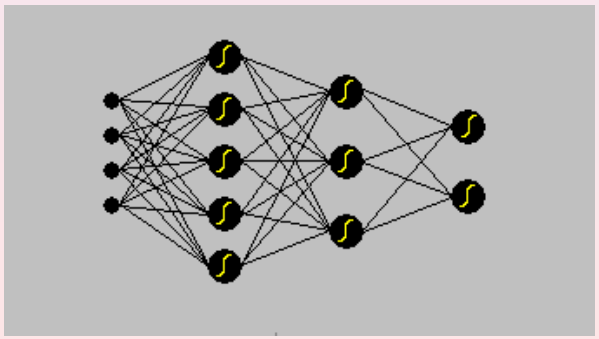
1. sinapsi o connessioni, caratterizzate da un peso positivo o negativo
2. un sommatore che somma i segnali in input pesati dalle sinapsi (input netto)
3. un valore soglia che aumenta/diminuisce l'input netto
4. una funzione di attivazione che limita l'ampiezza dell'output di un neurone rimappando l'input netto nell'intervallo $[0, 1]$ oppure $[-1, 1]$



$$v_k = \sum_{i=0}^{n} w_{ki} \cdot x_i \qquad y_k = \varphi(v_k)$$

Multilayer Perceptron

- È una rete neurale stratificata formata da 3 tipi di strato
  1. un insieme di ingressi (input layer)
  2. uno o più strati nascosti di neuroni (hidden layers)
  3. un insieme di neuroni di uscita (output layer)
- il segnale si genera nello strato di input, e si propaga attraverso gli strati nascosti fino allo strato di output
- la rete ha alta connettività
- funzione di attivazione continua
- un metodo efficiente per l'addestramento è il back-propagation



Back-propagation . . .

- Metodo per l'addestramento di una rete multilayer perceptron
- Distinguiamo 2 tipi di segnali
  1. segnale di funzione: è un segnale di input che si propaga in avanti attraverso la rete fino ad emergere come segnale di output
  2. segnale di errore: ùn segnale che ha origine nel layer di uscita e si propaga all'indietro attraverso la rete
- L'obiettivo è minimizzare l'errore della rete
  - errore totale relativo ad un esempio presentato alla rete (on-line training)
  - la media degli errori totali relativi ad ogni esempio nel training set (off-line training)
- (ci concentreremo sull' on-line training)

. . . Back-propagation. . .

- La minimizzazione dell'errore avviene mediante il metodo della discesa del gradiente
- L'errore da minimizzare è

$$E(n) = \frac{1}{2} \sum_j e_j(n)^2$$

dove $e_j(n) = d_j(n) - y_j(n)$ è la differenza tra l'output desiderato e quello ottenuto dal j-esimo neurone.

- I parametri da aggiustare sono i pesi w_{ji}.
- L'aggiornamento dei pesi avviene in direzione opposta al gradiente

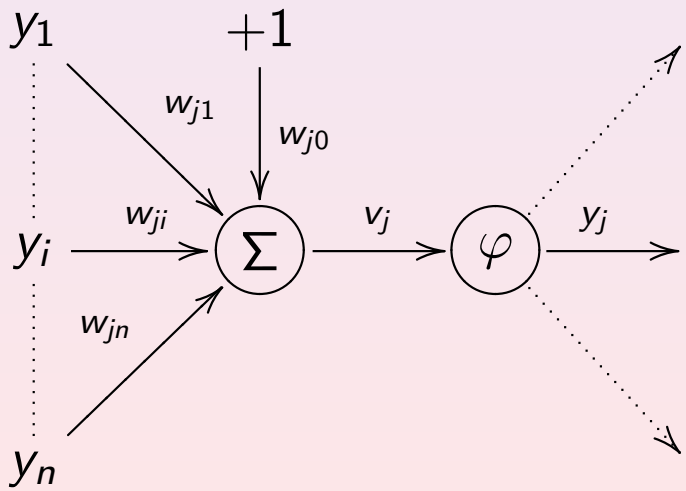
$$w_{ji}(t+1) = w_{ji}(t) - \eta \frac{\partial E(t)}{\partial w_{ji}}$$

