

Relazione di progetto - Visione Artificiale e Riconoscimento

Alessandro Marcantoni - 0000980506

Simone Romagnoli - 0000995763

Marta Spadoni - 0000978305

Aprile 2022

Indice

1	Introduzione	3
2	Stato dell'arte	4
2.1	Rilevamento delle mascherine	4
2.2	Controllo del distanziamento sociale	5
3	Requisiti	7
3.1	Rilevamento delle mascherine	7
3.1.1	Descrizione del problema	7
3.1.2	Analisi dei requisiti	7
3.2	Controllo del distanziamento sociale	8
3.2.1	Descrizione del problema	8
3.2.2	Analisi dei requisiti	8
4	Soluzione	9
4.1	Rilevamento delle mascherine	9
4.1.1	Svolgimento	9
4.2	Controllo del distanziamento sociale	11
4.2.1	Svolgimento	11
5	Valutazione delle prestazioni	14
5.1	Rilevamento delle mascherine	14
5.2	Controllo del distanziamento sociale	17
6	Sviluppi futuri e conclusioni	23

1 Introduzione

La situazione sanitaria in cui il nostro paese volge dal 2020 ad oggi è il contesto in cui si colloca questo progetto. Le lezioni, le riunioni, i seminari e molte altre attività che prima si svolgevano in presenza, in aule universitarie o in stanze di uffici, ad oggi purtroppo vengono tenute ancora per la maggior parte online attraverso video-chiamate. Il progetto ha l’obiettivo di realizzare, in collaborazione con il corso di **Smart City e Tecnologie Mobili**, un sistema per una *Smart Anti-Covid Room*, in modo da poter tornare in sicurezza a svolgere tali attività in presenza.

Nello specifico, una componente essenziale del progetto sarebbe quella di un sistema di videosorveglianza che monitori la stanza e verifichi il rispetto delle due condizioni che si sono dimostrate essere le più efficaci per combattere la trasmissione del COVID-19: l’uso delle mascherina e il rispetto del distanziamento sociale. Entrambe le verifiche avvengono attraverso l’uso di tecniche di *deep learning* adatte ad essere utilizzate in *real-time*.

Il sistema che si occupa di verificare che le persone all’interno della stanza indossino correttamente le mascherine si sviluppa su due fasi: la prima si occupa di realizzare la *Face Detection*, cioè l’individuazione delle facce presenti nell’immagine mentre la seconda, attraverso un classificatore, determina se ciascun viso è coperto in modo corretto o meno da una mascherina.

Anche il controllo del distanziamento sociale avviene su più fasi: nella prima si individuano tutte le persone presenti nell’immagine e le loro posizioni, nella seconda si trasformano tali posizioni nelle corrispondenti della visione *bird-eye* e nella terza e ultima si determina a quale distanza si trovi ciascuna persona dalle altre così da individuare i contatti a rischio.

2 Stato dell'arte

In questa sezione viene analizzato nel dettaglio lo stato dell'arte riguardante i modelli in questione. Le problematiche in analisi sono state oggetto di numerose ricerche a partire dalla diffusione su larga scala del COVID-19, quindi le soluzioni presenti in letteratura a riguardo sono relativamente recenti; tuttavia, offrono diversi spunti per idee e ampio spazio per studiare tali problemi.

2.1 Rilevamento delle mascherine

Per quanto riguarda il rilevamento delle mascherine, la maggior parte dei modelli allo stato dell'arte realizzano la soluzione in due passaggi: nello specifico, una prima parte di rilevamento di volti all'interno di un'immagine, seguita dalla classificazione delle relative regioni di interesse per determinare se un viso stia indossando una mascherina o meno.

Esistono diversi modelli per effettuare *detecting* di visi umani sia in spazi chiusi, sia *in-the-wild*. I modelli esistenti gestiscono efficientemente diversi tipi di occlusione e sfruttano sia *single-stage*, sia *multi-stage detector*. Inoltre, garantiscono la possibilità di applicazione del sistema in *real-time*, grazie a tempi medi di inferenza nell'ordine dei decimi di secondo. In particolare, tra i modelli più popolari per la risoluzione del problema si citano *ResNet50*, *AlexNet* e *MobileNet* che fungono da *backbone* per reti di *face detection*. Inoltre, è possibile ottenere un'ottima efficienza anche grazie all'utilizzo del *transfer learning*; tale tecnica consente di sfruttare le conoscenze già apprese da una rete pre-addestrata e trasferirle all'interno di una rete *ad hoc* appositamente strutturata in poco tempo.

Per quanto riguarda i dataset, i più famosi sono *WiderFace* [1] relativamente alla *Face Detection*, mentre, per quanto riguarda la *Mask Detection* esistono diversi riferimenti, dei quali si cita *MAFA* (MAsked FAces) [5].

Effettuando un confronto [11] dei modelli precedentemente citati, una delle migliori soluzioni prevede l'utilizzo di una *ResNet50* che permette di raggiungere un'accuratezza del 98.2% . Infatti, tra le reti riportate, la *ResNet50* è quella con una struttura più complessa (figura 1), e riesce ad effettuare *object detection* di oltre 1000 classi; tuttavia, utilizzare una rete più leggera come la *MobileNet* può favorire le prestazioni del modello in questione.

Ad ogni modo, quelle citate non sono le uniche soluzioni presenti allo stato dell'arte. Infatti, altre soluzioni prevedono l'utilizzo di una rete *YOLO-V4* [10] o anche *Faster RCNN*.

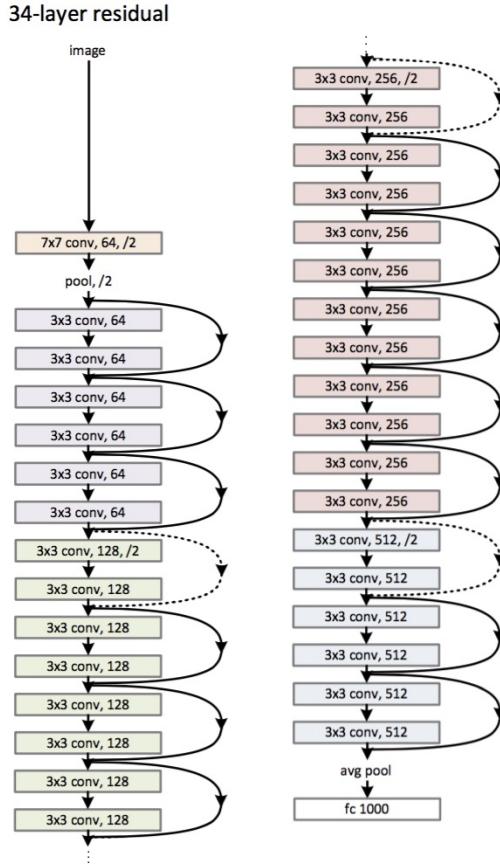


Figura 1: Struttura di una rete neurale *ResNet50*.

