Human Activity Recognition per la classificazione di balli caraibici tramite Pose Estimation

---

Candidato: Alberto Tontoni

Nome del corso: Machine Learning

Anno accademico: 2023-2024

Il problema

- Realizzare un classificatore binario di Time Series
- Dato un video di ballo, riconoscere tra due stili: salsa e bachata
- Idealmente, ogni ballerino è una Time Series
- I due stili presentano differenze e sovrapposizioni
- Tre principali tipologie di video disponibili sul web: demo, social, solo

Demo

(*Bachata)*

Social

(*Salsa)*

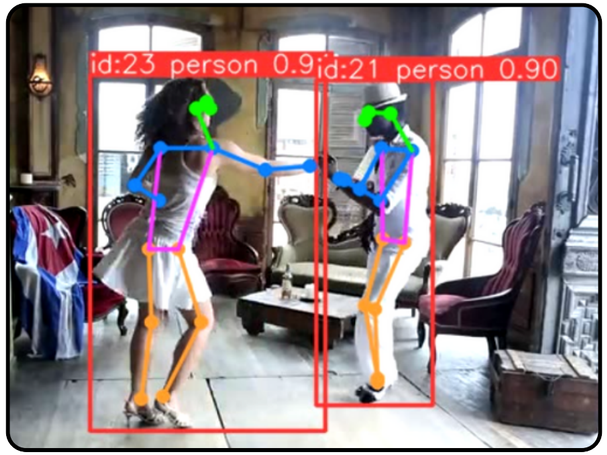
Solo

(*Salsa)*

Pose Estimation

Ogni video viene processato da YOLOv8, una rete neurale preaddestrata per le attività di object detection, pose estimation e tracking:

- **Object Detection**: riconoscere una persona in un'immagine
- **Pose Estimation**: riconoscere i movimenti di una persona
- **Tracking**: riconoscere la stessa persona in frames diversi di uno stesso video

