Università degli Studi di Perugia
Dipartimento di Matematica e Informatica

A.D. 1308 unipg
DIPARTIMENTO
DI MATEMATICA E INFORMATICA

Tesi triennale in Informatica

Sviluppo di un sistema di rilevamento in tempo reale di azioni violente tramite YOLO e LSTM

Relatore
Prof. Francesco Santini

Laureando
Filippo Notari

Anno Accademico 2024-2025

Indice

Capitolo 1

Introduzione

L'obiettivo di questa tesi è lo sviluppo di un sistema di rilevamento e riconoscimento automatico di azioni violente e la rilevazione di armi in tempo reale.

Oggi viviamo in contesti urbani e sociali dove la videosorveglianza è ormai onnipresente. Strade, piazze, scuole e luoghi di aggregazione producono costantemente una mole di dati visivi talmente vasta da risultare, di fatto, ingestibile per un operatore umano. È proprio qui che emerge il limite dei sistemi tradizionali: la capacità di archiviazione ha superato di gran lunga la nostra capacità di monitoraggio effettivo. Non si tratta più solo di registrare ciò che accade, ma di riuscire a interpretarlo mentre avviene.

L'idea è quindi quella di fornire un supporto concreto alla vigilanza attraverso l'automazione, permettendo di individuare tempestivamente comportamenti anomali o situazioni di potenziale pericolo che potrebbero sfuggire a un occhio stanco o distratto. Un approccio di questo tipo non ha solo una valenza tecnologica, ma anche un impatto diretto sulla sicurezza pubblica e sulla prevenzione della criminalità, migliorando la tutela delle persone nei contesti più disparati.

Tuttavia, passare dalla teoria alla pratica nel campo della computer vision comporta difficoltà notevoli. Analizzare un flusso video in uno scenario reale non è paragonabile al lavoro su dataset controllati: ci si scontra con variazioni di luce improvvise, angolazioni di ripresa spesso sfavorevoli e una densità di persone che rende la scena caotica. In molti casi, i soggetti coinvolti sono parzialmente coperti o ripresi da distanze tali da rendere i dettagli molto sfumati. Sono proprio queste complessità a

rendere insufficienti i vecchi approcci basati su regole rigide o sull'analisi statica dei singoli frame, spingendo la ricerca verso soluzioni più dinamiche e robuste, capaci di interpretare correttamente l'azione nel suo insieme.

Un'altra sfida è distinguere comportamenti violenti da azioni quotidiane che possono apparire simili. Movimenti rapidi, gesti improvvisi o interazioni fisiche non aggressive come giochi, abbracci o attività sportive possono essere erroneamente classificati come violenti. Per questo, è fondamentale un sistema in grado di analizzare il contesto temporale di un'azione, valutando la dinamica dei movimenti e la loro evoluzione nel tempo, riducendo così i falsi positivi e migliorando l'accuratezza del rilevamento.

Dal punto di vista applicativo, un sistema di riconoscimento di azioni violente deve anche essere efficiente e operare in tempo reale. In scenari di sorveglianza reale, i dati devono essere elaborati con latenza minima, permettendo interventi tempestivi da parte degli operatori o delle forze di sicurezza. Questo richiede modelli robusti ma ottimizzati, capaci di gestire flussi video continui senza compromettere le prestazioni complessive del sistema.

Considerando tutto ciò, diventa chiaro che rilevare automaticamente comportamenti violenti e la presenza di armi in contesti urbani è un compito complesso, che richiede di combinare diverse competenze, dalla visione artificiale all'analisi dei movimenti nel tempo.

Il nostro lavoro propone un metodo che combina l'analisi di cosa accade nello spazio e di come le azioni evolvono nel tempo in un video.

In particolare, il sistema utilizza modelli di object detection (YOLO) per individuare armi specifiche in questo caso, coltelli e algoritmi di pose estimation (YOLO-Pose) per rilevare i soggetti presenti nella scena e i relativi keypoint corporei. Le informazioni raccolte vengono poi elaborate da una rete neurale ricorrente di tipo Bi-LSTM, progettata per catturare le dipendenze temporali e distinguere movimenti quotidiani da vere e proprie aggressioni fisiche.

Il lavoro si articola in cinque capitoli, organizzati come segue: il **Capitolo 2** descriverà le tecnologie e i modelli selezionati per il progetto. Verranno approfonditi il funzionamento dell'algoritmo YOLO per l'Object Detection, la Pose Estimation e la struttura delle reti LSTM per l'analisi di sequenze temporali.

