Tesina per il corso di

Image Processing and Computer Vision

A cura di
Francesco Alagna s265350
Francesca Scarrone s267467

F&F Surveillance System

25 febbraio 2021

Panoramica

L'obiettivo di questa tesina è quello di realizzare **un'applicazione web client-server** in grado di riconoscere i volti per un sistema di sicurezza domestico.

Come funziona

Dal lato client un'interfaccia (in JavaScript, CSS e HTML) semplice e intuitiva permette all'utente di impostare i familiari, mentre dal lato server, invece, ci si occuperà del processo di **riconoscimento facciale** attraverso algoritmi e librerie in JavaScript.

Quando il sistema individua un volto, se questi non è conosciuto, verrà inviata una **notifica di allerta**.

Backend

L'applicazione utilizza un database in cui sono contenuti i dati e le foto delle persone che saranno divisi in profili di persone **conosciute** e **sconosciute**. La libreria face-api.js si occuperà del **riconoscimento** e il confronto dei volti.

Sommario

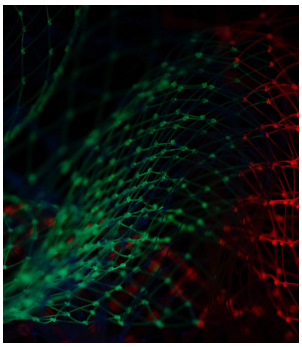
Testing e obiettivi

Il sistema è stato testato tramite **webcam**, utilizzando le nostre foto come membri della famiglia e abbiamo chiesto a persone a noi vicine di farsi riprendere per verificare che il sistema funzioni correttamente. Il nostro obiettivo era quello di superare almeno il **50% di accuratezza** nel riconoscimento, puntando però a un risultato ottimale superando il 70/80%.

Il progetto è disponibile su GitHub al seguente link.

Machine learning

I sistemi di machine learning sono ormai richiesti in molte applicazioni e il loro impiego è vario. L'idea è quella di estrarre **informazioni** dai dati costruendo delle regole, o meglio, dei **pattern**.



Pre-processing

Un sistema di machine learning affronta diverse fasi nel suo sviluppo, la fase di pre-processing, è quella in cui a partire dai dati si ottengono dei **vettori di caratteristiche**, il cui insieme va a definire un pattern, ossia un adattamento di queste caratteristiche che, in base a quelle scelte, può essere usato sia per definizioni **quantitative** (lunghezze, aree), sia per descrizioni **qualitative**.

Per cui è necessario scegliere accuratamente le caratteristiche da tenere in considerazione, questa scelta deve tener conto di tre aspetti:

- **Coverage**: la copertura che permette di distinguere le informazioni rilevanti
- **Concision**: la rimozione delle caratteristiche che non riducono la coverage
- **Directness**: caratteristiche ideali (chiare) sono utili indipendentemente dalla predizione

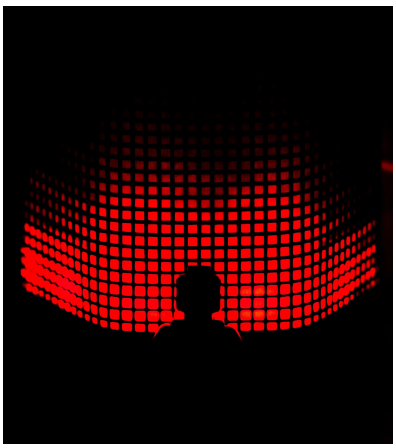
Una volta definite le caratteristiche è possibile definire i pattern, ai quali poi si possono applicare gli algoritmi di machine learning che costruiranno un modello attraverso algoritmi di **clustering** (raggruppamento) e **classification** (aggiungere etichette e fare delle predizioni).

Clustering

Lo scopo della **segmentazione** o clustering è di separare l'immagine in oggetti raggruppando gruppi **simili di pixel**, questi sono rappresentati con un unico token. Oltre a separare le regioni dell'immagine, permette anche di **riassumere i dati** (rappresentare grandi vettori continui con il numero di cluster), di **contare** (costruire istogrammi, colore) e di **predire** (immagini dello stesso cluster possono avere la stessa etichetta).

Classification

Gli algoritmi di machine learning analizzano e **pesano** le caratteristiche scegliendo dei parametri in modo da massimizzare la performance: tali parametri rappresentano **l'apprendimento**.



