Università degli Studi di Udine

Corso di Laurea Magistrale in Comunicazione Multimediale e Tecnologie dell'Informazione

Corso di Machine Learning

Pose2Seg: a reproducibility test. Adaptation and qualitative analysis of the Ski2Dpose Dataset.

Stanislao Bellantoni 159071
Marco Del Ben 128920

Anno Accademico 2022/2023

Indice

Elenco delle figure

1 Struttura del report

La prima sezione introduce il contesto del progetto e del paper "Pose2seg: Detection Free Human Instance Segmentation", delineando l'obiettivo del reproducibility test e la sua importanza nel contesto della ricerca in machine learning.

Successivamente, la sezione sullo *Stato dell'Arte* fornisce una panoramica delle metodologie esistenti nella detection-free human instance segmentation, mettendo in luce le sfide e le limitazioni dei metodi tradizionali.

La sezione *Metodologia* delinea il processo seguito per riprodurre il paper, compresa l'implementazione del codice proposto e l'analisi dei risultati ottenuti nelle fasi di train e test del modello.

Una particolare attenzione è rivolta all'adattamento e al testing del dataset Ski2dpose utilizzando il modello proposto da Pose2Seg, con un'analisi qualitativa dell'output ottenuto.

Infine, la sezione sulle *Future implementazioni* offre spunti per possibili sviluppi futuri del progetto, suggerendo miglioramenti, estensioni o nuove direzioni di ricerca nel campo della detection-free human instance segmentation.

Le *Conclusioni* riassumono i principali risultati ottenuti nel report, sottolineando le sfide affrontate.

2 Introduzione

Attraverso un approccio pratico, è stato posto l'obiettivo di trovare una soluzione innovativa in un contesto di applicazione reale, con la prospettiva di acquisire competenze concrete e prepararsi per possibili opportunità lavorative nel settore IT.

Con l'implementazione di questo progetto, si mira a mettere in pratica le conoscenze teoriche acquisite durante il percorso accademico, affrontando una sfida reale e lavorando su un problema significativo nel campo della computer vision e nello specifico nella segmentazione umana *detection-free*.

Il primo obiettivo del progetto è comprendere appieno il contenuto del paper "Pose2Seg: Detection Free Human Instance Segmentation" e cercare di riprodurne i risultati in modo accurato e affidabile.

Questo coinvolge diverse fasi:

1. *Comprensione del paper*: In questa fase, ci si è immersi nello studio del paper, analizzando i suoi concetti fondamentali, gli obiettivi, i metodi proposti e i risultati ottenuti;

2. *Implementazione del codice*: Successivamente, utilizzando Google Colab, verrà implementato il codice fornito dal paper per riprodurre gli esperimenti e i risultati presentati;

3. *Valutazione dei risultati*: Una volta completata l'implementazione, verranno valutati i risultati ottenuti durante le fasi di addestramento e test del modello.

   Questo consentirà di confrontare i risultati ottenuti con quelli riportati nel paper originale e di valutare l'accuratezza e l'affidabilità della riproduzione.

Il secondo obiettivo è quello di adattare e testare il dataset Ski2DPose utilizzando il modello proposto da Pose2Seg.

Questo permetterà di condurre un'analisi qualitativa sull'output ottenuto dal modello quando applicato a un contesto diverso da quello originale del paper.

Le fasi principali di questo obiettivo includono:

1. *Preparazione del dataset*: Si inizia preparando il dataset Ski2DPose per l'input nel modello, adattandone la struttura dati in modo tale da poterlo "dare in pasto" a Pose2Seg (in questa fase considerato come "black-box").

2. *Esecuzione del modello su Ski2DPose*: Successivamente, verrà eseguito il modello proposto da Pose2Seg sul dataset Ski2DPose preparato secondo gli standard richiesti dall'architettura originale proposta dal paper.

3. *Analisi qualitativa dell'output*: Infine, verrà condotta un'analisi qualitativa dell'output ottenuto dal modello, valutando la sua capacità di segmentare istanze umane nel contesto specifico del dataset Ski2DPose.

   Questo consentirà di identificare eventuali sfide o limitazioni nell'applicazione del modello e di suggerire possibili miglioramenti o adattamenti.

3 Stato dell'Arte

Nell'attuale stato dell'arte della computer vision, una delle sfide principali nell'ambito dell'object detection è rappresentata dall'occlusione parziale o totale degli oggetti nell'immagine.

Metodi tradizionali come Faster R-CNN[2] e Mask R-CNN[1] hanno ottenuto risultati significativi nella segmentazione delle istanze umane, ma possono essere influenzati da problemi come l'occlusione e la sovrapposizione, rendendo difficile ottenere risultati accurati in scenari complessi.

L'articolo "Pose2seg: Detection Free Human Instance Segmentation"[5] propone un approccio innovativo che affronta questa sfida senza la necessità di una fase di rilevamento separata.

Pose2seg sfrutta le informazioni di posa umana per guidare il processo di segmentazione, consentendo una segmentazione accurata anche in scenari con occlusione parziale o totale.

Un confronto con metodi tradizionali come Faster R-CNN e Mask R-CNN evidenzia le superiorità di Pose2seg nell'affrontare situazioni di occlusione e sovrapposizione, offrendo una soluzione più efficiente e accurata per la segmentazione delle istanze umane.