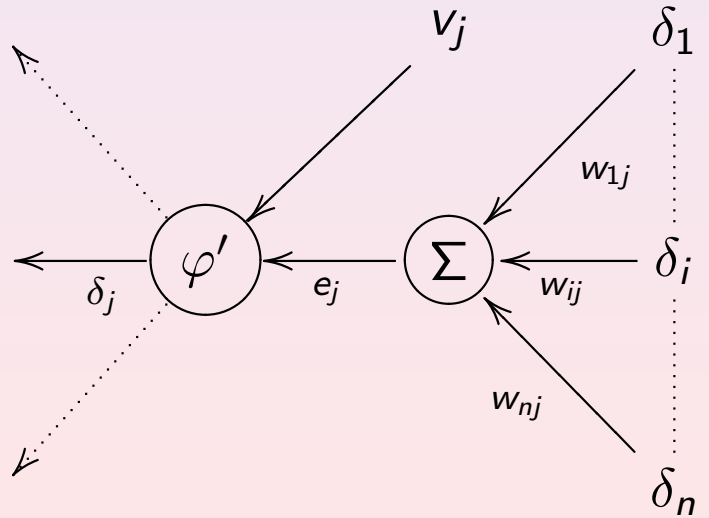
. . . Back-propagation

- Vediamo più in dettaglio come calcolare l'aggiornamento dei pesi

$$\Delta w_{ji} = -\eta \frac{\partial E}{\partial w_{ji}} = \eta \delta_j y_i$$

$$\delta_j = \begin{cases} \varphi'(v_j) e_j, & \text{se } j \text{ è un neurone di output} \\ \varphi'(v_j) \sum_k \delta_k w_{kj}, & \text{altrimenti} \end{cases}$$





Alcuni utili accorgimenti . . .

1. Aggiornamento dei pesi con momento
   - La scelta del fattore η di apprendimento influenza molto l'algoritmo di apprendimento (lentezza, oscillazioni)
   - Inserendo il momento nella regola di aggiornamento riusciamo ad ovviare a questi problemi

$$\Delta w_{ji}(t+1) = \underbrace{\alpha w_{ji}(t)}_{\text{momento}} + \eta \delta_j y_i$$

   - Il parametro α è preferibile sceglierlo nell'intervallo $[0, 1)$
2. Scegliere una funzione di attivazione antisimmetrica ovvero tale che $\varphi(-x) = -\varphi(x)$. Noi abbiamo usato la tangente iperbolica

