

Visione Artificiale - Assignment 3

Giovanni Castelli, Ivan Sollazzo, Gabriele Nicolò Costa

June 2024

Contents

1	Introduzione	2
2	Tracking System	3
2.1	Architettura	3
2.1.1	DETR	3
2.1.2	VGG-16	3
2.2	Logica di Track	4
2.2.1	Inizializzazione	4
2.2.2	Rilevamento pedoni	4
2.2.3	Matching	4
2.2.4	Reidentificazione	5
2.2.5	Discard delle track scomparse	5
3	TrackEval framework	6
3.1	HOTA Metrics	6
3.2	CLEARMOT metrics	6
4	Working Pipeline	8
4.1	TrackEval Pipeline	8
5	Validazione	9
5.1	Risultati	9
5.2	Considerazioni	9
6	Testing	10
7	Considerazioni finali	11
7.1	Come si potrebbe migliorare	11
7.2	Esempi di re-identificazione	11
7.3	Suddivisione del lavoro	12
7.4	Video migliore	12
7.5	Video peggiore	12

1 Introduzione

Scopo dell'Assignment è quello di realizzare un sistema di Tracking capace di rilevare pedoni, assegnare loro una identità e cercare di seguirli per la durata del video.

Per realizzare ciò abbiamo avuto bisogno di:

- Un pedestrian detector (che ci fornisce in output le **detections**)
- Un sistema di tracking che elabora le detections (che ci fornisce in output le **tracks**)
- Un framework di valutazione delle metriche di tracks

Per quanto riguarda i video, utilizzeremo il train set della MOT17-Challenge, dove splitteremo i video in validation e test set.

Di seguito verranno descritte l'architettura e la logica utilizzata e verrà fornita una descrizione delle metriche di valutazione utilizzate. Infine, passeremo alla fase di validazione del nostro modello e verranno inclusi i risultati su validation-set e test-set.

2 Tracking System

2.1 Architettura

Il nostro tracking system è composto da tre parti fondamentali:

- Un detector di pedestrian
- Una rete convoluzione per fare feature representation
- Un sistema di tracking

Difatti la nostra idea si basa sul paradigma tracking by detection, ovvero:

- Tramite il detector rileviamo delle detection in un determinato frame
- Tramite il sistema di tracking cerchiamo di "matchare" o comunque di inseguire le track presenti al frame precedente, eventualmente creando nuove track e/o segnalando opportunatamente le track scomparse, creandoci così un sistema di **identificazione**
- Un approccio simil "rete siamese", basata su rete convoluzionale VGG-16 per fare in modo di estrarre le features della track scomparsa per fare in modo di matcharla in seguito in un frame successivo, nel processo di **reidentificazione**

Per quanto riguarda le policy utilizzate, si fa riferimento alla sezione "Logica di Track". Di seguito vengono descritti il detector e la rete convoluzionale utilizzata.

2.1.1 DETR

Il modello DETR ([repository github](#)) (DEtection TRansformers) è un modello di deep learning introdotto da Facebook per affrontare il problema del rilevamento degli oggetti. Difatti è destinato al rilevamento di oggetti multipli in una scena. Sebbene sia addestrato su un elevato numero di classi, noi filtriamo le detection alla classe 1 corrispondente ai **pedestrian**.

Il modello DETR Utilizza una combinazione di reti neurali convoluzionali (CNN) (in particolare Resnet-50) e trasformatori (transformers) per migliorare l'efficienza e l'accuratezza nel rilevamento degli oggetti. Tuttavia, sebbene è capace di rilevare oggetti, associa una confidence di rilevamento ad ogni detection: questa confidence diventa quindi un iperparametro per il nostro problema, dal momento che vogliamo tracciare quanti più pedoni in una scena.

2.1.2 VGG-16

VGG-16 è una rete convoluzionale (CNN) usata per la classificazione di oggetti. Essa è formata da una parte puramente convoluzionale, destinata all'estrazione di features, e da una parte fully connected per effettuare classificazione. Noi useremo una rete VGG-16 pre-addestrata, ma prenderemo solo l'uscita del layer fully-connected prima della classificazione, ottenendo così un vettore di 4096 features: ci avvaleremo di questo vettore per le metriche di similarità tra due crop di immagini (due detections) al fine di fare re-identificazione.