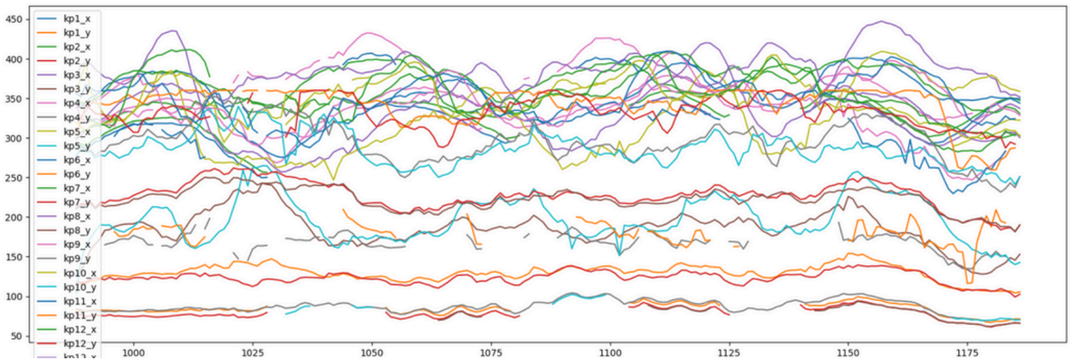


Un frame di video processato da yolov8-pose-p6

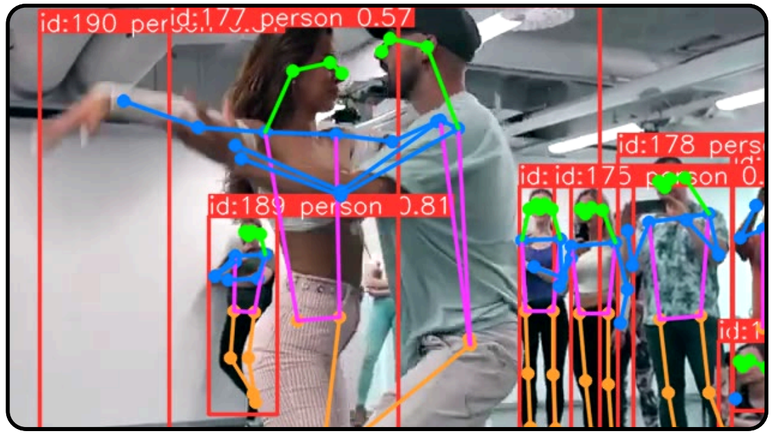
Data Cleaning

Sebbene YOLOv8 abbia delle potenzialità enormi, presenta anche delle limitazioni per il nostro task:

- Scambi frequenti di posizione tra i ballerini: problemi di tracciamento
- Parti del corpo non perfettamente visibili: missing values
- Presenza di spettatori: rumore di fondo



Time Series di un ballerino di salsa: sono riportati i punti (x,y) di 17 punti chiave. E' evidente la presenza di valori mancanti.

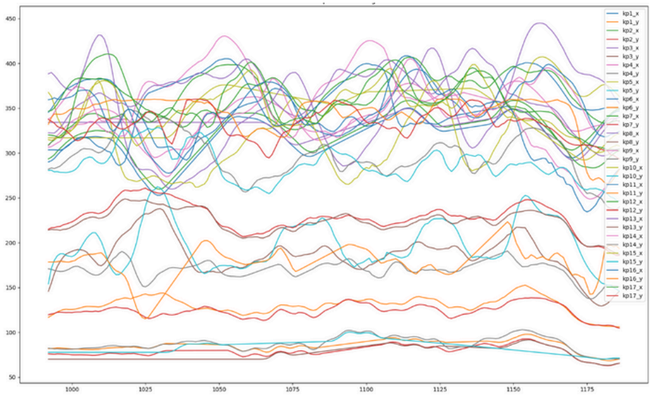


Video "demo" di bachata in cui, oltre ai due ballerini in primo piano, sono presenti numerosi spettatori.

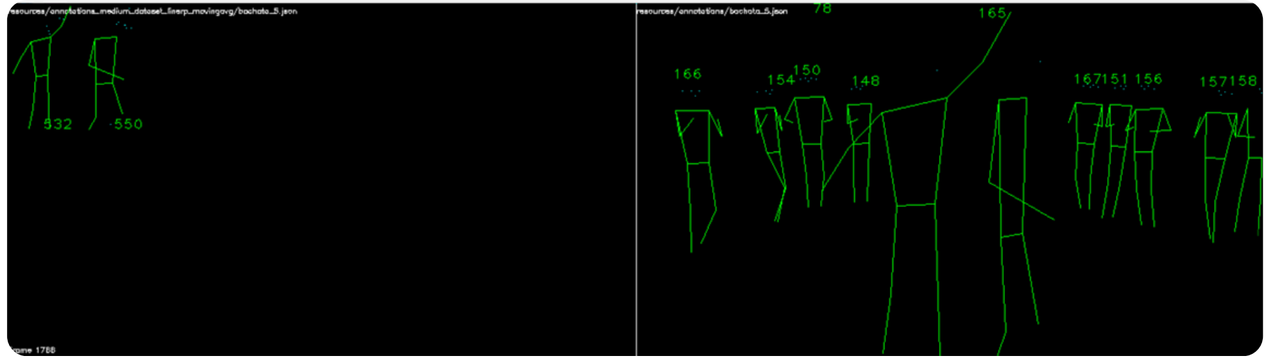
Data Cleaning

Tra le tecniche principali per la pulizia dei dati:

- Interpolazione lineare dei missing values
- Filtro moving average sugli ultimi 5 frames
- Eliminazione di sequenze troppo corte (<30 frames)
- Eliminazione di sequenze con *variazione media totale* troppo bassa (<0.2)



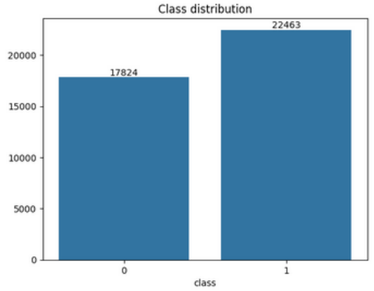
Time Series di prima, interpolata linearmente e filtrata.