La fase di classification è dove avviene il **training** dell'algoritmo, si usa un **dataset di immagini** molto ampio da cui estrarre le caratteristiche da etichettare e una volta addestrato il modello, si usano delle immagini di test che dovranno essere etichettate **automaticamente**.

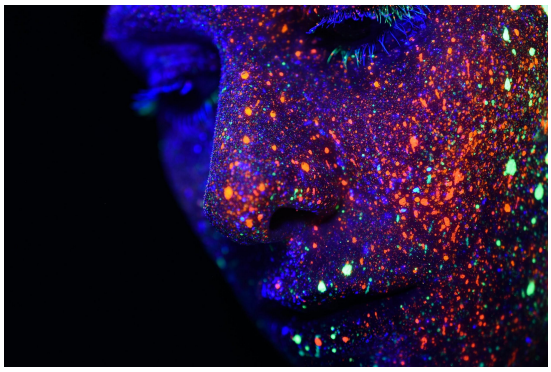
Il classificatore è quel "componente" che cerca di **apprendere** il peso e i vari parametri delle caratteristiche. Esistono diversi classificatori: k-nearest neighbor, Naive-Bayes, alberi di decisione, classificatore di Haar/Viola-Jones, Support Vector Machine, reti neurali, etc….

È importante sottolineare che le immagini possono contenere dati in grado di **disturbare** il riconoscimento (ad esempio le occlusioni) inoltre la stessa immagine scattata in condizioni diverse può portare a una **diversa etichettatura** (ad esempio le condizioni meteorologiche, l'illuminazione, la scala, occlusioni, etc...).

Quindi vige il **No Free Lunch Theorem**: secondo cui non esiste un classificatore intrinsecamente migliore di un altro, e si devono fare delle assunzioni per generalizzare.

Face recognition

Gli algoritmi di machine learning sono molto efficaci per riconoscere i **volti** presenti in un'immagine o in un flusso video (analizzandolo frame per frame).



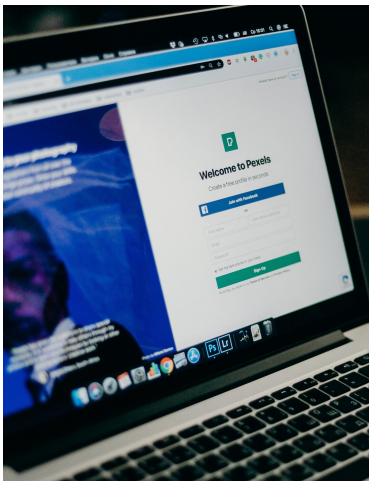
Ci sono per diversi **fattori** che però rendono il riconoscimento difficile: bilanciamento del bianco, illuminazione variabile, accessori (occhiali), età, make up, capelli, barba, etc….

Per cui ci sono diversi **approcci** per riconoscere i volti: alcuni basati sulla **conoscenza** (caratteristiche e relative dimensioni e posizione), altri su **caratteristiche invarianti** (come occhi, bocca, colore della pelle), altri si basano sul **template** matching, o sulle "eigenfaces", reti neurali.

Haar Feature-based Cascade Classifier

Utilizzato in OpenCV, è adatto al riconoscimento di **volti ed oggetti** (rigidi) e si basa sugli alberi di decisione usando come caratteristiche le wavelet di Haar. Le caratteristiche Haar-like sono solitamente **legate a bordi**: linee e zona centrale (center surround). Queste caratteristiche sono specificate da forma (spigoli), posizione (nell'immagine) e scale (può variare in base all'immagine, e in base al training) e sono particolarmente adatte per riconoscere appunto oggetti rigidi (squadrati) e volti ma ha bisogno di un **grande dataset di immagini** per il training.

Web applications



Il paradigma delle applicazioni web è sempre più complesso, la sua struttura è composta di **diversi layer** a partire "dall'interfaccia" a livello più alto (ad esempio un browser) fino al database server (a livello più basso) passando quindi attraverso l'architettura di internet e l'application server.

Questa architetture permette di **delegare al server** la gestione di dati e l'esecuzione di operazioni pesanti in modo da chiedere ai client meno specifiche possibili: il server deve quindi gestire e **rispondere alle richieste** dei possibili molteplici client attraverso il protocollo HTTP.

