SafeCar

Andrea Marcelli - Gabriele Russo

Introduzione

Lo scopo del progetto è quello di sviluppare un servizio che fornisca al guidatore di un autoveicolo un sistema di allarme in caso di colpo di sonno alla guida, contribuendo dunque alla risoluzione dell'annoso problema degli incidenti stradali.

Secondo l'American Sleep Foundation, circa la metà dei guidatori americani afferma di usare il proprio veicolo anche se si sente assonnata, circa il 20% di essi ha affermato di essersi addormentato alla guida nell'ultimo anno.
La statistica è estremamente preoccupante, e ci ha portati alla riflessione sulle tecnologie attuali che potessero essere in grado di arginare in un qualsiasi modo il problema.

Da questa riflessione e dalla ricerca che ne è conseguita è nato SafeCar.

SafeCare necessita di una semplice videocamera, ormai integrata in tutti gli smartphone o acquistabile anche a prezzi molto bassi.
Visto lo scopo didattico del progetto non è stato ancora implementato per i vari dispositivi mobile, che utilizzano iOS o Android.

SafeCar si basa principalmente sull'Eye Blinking e sulla Mouth Detection.
I vantaggi di queste tecniche stanno nella possibilità di prevedere (e non solo individuare) un colpo di sonno e sulla non invadenza della modalità di acquisizione dei dati (come detto, basta una qualsiasi videoamera).

La funzione principale di SafeCar dunque è quella di riconoscere lo stato di sonnolenza e avvisare il guidatore tramite segnale acustico nella speranza di indurre il guidatore ad effettuare una sosta nel luogo più vicino possibile (piazzola di sosta, parcheggio, area di servizio).
Inoltre viene aggiunto anche un servizio aggiuntivo di riconoscimento del guidatore, per segnalare eventuali furti del veicolo.

Installazione

Posizionamento videocamera

Per avere un risultato ottimale si dovrebbe posizionare la videocamera di fronte al guidatore, così da poter avere una visione frontale e abbastanza ravvicinata di occhi e bocca.
La distanza dal volto dovrà essere di circa 50cm.
È possibile ottenere ottimi risultati anche posizionandola leggermente di lato al guidatore, ad esempio dove sono presenti bocchette dell'aria o superfici dove poter fissare il supporto.

Questa immagine indica le zone d'efficacia del posizionamento della videocamera

Utilizzo software

Come prima cosa è necessario addestrare il sistema di riconoscimento facciale, inserendo nella cartella con il proprio nome quante più foto possibile (il minimo consigliato è di venti, la precisione dell'algoritmo aumenta in maniera importante in presenza di almeno trenta fotografie).

Eseguire da terminale (Linux, Powershell etc) il comando:

python extract_embeddings.py --dataset dataset --embeddings output/embeddings.pickle --detector

per estrarre gli embedding dalle foto.

successivamente eseguire il comando:

python train_model.py --embeddings output/embeddings.pickle --recognizer output/recognizer.pickle --le output/le.pickle

per allenare il modello.

Infine lanciare il programma:

python last.py --shape-predictor shape_predictor_68_face_landmarks.dat --detector face_detection_model --embedding-model openface_nn4.small2.v1.t7 --recognizer output/recognizer.pickle --le output/le.pickle --alarm alarm.wav