2.2 Controllo del distanziamento sociale

Per quanto riguarda il controllo del distanziamento sociale, viene principalmente adottata una tecnica che si suddivide in diversi passaggi. Innanzitutto, vengono rilevate le persone all'interno dell'immagine ottenendo le coordinate reali di ciascuna; in seguito, viene creato un modello virtuale che rappresenta la visione a *bird-eye* dell'inquadratura, nello specifico si definisce in primo luogo un'area rettangolare con coordinate reali sull'immagine in analisi e successivamente tale area viene trasposta nella corrispondente del modello a *bird-eye*. All'interno del modello così creato, viene infine calcolata la distanza tra ogni coppia di persone al fine di determinare quali di queste stanno effettivamente rispettando il vincolo del distanziamento sociale.

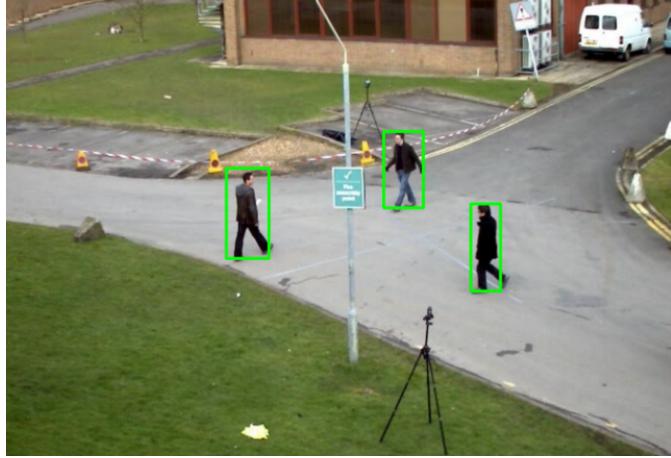


Figura 2: Esempio di segmentazione di istanze di persone utilizzando delle *bounding-box*.

Esistono diversi modelli per effettuare *human detection* sia in spazi chiusi, che in spazi aperti. Tali modelli gestiscono efficientemente diversi tipi di occlusione e garantiscono la possibilità di applicazione in *real-time* grazie a tempi medi di inferenza nell'ordine dei decimi di secondo. In particolare, tra i modelli più popolari per la risoluzione si citano *YoloV3* e *SSD-MobileNet-V2*.

Per quanto riguarda i dataset, i più famosi sono *WiderPerson* [3] e *P-DESTRE* [7]; si cita inoltre il dataset *PnPLO* [6] specializzato sull'individuazione di persone e figure *person-like*.

Per la creazione del modello virtuale a *bird-eye*, vengono solitamente adottate delle tecniche di trasformazione geometrica che, grazie ad una trasformazione prospettica, permettono di ottenere una matrice di traslazione: questa, se applicata a dei punti nell'immagine, permette di ottenere le coordinate relative alla vista a *bird-eye* desiderata.

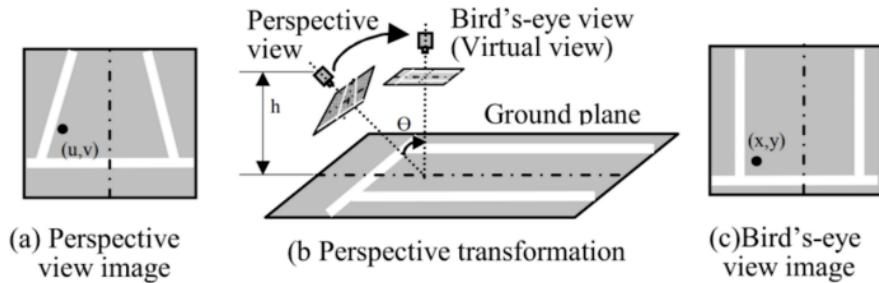


Fig. 1. Illustration of perspective transformation in a parking lot scene

Figura 3: Ottenimento del modello a *bird-eye* di coordinate reali.

Grazie alle informazioni ottenute si riesce poi ad applicare elementi rappresentativi (come delle *bounding-box* o delle linee colorate che congiungono le persone) sull'immagine originale per dare un risultato interpretabile intuitivamente dall'occhio umano. Si riporta un semplice esempio implementativo [9] del modello descritto.

3 Requisiti

In questa sezione si analizzano i problemi in questione per dedurre, di conseguenza, i requisiti progettuali che determineranno le funzionalità del sistema finale. Entrambe le problematiche descritte fanno riferimento all'analisi di uno streaming video in *real-time*: di seguito, vengono descritti i problemi dal punto di vista di una singola immagine, tuttavia bisogna tener conto delle prestazioni finali affinché queste permettano un *framerate* sufficientemente adatto ad un flusso di immagini in tempo reale.

3.1 Rilevamento delle mascherine

3.1.1 Descrizione del problema

Data un'immagine contenente diverse persone all'interno di un'aula, si vuole determinare quali di queste indossano la mascherina e quali no. Le complicazioni che questo problema presenta sono molteplici: innanzitutto, l'immagine deve essere di alta qualità per permettere di individuare persone anche in profondità; inoltre, l'inquadratura della videocamera deve permettere una visione frontale della stanza, in modo da riprendere maggiormente persone di fronte piuttosto che girate. Infine, non deve presentare alterazioni luminose in diverse parti e le facce delle persone devono presentare meno occlusioni possibili.

3.1.2 Analisi dei requisiti

Sulla base della problematica appena descritta, si elencano di seguito i requisiti essenziali che il sistema deve soddisfare:

1. Data l'immagine, è necessario risolvere il problema in due fasi:
 - **Face Detection** - rilevamento di tutti i volti, probabilmente con un'analisi multi-scala, che restituisce la *bounding box* e la confidenza per ogni volto trovato;
 - **Mask Classification** - date le *RegionOfInterest* contenenti tutti i volti individuati, si effettua una classificazione che determina se ciascuno sta indossando la mascherina, la sta indossando ma in modo scorretto, oppure non la sta indossando affatto.
2. L'inquadratura della videocamera deve permettere una visione frontale della stanza per riprendere più persone possibile frontalmente e non deve presentare distorsioni "a barilotto" o "a cuscinetto".
3. È necessario individuare un approccio (ad esempio basato su reti neurali) che permetta la soluzione del problema in tempi ridotti, ma con risultati accurati.
4. Vista la soluzione in due fasi del problema, è necessario scegliere due dataset adatti per addestrare e testare i modelli sviluppati: il primo deve contenere un elevato numero di facce di persone inquadrata da diverse angolazioni e che contengano anche diversi tipi di occlusione, in modo da riuscire a individuare visi umani dietro mascherine o non completamente visibili.

3.2 Controllo del distanziamento sociale

3.2.1 Descrizione del problema

Data un’immagine contenente diverse persone all’interno di un’aula, si vuole determinare quali di queste stanno rispettando il distanziamento sociale e quali no. Le complicazioni che questo problema presenta sono molteplici: innanzitutto, l’immagine deve essere di alta qualità per permettere di individuare persone anche in profondità; inoltre, l’inquadratura della videocamera deve provenire da un’angolazione adeguata in modo da ottenere la trasposizione della vista a *bird-eye* tramite un’opportuna trasformazione geometrica. In più, le persone devono sovrapporsi il meno possibile per evitare di occludersi a vicenda.