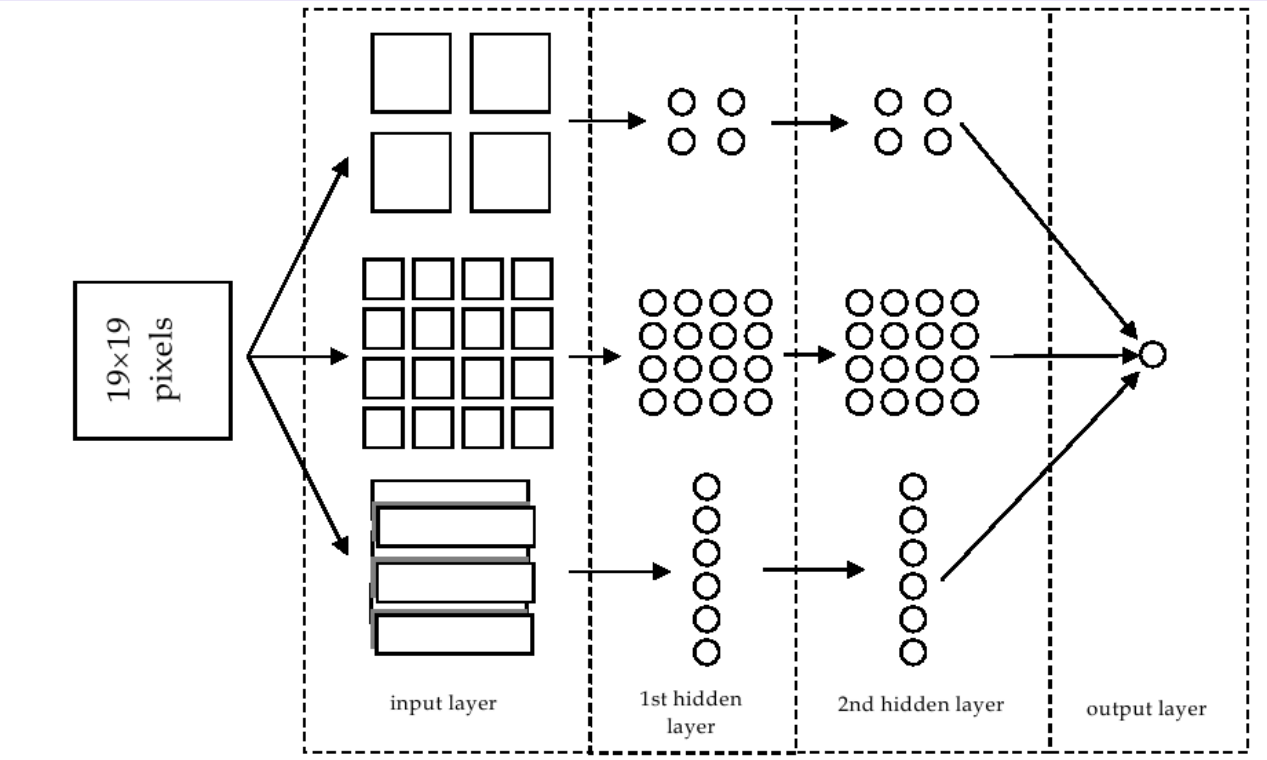
$$\varphi(x) = a \tanh(bx)$$

con $a = 1.7159$ e $b = \frac{2}{3}$. È inoltre facile ed efficiente calcolare $\varphi'(v_j) = \frac{b}{a}[a - y_j][a + y_j]$

. . . Alcuni utili accorgimenti

3. La risposta desiderata è consigliabile limitarla all'intervallo $[-1, +1]$ per evitare che l'algoritmo porti i pesi all'infinito.
4. È consigliabile anche rimappare l'input nell'intervallo $[-1, +1]$ anzichè limitarci ad avere solo valori positivi, per evitare fenomeni oscillatori della rete.
5. Anche l'inizializzazione dei pesi è critica. Pesi troppo alti portano a saturazione, mentre pesi troppo piccoli a lentezza. Il peso w_{ji} della rete si può ottenere da una variabile aleatoria uniformemente distribuita di media 0 e varianza $\frac{1}{m_j}$ con m_j il numero di connessioni entranti nel neurone j.

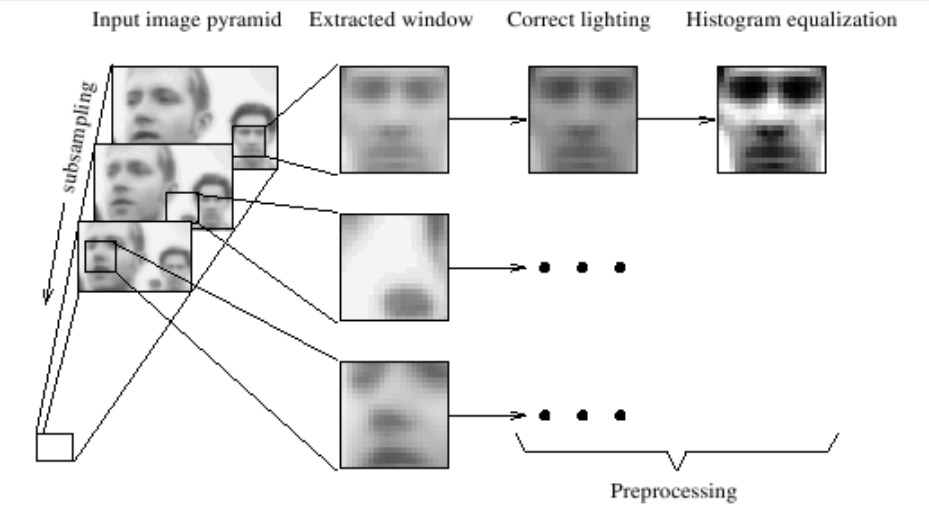
Struttura della rete neurale



- Funzione di attivazione: $\varphi(x) = 1.7519 \tanh\left(\frac{2}{3}x\right)$
- Input e risposta desiderata rimappati nell'intervallo $[-1, +1]$
- L'apprendimento avviene con learning rate $\eta = 0.05$ e momento $\alpha = 0.2$.