Un altro approccio rilevante nel campo della segmentazione degli oggetti è rappresentato dal paper "Pose-Guided Knowledge Transfer for Object Part Segmentation"[3].

Questo lavoro propone un metodo innovativo che sfrutta le conoscenze acquisite dalla posa per affrontare sfide come l'occlusione parziale o totale e migliorare l'accuratezza complessiva della segmentazione delle parti degli oggetti.

Infine, un ulteriore contributo nel panorama della computer vision è rappresentato dal paper "Pose-aware Multi-level Feature Network for Human Object Interaction Detection"[4]. Questo lavoro si concentra sulla detection delle interazioni tra persone e oggetti, utilizzando informazioni di posa per migliorare l'accuratezza e la precisione della detection. L'approccio proposto integra una rete neurale complessa, in grado di estrarre e combinare le caratteristiche delle immagini a diversi livelli di astrazione, per identificare le interazioni umano-oggetto in modo efficace anche in scenari complessi.

Nel seguente capitolo verrà illustrata la metodologia per la riproducibilità di Pose2Seg.

4 Metodologia

Per replicare il paper "Pose2seg: Detection Free Human Instance Segmentation", è stata seguita una metodologia che comprende diverse fasi, dalla preparazione dell'ambiente di lavoro all'adattamento del codice e all'esecuzione del train e test.

1. *Scaricamento della repository ufficiale del paper su GitHub*: È stata scaricata la repository ufficiale del paper da GitHub. Questo ha fornito accesso al codice sorgente necessario per replicare gli esperimenti.

2. *Scelta del framework per l'implementazione - Google Colab*: È stato scelto Google Colab come framework per l'implementazione del paper.

   Colab ha permesso di lavorare in modo collaborativo e da remoto, sfruttando le potenze di calcolo delle GPU messe a disposizione dalla piattaforma. Questo ha consentito di eseguire rapidamente le computazioni necessarie senza dover preoccuparsi dell'installazione di librerie o della gestione delle risorse hardware.

3. *Adattamento del codice allo standard di Colab*: Una volta scelto l'ambiente di lavoro, sono state adattare aclune parti del codice del paper per garantire la massima compatibilità con il framework.

   È stato scelto un framework in stile "jupyter notebook" per questioni legate all'organizzazione del lavoro. Questo, infatti, ha permesso di suddividere il codice in celle e di eseguire le computazioni in modo incrementale, controllato e documentato.

   Inoltre, lavorando da remoto, l'uso del notebook ha facilitato la visualizzazione e la comprensione dei risultati ottenuti.

Seguendo questa metodologia, è stato possibile preparare l'ambiente di lavoro, adattare il codice del paper ed eseguire gli esperimenti necessari per replicare gli esiti.

Il prossimo paragrafo riguarderà le fasi di train e test, in cui verranno presentati i risultati ottenuti e le analisi qualitative e quantitative condotte durante il processo di riproduzione.

4.1 Analisi delle fasi di Train e Test

Per il dataset COCO, è stato scaricato e installato l'API COCO, mentre per il dataset OCHuman, è stato clonato il repository OCHumanApi e installato l'API OCHuman.

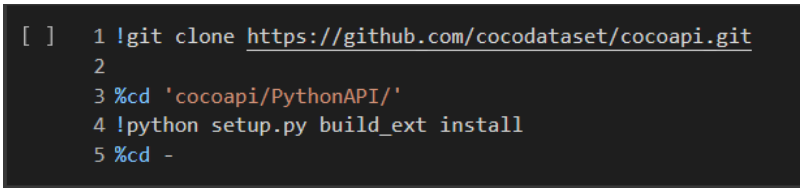

Figura 1: Esempio di adattamento del codice agli standard di Colab per l'installazione di cocoAPI

Entrambi questi passaggi sono stati essenziali per gestire correttamente le annotazioni e le immagini durante il processo di addestramento e valutazione del modello.

Dopo aver completato il download e l'installazione delle API necessarie, è stato eseguito lo script di addestramento *train.py* per allenare il modello, iterandolo per 15 epoche.

Durante questo processo, il modello ha utilizzato i dati disponibili e le funzioni fornite dalle API per apprendere i pattern e le caratteristiche necessarie per eseguire una segmentazione precisa delle istanze umane.

Al termine della fase di train, sono stati salvati i pesi del modello allenato in un file denominato *last.pkl*.

Questo file contiene i parametri ottimizzati del modello, che possono essere utilizzati in seguito per effettuare predizioni su nuovi dati o per continuare l'addestramento del modello.

Utilizzando i pesi ottenuti, è stata avviata la fase di test. Al termine di questa fase, è stato constatato che i risultati erano dello stesso ordine di grandezza di quelli presentati nel paper.

I punteggi ottenuti per *AP (Average Precision)*, *APm (Average Precision for Medium Objects)* e *APl (Average Precision for Large Objects)* sono risultati leggermente inferiori rispetto ai valori riportati nel paper.

