Relazione Progetto
Machine Learning
&
Sistemi Intelligenti per Internet

Autori: Benedetto Debora e Soldani Davide

Indice

Introduzione

Il progetto consiste nel riconoscimento di uno stato emotivo a partire dalla foto di una faccia.

Per svolgere il progetto è stata presa la decisione di utilizzare due tecnologie differenti di Machine Learning:

- Support Vector Machines
- Convolutional Neural Networks

In particolare le SVM sono state addestrate in modo da effettuare un riconoscimento fra sei classi:

- Anger
- Disgust
- Happiness
- Neutral
- Sadness
- Surprise

Per quanto riguarda le CNN invece sono state definiti 7 classi:

- Anger
- Disgust
- Happiness
- Neutral
- Sadness
- Surprise
- Fear

È stato inoltre aggiunta un'ulteriore funzione che permette il riconoscimento dello stato emotivo in real-time tramite l'uso della webcam.

Dataset

Inizialmente è stato utilizzato unicamente il dataset di espressioni facciali relativo a questo indirizzo:

https://github.com/muxspace/facial_expressions

Il dataset consiste in una raccolta di immagini per una competizione su kaggle avvenuta nel 2017.

I risultati ottenuti utilizzando esclusivamente tale dataset non hanno dato valori accettabili di accuracy,per questo motivo si è deciso di effettuare una ricerca al fine di individuare nuovi dataset da poter integrare con quello di kaggle.

In questo processo si ci è imbattuti più volte in pagine che richiedevano firme e accettazione di vincoli legali riguardo l'utilizzo delle foto, in alcuni casi non è stato possibile scaricare dataset perché erano a pagamento o forniti esclusivamente a scopo di ricerca per un professore universitario e quindi non disponibili per un normale studente.

Dopo un'intensa ricerca e un numero consistente di prove di addestramento sui sistemi (SVM e CNN) si è trovato ed utilizzato il dataset relativo a questo indirizzo:

http://www.consortium.ri.cmu.edu/ckagree/

SVM

Tecnologie utilizzate

Il progetto è stato realizzato utilizzando il tool Colaboratory offerto da Google, progettato per scopi educativi e di ricerca per machine learning. Il tool utilizza Jupiter come enviroment, e non è richiesto alcun setup.

Come detto sopra, nella prima fase del progetto è stato cercato il dataset adeguato da utilizzare. In particolare, per l'addestramento delle SVM c'è stata un'ulteriore lavorazione del dataset in modo tale da rendere omogeneo il numero di immagini per ogni emozione. Le emozioni "contempt" e "fear" non sono state considerate nell'addestramento, perché contenenti un numero di immagini eccessivamente limitato (circa 20). Il totale di immagini utilizzate è 1906.

Le immagini sono organizzate in cartelle, ognuna rappresentativa di una emozione.

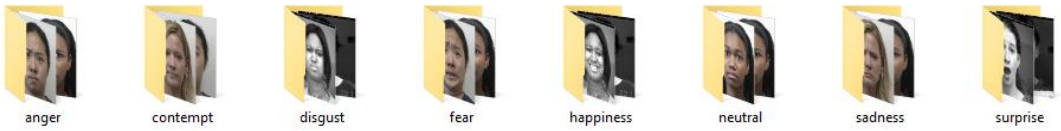


Figura 1: organizzazione del dataset in cartelle

La distribuzione del numero di immagini e le emozioni considerate sono quelle mostrate in figura:

```
dimensione del dataset:  anger 297
dimensione del dataset:  disgust 266
dimensione del dataset:  happiness 351
dimensione del dataset:  neutral 351
dimensione del dataset:  sadness 296
dimensione del dataset:  surprise 351
dim totale dataset:  1906
```

Figura 2: Dataset utilizzato per l'SVM

Si mostra inoltre esempi di immagine provenienti dal dataset:

Figura 2: example: happiness face

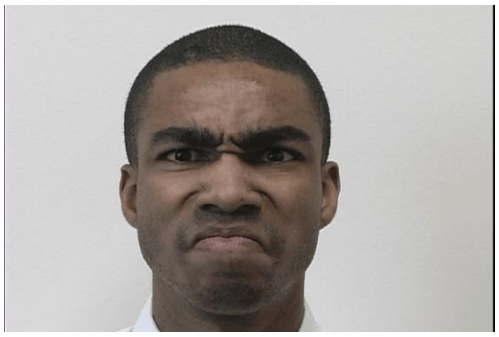


Figura 3: example anger face

Figura 3: example sadness face

Estrazione delle feature

Una volta realizzato il dataset, il prossimo obiettivo è stato quello di estrarre da ogni immagine le features adatte a rappresentarla.