3.2.2 Analisi dei requisiti

Sulla base della problematica appena descritta, si elencano di seguito i requisiti essenziali che il sistema deve soddisfare:

1. Data l’immagine, è necessario risolvere il problema in due fasi:
 - **Human Detection** - rilevamento di tutte le persone all’interno dell’inquadratura, possibilmente con un’analisi multi-scala, che restituisce la *bounding box* e la confidenza per ogni persona trovata;
 - **Transform & Distancing Check** - trasposizione delle posizioni delle persone trovate nel modello a *bird-eye* e successivo controllo delle distanze per ogni coppia di persone.
2. Bisogna individuare un’area rettangolare all’interno dell’inquadratura prospettica in modo da poter calcolare la matrice di trasformazione che consenta la mappatura dei punti dall’immagine iniziale a quella a *bird-eye*.
3. L’inquadratura della videocamera deve permettere una visione frontale della stanza per riprendere più persone possibile frontalmente e non deve presentare distorsioni ”a barilotto” o ”a cuscinetto”.
4. È necessario individuare un approccio (ad esempio basato su reti neurali) che permetta la soluzione del problema in tempi ridotti, ma con risultati accurati.
5. È necessario scegliere un dataset adatto per valutare le prestazioni dei modelli analizzati; in particolare, deve contenere un adeguato numero di persone inquadrate anche parzialmente e da diverse angolazioni, in modo da rilevare anche persone sovrapposte in prospettiva.

4 Soluzione

In questa sezione viene descritta la soluzione implementata per ognuno dei problemi descritti, ponendo particolare attenzione sul modello utilizzato e sui risultati ottenuti dal trattamento dei dataset scelti. Inoltre, verrà posta particolare attenzione sulle tecniche utilizzate per dare un contributo speciale allo stato dell'arte: nello specifico, si sottolineeranno i vantaggi che il progetto ha portato per il funzionamento del sistema all'interno di una stanza a partire da inquadrature ottimali.

4.1 Rilevamento delle mascherine

Il contributo principale del progetto per quanto riguarda il rilevamento delle mascherine verte essenzialmente sullo sviluppo di un modello che permetta di individuare mascherine anche in zone occluse, ma soprattutto in secondo piano, in *region of interest* con aree molto piccole. Esistono diversi modelli basati su analisi multi-scala che implementano già tale funzione, tuttavia non supportano l'analisi in *real-time*, oppure la supportano, ma riescono a rilevare solamente mascherine in primo piano. Nelle sezioni successive, viene spiegato come è stato possibile raggiungere tali obiettivi e come sono stati trattati i dataset per poter addestrare i modelli desiderati.

4.1.1 Svolgimento

Come anticipato nella sezione 3.1.2, il rilevamento delle mascherine avviene in due fasi: una prima di *face detection* e una seconda di *mask classification*.

Per quanto riguarda la fase di *face detection*, sono stati confrontati diversi modelli presenti allo stato dell'arte e si è scelto di usarne uno pre-addestrato. In particolare, è stata scelta la rete *RetinaFace*, che presenta una *SSD MobilenetV2* come *backbone* ed è addestrata su *WiderFace* [1]. La difficoltà principale è stata trovare un modello interpretabile dall'ultima versione della libreria Python *Tensorflow* e il cui output si basasse sull'individuazione di *bounding box* piuttosto che su segmentazione dei contorni. La rete trovata ([8]) ha permesso di ripristinare il suo stato tramite i *checkpoints* messi a disposizione, tuttavia è stata addestrata utilizzando la versione 2 di *Tensorflow*: quindi, è stata caricata con tale versione e successivamente salvata in formato *saved model* in modo da poter essere utilizzata anche con versioni successive della libreria.

La fase di *mask classification*, invece, è stata più complessa e articolata da affrontare. Innanzitutto, si è pensato di confrontare diversi classificatori per scegliere poi il più efficace ed efficiente; in seguito, si è discusso sulla natura del problema di classificazione in questione: nello specifico, se si trattasse di un problema di classificazione binaria o multi-classe. In seguito ad un'attenta analisi dei dataset a disposizione allo stato dell'arte, è stato concluso che esistono pochi modelli che trattano quello in questione come un problema multi-classe, considerando i casi di mascherina **indossata**, **non indossata** e **indossata non correttamente**. Quindi, per studiare al meglio la problematica e dare maggior contributo, è stato scelto di trattarlo come un problema multi-classe. Di conseguenza, sono stati analizzati e confrontanti due diversi modi per effettuare la classificazione:

- Rete neurale - per tale approccio è stata utilizzata una *MobileNetV2* pre-addestrata sul dataset *image-net* ed è stato effettuato un processo di *fine tuning*. Sono stati effettuati diversi addestramenti variando diversi elementi, tra cui alcuni iperparametri (*learning rate*, ottimizzatore, etc) e una serie di operazioni sul training set (*data augmentation* e doppia operazione di *resize*); infine, è stato scelto il modello con risultati migliori, nello specifico confrontando i valori di *accuracy* e *loss function* su un validation set.
- Support Vector Machine - per tale approccio è stato addestrato un classificatore SVM grazie alle librerie di *Scikit-Learn*; in particolare, i dati su cui viene addestrato tale classificatore sono gli *Histogram of Oriented Gradients (HOG)* derivati dalle immagini in bianco e nero; per la scelta dei migliori iperparametri, è stata effettuata una *GridSearch* che ha concesso di individuarli confrontando i diversi livelli di *accuracy* e *loss function*.

Per entrambi gli approcci è stato infine utilizzato un dataset ([4]) che, come anticipato, contenesse anche mascherine indossate scorrettamente. Inoltre, è stato deciso di effettuare un procedimento di *data transformation* per adattare le immagini al problema in questione: nello specifico, ogni immagine del dataset ha subito una doppia operazione di *resize*, prima a dimensione 30×30 poi a dimensione 224×224 ; questo perché i volti trovati all'interno dei frame delle inquadrature nelle aule hanno spesso dimensioni ridotte e vanno ridimensionati all'input della rete, che è appunto $224 \times 224 \times 3$.

Il listato 1 mostra la struttura della rete neurale addestrata.

```

1 input_shape = (224, 224, 3)
2
3 data_augmentation = keras.Sequential(
4     [
5         layers.RandomFlip("horizontal"),
6         layers.RandomRotation(0.1),
7         layers.RandomZoom(0.2)
8     ]
9 )
10 inputs = keras.Input(shape=input_shape)
11 augmentation = data_augmentation(inputs)
12 mobilenet = MobileNetV2(weights="imagenet", input_shape=input_shape)(augmentation)
13 maxpool = tf.keras.layers.GlobalMaxPooling2D()(mobilenet)
14 output = tf.keras.layers.Dense(3, activation='softmax')(maxpool)
15 model = tf.keras.Model(inputs=[inputs], outputs=[output])
16
17 model.compile(optimizer=Adam(lr=0.0001), loss='categorical_crossentropy')

```

Listing 1: Struttura della rete addestrata per *mask classification*.