In particolare, i punteggi originali del paper sono stati riportati come 0.582 AP, 0.539 APm e 0.679 APl, mentre i punteggi ottenuti dal test sono stati 0.564, 0.519 e 0.669 rispettivamente.

Methods	*Backbone*	*AP*	AP_M	AP_L
Mask R-CNN	Resnet50-fpn	0.532	0.433	0.648
PersonLab	Resnet101	-	0.476	0.592
PersonLab	Resnet101(ms scale)	-	0.492	0.621
PersonLab	Resnet152	-	0.483	0.595
PersonLab	Resnet152(ms scale)	-	0.497	0.621
Ours	Resnet50-fpn	**0.555**	**0.498**	**0.670**
Ours(GT Kpt)	Resnet50-fpn	**0.582**	**0.539**	**0.679**

Figura 2: Risultati originali del paper ottenuti sul COCOPersons val split [5]

```
[POSE2SEG]          AP|.5|.75| S| M| L|     AR|.5|.75| S| M| L|
[segm_score] cocoVal 0.564 0.911 0.641 -1.000 0.519 0.669 0.273 0.669 0.680 -1.000 0.629 0.750
```

Figura 3: Risultati riprodotti ottenuti sullo split test del dataset COCO

Questo lieve scarto potrebbe essere attribuito al fatto che il modello è stato allenato solamente per 15 epoche, rispetto alle 40 epoche nel paper originale. Tuttavia, i risultati sono considerati soddisfacenti e dimostrano l'efficacia del modello nella segmentazione delle istanze umane.

Data una disponibilità di maggiori risorse computazionali, sarebbe interessante verificare se i risultati ottenuti combaciano a parità di epoche, al fine di valutare ulteriormente l'efficacia del modello e la validità della metodologia di addestramento.

In conclusione, la fase di reproducibilità del paper è stata portata a termine con risultati promettenti.

È stato possibile replicare con successo il processo di addestramento del modello e i test effettuati hanno mostrato risultati coerenti e in linea con quelli presentati nel paper.

Nel complesso, i risultati indicano che la metodologia proposta può essere efficacemente riprodotta e applicata con successo.

Passando al secondo task, ci si concentrerà sull'adattamento del dataset *Ski2Dpose* e sull'analisi qualitativa del test.

In questa fase, verrà esplorato come il modello addestrato possa essere applicato a un diverso dominio di dati e sarà valutata la sua capacità di generalizzazione.

5 Adattamento e Test di Ski2dpose

Il capitolo successivo è incentrato sull'adattamento e il test del dataset *Ski2Dpose* nell'ambiente di lavoro, conformemente agli standard richiesti dal modello *Pose2Seg*.

L'obiettivo principale è illustrare le procedure per consentire l'interazione tra dataset e modelli con architetture e standard diversi.

La prima parte del capitolo si dedica all'adattamento del dataset Ski2Dpose, concentrando l'attenzione sull'integrazione nell'ambiente di lavoro e sull'aderenza agli standard specifici del modello Pose2Seg.

La seconda parte del capitolo verte sul test e sull'analisi qualitativa delle segmentazioni ottenute. Durante questa fase, saranno esplorate le caratteristiche qualitative delle segmentazioni prodotte per valutare la qualità e l'accuratezza del modello nell'identificare e segmentare le istanze umane nel dataset *Ski2Dpose*.

5.1 Adattamento

Inizialmente, sono state valutate diverse possibilità per integrare il dataset all'interno del modello. In particolare, sono state esaminate due vie principali:

1. *Modifica del codice sorgente di Pose2Seg:* In questa prima opzione, l'idea era di adattare direttamente il codice sorgente di Pose2Seg agli standard di Ski2DPose. Questo avrebbe richiesto modifiche sostanziali al codice esistente per integrare il nuovo dataset e renderlo compatibile con il modello.

2. *Modifica del dataloader fornito da Ski2DPose:* La seconda opzione prevedeva di lavorare sul dataloader fornito direttamente dal sito di Ski2DPose. In questo caso, l'obiettivo era modificare il dataloader per renderlo compatibile con gli standard di Pose2Seg. Questa via è stata preferita in quanto il dataloader era già disponibile online e pronto all'uso, riducendo così il rischio di errori e semplificando l'integrazione del nuovo dataset nel modello.

Il DataLoader SkiDataset è progettato per facilitare il caricamento e il pre-processing delle immagini e delle relative annotazioni presenti nel dataset Ski2DPose. Questo componente svolge diversi compiti chiave per garantire che i dati siano pronti per l'addestramento del modello.

Di seguito sono elencati gli obiettivi principali del DataLoader:

1. *Caricamento delle immagini e delle annotazioni:* Il DataLoader si occupa del caricamento delle immagini dal percorso specificato e delle relative annotazioni da un file JSON. Le immagini vengono lette e, se necessario, ridimensionate per adattarsi alle dimensioni desiderate. Le annotazioni vengono estratte dal file JSON e convertite in tensori PyTorch per rappresentare le posizioni dei giunti e la visibilità degli stessi.

2. *Normalizzazione delle immagini:* Se specificato, il DataLoader normalizza le immagini utilizzando la media e la deviazione standard dei canali BGR. Questi valori possono essere calcolati basandosi su tutti i dati disponibili o solo sui dati di addestramento, a seconda della modalità selezionata.