Inizialmente è stata utilizzata la libreria dlib, in grado di identificare a partire dall'immagine di una faccia 68 punti con i quali si identificano le varie parti del viso: le sopracciglia, il naso, la bocca e i contorni del viso. viso.

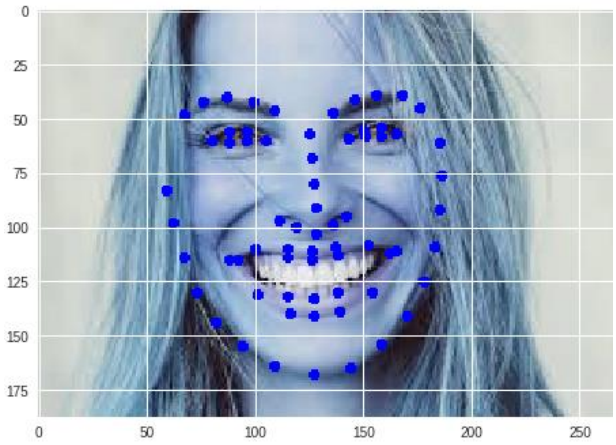


Figura 4: identificazione punti del viso da usare come features

Si identifica inoltre il punto centrale dell'immagine, da prendere come riferimento nell'elaborazione successiva. Nell'immagine il punto centrale viene disegnato in rosa.

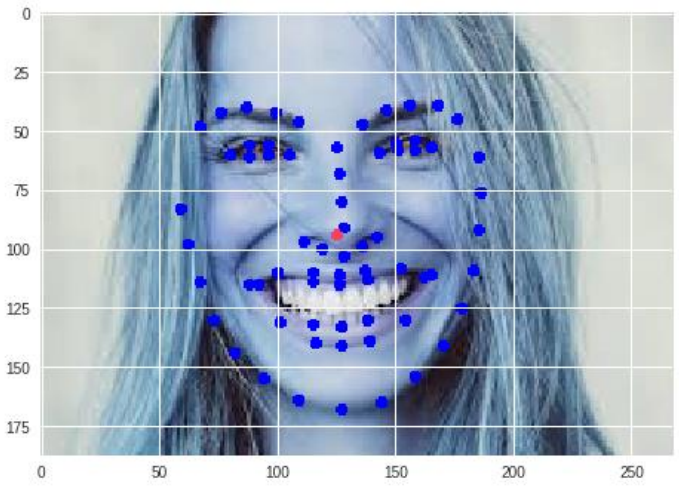


Figura 5: punto centrale dell'immagine, da usare come feature

Infine, a partire da punto centrale, si identifica per ogni punto (in figura, i punti sono disegnati in blu) la distanza dal punto centrale:

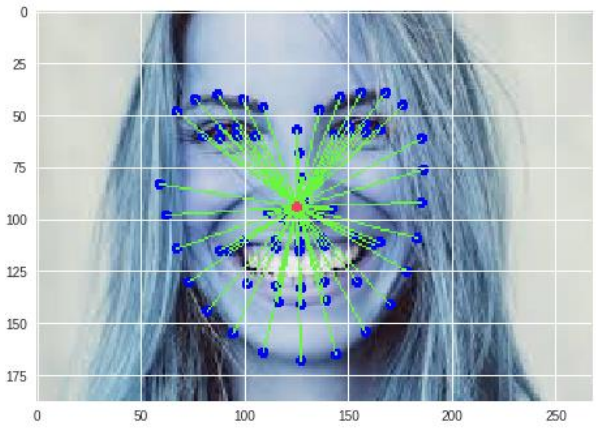


Figura 6: distanza dal punto centrale, da usare come feature

Addestramento SVM: tempistiche e risultati

Le immagini del dataset sono suddivise in cartelle, ognuna delle quali contiene immagini relative a una emozione. Le cartelle sono visitate una per volta, ogni immagine viene convertita in scala di grigi e vengono estratte le features dette sopra. A questa viene associala la label indicante l'emozione rappresentata.

Il classificatore utilizzato è una SVM.SVC con kernel lineare, e dopo vari tentativi è stato scelto il parametro C=1 come più adatto.

Il dataset comprende 1906 immagini, di cui 80% destinate al training set e il restante 20% destinate al validation set. In figura sono mostrate le metriche ottenute sul validation set:

	precision	recall	f1-score	support
anger	0.55	0.58	0.56	59
disgust	0.62	0.58	0.60	31
happiness	0.87	0.79	0.83	68
neutral	0.79	0.81	0.80	70
sadness	0.39	0.48	0.43	31
surprise	0.73	0.68	0.71	60
micro avg	0.69	0.69	0.69	319
macro avg	0.66	0.66	0.66	319
weighted avg	0.70	0.69	0.69	319

Figura 7: metriche SVM sul validation set

L'estrazione delle features impiega un tempo pari circa a 20minuti.