Tools

Lo sviluppo di applicazioni web, per quanto sia complesso, è però supportato da diversi **software e servizi** che permettono di realizzare applicazioni, interfacce e server in maniera intuitiva anche se un po' limitata (ad esempio Wordpress), ma anche, da diversi tool e linee guida che aiutano, chi non volesse essere "limitato", a strutturare la propria applicazione sia dal lato client (**front-end**, ad esempio il design dell'interfaccia grafica) sia dal lato server (**back-end**, la creazione di database o DBMS, la definizione della business logic, etc...).

Inoltre il progredire di **linguaggi di programmazione** (come JavaScript) e l'integrazione di **librerie** (ad esempio quelle in Python) permettono di aggiungere molte features alle applicazioni web in modo molto più semplice e veloce.

face-api.js library

Questa API (scritta in JavaScript) permette di risolvere i problemi legati sia alla **face detection** in un'immagine o in uno stream video sia alla **face recognition** vera e propria.

Face Detection

Per quanto riguarda il primo dei nostri problemi, la face detection, face-api.js permette di utilizzare diversi modelli di "**detectors**" utilizzabili in base agli obiettivi.

SSD Mobilenet V1

SSD (Single Shot Multibox Detector), è molto accurato e si basa su **CNN** (Convolutional Neural Network) del MobileNet V1. La rete neurale, per ogni faccia, calcola la **posizione** nell'immagine e ne ritorna i **bounding box**: questo detector, addestrato con WIDERFACE dataset e con i pesi dati dalla seguente repository, mira a ottenere **un'accuratezza elevata** che però comporta un maggiore tempo di deduzione.

Tiny Face Detector

Questo detector è molto **efficiente** e permette il rilevamento del volto in **real-time**, ovviamente è più veloce e richiede meno risorse rispetto al precedente, ma perde sensibilmente **l'accuratezza** per il rilevamento di **volti piccoli**. In ambienti mobile o web è il più indicato proprio per queste caratteristiche.

MTCNN

Il Multi-task Cascaded Convolutional Neural Networks rappresenta una **via di mezzo** tra i precedenti, offrendo più spazio per la configurazione, rendendolo quindi **adattabile** a diversi contesti (anche per riconoscere una grande varietà di dimensioni di bounding box). Si tratta di un 3-stage CNN che calcola simultaneamente sia i landmark (5 punti) sia i bounding box sia gli "scores" (la distanza) per ogni volto. MTCNN è stato presentato nel seguente paper, ma è ancora in **fase di sperimentazione**.

68 Point Face Landmark Detection Models

I landmarks sono i punti che vengono utilizzati nel calcolo della "distanza" tra il volto trovato nell'immagine e i volti con cui si fa il confronto, inoltre essi permettono di **centrare la bounding box** del volto in modo da rendere i calcoli ancora più accurati.

Face Recognition

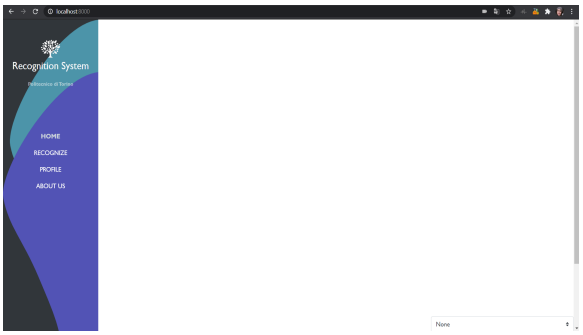
Risolto quindi il primo problema, resta solo il **confronto tra i risultati** ottenuti dai detectors. I **descriptors** sono dei vettori di caratteristiche usati per descrivere le caratteristiche di un volto per cui è possibile calcolare una semplice **distanza euclidea** tra due descrittori e usare una **soglia** per decidere se i volti appartengono alla stessa persona o meno.

La rete neurale è la medesima utilizzata in un'altra API (face-recognition.js) mentre i pesi per le caratteristiche sono stati addestrati da davisking e dagli archivi di modelli con accuratezza di predizione del 99.38%.

Face Expression Recognition Model

Questa API permette inoltre riconoscere con sufficiente accuratezza le **espressioni dei volti**, il modello usato è molto **leggero e veloce**, ed è stato addestrato con dataset pubblici e immagini prese dal web (in questo caso accessori come gli occhiali possono ridurre l'accuratezza).