Si fa notare che nel modello sono stati inclusi dei livelli di *image augmentation* attivi solamente durante l'addestramento, in particolare *RandomFlip*, *RandomRotation* e *RandomZoom*. In questo modo, la rete viene addestrata anche su casi non standard, riuscendo poi a generalizzare meglio in fase di *inference*.

4.2 Controllo del distanziamento sociale

Per quanto riguarda la parte di verifica del distanziamento sociale, l'obiettivo principale del progetto è quello di individuare il modello più adatto per eseguire il controllo in *real-time* ma mantenendo comunque un livello di accuratezza elevato. Inoltre, le soluzioni presenti allo stato dell'arte si riferiscono, per la maggior parte, a contesti all'aperto (strade e piazze), mentre il modello scelto deve essere in grado di valutare il distanziamento all'interno di una stanza dove le persone possono anche essere parzialmente occluse, ad esempio da banchi. Nella sezione successiva, viene descritto come è stato implementato il controllo del distanziamento sociale considerando gli obiettivi descritti.

4.2.1 Svolgimento

Il sistema di verifica del distanziamento sociale, come descritto nella sezione 3.2.2, si sviluppa su due fasi: la prima di *Human Detection* e la seconda di verifica del distanziamento attraverso opportune trasformazioni geometriche.

Lo sviluppo del sistema di verifica del distanziamento sociale ha riguardato in primo luogo l'implementazione di tutte le funzioni necessarie per creare il modello virtuale a *bird-eye*. A questo scopo, è stato implementato un primo script Python che consente all'utente di definire su un'immagine i quattro punti che individuano il rettangolo sulla scena reale in cui deve avvenire il controllo. Dati tali punti, lo script calcola la matrice che determina la trasformazione prospettica dei 4 punti nel modello a *bird-eye*, questo avviene mediante l'utilizzo della funzione *getPerspectiveTransform* della libreria OpenCV. La matrice individuata consentirà di convertire le coordinate dei centroidi delle persone sulla scena, nelle corrispondenti sul modello virtuale.

A seguito della prima fase di *Human Detection*, per ciascuna persona presente nella scena viene determinato un *bounding box* che ne individua la posizione. In seguito, da tale rettangolo viene determinato il centroide della persona, punto utilizzato per la verifica del distanziamento. Nelle principali soluzioni allo stato dell'arte, tale punto viene solitamente scelto come il baricentro del rettangolo, nel nostro caso però si è deciso di utilizzare il punto medio della base del rettangolo. Questo consente di avere nel modello a *bird-eye* una trasposizione più fedele della posizione reale dell'individuo.

Una volta calcolati tutti i centroidi delle persone nella scena, questi vengono mappati nel modello virtuale utilizzando la matrice di trasformazione citata in precedenza. Nello specifico, il *mapping* avviene grazie alla funzione *perspectiveTransform* di OpenCV.

Determinata la visione a *bird-eye* della scena si procede con la valutazione della distanza tra ogni coppia di centroidi, al fine di determinare quali sono i contatti a rischio. Nello specifico, viene calcolata la distanza euclidea tra due punti e, come soglia per determinare se due persone sono troppo vicine, viene utilizzato un valore che tiene conto delle dimensioni reali della stanza in cui si sta applicando il sistema. Dunque, al termine di questa fase, viene determinato per ogni coppia se stanno violando il distanziamento sociale o meno.

La capacità del sistema di individuare correttamente i contatti a rischio dipende fortemente dalle prestazioni della rete neurale utilizzata per effettuare la *Human Detection*. Infatti, come descritto, tutta la fase di verifica dipende dalle *bounding box* determinate dal modello utilizzato.

Nella prima fase del progetto si è quindi scelto di utilizzare la rete SSD (Single-Shot multibox Detector), che ha come backbone una VGG-16. Tale rete infatti, consente una localizzazione multi-scala e un *processing* delle immagini in real-time oltre che essere più veloce e accurata della rete YOLO. Nello specifico, l'obiettivo iniziale era quello di realizzare l'addestramento della rete utilizzando il dataset *Wider Person*[1].

Il dataset WiderPerson è un dataset di riferimento per il rilevamento dei pedoni, le immagini sono selezionate da una vasta gamma di scenari e non solo quello del traffico come per altri dataset. In totale comprende 13.382 immagini e circa 400mila annotazioni con vari tipi di occlusioni. La suddivisione in training e validation set predisposta dagli autori del dataset prevede 8000 immagini per il primo e 1000 per il secondo, le restanti immagini sono dedicate a comporre il testset e di queste non vengono fornite le etichette.

Purtroppo, nonostante le molteplici epoches di addestramento effettuate non è stato possibile raggiungere il livello di accuratezza desiderato probabilmente a causa di problemi della libreria scelta per eseguire la procedura. Dunque, si è deciso di proseguire andando a valutare alcuni modelli pre-addestrati messi a disposizione da Tensorflow, al fine di determinare quale di questi sia il più adatto per il sistema sviluppato.

Il *TensorFlow2 Detection Model Zoo* [12] è una collezione di modelli per *object-detection* pre-addestrati sul dataset COCO [2]. Quest'ultimo avendo tra le sue 80 classi anche quella "persona" ha reso possibile l'impiego di tali modelli per la risoluzione della problematica studiata. La principale accortezza necessaria quando si esegue l'operazione di *inference* è quella di filtrare i risultati solo per la classe di interesse, in questo caso, quella di indice 1. Per ogni modello contenuto nel *Model Zoo* viene riportata la mAP sul dataset di riferimento e la velocità di inferenza. Considerando che il sistema sviluppato deve essere in grado di elaborare le immagini in tempo reale, si è scelto di valutare i modelli con una *speed* inferiore ai 40 ms, questo ovviamente comporta una minore *mean Average Precision*.

I modelli del *Model Zoo* che si è scelto di valutare sono:

- **SSD MobileNet V2 FPNlite**: inference speed: 22ms COCO mAP: 22.2
- **Centernet Resnet50 V2**: inference speed: 27ms COCO mAP: 29.5
- **Centernet Resnet50 V1 FPN**: inference speed: 27ms COCO mAP: 31.2
- **SSD MobileNet V2**: inference speed: 19ms COCO mAP: 20.2
- **EfficientDet D0**: inference speed: 39ms COCO mAP: 33.6
- **Centernet Resnet101 V1 FPN**: inference speed: 34ms COCO mAP: 34.2

Inoltre si è deciso di valutare anche una versione pre-addestrata su Pascal VOC 2007 e 2012 della rete scelta inizialmente, la SSD300.

In figura 4 viene riportato un riassunto delle fasi del sistema appena descritto.