L'addestramento del classificatore e la sua valutazione impiega un tempo pari a 0,21 minuti.

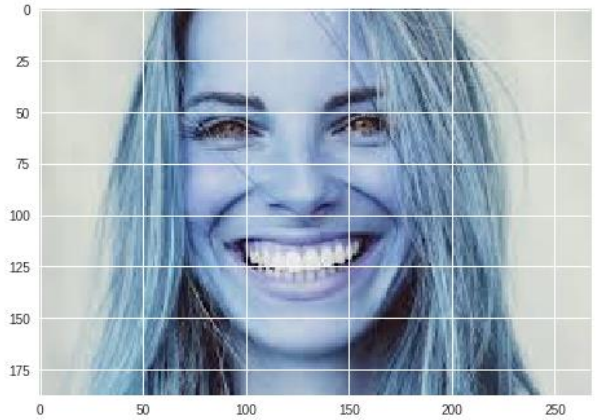
L'accuratezza è del 68%, se si utilizza la cross-validation 3fold allora l'accuratezza diminuisce a 56%.

Nel grafico in figura viene mostrato l'andamento dell'accuratezza utilizzando sia la cross-validation 3fold sia senza l'utilizzo della cross validation:

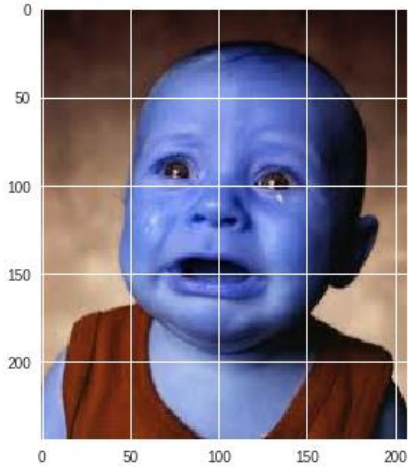
Figura 8: learning curves SVM con C=1

Risultati su immagini test prese su Google

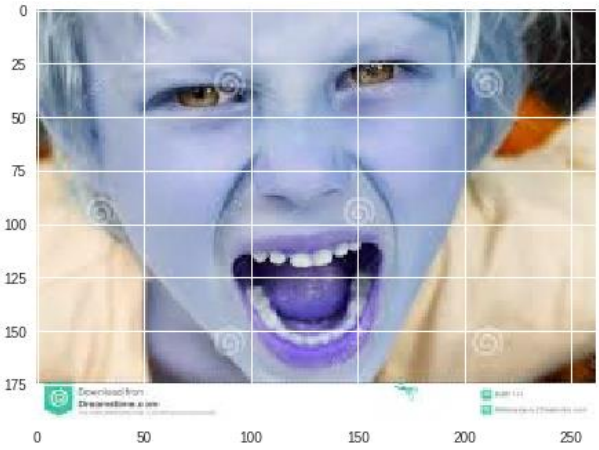
Sono infine mostrati i risultati del classificatore su immagini test prese da Google, indicando le probabilità di appartenenza a ciascuna classe e la classe predetta dal classificatore.



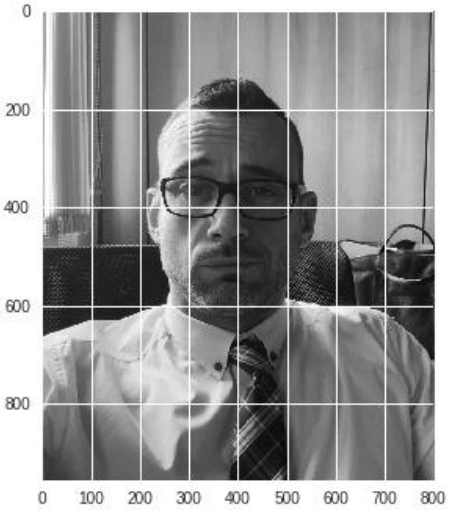
```
classes:  ['anger', 'disgust', 'happiness', 'neutral', 'sadness', 'surprise']
classes_prob:  [[ 2.00629997  3.01094032  5.5         0.75619119  4.10874228 -0.38217374]]
emotion_predicted:  happiness emotion_expected:  happiness
```



```
classes:  ['anger', 'disgust', 'happiness', 'neutral', 'sadness', 'surprise']
classes_prob:  [[ 2.17299767  5.29259708  2.12517798  1.57459607  4.33463119 -0.5
emotion_predicted:  disgust emotion_expected:  sadness
```



```
classes:  ['anger', 'disgust', 'happiness', 'neutral', 'sadness', 'surprise']
classes_prob:  [[ 5.1829402   3.07233383  3.35255099 -0.5         3.15606395  0.73611102]]
emotion_predicted:  anger emotion_expected:  anger
```



```
classes:  ['anger', 'disgust', 'happiness', 'neutral', 'sadness', 'surprise']
classes_prob:  [[ 3.06790354  1.97423304  1.8508708   5.43583727 -0.5         3.17115535]]
emotion_predicted:  neutral emotion_expected:  neutral
```