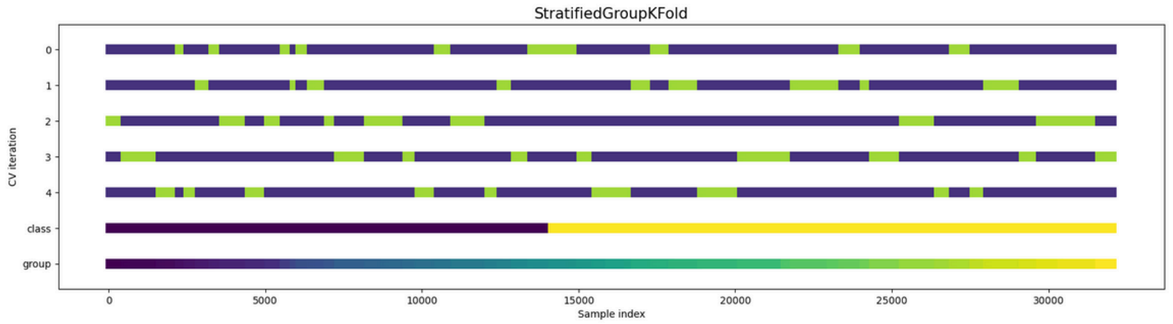
A destra, le pose estratte da YOLOv8. A sinistra, le pose rimanenti dopo il filtraggio.

Il Dataset

- Sono stati collezionati 66 video - 50% salsa, 50% bachata
- Si applicano tecniche di feature engineering tipiche delle Time Series:
  - Sliding window di 30 frames, overlap 50% e step size pari a 2
  - Velocità e accelerazioni derivate dai punti chiave stimati da YOLOv8
  - Derivazione di angoli tra punti chiave, distanze mutue tra punti
- Train/Test Split - 80% Training, 20% Test
- Cross-Validation a 5 fold preservando il bilanciamento tra classi e tra i “gruppi” (ie. osservazioni di video diversi)