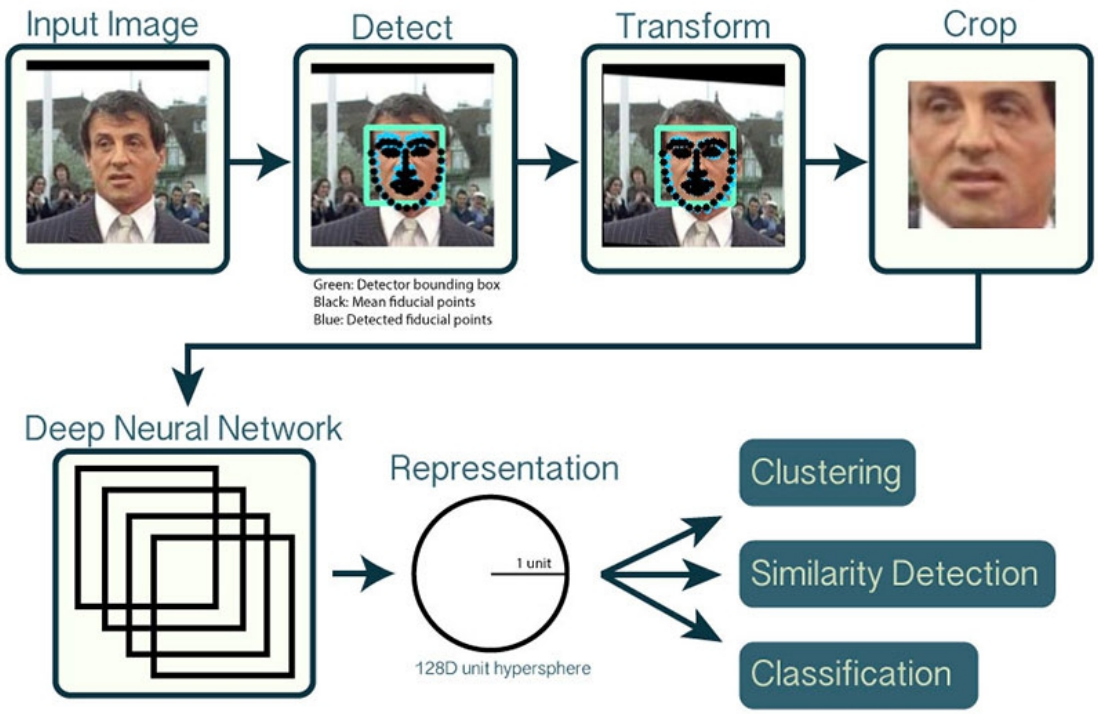
Si è scelto di esplicitare i singoli file da caricare, invece di caricarli di default, per rendere il programma più modellabile e personalizzabile possibile, lasciando libera possibilità di sperimentare.

Come funziona

L'applicazione rileva tramite un flusso video in tempo reale il viso del guidatore per:
- effettuare il riconoscimento facciale in tempo reale
-estrapolare il movimento di occhi e bocca per decidere tramite un algoritmo se il guidatore è in stato di sonnolenza

Per realizzare il nostro sistema di riconoscimento facciale, eseguiremo prima il rilevamento del volto, estrarremo le facciate da ogni volto utilizzando il deep learning, formeremo un modello di riconoscimento facciale sulle immersioni e infine riconosciamo i volti in entrambe le immagini e i flussi video con OpenCV.

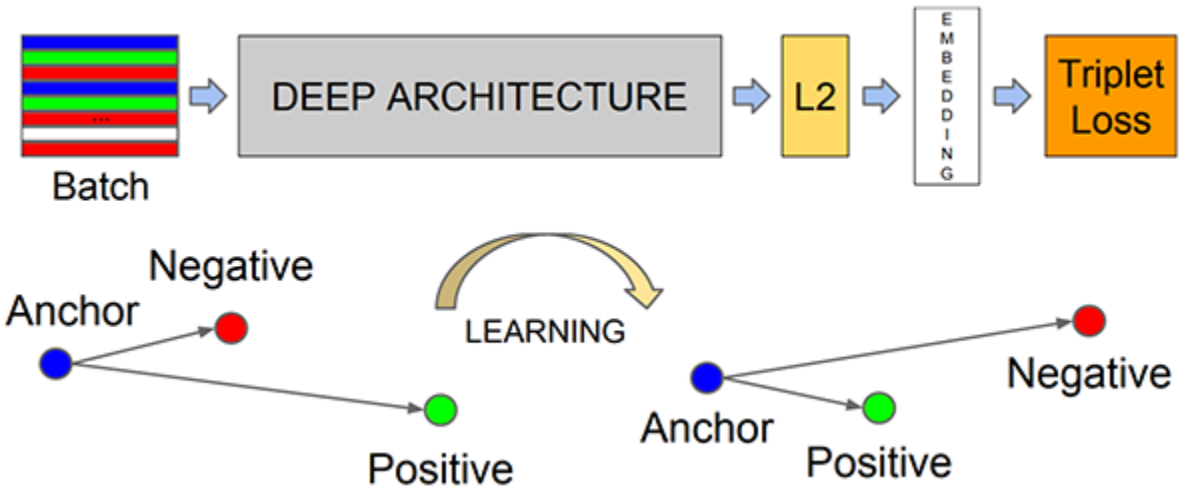
Come funziona il riconoscimento facciale di OpenCV



Per costruire la nostra pipeline di riconoscimento facciale OpenCV, applicheremo l'apprendimento approfondito in due passaggi chiave:
Applicare il rilevamento del volto , che rileva la presenza e la posizione di un volto in un'immagine, ma non lo identifica

Per estrarre i vettori di caratteristiche 128-d (chiamati "embeddings") che quantificano ogni faccia in un'immagine