CNN

Tecnologie Utilizzate

Per la realizzazione della rete neurale convolutiva e l'elaborazione del dataset sono state utilizzate le seguenti tecnologie e librerie:

- Python 3.7
- Tensorflow
- Keras
- Tensorboard
- Numpy
- Pickle
- Open-CV
- Os
- Tqdm

Dataset

Per l'addestramento è stato inizialmente utilizzato unicamente il dataset di espressioni facciali di Kaggle.

I risultati ottenuti utilizzando esclusivamente tale dataset non hanno dato valori accettabili di accuracy. In particolare facendo più prove di addestramento l'accuracy rimaneva intorno al 66%.

Utilizzando in modo congiunto il dataset di kaggle ed il dataset ck i risultati di accuracy hanno subito un innalzamento arrivando intorno al 75%.

A questo punto si è iniziato a lavorare sull'architettura della CNN al fine di migliorare ancor di più l'accuracy.

Metodologia Iniziale e Rete Convolutiva addestrata da Zero

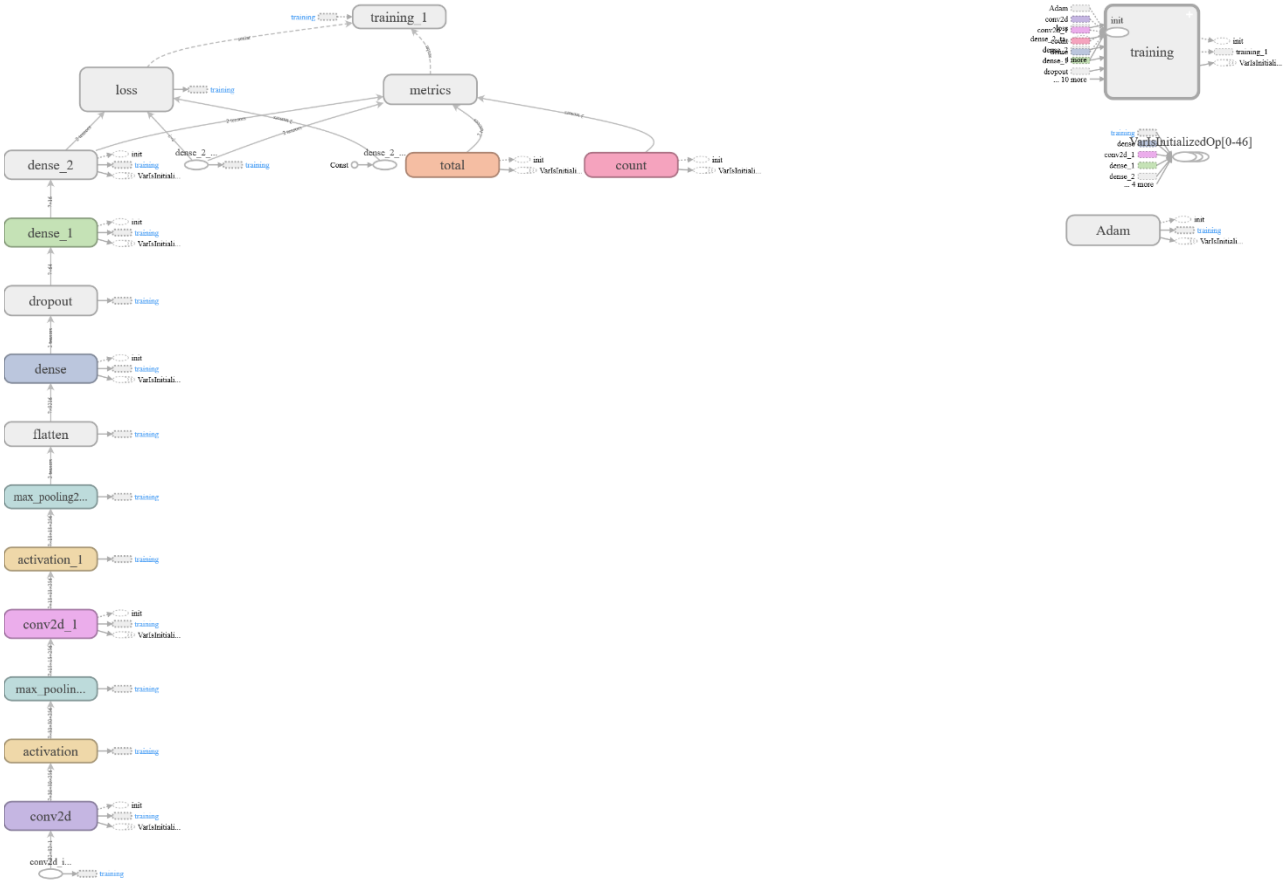
Inizialmente per capire l'uso ed il funzionamento delle reti neurali convolutive si è fatto uso del tutorial messo a disposizione da tensorflow e disponibile a questo indirizzo:

https://codelabs.developers.google.com/codelabs/tensorflow-for-poets/#0

Il tutorial fa utilizzare uno script "retrain.py" che effettua un addestramento basato su transfer learning della rete Mobilenet su un dataset passato come parametro