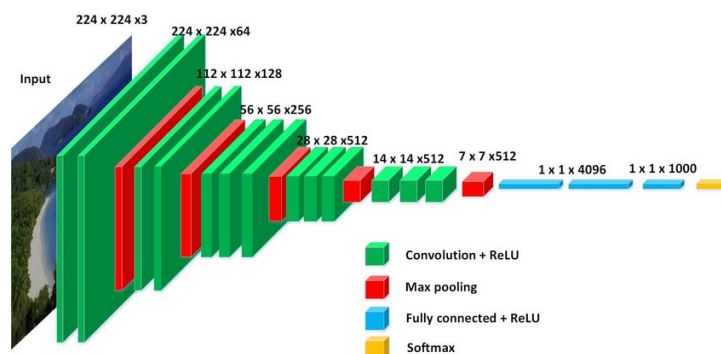


Figure 1: Architettura tipica di una VGG-16

Il nostro approccio è uno dei più usati in letteratura per fare feature representation, quindi per cercare di estrarre le caratteristiche che permettono poi effettivamente di misurare in maniera qualitativa la similarità tra due immagini o due pedestrian nel nostro caso.

2.2 Logica di Track

La logica di track si divide in cinque parti fondamentali:

- Inizializzazione del **tracker**
- Rilevamento dei pedoni (fase di **detection** al frame corrente)
- Associazione delle identità tra frame (fase di **matching**), con conseguente gestione di eventuali track perse e soglia sul matching delle identità, risolvendo un problema di **maximum bipartite matching**
- Reidentificazione delle identità con le detections non matchate con il frame precedente con le track che sono scomparse ma che risultano essere ancora "vive"
- Gestione delle track morte e rimozione delle track morte da quelle perse.

In sintesi, per tutta la durata del tracciamento noi lavoriamo con tre tipologie di track:

- Track vive (track matchate tra frame consecutivi)
- Track perse (track non matchate tra frame consecutivi ma che comunque non hanno superato la **memoria** del tracker, ovvero il numero massimo di frame in cui una track risulta essere scomparsa)
- Track morte (track che ho perso e non sono stato in grado di matcharle con le detections perché ho superato la memoria del sistema)

2.2.1 Inizializzazione

Inizialmente il nostro sistema di tracking non ha nessuna track istanziata, dunque assegniamo in fase iniziale un id ad ogni detection del primo frame.

2.2.2 Rilevamento pedoni

Usando il nostro DETR rileviamo le detection al frame corrente, pronte per essere assegnate ad un ID o eventualmente riassegnate ad un ID.

2.2.3 Matching

Nella fase di matching noi abbiamo a disposizione t tracks vive (provenienti dal frame precedente) e d detections (rilevate con il DETR). Per associare le tracks alle detections dobbiamo risolvere un problema di **maximum bipartite matching**, tramite algoritmo ungherese. Tuttavia questo algoritmo accetta in input una matrice di costo e ci restituisce i matching tra righe (tracks) e colonne (detections).

La matrice di costo viene realizzata usando come metriche la **IoU**: decidiamo di usare la IoU come metrica di matching perché anche se il soggetto si sposta velocemente, tra un frame e l'altro non può completamente trovarsi da un'altra parte, ma sicuramente nel frame successivo si troverà in una zona "sovrapposta" a quella del frame precedente. Tuttavia, ci sono tre casistiche da tenere in considerazione:

- La matrice di costo è quadrata (allora abbiamo che idealmente ogni track è associata ad ogni detection, nella pratica inseriamo un valore di soglia).
- La matrice di costo ha più righe che colonne, quindi alcune track sono andate perse.
- La matrice di costo ha più colonne che righe, quindi sono presenti nuove identità.