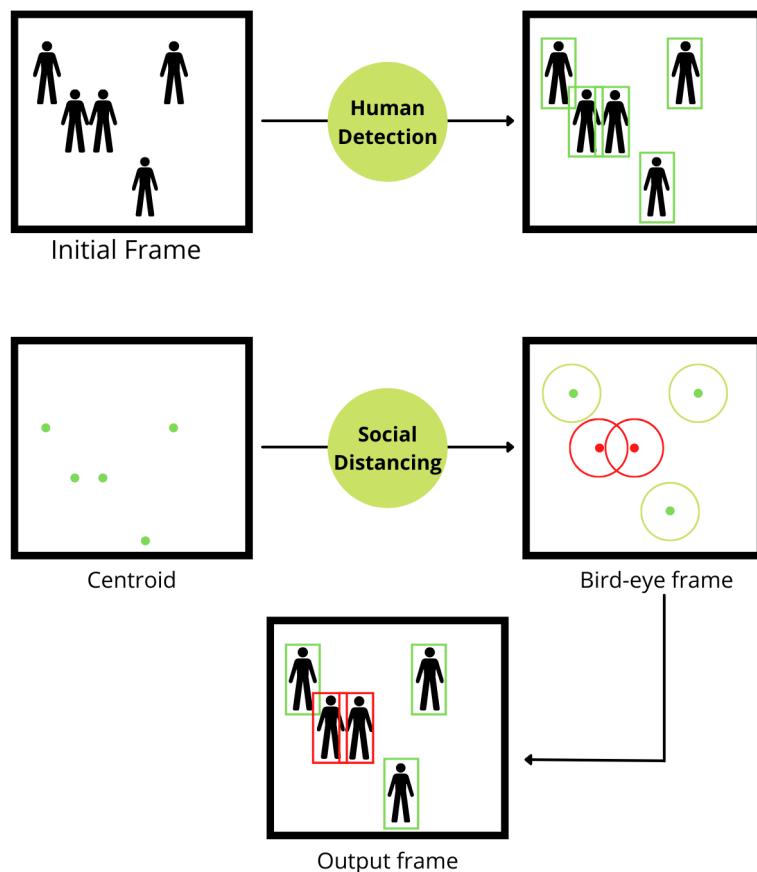


Figura 4: Fasi del sistema di verifica del distanziamento sociale

I risultati della valutazione delle prestazioni e la descrizione del test set utilizzato sono riportati nella sezione 5.

5 Valutazione delle prestazioni

In questa sezione vengono valutate le prestazioni dei modelli sviluppati, ponendo particolare attenzione sulla precisione delle reti ed effettuando un'attenta analisi degli errori.

5.1 Rilevamento delle mascherine

Per quanto riguarda il rilevamento delle mascherine, è stato necessario testare e valutare le prestazioni di entrambe le reti neurali utilizzate per il procedimento descritto nella sezione 4.1. In particolare, è stato prodotto un dataset etichettato da utilizzare come test set, con diverse persone sparse all'interno di un'aula universitaria; affinché il test set rappresenti a sufficienza il problema in analisi, oltre ad avere un'inquadratura ottimale, le persone riprese sono situate in diverse zone della stanza per vedere come reagisce il sistema sviluppato alla profondità e, inoltre, indossano la mascherina a intermittenza per verificare il corretto comportamento del modello.

Per valutare le prestazioni del modello di *Face Detection* è stato necessario controllare l'*Intersection Over Union (IoU)* tra le *bounding box* prodotte dalla rete e quelle etichettate. A partire da questi valori, se superiori ad una determinata soglia, sono stati determinati i *True Positive* individuati dalla rete, per poi calcolare **Precision** e **Recall** sull'intero dataset.

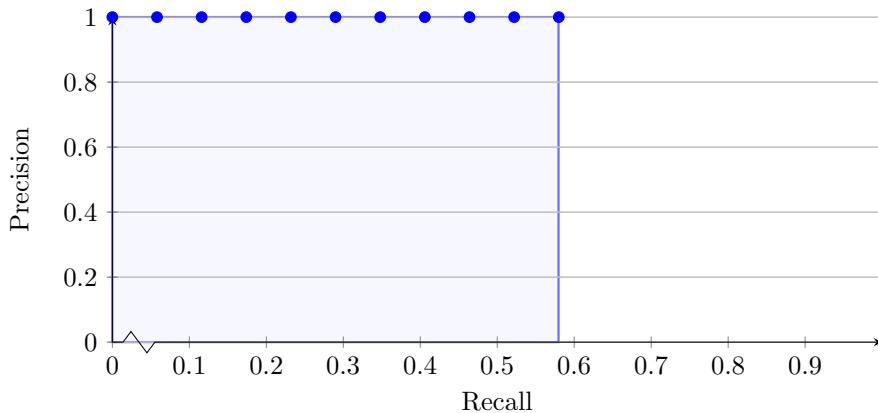


Figura 5: Andamento della curva *precision-recall*.

In figura 5 viene mostrata la curva *precision/recall* derivata dal test set: come si può notare, la *precision* non cala mai poichè non vengono mai individuati dei **falsi positivi**; mentre la *recall* si ferma a 0.58 in quanto si verificano dei **falsi negativi** (non vengono rilevati alcuni elementi del test set). Da questa curva si evince anche il valore dell'*average precision*:

$$\text{Average Precision} = 0.580$$

Si ritiene che il risultato ottenuto sia soddisfacente in quanto i volti che non vengono individuati appartengono a persone molto in profondità nell'aula oppure a persone in posizioni non frontali all'inquadratura; quindi con stanze con capienza minore aumenterebbe la precisione del modello.

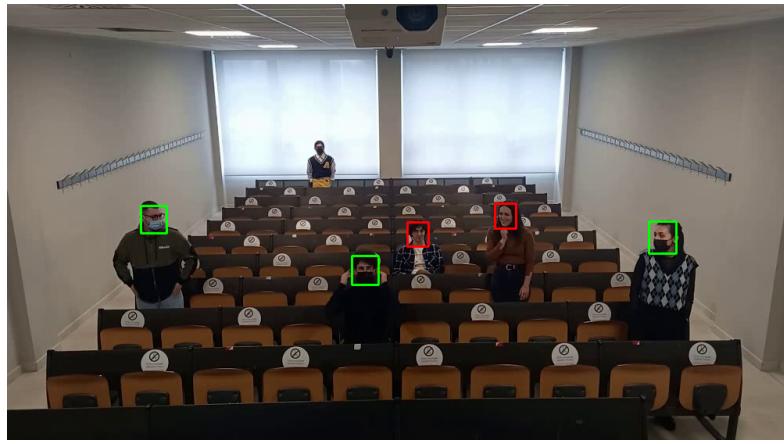


Figura 6: Immagine del test set con *bounding box* verdi sui volti con mascherina e rosse su quelli senza.

Per valutare le prestazioni per la *Mask Classification*, il modello addestrato è stato testato con il test set etichettato. In particolare, sono stati calcolati il valore di **Accuracy** e la **Confusion Matrix** relativi al test set in questione.