I risultati ottenuti con questa metodologia attestavano un'accuracy intorno al 70%

Si è quindi passati alla pratica andando a definire da zero una nuova rete neurale convolutiva la cui architettura è descritta nella seguente immagine:



Sono state effettate più prove variando i parametri della rete (lr, decay, momentum), metodo di ottimizzazione (adam, SGD) ed eseguendo più addestramenti andando a modificare a volte anche la grandezza delle immagini in input alla rete.

A seconda della scelta dei parametri e soprattutto della grandezza delle immagini sono stati ottenuti tempi di addestramento che variavano dai 15 minuti alle 8 ore.

In ogni caso con l'utilizzo di questa rete non si è riusciti ad ottenere un'accuracy che superasse il 79%

Transfer Learning

Nota: In questo paragrafo e in generale quando si ci riferisce ad "addestramento" di una rete con transfert learning, ci riferiamo esclusivamente all'addestramento dei nuovi strati aggiunti alla rete, l'addestramento per i restanti strati è bloccato

A questo punto non potendo aumentare più di tanto gli strati della rete convolutiva da addestrare "da zero" (poiché si sarebbero ottenuti tempi di addestramento non accettabili) si è iniziato ad effettuare delle prove di utilizzo del tranfer learning, in particolare fra le tante reti disponibili inizialmente sono state effettuare prove di utilizzo della rete Mobilenet ed in seguito si ci è concentrati sull'utilizzo della rete VGG-Face.

Mobilenet

MobileNet è una rete neurale addestrata per riconoscere e classificare 1000 oggetti "generici". Per vere un'idea della profondità della rete di seguito è mostrata un'immagine di tutti gli strati utilizzati:

Alla rete mobilenet è stato eliminato l'ultimo strato e sono stati aggiunti i seguenti strati:

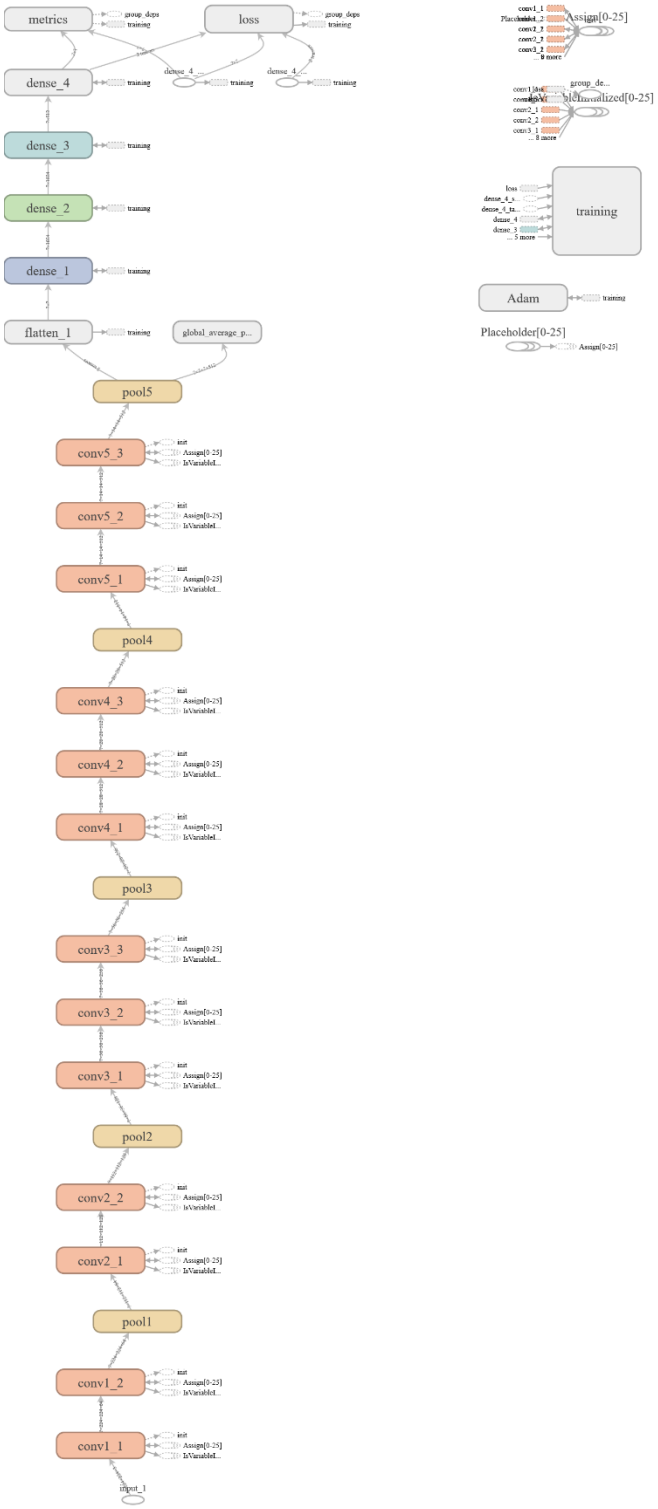
- GlobalAveragePooling2D()
- Dense(1024,activation='relu')
- Dense(1024,activation='relu')
- Dense(512,activation='relu')
- Dense(7,activation='softmax')