Innanzitutto, inseriamo un'immagine o un fotogramma video nella nostra pipeline di riconoscimento facciale. Data l'immagine di input, applichiamo il rilevamento del volto per rilevare la posizione di una faccia nell'immagine.
Poi passiamo la faccia di input attraverso la nostra rete neurale profonda:



Il modello di deep learning di FaceNet calcola l'incorporamento a 128-d che quantifica il volto stesso.
Ma in che modo la rete calcola l'incorporamento del volto?
La risposta sta nel processo di formazione stesso, tra cui:
1.I dati di input alla rete
2.La funzione di perdita di triplette
Per addestrare un modello di riconoscimento facciale con apprendimento approfondito, ogni batch di dati di input include tre immagini:
1.L'ancora
2.L'immagine positiva
3.L' immagine negativa
L'ancora è il nostro volto attuale e ha un'identità
La seconda immagine è la nostra immagine positiva - questa immagine contiene anche un volto della persona.
L'immagine negativa, d'altra parte, non ha la stessa identità e potrebbe appartenere alla persona B, C o anche Y!

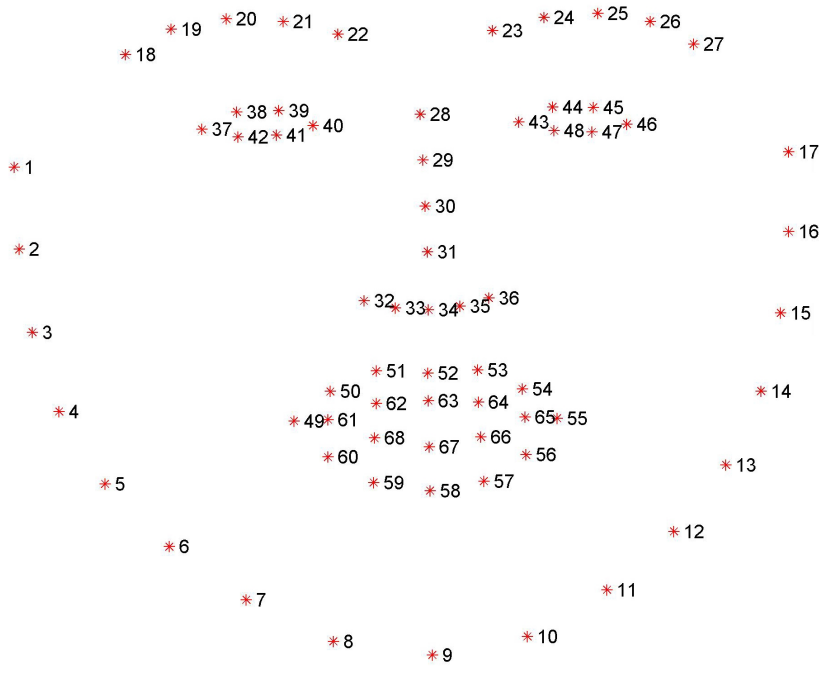
Il punto è che l'ancoraggio e l'immagine positiva appartengono entrambi alla stessa persona / faccia mentre l'immagine negativa non contiene la stessa faccia.
La rete neurale calcola gli incastri di 128 d per ciascuna faccia e quindi modifica i pesi della rete (tramite la funzione di perdita di triplette) in modo tale che:
1.Gli incastri a 128 gradi dell'ancora e l'immagine positiva si trovano più vicini
2.Mentre allo stesso tempo, spingendo via le orme per il padre dell'immagine negativa
In questo modo, la rete è in grado di imparare a quantificare i volti e restituire incastri altamente robusti e discriminanti adatti al riconoscimento facciale.

Il set di dati fornito è composto da 2 persone, Andrea e Gabriele.

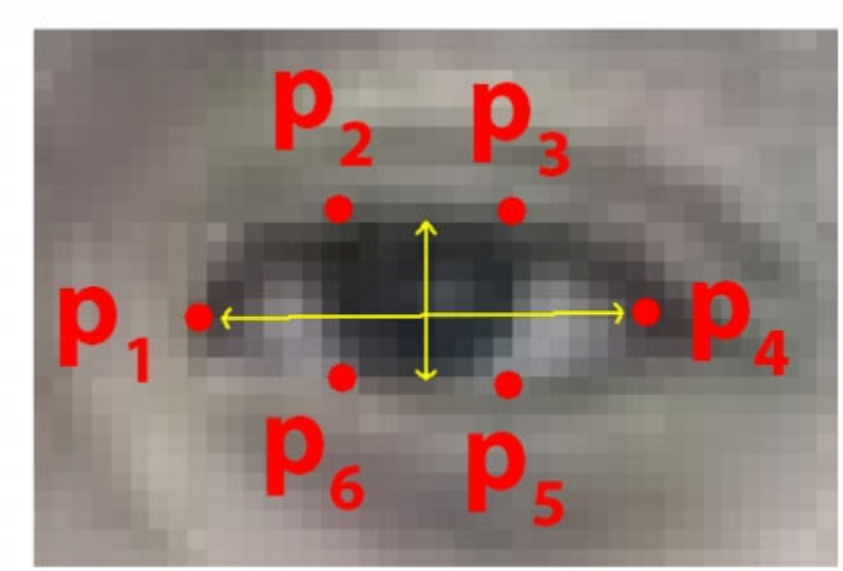
Riconoscimento del battito degli occhi e rilassamento della bocca