2.2.4 Reidentificazione

Nel caso in cui una track andasse persa (perché per esempio il valore di matching è superiore alla soglia fissata o perché per esempio rientriamo nel caso in cui ci siano più righe che colonne nella matrice di costo), allora le inseriamo in una lista di track scomparse, avviando un "contatore" di frame consecutivi in cui la track risulta essere scomparsa.

Cerchiamo inoltre di riassegnare ad ogni frame le track scomparse con le detections non matchate, risolvendo ancora una volta un problema di **maximum bipartite matching**. Tuttavia stavolta la matrice di costo viene realizzata tenendo conto sia della similarità delle features (estratte con VGG16) che di una eventuale intersezione. L'idea di usare una metrica combinata deriva dal fatto che potrebbero esserci soggetti quasi identici nello stesso frame, ma grazie alla IoU valuto eventualmente se siamo in prossimità di un determinato soggetto. Il contributo di IoU e similarità viene pesato tramite un parametro di combinazione lineare, empiricamente stabilito tramite test su diversi video e valutazioni diretta sui valori restituiti sia da IoU che da Similarity. Eventuali detections che non vengono matchate neanche con le identità scomparse allora risultano essere nuove identità.

2.2.5 Discard delle track scomparse

Come discusso precedentemente, avviamo un contatore di frame in cui una track risulta essere scomparsa e appena superiamo la memoria del Tracker (ovvero il numero massimo di frame in cui una track può risultare scomparsa), allora eliminiamo dalle track scomparse le track con valore di lost superiore alla memoria, inserendole in una lista di track morte.

3 TrackEval framework

Per misurare il nostro modello di Tracker utilizziamo sostanzialmente le HOTA metrics e le CLEARMOT metrics, di seguito viene fornita una breve descrizione delle metriche utilizzate in fase di validazione.

3.1 HOTA Metrics

HOTA (Higher Order Tracking Accuracy) è una metrica che combina aspetti di accuratezza del rilevamento e dell'associazione per fornire una valutazione completa delle prestazioni di un tracker.

- **HOTA (Higher Order Tracking Accuracy):** Questa è la metrica principale che combina le informazioni di rilevamento e associazione. È una misura complessiva dell'accuratezza di tracciamento, che considera sia la capacità di rilevare correttamente gli oggetti che quella di associarli correttamente nel tempo.
- **DetA (Detection Accuracy):** Misura l'accuratezza del rilevamento degli oggetti. Indica quanto bene l'algoritmo riesce a rilevare gli oggetti presenti nelle scene.
- **AssA (Association Accuracy):** Misura l'accuratezza dell'associazione degli oggetti rilevati. Indica quanto bene l'algoritmo riesce a mantenere l'identità degli oggetti attraverso diversi frame.
- **LocA (Localization Accuracy):** Misura la precisione con cui gli oggetti rilevati sono localizzati nello spazio. Si riferisce alla capacità dell'algoritmo di stimare correttamente le posizioni degli oggetti.
- **DetPr (Detection Precision):** Misura la precisione del rilevamento, ovvero la frazione di oggetti rilevati correttamente rispetto a tutti gli oggetti rilevati.
- **DetRe (Detection Recall):** Misura il recall del rilevamento, ovvero la frazione di oggetti rilevati correttamente rispetto a tutti gli oggetti presenti nella scena.
- **AssPr (Association Precision):** Misura la precisione dell'associazione, ovvero la frazione di associazioni corrette rispetto a tutte le associazioni effettuate.
- **AssRe (Association Recall):** Misura il recall dell'associazione, ovvero la frazione di associazioni corrette rispetto a tutte le possibili associazioni corrette.

Le metriche HOTA forniscono una valutazione dettagliata delle prestazioni di un algoritmo di tracciamento degli oggetti, considerando sia l'accuratezza del rilevamento che quella dell'associazione degli oggetti nel tempo. Queste metriche aiutano a comprendere meglio le capacità e le limitazioni di un tracker in diversi scenari.

3.2 CLEARMOT metrics

Le metriche CLEARMOT sono comunemente utilizzate per valutare le prestazioni degli algoritmi di tracciamento multiplo di oggetti (Multi-Object Tracking, MOT).