L’ultimo presenta 7 neuroni in quanto è il numero delle classi da identificare.

Sono state eseguite più prove anche in questo caso, modificando parametri e dimensioni dell’input elaborato ma pur avendo utilizzato una rete preaddestrata, i risultati sono stati uguali se non peggiori a quelli ottenuti con la rete neurale che si era definita a mano “da zero”.

VGG-Face

A questo punto si è passati quindi alla ricerca di una rete neurale che fosse addestrata in particolare sul riconoscimento facciale ed è stata trovata la rete VGG-Face la cui architettura (con l’aggiunta di nuovi strati) è mostrata nella prossima pagina.



Alla rete come solito per il transfer learning è stato eliminato l'ultimo strato e sono stati aggiunti in prima istanza i seguenti strati:

- Flatten()(x)
- Dense(1024,activation='relu')
- Dense(512,activation='relu')
- Dense(7,activation='softmax')

Eseguendo degli addestramenti “di prova” con immagini a bassa risoluzione (addestramenti che quindi impiegavano circa 15 minuti l’uno) si è passati ad utilizzare la seguente struttura le cui prestazioni risultavano migliori:

- Flatten()(x)
- Dense(1024,activation='relu')
- Dense(1024,activation='relu')
- Dense(512,activation='relu')
- Dense(7,activation='softmax')

Utilizzando questa rete ed eseguendo un addestramento con immagini di grandezza (224, 224, 3) e di durata 10 ore sono stati ottenuti risultati ottimi. Tali risultati sono mostrati nella seguente tabella:

	precision	recall	f1-score	support
Class 0	0.73	0.76	0.75	207
Class 1	0.77	0.83	0.79	230
Class 2	0.84	0.53	0.65	30
Class 3	0.98	0.92	0.95	4034
Class 4	0.92	0.99	0.95	5019
Class 5	0.87	0.55	0.67	210
Class 6	0.89	0.69	0.78	318
micro avg	0.93	0.93	0.93	10048
macro avg	0.86	0.75	0.79	10048
weighted avg	0.93	0.93	0.93	10048
samples avg	0.93	0.93	0.93	10048

