Report

**Introduzione**

Il progetto realizzato si occupa del riconoscimento di esercizi adatti agli anziani attraverso l'utilizzo di librerie definite per il campo della Computer Vision, in particolare OpenCV, OpenPose e Scikit-Learn.

Prima di realizzare ed eseguire gli script presenti all'interno del progetto, è necessaria l'installazione delle dipendenze, ovvero strumenti necessari al corretto funzionamento degli stessi (tra questi vi sono alcuni tra quelli citati precedentemente).

Il primo step è stato quello di individuare quali esercizi potessero essere adatti per gli anziani e, dopo una ricerca approfondita su internet, abbiamo concordato sui seguenti sei esercizi per la “ginnastica dolce”, utili per l’addestramento del sistema:

* circonduzione delle braccia in piedi per allenare le braccia;
* jumpingjack per allenare braccia, gambe e addominali;
* sollevamento di una gamba da seduti per allenare le gambe;
* movimento delle braccia a 90 gradi per sollecitare i muscoli pettorali;
* squat per allenare le gambe;
* portare le gambe al petto alternate in piedi per allenare i muscoli dell’equilibrio.

Conseguentemente abbiamo raccolto e realizzato numerosi video per ogni esercizio, dei quali l’ 80% verranno destinati nei passaggi finali al training set e la restante percentuale al test set (i filmati considerati hanno 30 fps). A seguire, i video sono stati convertiti in una sequenza di frame numerati (attraverso l’esecuzione di uno script *python*) e salvati in cartelle dedicate contrassegnate dal nome dell’esercizio e da un numero. Infine, è stato realizzato un file di testo (“*valid\_images.txt*”) che specifica per ogni cartella il nome e l’intervallo dei frame contenuti, ovvero il numero corrispondente al frame iniziale e al frame finale.

(Le cartelle realizzate vengono date in input al primo script)

*Legs\_1*

*166 675*

*Legs\_2*

*1 638*

*Legs\_3*

*34 488*

*…*

Nel file di configurazione del progetto(“config/*config.txt”)* inizialmente sono stati inseriti i seguenti parametri:

* *image\_filename\_format:* il formato dei frame;
* *skeleton\_filename\_format:* il formato dei documenti di testo che vengono generati dopo il riconoscimento dello scheletro della persona presente all’interno del frame;
* *window\_size:* il numero di frame adiacenti per l’estrazione delle features;
* *classes:* le classi che verranno considerate nel progetto ( *'jumpingjack', 'squat', 'arms', 'legs', 'foldedLegs', 'shoulders'* ).

**s1\_get\_skeletons\_from\_training\_imgs.py**

Il codice si occupa di prendere in input ciascuna cartella all'interno dataset e, per ciascuno dei frame all'interno di essa, identifica lo scheletro della persona nell'immagine e salva in un documento di testo le seguenti informazioni:

- un numero identificativo associato a ciascuna classe(esercizio)

- un numero identificativo associato ad ogni video

- un numero identificativo associato ad ogni frame del video

- la stringa relativa alla classe

- la directory del video e il nome del frame in questo modo: <video\_directory>/<nome\_frame>.jpg

- per ciascun joint dello scheletro vengono salvate le relative coordinate

Tutte le informazioni (tranne le coordinate dei joints) relative ad ogni frame vengono salvate in un file txt definito “*images\_info.txt”*, ma questo risulta essere inutilizzato in tutto il progetto.

Il riconoscimento dello scheletro viene effettuato utilizzando la libreria OpenPose, in particolare scegliendo il modello e la dimensione dell'immagine.

Le immagini di cui effettuare il riconoscimento vengono indicate dal file "*valid\_images.txt*" contenuto in entrambi i set.

Lo script fa riferimento al file di configurazione per i seguenti parametri:

* da openpose:
  + *model*: si definisce il modello da cui effettuare il riconoscimento dello scheletro
  + *img\_size*: dimensioni dell’immagine da analizzare
* input:
  + *images\_description\_txt*: percorso dove salvare il file *valid\_images.txt*
  + *images\_folder*: percorso da cui prendere le cartelle contenenti i frame da analizzare
* output:
  + *images\_info\_txt*: percorso dove salvare il file *images\_info.txt*
  + *detected\_skeletons\_folder*: percorso dove salvare i file relativi a ciascun frame
  + *viz\_imgs\_folders*: percorso dove salvare le immagini che mostrano gli scheletri riconosciuti

Funzione da citare: *process\_images*

 input:

* *images\_loader*: Lettore di immagini, prende il riferimento di ciascuna immagine di cui effettuare il riconoscimento
* *DST\_DETECTED\_SKELETONS\_FOLDER*: directory dove sono presenti tutte le cartelle associate a ciascun video (all'interno contengono i frame)

         elaborazione:

Per ciascun frame contenuto in ciascuna cartella si effettua il riconoscimento dello scheletro della persona contenuta all'interno del frame attraverso il modello offerto da OpenPose. I joint e le relative connessioni vengono disegnate sul frame che viene visualizzato a schermo, successivamente le informazioni definite precedentemente vengono salvate in un file txt dedicato.