3. *Adattamento dei dati:* Il DataLoader offre la flessibilità di selezionare la modalità di caricamento dei dati, consentendo di caricare solo i dati di addestramento, solo i dati di validazione o entrambi. Questa caratteristica è utile per personalizzare l'addestramento del modello in base alle esigenze specifiche del progetto.

4. *Annotazione delle immagini:* Il DataLoader fornisce un metodo per annotare le immagini con i giunti estratti dalle annotazioni. Questo processo consente di evidenziare visivamente i giunti visibili con un cerchio rosso e quelli non visibili con un cerchio blu, facilitando l'analisi visiva dei dati.

5. *Visualizzazione delle immagini:* Infine, è possibile utilizzare il DataLoader per visualizzare le immagini annotate e consentire la navigazione tra di esse premendo i tasti della freccia sinistra e destra. Questa funzionalità permette di esplorare rapidamente il dataset e verificare la correttezza delle annotazioni.

In Pose2Seg è stato utilizzato il formato COCO (Common Objects in Context) come convenzione per rappresentare le annotazioni delle pose. COCO è un formato standard utilizzato per rappresentare dataset di segmentazione e rilevamento di oggetti. È ampiamente utilizzato nella visione artificiale e fornisce una struttura unificata per memorizzare informazioni dettagliate su immagini, oggetti, annotazioni e altro ancora.

Inizialmente, il processo di adattamento del DataLoader originale di Ski2DPose al formato COCO è stato un lavoro che ha richiesto diversi tentativi e prove. Il processo è stato condotto in modo progressivo, considerando piccoli estratti di codice alla volta, al fine di minimizzare gli errori globali e garantire una transizione fluida verso il formato COCO.

In particolare le modifiche apportate al DataLoader includono principalmente:

1. *Modifica delle annotazioni dei giunti e dei segmenti:* È stato necessario affrontare il fatto che il numero di keypoints nel dataset Ski2DPose è maggiore rispetto a quelli tipicamente presenti nel formato COCO. Pertanto, è stata condotta un'adeguata revisione dei nomi dei giunti e dei segmenti per allinearli con il formato COCO. Questo processo ha coinvolto la mappatura dei giunti originali a quelli specificati nel formato COCO e, inoltre, l'eliminazione dei giunti relativi agli sci e ai bastoni da sci, che non sono presenti nel formato COCO.

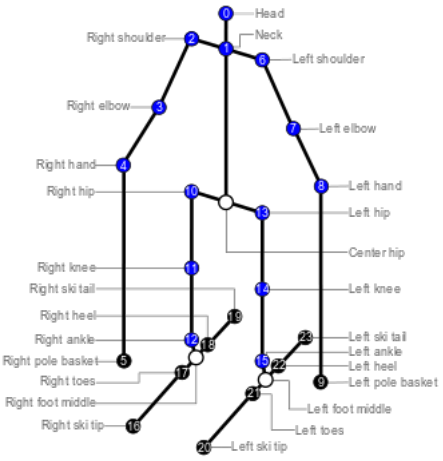

Figura 4: Diagramma dei keypoints di Ski2dPose

```python
# Create index to remove nodes 5, 9 (ski poles) and 16 to 24 (ski elements)
indice = [i for i in range(16) if i not in [5, 9]]
keypoints = an[indice]
```

Figura 5: Codice per la rimozione dei keypoints non richiesti da COCO

```python
# Loop through first 17 elements and set vis from 1 to 2 (coco requirements)
for i in range(17):
    point, v = an[i], vis[i]
    v_coco = 2 if v == 1 else 0

    if i == 5 or i == 9 or i == 16:
      keypoints_coco.extend([0, 0, 0])
    else:
      keypoints_coco.extend([point[0].item(), point[1].item(), v_coco])
```

Figura 6: Codice per l'impostazine della visibilità dei keypoints

2. *Aggiunta di informazioni sulle immagini COCO:* È stata introdotta la rappresentazione di ogni immagine nel dataset come un'entry COCO. Questa rappresentazione includeva informazioni cruciali come l'ID dell'immagine, la larghezza, l'altezza, il nome del file e la licenza.

```python
# Process each image and annotation in the dataset
for img, an, vis, bbox, segmx, info in ski_dataset:
    # Define COCO image entry
    coco_img_entry = {
        "id": image_id_counter,
        "width": img.shape[1],
        "height": img.shape[0],
        "file_name": f"{info[2]}.{img_extension}",
        "license": 1,
    }
```

Figura 7: Codice per le image entry di COCO

3. *Aggiunta di annotazioni COCO:* È stata una parte significativa del processo, in cui sono state generate le annotazioni COCO corrispondenti per ciascuna immagine. Queste annotazioni comprendevano le coordinate dei giunti (con visibilità modificata secondo le specifiche COCO), la bounding box e la segmentazione della persona.

```
# Define COCO annotation entry
coco_annotation_entry = {
    "id": image_id_counter,
    "image_id": image_id_counter,
    "category_id": 1,
    "keypoints": keypoints_coco,
    "num_keypoints": int(vis.sum()),
    "area": (bbox[2]-bbox[0]) * (bbox[3]-bbox[1]),
    "bbox": bbox,
    "iscrowd": 0,
    "segmentation": segmx
}
```