L’accuracy media stimata sul test_set è del 93%

Grafici ultimo addestramento

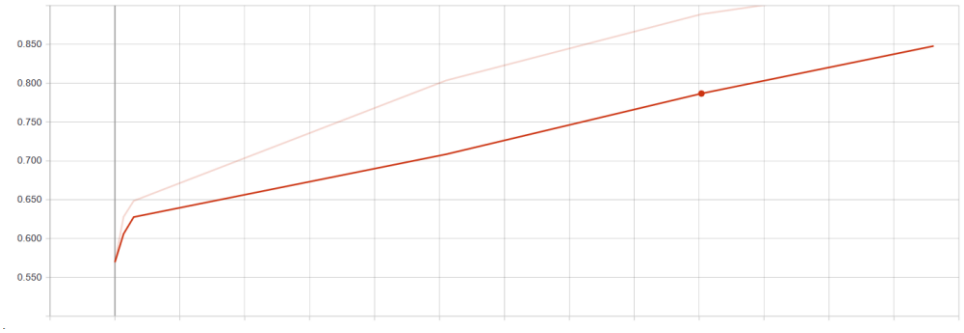


Grafico Accuracy su training set durante l'addestramento

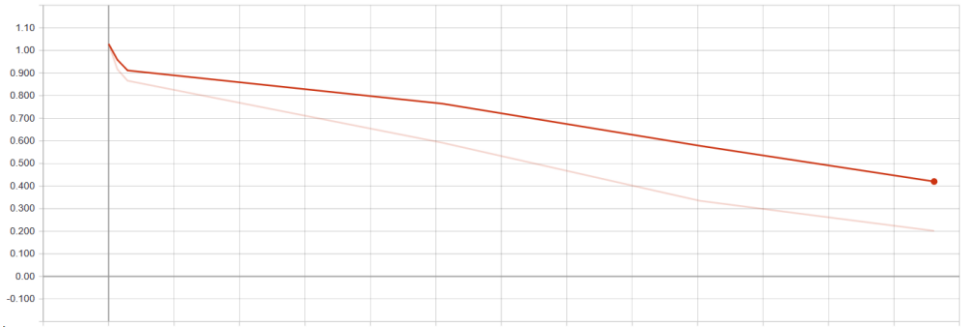


Grafico Loss su training set durante l'addestramento

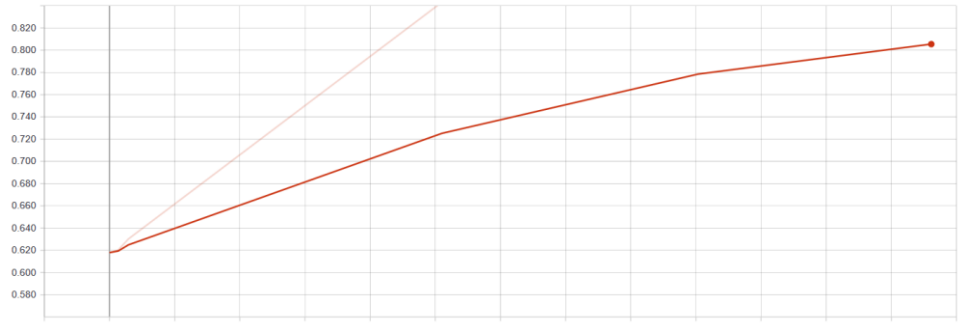


Grafico Accuracy su validation set durante l'addestramento

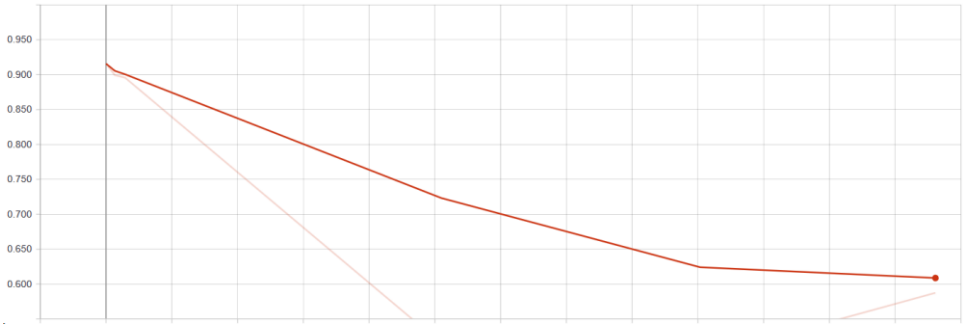
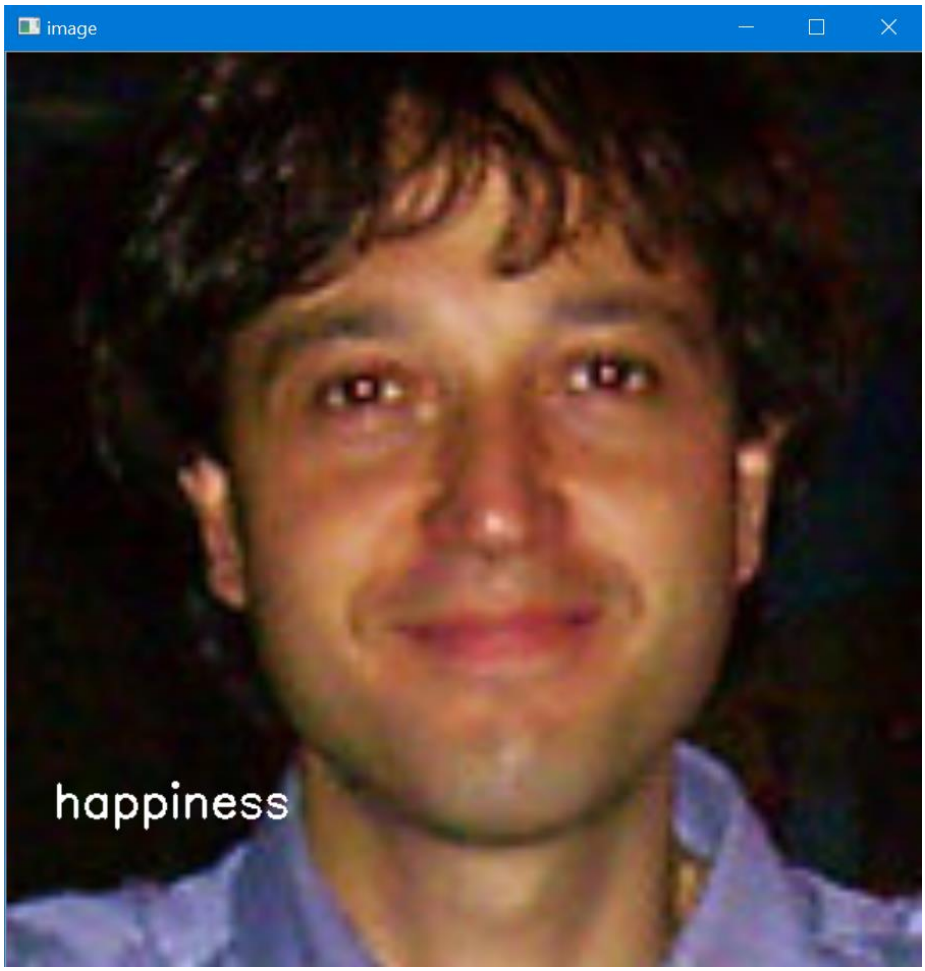
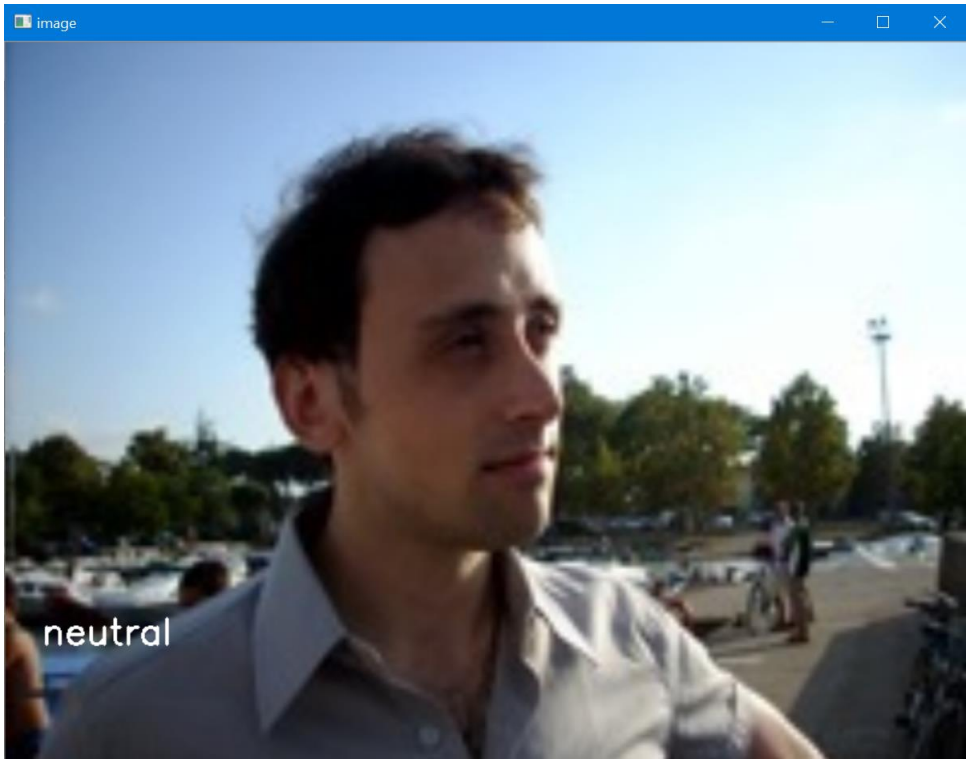


Grafico Loss su validation set durante l'addestramento

Test su foto trovate in rete





Uso della CNN in coordinamento con una videocamera e riconoscimento in real-time

Facendo uso della libreria Open-CV è stato possibile utilizzare la webcam del pc.
Caricando il file .model della rete neurale addestrata è stato possibile prevedere e stampare a schermo in real-time l'espressione facciale catturata dalla webcam.

Conclusioni

Le difficoltà dell'utilizzo delle SVM confrontate con le CNN riguardano principalmente l'estrazione delle feature e la decisione di quali di esse utilizzare al fine di effettuare una corretta classificazione dell'immagine. Di contro le CNN estraendo automaticamente le feature non incorrono in questa problematica.

Per quanto riguarda i tempi di addestramento abbiamo ottenuto tempi nettamente inferiori utilizzando le SVM, mentre con le CNN alcuni addestramenti hanno richiesto fino a 10 ore.

Per quanto riguarda la precisione le CNN si sono mostrate più accurate nel predire l'emozione rispetto alle SVM.