Global Accuracy = 0.848

Mean Class Accuracy = 0.591

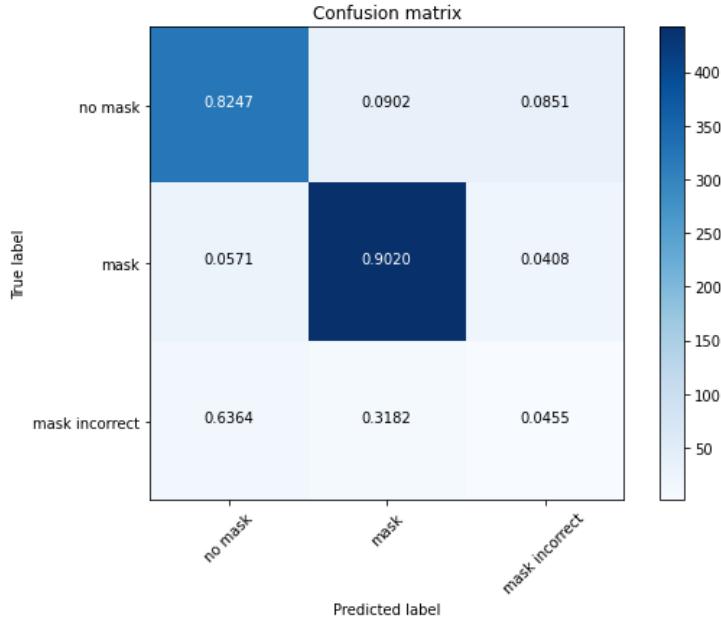


Figura 7: Matrice di confusione del modello di *mask classification* con approccio a rete neurale sul test set.

I risultati ottenuti si ritengono soddisfacenti, in quanto l'unica accuratezza con un valore basso (al di sotto di una soglia del 80%) risulta essere quella della classe **mascherina indossata non correttamente**. Infatti, tale valore fa decrescere quello della **Mean Class Accuracy**; inoltre, si ritiene che tale percentuale sia giustificata per diversi motivi:

- Scarsa disponibilità di istanze di tale classe all'interno del test set; infatti, in questo modo, un errore incide maggiormente sulla percentuale finale.
- Soggettività del criterio di discriminazione tra tale classe e le altre; infatti, anche l'etichettatura stessa del test set ha fatto scaturire opinioni divergenti tra i vari membri del gruppo.

Inoltre, le istanze erroneamente classificate appartenenti alle prime due classi sono giustificate da occlusioni, come mani sul volto, ma anche dall'imperfetta etichettatura effettuata: le *bounding box* manualmente individuate non coincidono sempre con il volto sottostante per via della bassa risoluzione.

Si riportano gli stessi dati ottenuti utilizzando un classificatore di tipo *Support Vector Machine*.

Global Accuracy = 0.541

Mean Class Accuracy = 0.335

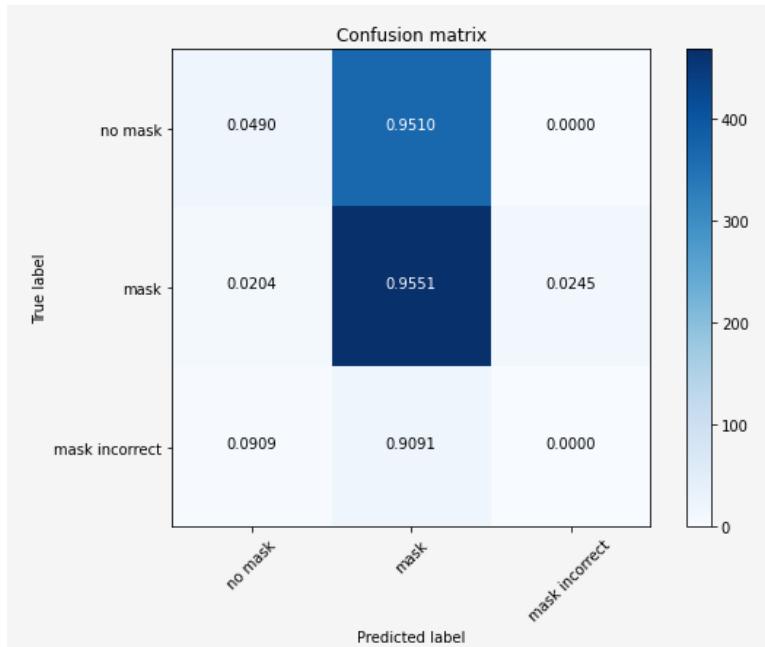


Figura 8: Matrice di confusione del modello di *mask classification* con classificatore SVM sul test set.

Come si può notare dalla matrice di confusione, il classificatore SVM è fortemente sbilanciato sulla classe **mascherina indossata correttamente**: nonostante l'utilizzo dello stesso dataset

per l’addestramento e il processo di *data augmentation*, le sue prestazioni sul test set prodotto non sono soddisfacenti. Forse potrebbero migliorare su un test set con diversa illuminazione ed etichettatura più precisa, ma siccome le prestazioni ottenute da questo secondo approccio non raggiungono i livelli del primo, le successive valutazioni sull’intero modello verranno effettuate utilizzando l’approccio a rete neurale.

Infine, per valutare le prestazioni di tutto il modello, testando in sequenza i task di *Face Detection* e *Mask Classification*, è stato calcolato il valore di **Average Precision** per ogni classe, sempre sullo stesso test set; da tali dati, è stata infine ottenuta la **Mean Average Precision**, indicativa delle prestazioni globali dell’intero software.

Average Precision on class 0 (no mask) = 0.392

Average Precision on class 1 (mask) = 0.526

Average Precision on class 2 (incorrect mask) = 0.046

Mean Average Precision = 0.321

Average Inference Time = 0.765

Il risultato finale non risulta essere particolarmente soddisfacente. Tuttavia, si sottolinea che le basse percentuali ottenute sono influenzate principalmente dalla bassa **Average Precision** del modello di *Face Detection*: infatti, i risultati ottenuti su ogni classe rispecchiano, in termini di proporzioni, quelli ottenuti nel task di *Mask Classification*, seppure più bassi. Quindi, ci si ritiene parzialmente soddisfatti, considerando anche il fatto che il test set appositamente etichettato, oltre a non avere condizioni di luce e contrasto ideali, contiene istanze particolarmente complesse come persone in profondità o in posizione non frontale all’inquadratura.

5.2 Controllo del distanziamento sociale

Per effettuare un’attenta analisi delle prestazioni dei modelli scelti per affrontare il problema, si è deciso di utilizzare un test set creato appositamente per il progetto. Le immagini del dataset sono infatti dei frame tratti da alcuni video registrati dal gruppo all’interno di un’aula universitaria. In questo modo, i modelli sono valutati su immagini simili al contesto reale di applicazione del sistema.

Il test set è composto da 163 immagini, ognuna delle quali contiene almeno 5 persone sparse all’interno di una stanza. I soggetti sono tutti in piedi ma parzialmente coperti dai banchi presenti nell’aula. Inoltre, le persone indossano tutte la mascherina e non sono sempre frontali alla fotocamera. Queste caratteristiche rendono più complessa l’individuazione delle persone da parte delle reti non addestrate su contesti simili.

Le annotazioni del test set sono state realizzate dal gruppo; per ogni persona viene riportato l’angolo in alto a sinistra e quello in basso a destra del rettangolo che la circoscrive. In aggiunta, per ciascun frame è stata inserita un’etichetta che indica se nell’immagine viene rispettato o meno il distanziamento sociale; nello specifico, viene inserito “1” nel primo caso “0” altrimenti.

In figura 9 vengono mostrati due esempi di immagini contenute nel test set con le relative annotazioni.