Distribuzione tra classi dell’intero dataset

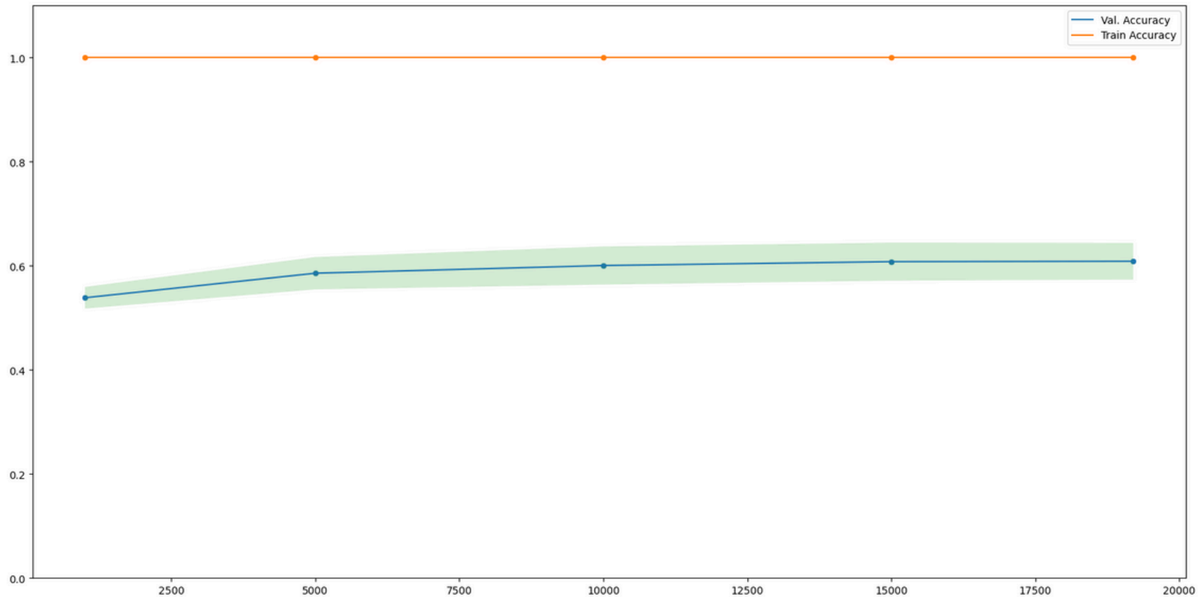


Rappresentazione della 5-Fold CV sulla base degli indici del training set: in verde i campioni del validation set

Curva di apprendimento

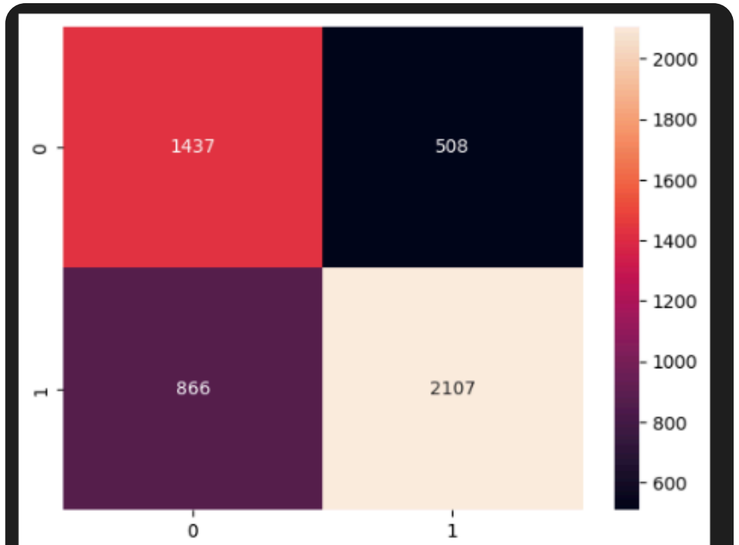
- Ad ogni fold, Random Undersampling per preservare il bilanciamento fra classi
- PCA per ridurre la varianza tra le diverse fold (exp.variance 99.9%)
- Addestramento di una Random Forest su training set di dimensioni crescenti:
  - Minimo: 1000 osservazioni
  - Massimo: 19200 osservazioni

Curve di apprendimento in termini di f1-score macro average (erroneamente indicata come Accuracy).
Con 1000 datapoint siamo intorno al 53%, mentre con 19200 datapoint siamo intorno al 59.5%.



Conclusioni

- I dati effettivamente utilizzati per il training sono circa la metà di quelli effettivamente disponibili - ciò è causato dal Train/Test Split, dalla 5-Fold CV e soprattutto dal Random Undersampling
- Le prestazioni migliorano se si salta la CV: in questo caso si ha macro f1 >70% per: Random Forest, Histogram Gradient Boosting, SVM RBF
- Le prestazioni sembrano migliorare se si utilizzano features high-level insieme a quelle raw stimate da YOLOv8
- Altri approcci tentati (SMOTE, UMAP, t-SNE) non hanno migliorato la situazione
- In tre parole: servono più dati



	precision	recall	f1-score	support
0	0.62	0.74	0.68	1945
1	0.81	0.71	0.75	2973
accuracy			0.72	4918
macro avg	0.71	0.72	0.72	4918
weighted avg	0.73	0.72	0.72	4918

Confusion Matrix e Classification Report per SVM RBF addestrato sull'intero training set, calcolando le features di alto livello ma non le statistiche per singole finestre. Come possiamo vedere, con più dati si hanno prestazioni migliori.