- **MOTA (Multi-Object Tracking Accuracy):** Questa metrica combina diversi fattori di errore, tra cui falsi positivi, falsi negativi e errori di associazione, per fornire una misura complessiva dell'accuratezza del tracciamento. Viene calcolata come:

$$MOTA = 1 - \frac{\sum_t (FP_t + FN_t + IDSW_t)}{\sum_t GT_t}$$

dove FP_t è il numero di falsi positivi al tempo t , FN_t è il numero di falsi negativi, $IDSW_t$ è il numero di cambiamenti di ID, e GT_t è il numero di oggetti nel ground truth al tempo t .

- **MOTP (Multi-Object Tracking Precision):** Questa metrica misura la precisione con cui i tracker stimano le posizioni degli oggetti. Viene calcolata come la media della distanza tra le posizioni stimate e quelle reali degli oggetti tracciati:

$$MOTP = \frac{\sum_{i,t} d_{i,t}}{\sum_t c_t}$$

dove $d_{i,t}$ è la distanza tra la posizione stimata e quella reale dell'oggetto i al tempo t , e c_t è il numero di corrispondenze tra gli oggetti stimati e quelli reali al tempo t .

- **MT (Mostly Tracked):** Rappresenta la percentuale di oggetti che sono stati tracciati per almeno l'80% della loro durata nel ground truth.
- **ML (Mostly Lost):** Rappresenta la percentuale di oggetti che sono stati tracciati per meno del 20% della loro durata nel ground truth.
- **Frag (Fragmentations):** Il numero di volte che un singolo oggetto nel ground truth ha un tracciamento frammentato, cioè il tracciamento è stato interrotto e poi ripreso.
- **FP (False Positives):** Il numero totale di rilevamenti erronei dove un oggetto è stato rilevato ma non esiste nel ground truth.
- **FN (False Negatives):** Il numero totale di oggetti nel ground truth che non sono stati rilevati dal tracker.
- **IDSW (ID Switches):** Il numero di volte che l'ID assegnato ad un oggetto cambia durante il tracciamento.

4 Working Pipeline

Per la realizzazione del sistema di tracking ci serviamo di due file:

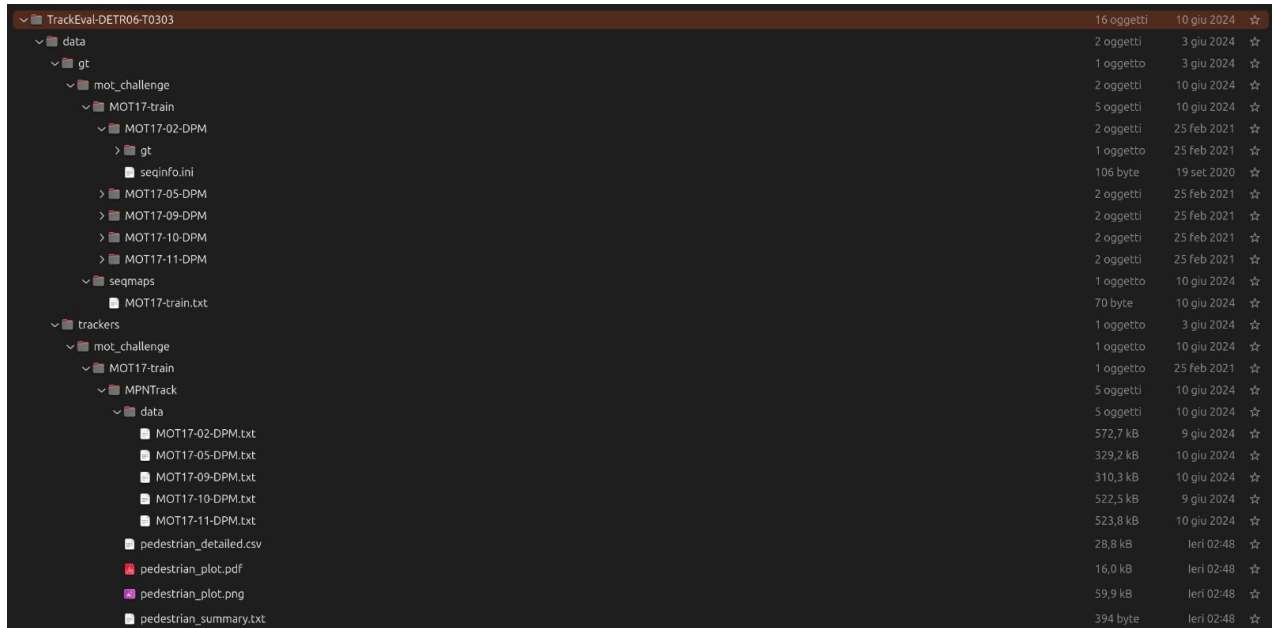
- Un file per estrarre le detections da una determinata sequenza di immagini, impostando in maniera opportuna la soglia del detector DETR (**detector.ipynb**)
- Un file per elaborare il video sulla base delle detections pre caricate (precedentemente calcolate), che ci permette di impostare soglie di matching e di reidentificazione (**tracker.ipynb**)