Per costruire il nostro rilevatore di battito, calcoleremo una metrica chiamata *rapporto d'aspetto dell'occhio* (EAR), introdotta da Soukupová e Čech nel loro documento del 2016, *Real-Time Eye Blink Detection Using Facial Landmarks*.
La proporzione dell'occhio è invece una soluzione molto più elegante che comporta un calcolo molto semplice basato sul rapporto tra le distanze tra i punti di riferimento del viso degli occhi.

L'idea di base è che su ogni volto sono presenti 68 punti specifici (chiamati punti di interesse o Landmark)—la parte superiore del mento, il bordo esterno di ciascun occhio, il bordo interno di ogni sopracciglio, etc... Non dobbiamo far altro che insegnare all'algoritmo di Machine Learning a cercare e identificare questi 68 punti su qualsiasi volto:
Questo è come il volto viene tradotto in termini di coordinate:



In termini di rilevamento dei battiti di ciglia, siamo interessati solo a due serie di strutture facciali: gli occhi.
Ogni occhio è rappresentato da 6 cordinate [x,y]



Sulla base del lavoro di Soukupová e Čech nel loro documento del 2016, *Real-Time Eye Blink Detection che utilizza i punti di riferimento del viso*, possiamo quindi derivare un'equazione che riflette questa relazione chiamata rapporto tra le dimensioni dell'occhio (EAR) :

$$\text{EAR} = \frac{\|p_2 - p_6\| + \|p_3 - p_5\|}{2\|p_1 - p_4\|}$$

Partendo da questa formula abbiamo svolto vari test.
Questa risultava ottima per rilevare il battito di ciglia, ma non perfetta per identificare l'occhio succhiuso.
Per questo abbiamo scelto di non usare la proporzione dell'occhio e di affidarci al calcolo dell'apertura, ignorando quindi di dividere per l'ampiezza dell'occhio

Sulla stessa linea d'onda per fornire un supporto a questa tecnica abbiamo pensato di combinarla al rilevamento della bocca.
Perchè? Molto spesso prima di addormentarsi si sbadiglia! Inoltre prima di prendere sonno si tende a rilassare la mascella lasciando quindi la bocca socchiusa.

Ovviamente essendo in macchina si può parlare, cantare ed altro, quindi questo pone un ostacolo.
Le analisi effettuate dalla bocca quindi non le reputiamo determinanti al fine di far scattere l'allarme, ma solo per allertare il sistema di essere più sensibile.

Per calcolare l'ampiezza dalla bocca usiamo la stessa tecnica degli occhi.