Normalizzazione dell'immagine

- Conversione in scala di grigi dell'intera immagine
- Per ogni campione da analizzare
  1. Fase di normalizzazione dell'illuminazione, per eliminare le differenze che un volto può avere se visto in condizioni di illuminazione differenti.
  2. Fase di equalizzazione dell'immagine, per utilizzare a pieno lo spettro delle intensità luminose ed aumentare il contrasto dell'immagine che evidenzia i dettagli del volto (occhi, bocca, naso, etc. . .)



Normalizzazione dell'illuminazione

- Si suppone che il contributo d'intensità luminosa dall'ambiente esterno sia lineare.
- Si calcola un piano di luminosità mediante approssimazione ai minimi quadrati dell'intensità luminosa dell'immagine. I coefficienti a_0, a_1, a_2 di questo piano devono minimizzare quindi la seguente funzione

$$\varphi(a_0, a_1, a_2) = \sum_{i=0}^{h-1} \sum_{j=0}^{w-1} \{Y[i,j] - (a_0 + a_1 i + a_2 j)\}^2$$

con w, h la larghezza e altezza dell'immagine e $Y^{h \times w}$ una matrice dell'intensità luminosa.

- Posto $Z[i,j] = 1 - (a_0 + a_1 i + a_2 j)$, si aggiusta l'immagine facendo la media aritmetica tra Y e Z.

Database di volti e addestramento

- 

MIT CBCL Face Database #1 (http://cbcl.mit.edu/software-datasets/FaceData2.html)
- immagini in formato 19×19 PGM
- Volti: 2.901
- Non-volti: 28.121
- volti di soggetti differenti ad orientamenti, illuminazione e scale diverse.
- un altro database è il Yale Face Database B (http://cvc.yale.edu/projects/yalefacesB/yalefacesB.html)
- Il training è stato fatto indicativamente per 20 epoche
- Ad ogni epoca è stato partizionato in modo random il dataset in 80% per il training e 20% per la validazione.
- Come test set abbiamo applicato il sistema a varie immagini.

Applicazione del filtro neurale

- Abbiamo a disposizione un filtro che analizza finestre di immagine di dimensione 19×19 e ritorna un valore ψ.
- Indicativamente se $\psi \longrightarrow 1$ abbiamo un volto, se $\psi \longrightarrow -1$ abbiamo un non-volto.
- Il filtro viene applicato all'immagine a scale diverse con riduzioni progressive del 20% (il training è stato fatto in modo da avere una rete tolleranete riduzioni di scala inferiori al 20%).
- Ad ogni scala, si applica il filtro ad ogni finestra 19×19 dell'immagine.
- Non è efficiente! Si può creare un filtro tollerante a spostamenti del volto entro un certo limite, in modo da ridurre il numero di finestre da analizzare.

Filtraggio di falsi positivi e negativi . . .

- Assumiamo che i volti generino un numero di rilevamenti maggiori nell'intorno del suo centro, rispetto ai falsi rilevamenti, e con un'intensità maggiore.
- Associamo ad ogni pixel p dell'immagine 3 informazioni: intensità di volto $I[p]$, numero di rilevamenti $\#p$, dimensione della finestra di rilevamento $D[p]$.
- Un rilevamento avviene su un certo pixel, se questo rappresenta il centro della finestra del filtro.
- Abbiamo poi due soglie: sull'intensità di volto γ e sul numero di rilevamenti π minimi necessari per avere un volto potenziale.
- Abbiamo più rilevamenti su uno stesso pixel p solo se sono avvenuti a scale diverse. In quel caso consideriamo come intensità $I[p]$ la loro media e per lo stesso discorso la dimensione della finestra $D[p]$ sarà una media delle finestre alle varie scale.