Inoltre, viene implementata anche una funzione per creare un video a partire dalla sequenza di immagini e dai tracks estratti dal secondo file.

In fase di validazione quindi proseguiamo prima con l'estrazione delle detections e dopo con la rilevazione dei track al variare degli iperparametri di validazione:

- Confidence sul DETR: conf 0.6 e conf 0.7
- Soglia di matching: 0.3, 0.4, 0.5
- Soglia di reidentificazione: 0.3, 0.4, 0.5

4.1 TrackEval Pipeline



TrackEval-DETR06-T0303	16 oggetti	10 giu 2024	☆
data			
gt	2 oggetti	3 giu 2024	☆
mot_challenge	1 oggetto	3 giu 2024	☆
MOT17-train	2 oggetti	10 giu 2024	☆
MOT17-02-DPM	5 oggetti	10 giu 2024	☆
gt	2 oggetti	25 feb 2021	☆
seqinfo.ini	1 oggetto	25 feb 2021	☆
MOT17-05-DPM	106 byte	19 set 2020	☆
MOT17-09-DPM	2 oggetti	25 feb 2021	☆
MOT17-10-DPM	2 oggetti	25 feb 2021	☆
MOT17-11-DPM	2 oggetti	25 feb 2021	☆
seqmaps	2 oggetti	25 feb 2021	☆
MOT17-train.txt	1 oggetto	10 giu 2024	☆
trackers	70 byte	10 giu 2024	☆
mot_challenge	1 oggetto	3 giu 2024	☆
MOT17-train	1 oggetto	10 giu 2024	☆
MPNTrack	1 oggetto	25 feb 2021	☆
data	5 oggetti	10 giu 2024	☆
MOT17-02-DPM.txt	5 oggetti	10 giu 2024	☆
MOT17-05-DPM.txt	572,7 kB	9 giu 2024	☆
MOT17-09-DPM.txt	329,2 kB	10 giu 2024	☆
MOT17-10-DPM.txt	310,3 kB	10 giu 2024	☆
MOT17-11-DPM.txt	522,5 kB	9 giu 2024	☆
pedestrian_detailed.csv	523,8 kB	10 giu 2024	☆
pedestrian_plot.pdf	28,8 kB	1eri 02:48	☆
pedestrian_plot.png	16,0 kB	1eri 02:48	☆
pedestrian_summary.txt	59,9 kB	1eri 02:48	☆
	394 byte	1eri 02:48	☆

Figure 2: Gerarchia delle directory

5 Validazione

Di seguito vengono riportate le metriche al variare degli iperparametri sui video di validazione:

- MOT17-02
- MOT17-05
- MOT17-09
- MOT17-10
- MOT17-11

Di seguito verranno riportate le tabelle in relazione alla confidence del DETR utilizzata e alle soglie su matching e reidentification usate, descritte nella sezione precedente

5.1 Risultati

Per le tabelle si faccia riferimento al Notebook python allegato

5.2 Considerazioni

Dall'analisi delle statistiche, notiamo come i migliori iperparametri sulla base delle migliori metriche di HOTA e MOTA sono:

- **Confidence detector:** 0.7
- **Soglia matching:** 0.5
- **Soglia re-id:** 0.5

Notiamo inoltre, tuttavia, che la scelta non è stata affatto facile in quanto più combinazioni di iperparametri hanno valori di MOTA ed HOTA simili tra di loro, dunque decidiamo di prendere il miglior iperparametro mediano sulla base di HOTA e MOTA.