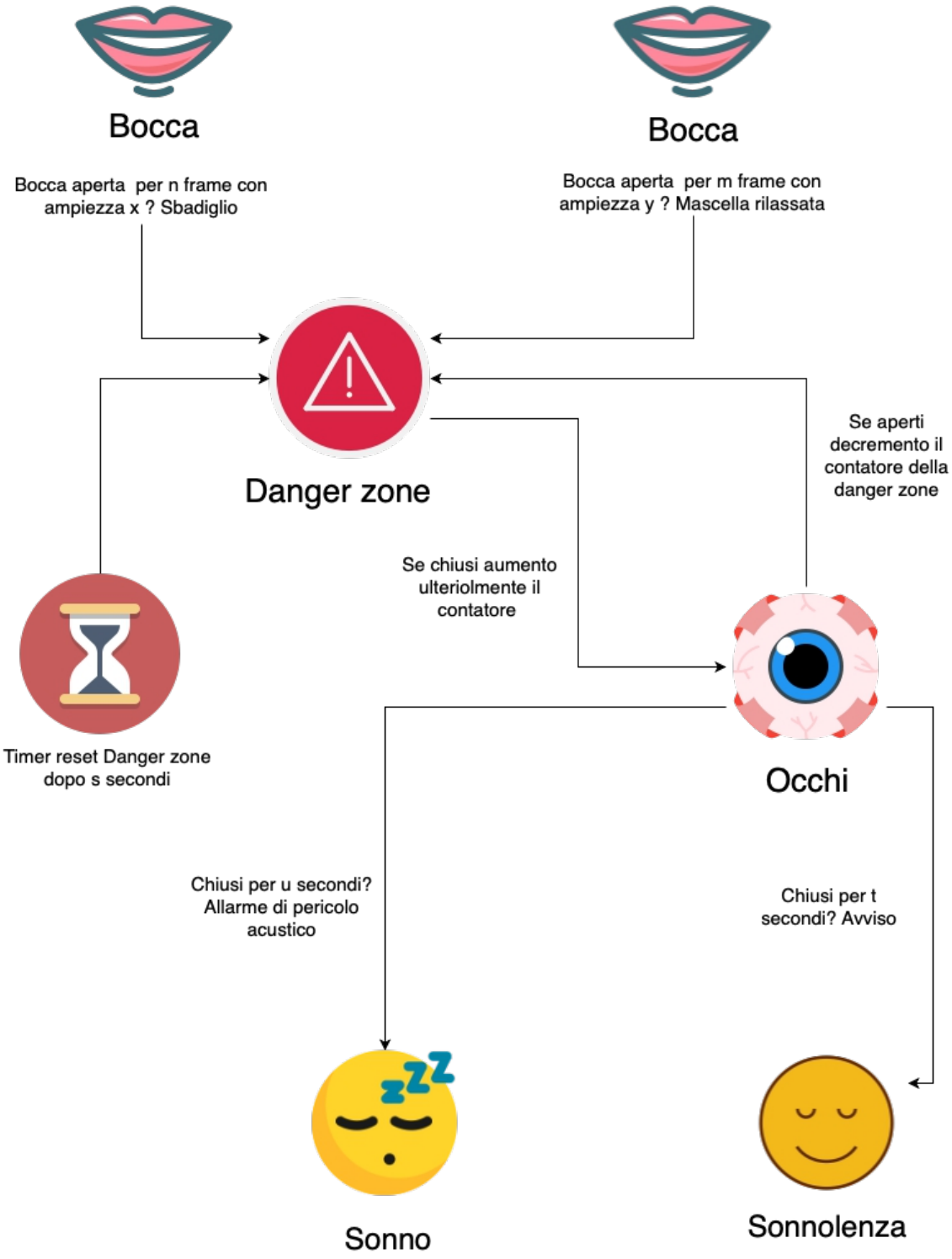
I thresold sono stati valutati in base ad alcuni test, aiutati molto dal fatto che la distanza in cui si può posizionare la fotocamera dal guidatore è molto simile per ogni autoveicolo.

Struttura della cartella

Il nostro progetto ha 3 directory nella cartella principale:
•dataset: contiene le immagini dei volti organizzate in sottocartelle per nome.
•face_detection_model: contiene un modello di deep learning pre-addestrato fornito da OpenCV per rilevare e localizzare i volti.
•output: contiene i file di pickle di output.

•**extract_embeddings.py**: è responsabile dell'utilizzo di un estrattore di funzioni di deep learning per generare un vettore 128-D che descrive una faccia.
•**openface_nn4.small2.v1.t7**: Un modello di deep learning pre addestrato
•**train_model.py**: il nostro modello SVM lineare verrà addestrato da questo script
•**detect.py**: lil core del progetto
•**recognize_video.py**: Iriconoscere dai fotogrammi di un flusso vide

L'algoritmo decisionale



Danger zone: stato scaturito dal rilevamento di uno sbadiglio o dal rilassamento della mascella. In questo stato il contatore di allarme che si avvia appena gli occhi sono socchiusi/chiusi si incrementa più velocemente.

Dati i seguenti threshold:

M1: ampiezza apertura bocca per considerare un rilassamento della mascella
FM1: numero di frame consecutivi in cui l'ampiezza della bocca deve superare M1 per essere considerata una mascella rilassata

M2: ampiezza apertura bocca per considerare uno sbadiglio
FM2: numero di frame consecutivi in cui l'ampiezza della bocca deve superare M2 per essere considerato uno sbadiglio