**s2\_put\_skeleton\_txts\_to\_a\_single\_txt.py**

Il codice si occupa di importare le informazioni contenute nei file di testo relativi a ciascun frame e di unirle all'interno di un unico documento di testo: “*skeletons\_info.txt*”.

Lo script fa riferimento al file di configurazione per i seguenti parametri:

* Input:
  + *detected\_skeletons\_folder*: percorso dove sono presenti i file txt relativi a ciascun frame
* output:
  + *all\_skeletons\_txt:* percorso dove salvare il file txt unificato *skeletons\_info.txt*

**s3\_preprocess\_features.py**

Questo script di codice si occupa di caricare i dati da *'skeletons\_info.txt'*, prendere i dati dal dataset ed effettuare il pre-processing delle features. I dati elaborati vengono inseriti in un due file .csv uno per le features non predette e uno per le features da predire.

Lo script fa riferimento al file di configurazione per i seguenti parametri:

* Input
  + *all\_skeletons\_txt*: percorso del file di testo unificato contenente i dati relativi a tutti i frame analizzati
* output
  + *processed\_features*: percorso del file csv delle features da non predire
  + *processed\_features\_labels*: percorso del file csv della feature da predire

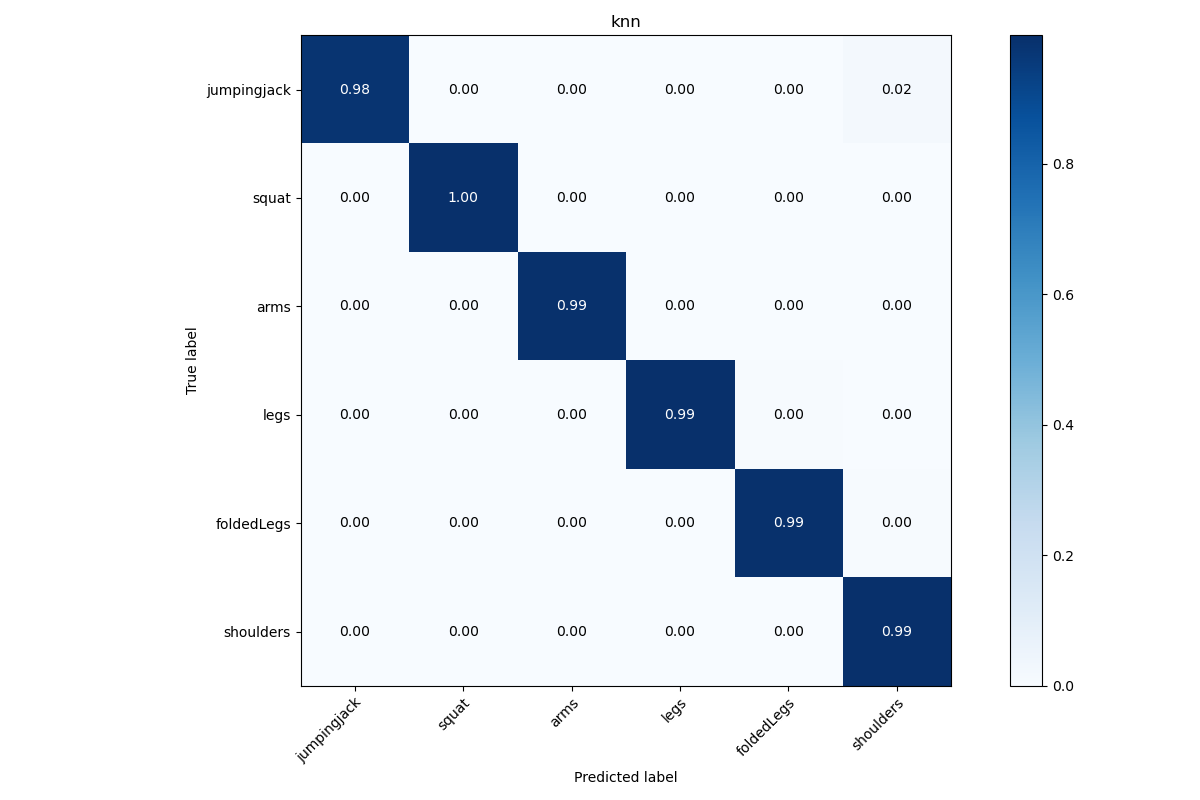
**s4\_train.py**

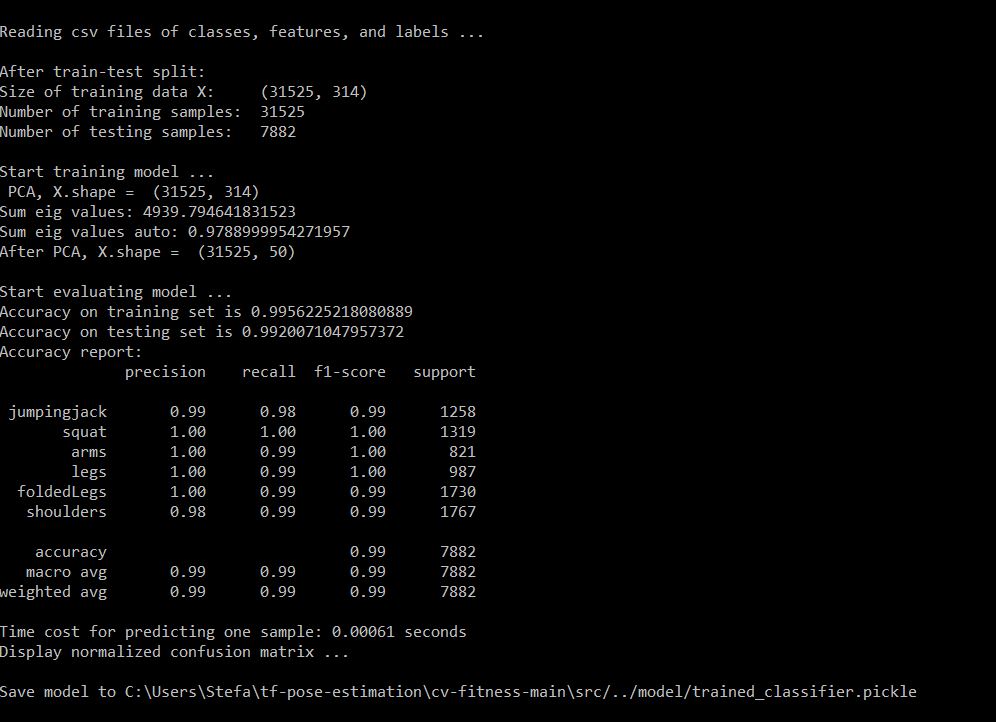
Lo script si occupa di caricare i file csv “*features\_X.csv”* e *“features\_Y.csv”*, effettuare lo split di entrambi i dataset, in modo tale da avere l’80% dei dati destinati al training set e i restanti per il test set. Il modello, quindi, viene sottoposto a fase di training e test con i nuovi dataset realizzati e viene elaborata l’accuratezza del modello in entrambe le fasi. Infine, viene elaborata e salvata la matrice di confusione relativa al modello e salvato il modello stesso. Il modello preso in considerazione in questo progetto è il KNN.

Lo script fa riferimento al file di configurazione per i seguenti parametri:

* input:
  + *processed\_features*: file csv che contiene i dati relativi alle features non predette
  + *processed\_features\_labels*: file csv che contiene i dati relativi alla feature da predire
* output:
  + *model\_path*: percorso di salvataggio del modello appreso