Figura 8: Codice per le annotation entry di COCO

```
coco_annotations["images"].append(coco_img_entry)
coco_annotations["annotations"].append(coco_annotation_entry)
```

Figura 9: Estratto di codice per l'aggiunta delle annotazioni per ciascuna immagine

4. *Salvataggio delle annotazioni COCO:* È stata implementata una funzionalità per salvare le annotazioni COCO in un file JSON separato, nel formato richiesto COCO.

```
# Save COCO annotations to a JSON file
coco_output_path = '/content/Pose2Seg/data/Ski2DPose/annotations/ski2dpose_labels_coco.json'
with open(coco_output_path, 'w') as coco_file:
    json.dump(coco_annotations, coco_file)
```

Figura 10: Le annotazioni vengono salvate in un file JSON

Figura 11: Esempio di frame con bbox (rosso), scheletro (verde) e keypoints (rossi: visibili e blu: non visibili) basati sulle annotazioni convertite in formato COCO

5.2 Test

La sezione dedicata alla fase di test si suddivide in tre parti principali:

1. *Generazione delle maschere di segmentazione di ground truth:* In questa prima parte viene illustrato il processo di creazione delle maschere di segmentazione di ground truth richieste dal modello.

   Grazie all'utilizzo della libreria Polygon, le coordinate dei keypoints delle pose sono state convertite in poligoni. Questo è stato necessario poiché il dataset Ski2DPose originale non includeva tali maschere mentre il modello di Pose2Seg richiede esplicitamente questa forma di rappresentazione dei dati.

2. *Modifica e integrazione dello script test.py*: La seconda sezione è dedicata all'illustrazione delle modifiche apportate allo script *test.py* fornito dal codice originale di Pose2Seg.

   Le modifiche sono state effettuate per integrare correttamente le annotazioni e le relative immagini necessarie per condurre la fase di test. Questo adattamento è stato essenziale per garantire che il modello elabori correttamente i dati e predica la segmentazioni.

3. *Analisi qualitativa delle segmentazioni predette*: La terza parte si concentra sull'analisi qualitativa delle segmentazioni predette dal modello.

   Durante questa fase, le segmentazioni generate dal modello vengono confrontate visivamente con le immagini originali del dataset. Questo confronto non utilizza metriche quantitative, ma si basa sull'osservazione diretta per valutare l'accuratezza e la coerenza delle segmentazioni rispetto alle pose effettive presenti nelle immagini.

   Tale analisi fornisce un'indicazione intuitiva delle prestazioni complessive del modello e identifica eventuali punti di forza e di debolezza.

5.2.1 Generazione delle maschere di segmentazione di ground truth

Durante il processo di adattamento del DataLoader originale di Ski2DPose al formato COCO, è emersa la necessità di generare maschere di ground truth basate sulle annotazioni dei keypoints disponibili nel dataset.

Questo è stato realizzato attraverso l'uso della libreria *Polygon*, che offre funzionalità per la manipolazione e la creazione di poligoni.

Le coordinate dei keypoints delle pose estratte dalle annotazioni sono state utilizzate per creare i poligoni corrispondenti, rappresentativi delle persone nelle immagini.

Questi poligoni sono stati quindi utilizzati per generare le maschere di ground truth, che forniscono informazioni dettagliate sulla posizione e sulla forma delle persone presenti nelle immagini.

Il codice aggiunto nel DataLoader per compiere queste operazioni consiste nella creazione di poligoni sulla base delle coordinate dei keypoints delle pose, e nella conversione di questo poligono in una lista di coordinate per la segmentazione.

Questa lista viene quindi utilizzata per generare le maschere di ground truth necessarie per la fase di test del modello Pose2Seg.

```
# Create polygon object from these coordinates
polygon = Polygon(keypoint_coords)
polygon_points = list(polygon.exterior.coords)

# List of coordinates
segmentation = [float(coord) for point in polygon_points for coord in point]
segmx = [segmentation]
```

Figura 12: Estratto del codice relativo alla generzione di poligoni basati sui keypoints

5.2.2 Modifica e integrazione dello script *test.py*

Lo script *test.py* è un componente fondamentale del modello Pose2Seg.

In questa sezione, esploreremo le principali funzioni di questo script e successivamente, illustreremo le modifiche apportate allo script per consentire l'esecuzione del test anche sul dataset Ski2dpose.

Di seguito le funzioni principali:

1. *Importazione dei moduli necessari:* Il codice inizia importando i moduli necessari, inclusi argparse, numpy, tqdm, e alcune funzioni personalizzate da altri file.

2. *Definizione della funzione di test:* La funzione *test* prende come argomenti il modello da testare, il nome del dataset su cui testare e un logger per la registrazione dei messaggi.

   Inizia leggendo le immagini e le annotazioni dal dataset specificato. Utilizza la classe *CocoDatasetInfo* per caricare le informazioni sul dataset COCO o OCHuman, incluso il percorso delle immagini e i file di annotazione. Successivamente, itera su ogni immagine del dataset, eseguendo l'inferenza del modello sulle immagini e generando le maschere di segmentazione previste. Queste maschere vengono quindi convertite in un formato compatibile con le API *pycocotools.mask*.