E1: ampiezza apertura minima media degli occhi per considerare gli occhi chiusi/socchiusi
FE1: numero di frame consecutivi in cui l'ampiezza media degli occhi deve essere inferiore a E1 per essere considerato un pericolo lieve
FED1: numero di frame consecutivi in cui l'ampiezza media degli occhi deve essere inferiore a E1 per essere considerato un pericolo grave

ed **M** l'ampiezza di apertura della bocca ed **E** quella degli occhi

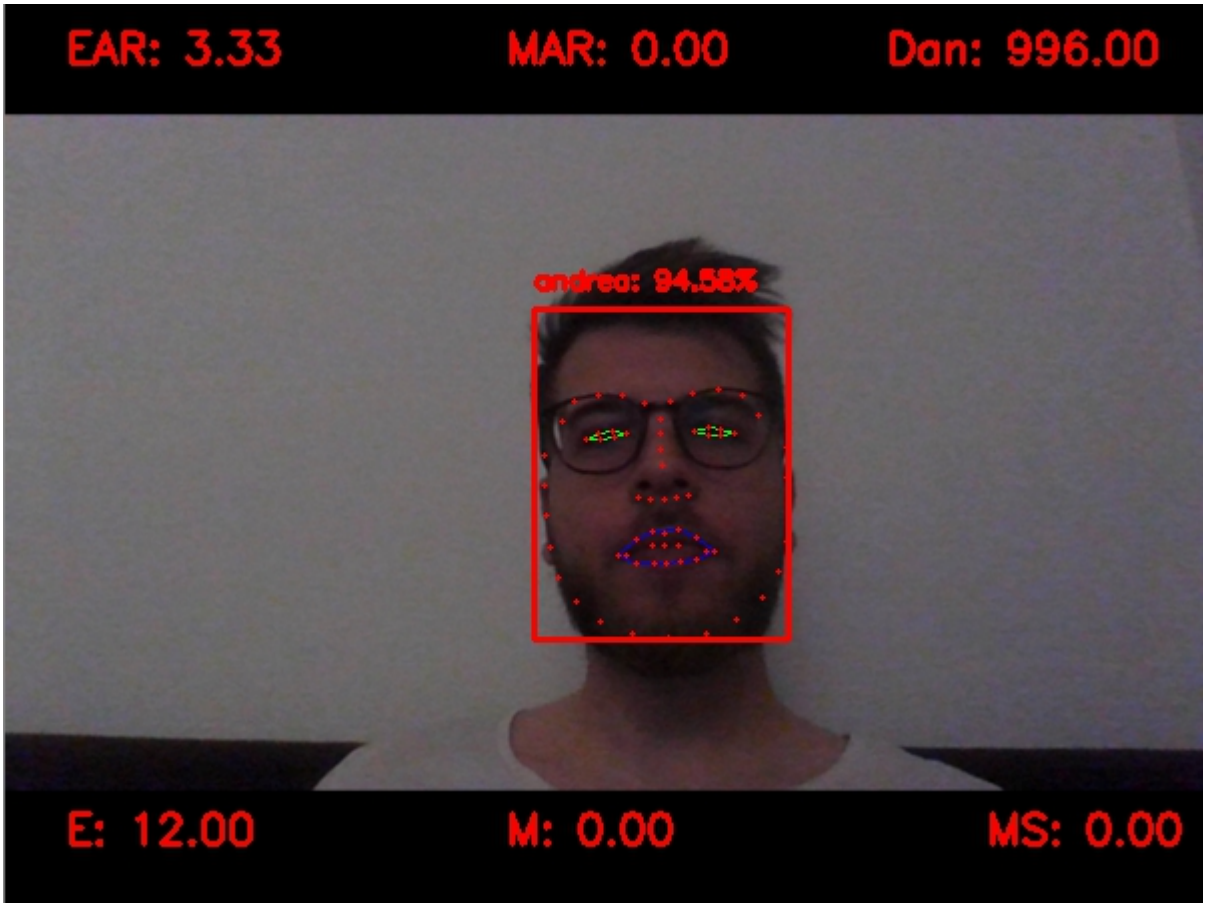
Se M è superiore a M1 per FM1 frame si è verificato un rilassamento della mascella, quindi potrebbe esserci uno stato di sonnolenza, quindi allerto il sistema di essere più rigido sull'apertura degli occhi (Danger zone).
Se M è superiore a M2 per FM2 frame si è verificato uno sbadiglio, effetto quindi le stesse azioni.
Se E è inferiore di E1 per FE1 frame significa che gli occhi sono chiusi per un periodo ti tempo che indica uno stato di sonnolenza, quindi avviso il guidatore.
Se invece E è inferiore di E1 per FE2 frame significa che il guidatore si è addormentato, faccio quindi scattare l'allarme.
In Danger zone vengono dimezzati i valori di FE1 e FE2.