. . . Filtraggio di falsi positivi e negativi

- Per ogni pixel p si considera il suo intorno immediato (9 pixel), eliminando quelli con $I[p] < \gamma$. La nuova intensità del pixel è data dalla media delle intensità rimanenti. Il nuovo numero di rilevamenti come la somma dei rilevamenti nell'intorno.
- A questo punto consideriamo i rilevamenti in ordine decrescente di intensità eliminando quelli per cui $\#p < \pi$. Il primo rilevamento lo consideriamo buono ed eliminiamo ogni rilevamento successivo che si sovrappone ad esso entro una certa soglia di tolleranza. Iteriamo il procedimento.
- I rilevamenti considerati buoni sono i volti trovati.

Risultati sperimentali

- È stato effettuato un test su 12 immagini di vario tipo con un numero di volti diversi (#V=n. volti; #NV=n. non volti; #F=n. filtraggi)

Imm.	#V	#NV	#F	Imm.	#V	#NV	#F
1.jpg	33/36	4	1.706.018	2.jpg	03/07	5	603.133
3.jpg	19/22	2	1.137.523	4.jpg	17/31	3	706.913
5.jpg	11/11	0	645.128	6.jpg	01/01	2	379.426
7.jpg	01/01	0	120.208	8.jpg	06/07	0	691.796
9.jpg	07/11	4	1.320.321	10.jpg	17/27	2	1.419.163
11.jpg	00/00	2	760.204	12.jpg	00/00	0	161.063

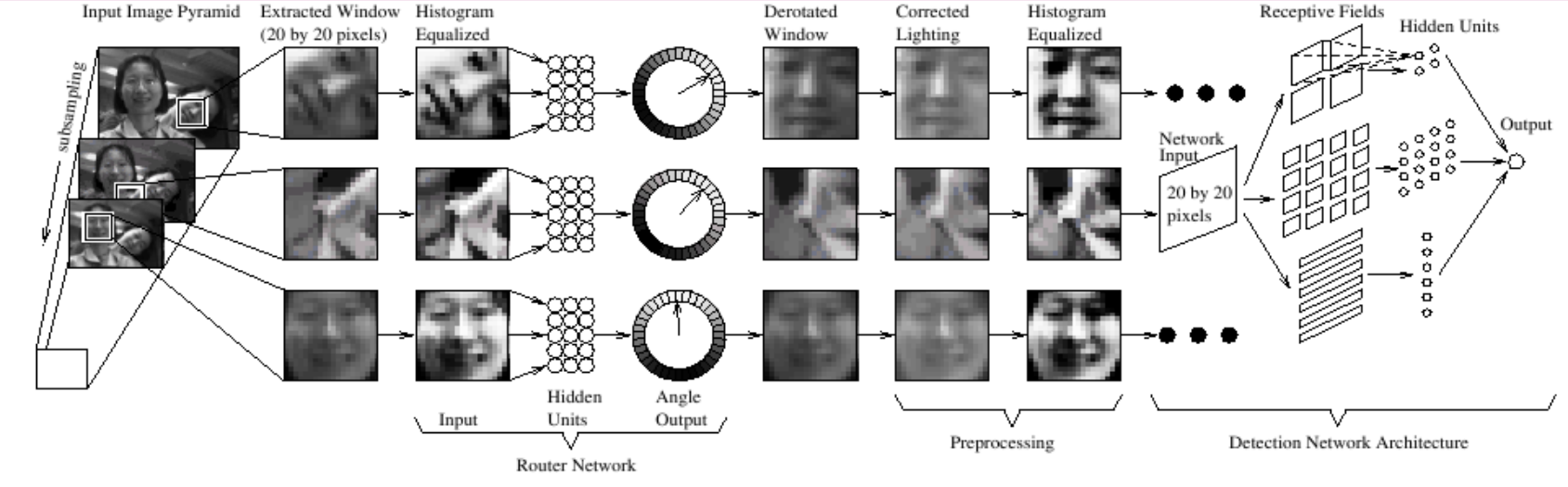
- Volti rilevati: 74,6%; Rilevamenti errati: 2.5E-6%
- Il numero di test fatti è piccola per poter stimare bene le prestazioni della rete, comunque possiamo vedere che il training ha prodotto una rete la cui priorità è non produrre falsi positivi.