   Infine, i risultati vengono salvati in una struttura dati per l'elaborazione successiva.

3. *Definizione della funzione di valutazione COCO:* La funzione *do_eval_coco* è utilizzata per valutare le prestazioni del modello utilizzando le metriche COCO, come AP (average precision) e AR (average recall). Viene quindi utilizzata la classe *COCOeval* dalle API *pycocotools* per calcolare queste metriche.

4. *Parsing degli argomenti della riga di comando:* Viene utilizzato il modulo *argparse* per analizzare gli argomenti passati dalla riga di comando quando si esegue lo script.

   Gli argomenti possono includere il percorso del file di pesi del modello (–weights), nonché flag opzionali per specificare se eseguire il test sul set di dati COCO (–coco) o sui set di dati OCHuman (–OCHuman).

5. *Caricamento del modello e avvio del test:* Dopo il parsing degli argomenti, il codice carica il modello Pose2Seg, inizializza i pesi del modello utilizzando il percorso specificato e avvia il test sui set di dati COCO e/o OCHuman a seconda dei flag specificati.

Di seguito le modifiche apportate allo script *test.py* per eseguire il test sul Dataset *Ski2DPose*:

1. *Aggiunta del dataset Ski2DPose:* È stata aggiunta una nuova condizione elif dataset == 'Ski2DPose' nel blocco test per gestire il nuovo dataset. Questa condizione imposta le variabili ImageRoot e AnnoFile per puntare alle directory corrette contenenti le immagini e i file di annotazione del Dataset Ski2DPose.

```
    elif dataset == 'Ski2DPose':
        ImageRoot = './data/Ski2DPose/images'
        AnnoFile = './data/Ski2DPose/annotations/ski2dpose_labels_coco.json'
```

Figura 13: Codice relativo alla nuova condizione per la gestione del nuovo dataset

2. *Visualizzazione delle immagini di esempio:* È stata aggiunta una parte del codice per visualizzare le immagini di esempio, i punti chiave e le maschere predette durante il ciclo di test. Questo è stato fatto utilizzando la libreria matplotlib per creare un subplot con tre immagini: la maschera ground truth, l'immagine originale con i punti chiave e la maschera predetta. Questa visualizzazione aiuta a comprendere visivamente le prestazioni del modello.

```
# View images, keypoints and pred mask
if i in random_indices:
    fig, axs = plt.subplots(1, 3, figsize=(20, 20))
    axs[0].imshow(gt_masks[0])  # assuming gt_masks is a list of masks
    axs[0].set_title('Ground Truth Mask')

    # Draw keypoints
    for kpt in gt_kpts[0]:
        x, y, _ = kpt
        cv2.circle(img, (int(x), int(y)), 5, (0, 255, 0), -1)  # Cerchio verde
    axs[1].imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
    axs[1].set_title('Original Image w/ Keypoints')

    axs[2].imshow(pred_masks[0])
    axs[2].set_title('Predicted Mask')
    plt.show()
```

Figura 14: Codice per la visualizzazione delle immagini.

3. *Stampa dei messaggi informativi:* È stata aggiunta la stampa di messaggi informativi per indicare quale dataset viene testato.

4. *Chiamata alla funzione di test:* Infine, sono state aggiunte le chiamate alla funzione test per eseguire il test del modello sui dataset selezionati.

```
print('===========> LOADING MODEL <===========')
model = Pose2Seg().cuda()
model.init(weights)

print('===========>   TESTING    <===========')
if Ski2DPose:
    print('-----------------------------------')
    print('TEST SKI2DPOSE')
    test(model, dataset='Ski2DPose')
if coco:
    print('-----------------------------------')
    print('TEST COCO VAL')
    test(model, dataset='cocoVal')
if OCHuman:
    print('-----------------------------------')
    print('TEST OCHUMAN VAL')
    test(model, dataset='OCHumanVal')
    print('-----------------------------------')
    print('TEST OCHUMAN TEST')
    test(model, dataset='OCHumanTest')
```

Figura 15: Codice relativo all'esecuzione del test sul Dataset.

5.2.3 Analisi qualitativa delle segmentazioni predette

Grazie alle maschere di ground truth ottenute dalla conversione delle coordinate dei keypoints in poligoni e alle modifiche apportate allo script *test.py*, è stato possibile avviare la fase di test e quindi condurre un'analisi qualitativa delle segmentazioni predette sul dataset Ski2dpose.

Le immagini di esempio seguenti illustrano le potenzialità del modello nell'identificare e segmentare con precisione le persone nelle immagini del dataset Ski2dpose.

La visualizzazione delle maschere predette rispetto alle maschere di ground truth evidenzia la capacità del modello di catturare dettagli importanti e di produrre segmentazioni accurate.

Ovviamente, è emerso che alcune predizioni non sono risultate molto accurate; tuttavia, esse hanno comunque consentito l'individuazione della sagoma umana all'interno delle immagini del dataset.