Figura 9: Esempi di immagini del test set con i relativi *bounding box*

Dato il contesto di applicazione del sistema e la conseguente necessità di un modello in grado di elaborare le immagini in *real-time*, si è in primo luogo valutata la velocità di elaborazione dei frame, comprendendo sia la fase di *Human Detection* svolta dal modello in esame, sia quella di *Transform* e *Distancing Check*. Nella tabella seguente vengono riportati i modelli ordinati per *Execution Speed* crescente: la colonna *Detection Speed* rappresenta invece la velocità di *predict* del modello per singolo frame espresso in secondi; infine, la colonna *FPS* rappresenta il numero di frame che si riescono ad elaborare in un secondo.

Model	Detection Speed	Execution Speed	FPS
centernet resnet50V1 fpn	0.0438188	0.0438496	23
centernet resnet50V2	0.0555336	0.0555633	18
centernet resnet101V1 fpn	0.0571568	0.0571863	17
ssd300	0.0897187	0.0897409	11
ssd mobilenetV2 fpnlite	0.102447	0.102477	10
ssd mobilenetV2	0.116552	0.116584	9
efficientdet D0	0.296755	0.296787	3

Come si può notare dalla tabella, la velocità di elaborazione è determinata totalmente dalla velocità del modello nell'eseguire la predizione, mentre tutte le elaborazioni effettuate per verificare il rispetto del distanziamento sociale aggiungono un ritardo trascurabile.

Come già descritto, la capacità del sistema di realizzare un adeguato controllo del distanziamento sociale dipende fortemente dalla precisione della rete nell'individuare le persone nella stanza sorvegliata. La metrica utilizzata per valutare le prestazioni dei modelli che effettuano *Object Detection* è la *Mean Average Precision*; nel nostro caso però i modelli sono utilizzati unicamente per eseguire *Human Detection*: si è quindi scelto di valutare la *Average Precision* solo per tale classe.

Come per il *Face Detection*, l'*Average Precision* è stata calcolata partendo con il determinare i *True Positive* e i *False Positive* per ciascun frame del test set. Da questi sono poi state calcolate

Precision e *Recall*, le curve necessarie per determinare l' *Average Precision*.

Di seguito si riporta una tabella che mostra i modelli ordinati per *Average Precision* decrescente.

Model	AP
centernet resnet101V1 fpn	0.921227
centernet resnet50V1 fpn	0.913972
efficientdet D0	0.907447
centernet resnet50V2	0.853472
ssd mobilenetV2 fpnlite	0.788238
ssd mobilenetV2	0.654942
ssd300	0.443307

Per quanto riguarda i valori di *AP* riportati è bene sottolineare che questi sono fortemente influenzati da due valori di soglia: quello relativo alla confidenza di predizione (lo *score*), utilizzato per filtrare i risultati della *predict*, e quello relativo al valore dell'*Intersection Over Union*, utilizzato per determinare *True and False Positive*. A seguito di vari test, si sono ritenuti adatti al problema e ai modelli utilizzati i seguenti valori di soglia: 0.45 per lo *score* e 0.5 per l'*IoU*.

Al fine di valutare l'effettiva capacità del sistema di individuare nelle varie immagini i contatti a rischio, si è deciso di far produrre al sistema una *label* per ogni frame. L'etichetta indica se nella relativa immagine è presente o meno almeno una violazione al distanziamento sociale. In questo modo, confrontando le etichette del test set con quelle prodotte dal sistema, è stato possibile valutare le prestazioni di classificazione per ciascun modello.

Di seguito si riportano le *Confusion Matrix* per ciascuno dei modelli in esame.

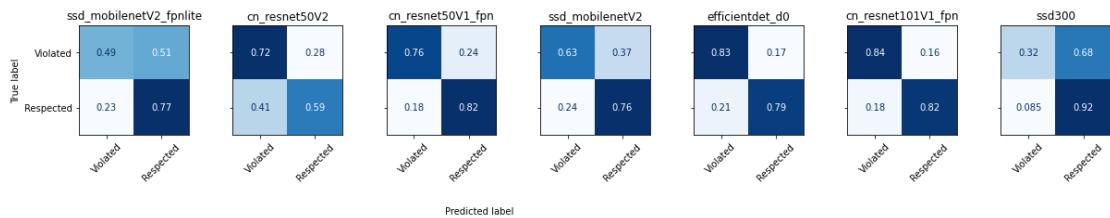


Figura 10: *Confusion Matrix* dei modelli

Dalle matrici di confusione mostrate è possibile notare come i modelli con bassa *Average Precision* determinino un maggior numero di falsi positivi rispetto agli altri. Un *False Positive* si verifica quando un frame in cui vi è una violazione del distanziamento sociale non viene classificato come tale. La correlazione tra *Average Precision* bassa e un alto numero di falsi positivi è data dal fatto che il modello non riesce ad individuare tutte le persone presenti all'interno dell'immagine e dunque alcuni dei contatti non vengono neanche esaminati dalla fase del sistema responsabile di individuare quelli a rischio.

Per quanto riguarda i falsi negativi, cioè quei frame in cui viene individuata una violazione del distanziamento sociale inesistente, è possibile notare che la percentuale si mantiene in generale bassa. Questo indica che il sistema per la verifica è stato calibrato adeguatamente, sia per quanto riguarda la matrice di trasformazione utilizzata per determinare la visione a bird-eye, sia per il valore utilizzato per individuare le persone troppo vicine.

Solo per la CenterNet ResNet50 V2 notiamo una percentuale di *False Negative* più elevata, analizzando tali errori si è dedotto che ciò è dovuto all'assenza di un'ulteriore fase di soppressione dei non massimi. Infatti per alcuni degli individui presenti nelle immagini vengono prodotti più di un solo *bounding box* e, data la loro intersezione, per il sistema di verifica questi determinano un contatto a rischio che in realtà non sussiste. Per chiarezza, viene riportato di seguito un falso negativo prodotto dalla rete in questione, i rettangoli rossi sono quelli che determinano la falsa violazione del distanziamento sociale.

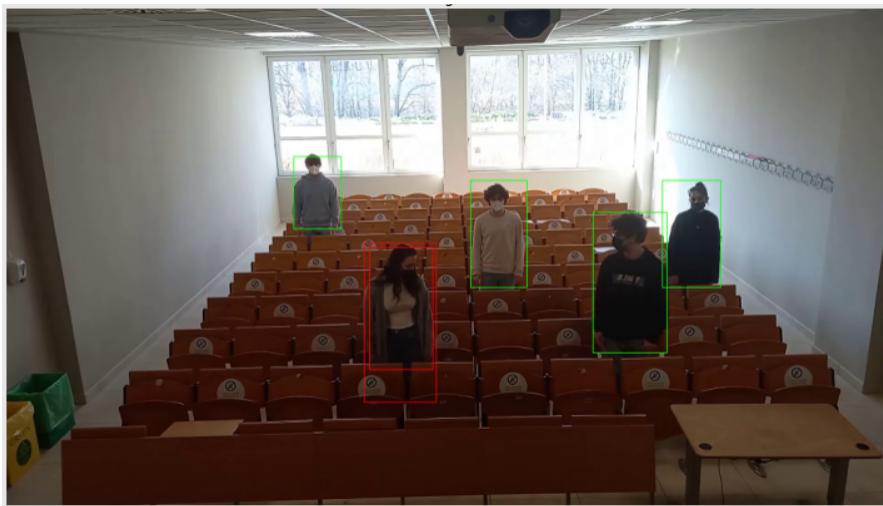


Figura 11: Esempio di Falso Negativo prodotto dalla CenterNet ResNet50 V2.