Implementazione dell'algoritmo

```
# check to see if the mouth aspect ratio is over...
if mouthMAR > MOUTH_AR_THRESH:
    MCOUNTER += 1

    # if the mouth were closed for a sufficient number of frame...SBADIGLIO
    if MCOUNTER >= MOUTH_AR_CONSEC_FRAMES:
        cv2.putText(frame, "SBADIGLIO", (50, 150),
            cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 150, 0), 2)
        DANGER_ZONE=DANGER_ZONE_MAX

else:
    MCOUNTER = 0

# check to see if the mouth aspect ratio is over...MASCELLA RILASSATA
if mouthMAR > SLEEP_MOUTH_AR_THRESH:
    SLEEP_MOUTH_COUNTER += 1

    # if the mouth were closed for a sufficient number of...
    if SLEEP_MOUTH_COUNTER >= SLEEP_MOUTH_AR_CONSEC_FRAMES:
        cv2.putText(frame, "RELAX", (350, 150),
            cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 250, 0), 2)
        DANGER_ZONE=DANGER_ZONE_MAX

else:
    SLEEP_MOUTH_COUNTER = 0


if ear < EYE_AR_THRESH:
    COUNTER += 1
    if DANGER_ZONE:
        COUNTER+=1

    # if the eyes were closed for a sufficient number of frame
```

```
if ear < EYE_AR_THRESH:
    COUNTER += 1
    if DANGER_ZONE:
        COUNTER+=1

    # if the eyes were closed for a sufficient number of frame
    # then write alert
    if COUNTER >= EYE_AR_CONSEC_FRAMES:
        cv2.putText(frame, "APRI GLI OCCHI", (50, 300),
            cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 50, 200), 2)

    # if the eyes were closed for a sufficient number of frame
    # then enter in danger zone and play alarm sound
    if COUNTER >= EYE_AR_CONSEC_FRAMES_DANGER:
        # if the alarm is not on, turn it on
        if not ALARM_ON:
            ALARM_ON = True

            # check to see if an alarm file was supplied,
            # and if so, start a thread to have the alarm
            # sound played in the background
            if args["alarm"] != "":
                t = Thread(target=sound_alarm,
                    args=(args["alarm"],))
                t.deamon = True
                t.start()

        # draw an alarm on the frame
        cv2.putText(frame, "SVEGLIATI!", (350, 300),
            cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)

# otherwise, the eye aspect ratio is not below the blink
# threshold, so reset the counter and alarm
else:
    DANGER_ZONE-=2
    COUNTER = 0
    ALARM_ON = False
```

Interfaccia



L'interfaccia è composta dall flusso video, e dalla visualizzazione di alcuni valori, che sono:

EAR: l'attuale eye aspect ratio
MAR: l'attuale mouth aspect ratio
Dan: indica quanti frame mancano all'uscita dalla danger-zone
E: indica da quanti frame l'EAR supera il valore di threshold
M: indica da quanti frame la MAR supera il valore di threshold per lo sbadiglio

MS: indica da quanti frame la MAR supera il valore di threshold per il rilassamento della mascella.

Inoltre a schermo vengono visualizzati i seguenti messaggi quando si verificano determinati eventi:

“Sbadiglio”: quando viene rilevato uno sbadiglio
“Relax”: quando viene rilevato il rilassamento della mascella
“Apri gli occhi”: quando gli occhi vengono rilevati chiusi per tot tempo
“Svegliati!”: quando il precedente messaggio non ha avuto effetto ed è passato altro tempo, e conseguentemente attiva un effetto sonoro per svegliare il guidatore
“Antifurto”: quando il riconoscimento facciale riscontra che il soggetto inquadrato non è nel sistema.

LIBRERIE

OPENCV

Abbiamo utilizzando il modulo di Deep Neural Network chiamato Caffe per identificare e localizzare i volti.
Perchè Caffe? É un algoritmo pre-addestrato che faceva al caso nostro.

DLIB

La libreria DLIB usa un modello di predizione facciale pre addestrato, basato su una modifica del modello Histogram of Oriented Gradients + Linear SVM.
La useremo in fase di face detector e facial landmark detector per rilevare occhi e bocca.

Perchè non usarla anche per il riconoscimento facciale? Avremmo potuto, ma ci sarebbe servito l'aiuto di altre librerie esterne, ed usare OpenCV solo per la detection del volto.
Ad esempio utilizzando questa libreria.

Test

Test per stabilire un valore di default per l'EAR e MAR ottimali,così da poter stabilire con precisione quando un gli occhi sono da considerare chiusi e la bocca aperta.
Soltanto in questa fase abbiamo chiesto ai tester di tenere gli occhi aperti normalmente e di simulare uno sbadiglio, tuitti gli altri test sono stati svolti senza influenzare l'utente.