Di seguito alcune immagini d'esempio in cui la predizione è risultata accurata:

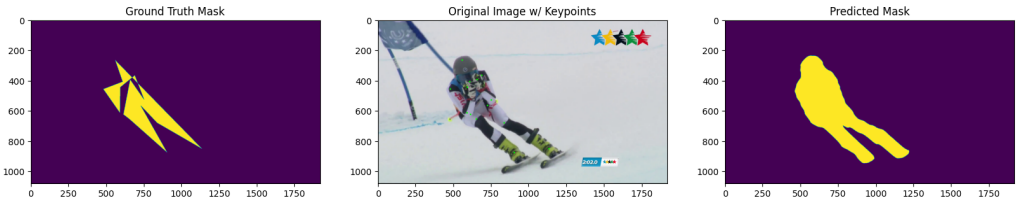

Figura 16: Esempio n° 1 di output del test; (sx) maschera di gorund truth creata con *Polygon*, (ct) immagine originale, (dx) maschera di segmentazione predetta

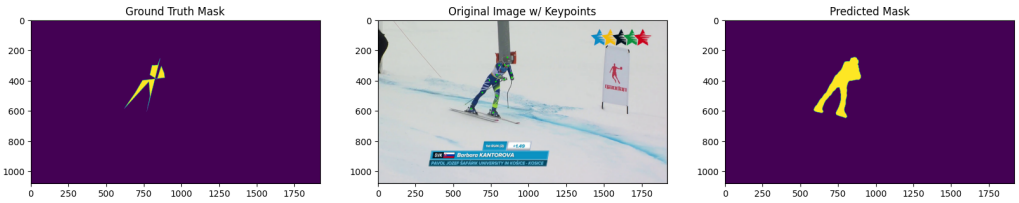

Figura 17: Esempio n° 2

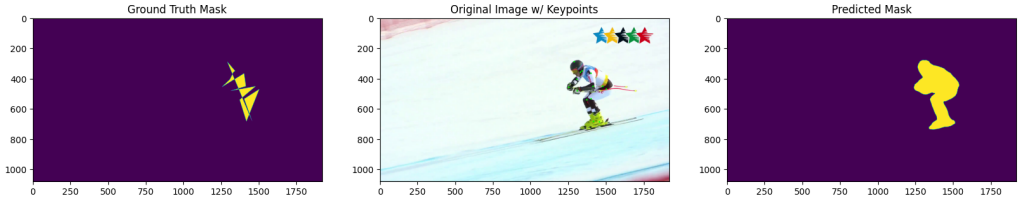

Figura 18: Esempio n° 3

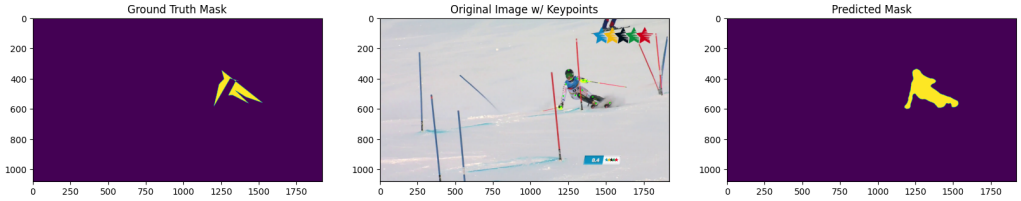

Figura 19: Esempio n° 4

A seguire alcuni esempi in cui, nonostante la predizione non sia avveuta in modo accurato, risulta ancora possibile distinguere le figure degli sciatori:

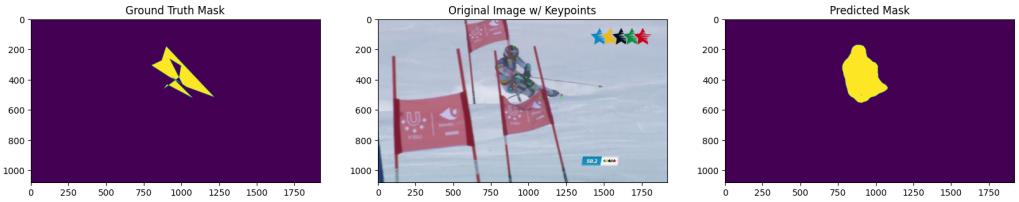

Figura 20: Esempio n° 5

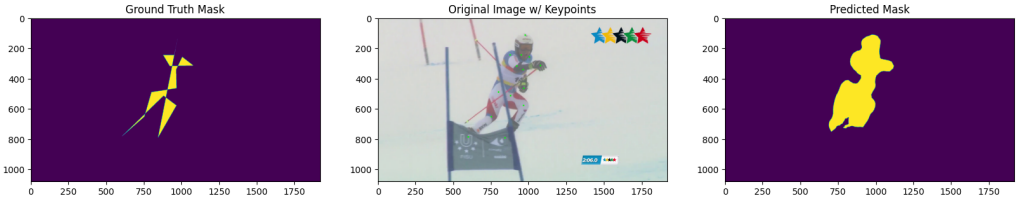

Figura 21: Esempio n° 6

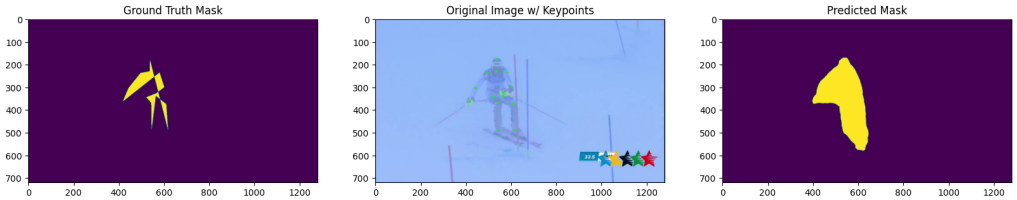

Figura 22: Esempio n° 7

6 Future Implementazioni

Per migliorare ulteriormente il progetto, ci sono diverse future implementazioni che potrebbero essere considerate.