In generale, ci sono due principali motivazioni che generano entrambe le tipologie di errori (falsi positivi e falsi negativi): errate predizioni e errori nel calcolo della distanza tra le varie coppie di *bounding box*.

Per quanto riguarda i *False Positive*, questi sono generati principalmente da errori dovuti a un'errata fase di *Human Detection*; infatti quando due persone sono eccessivamente vicine o leggermente sovrapposte i modelli tendono o a non individuare i soggetti o ad assegnare un unico *bounding box* causando inevitabilmente l'impossibilità di individuare il contatto a rischio. Gli errori dovuti ad un calcolo errato della distanza, generati da un'imprecisa conversione tra distanza in pixel e distanza in metri, sono presenti in numero esiguo.

Nel caso dei *False Negative*, come è già stato descritto, gli errori relativi alle predizioni riguardano l'individuazione di molteplici *bounding box* per la medesima persona o la produzione di rettangoli eccessivamente grandi che portano ad un centroide errato. Gli errori commessi

nella stima della distanza interpersonale avvengono invece, perlopiù quando si hanno persone "in fila", indice del fatto che la produzione del modello virtuale a bird-eye può essere ulteriormente migliorata.

Di seguito si riportano per ciascun modello un esempio di falso positivo e di falso negativo



Figura 12: Esempi di errori



Figura 13: Esempi di errori

Considerando il contesto di applicazione del sistema e le performance di ciascun modello, confrontando le reti tenendo conto del frame-rate di elaborazione, l'*Average Precision* e il numero di errori prodotti sul test set, si ottiene la seguente classifica:

Model	FPS	AP	errors
centernet resnet50V1 fpn	23	0.913972	80
centernet resnet50V2	18	0.853472	68
centernet resnet101V1 fpn	17	0.921227	73
ssd300	11	0.443307	128
ssd mobilenetV2 fpnlite	10	0.788238	102
ssd mobilenetV2	9	0.654942	88
efficientdet D0	3	0.907447	72

In conclusione il modello più adatto risulta essere: **CenterNet ResNet50 v1 FPN**.

6 Sviluppi futuri e conclusioni

L'elaborato nasce come progetto integrato con il corso di **Smart City e Tecnologie Mobili**, con lo scopo di analizzare l'utilizzo di tecnologie hardware e software per aiutare il contrasto della pandemia. I possibili sviluppi futuri dell'intero progetto sono molteplici: in particolare, sarebbe interessante espandere l'applicazione del sistema a contesti diversi da quello universitario, ad esempio conference room o negozi. In ambito di visione artificiale, un possibile sviluppo futuro potrebbe riguardare l'unione dei due modelli sviluppati: infatti, uno dei problemi del task di *Mask Detection* è quello di non riuscire ad individuare tutti i volti presenti nell'immagine poiché troppo in profondità o coperti da mani; unendo i modelli sviluppati si potrebbe forzare la ricerca del volto all'interno delle *region of interest* individuate dal task di *Human Detection*, così da rendere più efficace la prima soluzione.

Per concludere, l'idea del progetto è partita da un interesse personale dei membri del gruppo; portando avanti il lavoro, oltre ad essersi consolidata la cooperazione dei membri del team, sono state rafforzate le conoscenze di ciascuno riguardanti l'ambito del progetto. Inoltre, sono stati appresi a pieno gli argomenti del corso in questione e le nuove tecnologie che questo richiedeva di saper utilizzare. I requisiti individuati in fase di analisi sono stati raggiunti e il sistema è funzionante nella sua totalità. Infine, ci si ritiene soddisfatti del risultato ottenuto, consapevoli di poter riutilizzare quanto imparato in futuro, sia in ambito accademico, sia in ambito lavorativo. In particolar modo, ci si ritiene soddisfatti del software sviluppato in quanto ha dimostrato con efficacia le potenzialità offerte dal corso in questione, riuscendo a mettere in gioco abilità e creatività dei membri del gruppo.

Per ulteriori dettagli e informazioni, consultare il repository del progetto.

Riferimenti bibliografici

- [1] Shuo Yang et al. «WIDER FACE: A Face Detection Benchmark». In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.
- [2] COCO Consortium. *COCO 2017*. 2017. URL: <https://cocodataset.org/#home>.
- [3] Shifeng Zhang et al. «WiderPerson: A Diverse Dataset for Dense Pedestrian Detection in the Wild». In: *IEEE Transactions on Multimedia (TMM)* (2019).
- [4] Vijay Kumar. *Face Mask Detection*. 2020. URL: <https://www.kaggle.com/vijaykumar1799/face-mask-detection>.
- [5] *MAFA Dataset*. 2020. URL: <https://www.kaggle.com/revanthrex/mafadataset>.
- [6] Chandran Saravanan NJ Karthika. «Addressing False Positives in Pedestrian Detection». In: *International Conference on Electronic Systems and Intelligent Computing (ESIC 2020)*. Mar. 2020. DOI: https://doi.org/10.1007/978-981-15-7031-5_103. URL: <https://www.kaggle.com/karthika95/pedestrian-detection>.
- [7] *P-DESTRE Dataset*. 2020. URL: <http://p-destre.di.ubi.pt/index.html>.
- [8] peteryuX. *RetinaFace for Tensorflow2*. 2020. URL: <https://github.com/petaryuX/retinaface-tf2>.
- [9] Basile Roth. *A social distancing detector using a Tensorflow object detection model, Python and OpenCV*. 2020. URL: <https://towardsdatascience.com/a-social-distancing-detector-using-a-tensorflow-object-detection-model-python-and-opencv-4450a431238>.
- [10] Rutuja R. Mahurkar e Naresh G. Gadge. «Real-time Covid-19 Face Mask Detection with YOLOv4». In: *2021 Second International Conference on Electronics and Sustainable Communication Systems (ICESC)*. 2021, pp. 1250–1255. DOI: [10.1109/ICESC51422.2021.9533008](https://doi.org/10.1109/ICESC51422.2021.9533008).
- [11] Shilpa Sethi, Mamta Kathuria e Trilok Kaushik. «Face mask detection using deep learning: An approach to reduce risk of Coronavirus spread». In: *Journal of Biomedical Informatics* 120 (2021), p. 103848. ISSN: 1532-0464. DOI: <https://doi.org/10.1016/j.jbi.2021.103848>. URL: <https://www.sciencedirect.com/science/article/pii/S1532046421001775>.
- [12] Tensorflow. *TensorFlow 2 Detection Model Zoo*. 2021. URL: https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/tf2_detection_zoo.md.