6 Testing

Di seguito si riportano le statistiche sui video di test sulla base dei migliori iperparametri determinati, ovvero **valore di confidence 0.7, soglie a 0.5 per matching e reidentification**. I video utilizzati come test sono stati:

- MOT17-04
- MOT17-13

Anche qui, per le tabelle con i risultati si fa riferimento al notebook python allegato.

7 Considerazioni finali

7.1 Come si potrebbe migliorare

Da una analisi dei valori ottenuti e da una analisi visiva notiamo come effettivamente siano presenti i problemi tipici di un sistema di tracking:

- In caso di occlusione di pedoni, si rischia di assegnare un nuovo id a causa della "lost" di un track
- La memoria del sistema di track si limita a 30 frame, il ch   si    rivelato utile in molti frame consecutivi, nel caso in cui per esempio una persona non veniva rilevata correttamente dal detector, ma veniva rilevata qualche frame pi   avanti. Dunque potrebbe essere necessario aumentare la memoria
- I valori di sogliatura sul matching sono adatti a seguire spostamenti "lenti" di pedoni, ma in caso di spostamenti pi   veloci risultano essere molto stringenti (questo viene spiegato in parte dal numero di ID creati)

7.2 Esempi di re-identificazione

Ci   nonostante, viene comunque riportato un esempio di re-identificazione tra frame successivi rilevato durante la fase sperimentale e di test: Come possiamo notare dalle quattro immagini, inizialmente



(a) Frame 1



(b) Frame 2



(c) Frame 3



(d) Frame 4

Figure 3: Esempio di reidentificazione in fase sperimentale

l'identit   19 viene rilevata (a destra nella immagine). Dopo un frame il nostro detector la perde, per due frame consecutivi. Dopo due frame per   il pedone viene correttamente rilevato e quindi viene fatta correttamente re-identificazione.

7.3 Suddivisione del lavoro

Per quanto concerne la suddivisione del lavoro tra i componenti del gruppo si è proceduti nel seguente modo:

- Algoritmo di matching dei track: Sollazzo
- Algoritmo di re-identificazione dei track: Costa
- Esperimenti e ottimizzazione: Sollazzo, Costa
- Estrazione detections e tracks dai video: Castelli, Costa
- Validazione del tracker: Sollazzo, Castelli
- Test video tracker: Castelli
- Relazione tabelle e documento finale: Costa, Castelli, Sollazzo.

7.4 Video migliore

Sulla base delle metriche valutate (Hota e Clearmote) il video migliore è il MOT17-4.

Complessivamente notiamo che il sistema di tracking funziona, poiché la telecamera è fissa ed i pedoni non si muovono troppo velocemente.

Inoltre, notiamo come è presente un problema di occlusione in quanto notiamo dal video che se è presente un pedone che attraversa completamente un altro pedone, nella maggior parte dei casi viene associato l'id del secondo pedone a quello del primo (Switching Id).

Inoltre, se visualizziamo il video al rallentatore, noteremo che molte track all'inizio perse vengono re-identificate correttamente

Si allega il video: [Im Hereeee](#)

7.5 Video peggiore

Sulla base delle metriche valutate (Hota e Clearmote) il video peggiore è il MOT17-13.

In maniera Empirica abbiamo notato che a partire dal frame 487 il detector non riesce più a rilevare persone, inoltre il video presenta sia pedoni che camera in movimento, il che rende più complesso il meccanismo di matching e reidentificazione.

Si allega il video: [è lo stesso di sopra](#)