Qui di seguito sono riportati i grafici che mostrano la matrice di confusione relativa al modello appreso e il report sull’accuratezza del modello durante le fasi di training e test:



****

**s5\_run.py**

Lo script visualizza in real-time il flusso della videocamera, mostra i joints riconosciuti della persona presente nel filmato e usa il modello salvato per il riconoscimento degli esercizi (il nome dell’esercizio verrà visualizzato a schermo). Vi è inoltre la possibilità di effettuare il riconoscimento partendo da una cartella di immagini o da un video pre-caricato. A lato dello schermo sono stati visualizzati i possibili esercizi da effettuare e che vengono riconosciuti dal modello.

La libreria OpenPose viene utilizzata per il riconoscimento dei joints della persona presente nel filmato.

Lo script fa riferimento al file di configurazione per i seguenti parametri:

settings:

* source:
  + *webcam\_max\_framerate*: framerate massimo acquisito dal programma
* openpose:
  + *model*: percorso del modello utilizzato per il riconoscimento degli esercizi
  + *img\_size*: dimensioni dell’immagine visualizzata a schermo
* display:
  + *desired\_rows*: parametro di ridimensionamento dell’immagine per la visualizzazione a schermo

output:

* *video\_name*: percorso del video salvato durante l’esecuzione del programma
* *video\_fps*: fps del video salvato
* *skeleton\_folder\_name*: percorso dei file txt salvati, contenenti le informazioni relative ai frame durante l’esecuzione del programma

funzioni da citare:  *Draw\_result\_img*

Input:

* *img\_disp*: frame della videocamera
* *humans*: oggetto che fa riferimento agli scheletri da visualizzare a video
* *skeleton\_detector*: oggetto che permette di “disegnare” sul frame i joint riconosciuti
* *label\_class*: il nome dell’esercizio predetto
* *multi\_person\_classifier:* oggetto che fa riferimento al modello utilizzato per il riconoscimento

output:

*img\_disp*: immagine che visualizza i joint riconosciuti e il nome dell’esercizio predetto

Elaborazione:

Funzione che ridimensiona la finestra per la visualizzazione, disegna lo scheletro delle persone e visualizza il nome dell’esercizio su una barra bianca posta al di sopra della videocamera.