La nostra pagina web



Interfaccia grafica F&F Surveillance System

La pagina web ha uno stile semplice, intuitivo e minimalista realizzato con l'utilizzo di linguaggi quali JavaScript, HTML e CSS.

Una volta aperta la pagina web, dal menu laterale possiamo accedere alle schede Home, Recognize, Profile, About Us.

La scheda Home è la schermata principale, dove potremo scegliere la sorgente video e visualizzare lo stream video.

Dalla sezione Recognize abbiamo la possibilità di visionare le immagini dei volti delle persone che non sono state riconosciute. L'utente, selezionando l'immagine, ha la possibilità di svolgere diverse azioni: può cancellare la foto, associarla a un membro della famiglia esistente oppure decidere di creare un nuovo profilo, qualsiasi di queste azioni ricaricherà la pagina in modo da avere tutti i dati aggiornati.

Nella schermata Profile si possono trovare tutti i membri della famiglia ed è possibile modificare i dati inseriti per quell'account, eliminarlo e visualizzare l'accuratezza dell'algoritmo: per valutare quest'ultima calcoliamo il rapporto tra il numero di volte in cui il volto è stato rilevato e il numero di volte in cui il volto non è stato riconosciuto e specificato dall'utente nella pagina Recognize.

Per il rilevamento e il riconoscimento dei volti abbiamo integrato nel nostro progetto l'API face-api.js e i seguenti modelli descritti precedentemente: il detector SSD Mobilenet V1, il 68 Point Face Landmark Detection e infine il modello di Face Recognition.

Innanzitutto abbiamo creato due funzioni che ci permettessero di etichettare le foto associate ai profili e quelle dei volti sconosciuti già presenti, questo per evitare un numero eccessivo di screenshot raffiguranti la medesima persona.

Una volta avviato lo stream video, il sistema inizierà a cercare tutti i volti presenti, quando ne individua uno, confronta i descriptor di quel volto con quello dei membri della famiglia e, se non trova alcun riscontro, dopo aver controllato che questa persona non sia già presente in Recognized, notifica il rilevamento e scatta un'istantanea del video in riproduzione.

Oltre alle notifiche push dal browser, sono state implementate le notifiche di intrusione via mail e sms all'indirizzo e al numero inseriti nel profilo utente, purtroppo però l'invio di email da questo tipo di applicazioni richiede la verifica del dominio e diversi passaggi per i quali sono richieste credenziali aziendali (non a nostra disposizione) e anche i servizi sms sono a pagamento.

Performance e risultati del testing

Per testare il nostro sistema di Face Detection and Recognition abbiamo impostato i nostri profili come familiari e chiesto a nostri parenti di farsi riprendere per verificare il funzionamento e la precisione degli algoritmi utilizzati. Inoltre, per avere un maggiore dataset di volti, abbiamo inquadrato anche foto di persone sconosciute, notando che in questi casi il riconoscimento facciale è stato maggiormente impreciso.

La fase di testing ha rivelato che con un'illuminazione uniforme e in assenza di movimenti rapidi gli algoritmi utilizzati sono molto precisi, in caso contrario però, il volto viene individuato, ma talvolta questi non viene riconosciuto. Quando la sorgente video rileva molti volti sconosciuti, oppure se siamo in presenza di movimenti rapidi davanti alla webcam, le sezioni Recognized e Profile non riescono a caricarsi correttamente, questo è dovuto al fatto che se un volto non viene riconosciuto, dopo aver fatto lo screenshot, bisogna etichettare nuovamente le foto degli sconosciuti e in questo modo il sistema risulta sovraccaricato. Abbiamo riscontrato che l'utilizzo o meno di occhiali da vista, non influisce sul riconoscimento facciale, mentre talvolta si ricorre in errori quando il volto non è ripreso frontalmente dalla sorgente video.

Abbiamo comunque raggiunto una percentuale di successo superiore all'80% per diversi volti.

Tools & bibliography

- Repository del progetto
- face-api.js (per il riconoscimento dei volti)
- Express (per il server, scritto in JavaScript)
- Bootstrap (interfaccia grafica definita in HTML e CSS)
- Pexels (per le immagini in questo documento)
- Google Documents (per il template del documento)
- Slides del corso di Image Processing and Computer Vision
- Slides del corso di Web Application I