Conclusioni

- Il rilevamento di volti è una area di ricerca molto attiva, negli ultimi anni abbiamo assistito ad un notevole avanzamento degli algoritmi per risolvere il problema, ma alcuni dei migliori algoritmi sono ancora computazionalmente troppo costosi per essere applicabili a sistemi real-time.
- Le reti neurali sono considerate un potente strumento per risolvere problemi di pattern recognition, tanto da essere applicabili in ogni parte di un sistema di riconoscimento di volti. [1]
- Il loro pregio è di essere robuste alle variazioni delle condizioni ambientali, e particolarmente performanti.
- Un loro difetto è che la loro progettazione comporta uno sforzo notevole nel trovare una struttura, un tuning dei parametri e una politica di training ottimali.
- Abbiamo poi visto solo un tipo ti applicazione delle reti neurali al face detection ma ce ne sono molte altre. [4, 7]

Sviluppi futuri

- Il sistema che è stato presentato è stato esteso per funzionare anche in presenza di volti ruotati [11].
- Esso è caratterizzato dalla presenza di reti neurali multiple.
  1. Una rete "router" processa l'input per determinare l'orientamento del volto
  2. Sulla base dell'orientamento si ruota l'immagine in modo che risulti dritta
  3. Si applica la tecnica vista in questa presentazione.



Bibliografia I


R. Chellappa, C. L. Wilson, and S. Sirohey.
Human and machine recognition of faces. a survey.
In *Proc. IEEE*, volume 83, 1995.


T. F. Cootes and C. J. Taylor.
Active shape models – smart snakes.
In *Proc. of British Machine Vision Conference*, pages 266–275, 1992.


L. C. De Silva, K. Aizawa, and M. Hatori.
Detection and tracking of facial feature by using facial feature model and deformable circular template.
IEICE Trans. Inform. Systems, E78-D(9):1195–1207, 1995.


R. Feraud, O. Bernier, and D. Collobert.
A constrained generative model applied to face detection.
Neural Process. Lett., 5:73–81, 1997.

Bibliografia II


S. H. Jeng, Y. M. Liao, C. C. Han, M. Y. Chern, and Y. T. Liu.
Facial feature detection using geometrical face model: An efficient approach.
Pattern Recog., 31, 1998.


M. Kass, A. Witkin, and D. Terzopoulos.
Snakes: active contour models.
In *Proc. of 1st Int. Conf. an Computer Vision, London*, 1987.


S. H. Lin, S. Y. Kung, and L. J. Lin.
Face recognition/detection by probabilistic decision-based neural network.
IEEE Trans. Neural Networks, 8:144–132, 1997.


D. Maio and D. Maltoni.
Real-time face location on gray-scale static images.
Pattern Recog., 33:1525–1539, 2000.

Bibliografia III


B. Moghaddam and A. Pentland.
Face recognition using view-based and modular eigenspaces.
In *Automatic Systems for the identification of Humans*, volume 2277, 1994.


H. A. Rowley, S. Baluja, and T. Kanade.
Neural network-based face detection.
IEEE Trans. Pattern Anal. March. Intell., 20:23–28, 1998.


H. A. Rowley, S. Baluja, and T. Kanade.
Rotation invariant neural network-based face detection.
In *Proc. IEEE Intl. Conf. on Computer Vision and Pattern Recog.*, pages 38–44, 1998.


T. Sakai, M. Nagao, and T. Kanade.
Computer analysis and classification of photographs of human faces.
In *Proc. First USA – Japan Computer Conference*, page 2.7., 1972.

Bibliografia IV


A. Samal and P. A. Iyengat.
Human face detection using silhouttes.
Int. J. Pattern Recog. Atificial Intell., 9(6), 1995.


K. C. Yow and R. Cipolla.
Feature-based human face detection.
Image Vision Comput., 15(9), 1997.


A. L. Yuille, P. W. Hallinan, and D. S. Cohen.
Feature extraction from faces using deformable templates.
Int. J. Comput. Vision, 8:99–111, 1992.