Distanza di 40 cm e angolazione di 0°:

Utente	EAR	MAR
1	6,5	7,7
2	6,4	7,4
3	6,5	7,4
4	6,1	6,8
5	6,4	6,5
6	6,2	6,9
7	6,1	6,7
8	6,1	7,1
Andrea	6,5	7,2
Gabriele	6,6	7,4

Test per vedere come cambia l'EAR e la MAR a varie angolazioni e distanza, così da poter raccomandare come ottenere risultati ottimali.
Ovviamente le varie angolazioni sono dovute alle diverse disposizioni delgli interni delle autovetture.

Utente	Distanza	Angolazione	EAR	MAR
1	40	0°	6,5	7,5
2	43	0°	6,4	7,2
3	38	0°	6,5	7,1
4	50	30°	5,5	6,1
5	50	30°	5,4	5,8
6	45	20°	6,1	6,5
7	60	45°	4,5	4,3
8	48	10°	5,9	6,8
Andrea	50	0°	6,3	7,2
Gabriele	45	10°	6,4	7,4

Risultati rilevamento sonnolenza in condizioni consigliate (distanza di circa 50 cm e angolazione massima di 30°)
Il valore "/" nella colonna del rilassamento della mascella indica che l'utente non l'ha eseguito.

Utente	Numero sbadigli	Numero rilassamenti mascella	Sonnolenza
1	80,00%	/	100,00%
2	75,00%	100,00%	100,00%
3	100,00%	/	100,00%
4	100,00%	/	100,00%
5	80,00%	100,00%	100,00%
6	100,00%	/	100,00%
7	100,00%	/	100,00%
8	100,00%	/	100,00%
Andrea	100,00%	100,00%	100,00%
Gabriele	100,00%	100,00%	100,00%

Affidabilità riconoscimento facciale in base al numero di foto per addestrare il modello:

Utente	Numero foto	Successo	Affidabilità media
1	6	0	<50%
2	9	0	<50%
3	15	1	78,00%
4	27	1	85,00%
5	25	1	84,00%
6	31	1	90,00%
7	37	1	93,00%
8	34	1	90,00%
Andrea	50	1	97,00%
Gabriele	40	1	95,00%

Progressi futuri e possibili miglioramenti

Abbiamo individuato una serie di possibili migliorie operabili per favorire l'uso di SafeCar.

Interfaccia grafica

Il sistema attuale è avviabile mediante un qualsiasi terminale, ma un'eventuale implementazione in ambiente iOS o Android implicherebbe la realizzazione di un'interfaccia grafica dotata di tutorial iniziale dove inserire i dati necessari all'utilizzo. Al primo avvio il software potrebbe chiedere all'utente di scattare un dato numero di foto o di posizionare correttamente la fotocamera.

Personalizzazione di EAR e MAR

I valori EAR e MAR possono non essere perfettamente calzanti a seconda della forma dell'occhio dell'utente che sta guidando. Sarebbe possibile affinare ed equilibrare tali valori a seconda della persona che viene identificata.

Face Alignment

L'allineamento facciale consiste nell'identificazione di una struttura geometrica della faccia e nell'ottenimento di un allineamento della faccia basato sulla traslazione, rotazione e rappresentazione in scala di essa.
Grazie ad esso sarebbe possibile migliorare drasticamente il rate del riconoscimento, al caro prezzo di un eccessivo dispendio di risorse hardware che non tutti i telefoni/tablet sono in grado di sopportare senza avere gravi cali di framerate.

Regolare gli iperparametri

La rigorosità dell'algoritmo nelle classificazioni errate è stabilita da un valore C all'interno di una SVM lineare.
Valori molto alti di C comporteranno una maggiore rigorosità, espondendo l'algoritmo ai rischi dell'overfitting.
Viceversa, valori molto bassi causeranno una maggiore permissività.

Antifurto online

Seppure si tratti di una funzione esterna all'obiettivo del progetto, sarebbe semplice implementare un antifurto online che riconosca il volto del guidatore, magari collegandolo a un sistema di sorveglianza online o alla compagnia assicurativa presso cui il veicolo è registrato. In caso di procedimenti civili/penali l'antifurto potrebbe semplificare di molto i processi di risarcimento.

Distrazione

Pur essendo una limitazione dell'algoritmo (che incontra difficoltà nel momento in cui il volto si gira), si può sfruttare questa caratteristica per segnalare eventuali movimenti inconsulti o pericolosi del guidatore (guardare fuori dal finestrino, guardare in basso) che lo distraggano dalla guida.