Poichè il modello è stato addestrato con una GPU Nvidia T4, una delle opzioni potrebbe essere l'utilizzo di altre GPU per confrontare i risultati del test ottenuti a seguito della fase di training del modello.

Inoltre, aumentare il numero di epoche dell'allenamento potrebbe portare ad un miglioramento delle prestazioni del modello consentendo l'apprendimento di caratteristiche più complesse dei dati e migliorarando le capacità di generalizzazione.

Un'altra possibile implementazione potrebbe riguardare l'uso di tool online per la segmentazione di oggetti, al fine di confrontare gli output predetti dal modello con segmentazioni generate da altri metodi.

Questo confronto potrebbe fornire una valutazione più completa delle prestazioni del modello e identificare eventuali aree in cui il modello potrebbe essere migliorato.

7 Conclusioni

Questo progetto è stato essenziale per acquisire competenze nell'affrontare problemi legati all'integrazione di dataset con standard diversi tra loro.

Tale compito ha richiesto competenze sia di tipo teorico che pratico. Durante lo svolgimento, sono stati incontrati diversi ostacoli, legati a competenze informatiche e alla discontinuità dei lavori causata principalmente dalla sovrapposizione con sessioni d'esame.

Tuttavia, il superamento delle difficoltà sopracitate è stato motivo di soddisfazione, consapevoli che il lavoro svolto potrà costituire un punto di partenza per ulteriori implementazioni.

Questo lavoro è stato importante poiché ha fornito l'opportunità di interfacciarsi con lo studio di paper scientifici e di acquisire familiarità con le metodologie di ricerca nel campo della computer vision e del machine learning.

Inoltre, l'esperienza acquisita nel gestire problemi come l'integrazione di dataset con standard diversi e l'implementazione di modelli di segmentazione di oggetti è rilevante, considerando che tali task sono all'ordine del giorno sia in ambito accademico che professionale.

Le competenze sviluppate durante questo progetto saranno sicuramente utili nel futuro professionale, consentendo di affrontare sfide simili in contesti lavorativi e di ricerca.

In questo progetto, il lavoro di squadra ha giocato un ruolo fondamentale nel raggiungimento dei risultati. Grazie alla collaborazione e alla condivisione delle competenze tra i membri del team, siamo stati in grado di affrontare sfide e superare ostacoli in modo efficace.

Ognuno ha contribuito con le proprie competenze e prospettive, creando un ambiente collaborativo e stimolante in cui condividere idee, risolvere problemi e prendere decisioni.

La comunicazione aperta e la capacità di lavorare insieme sono state fondamentali per il progetto, nonostante le sfide e i ritardi incontrati lungo il percorso.

Grazie a questo lavoro è stato possibile riprodurre i risultati proposti dal paper *"Pose2Seg: Detection Free Human Human Instance Segmentation"* e integrare il dataset *Ski2DPose* per condurre un'analisi qualitativa completa.

Grazie all'implementazione del modello Pose2Seg e all'utilizzo dei dataset disponibili, è stato possibile ottenere risultati complessivamente positivi.

Sebbene ci sia ancora ampio margine di miglioramento, consideriamo il risultato ottenuto soddisfacente, in quanto è stato possibile ottenere segmentazioni di persone con un buon livello di precisione.

Questa esperienza ha fornito una solida base per affrontare sfide future nel campo della computer vision ed ha rappresentato un arricchimento a livello personale.

Riferimenti bibliografici

[1] Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[2] Bin Liu, Wencang Zhao, and Qiaoqiao Sun. Study of object detection based on faster r-cnn. In *2017 Chinese Automation Congress (CAC)*, pages 6233–6236, 2017.

[3] Shujon Naha, Qingyang Xiao, Prianka Banik, Md Alimoor Reza, and David J. Crandall. Pose-guided knowledge transfer for object part segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2020.

[4] Bo Wan, Desen Zhou, Yongfei Liu, Rongjie Li, and Xuming He. Pose-aware multi-level feature network for human object interaction detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[5] Song-Hai Zhang, Ruilong Li, Xin Dong, Paul L. Rosin, Zixi Cai, Han Xi, Dingcheng Yang, Hao-Zhi Huang, and Shi-Min Hu. Pose2seg: Detection free human instance segmentation. *CVPR*, pages 1–8, 2019.

Sitografia

1. Pose2seg code on GitHub 1: `https://github.com/liruilong940607/Pose2Seg`
2. Ski2Dpose Dataset: `https://www.epfl.ch/labs/cvlab/data/ski-2dpose-dataset/`