Nel **Capitolo 3** si presenterà l'implementazione pratica del sistema. Verrà detta-

gliata la creazione dei dataset e dei modelli che verranno usati. Poi verrà descritta l'architettura della pipeline di elaborazione, descrivendo l'integrazione tra la parte di rilevamento e il classificatore temporale.

Il **Capitolo 4** sarà dedicato alle metriche e alla validazione del sistema. Verranno illustrate le metriche di valutazione adottate e si discuteranno i risultati ottenuti.

Infine il **Capitolo 5** trarrà le conclusioni del lavoro svolto, riassumendo i traguardi raggiunti e delineando possibili sviluppi futuri per migliorare ulteriormente la robustezza e l'efficienza del sistema in scenari reali.

Capitolo 2

Background

2.1 Human Activity Recognition

La Human Activity Recognition (HAR) è un ambito di ricerca fondamentale nella Computer Vision e nell'Intelligenza Artificiale, con applicazioni critiche nella videosorveglianza, nell'assistenza sanitaria e nell'interazione uomo-macchina [1]. L'obiettivo principale di questi sistemi è sviluppare algoritmi in grado di interpretare automaticamente azioni, movimenti e gesti umani a partire da dati grezzi (come sequenze video o segnali di sensori), permettendo alle macchine di comprendere il comportamento umano in contesti complessi [2].

Le attività umane possono essere categorizzate in quattro livelli gerarchici di complessità:

Azioni individuali (Atomic Actions). Si riferiscono a movimenti elementari o posturali di un singolo soggetto. Rappresentano le azioni fondamentali del movimento, dove lo stato del corpo cambia da statico a dinamico.

Human-Human Interaction (HHI). Comprende interazioni che coinvolgono due o più persone. Questa categoria è significativamente più complessa poiché richiede non solo il rilevamento dei singoli soggetti, ma anche l'analisi della relazione spazio-temporale tra di essi.

Human-Object Interaction (HOI). Riguarda le interazioni tra un essere umano e un oggetto specifico per raggiungere uno scopo. Questo è cruciale per discriminare la natura di un'azione in base al contesto strumentale.

Attività di gruppo. Rappresenta la complessità maggiore, implicando l'analisi simultanea di più soggetti che agiscono, cooperano o competono in uno scenario comune.

Dal punto di vista dell'acquisizione dei dati, l'HAR può essere classificata in due approcci metodologici principali:

Basata su Sensori Indossabili (Wearable). Utilizza dispositivi indossabili applicati direttamente sul corpo per acquisire segnali inerziali relativi al movimento. Sebbene precisi, sono spesso invasivi e poco pratici [3].

Basata su Sensori Non Indossabili (Non-wearable). Si basa su sensori ambientali o dispositivi fissi, come radar, Wi-Fi, sensori di pressione o infrarossi, che permettono di riconoscere le attività senza il contatto diretto con l'utente. Tali soluzioni risultano meno invasive rispetto ai wearable, ma possono soffrire di limitazioni in termini di risoluzione spaziale e robustezza al rumore [4].

Basata su Visione (Vision-based). Utilizza sensori visivi esterni, come telecamere RGB, di profondità (RGB-D) o termiche. Questo approccio è non invasivo e ideale per la sicurezza. All'interno di questa categoria, si distingue ulteriormente tra metodi basati sui pixel (che analizzano l'intero frame) e metodi *Skeleton-based* (che analizzano solo la struttura scheletrica) [5].

2.2 YOLO

YOLO (You Only Look Once)[1] rappresenta una famiglia di architetture per il rilevamento di oggetti in tempo reale basata su reti neurali convoluzionali (CNN). A differenza dei metodi tradizionali a due stadi (come R-CNN), che propongono regioni di interesse e successivamente le classificano, YOLO adotta un approccio unificato:

[1]Sito di YOLO https://docs.ultralytics.com/.

una singola rete neurale prevede simultaneamente le *bounding box* e le probabilità di classe direttamente dall'immagine intera [6].

Questa architettura elabora l'intera immagine sia durante la fase di addestramento che durante l'inferenza. Ciò consente al modello di codificare implicitamente le informazioni contestuali relative alle classi e alla loro disposizione spaziale. Grazie a questa visione globale, YOLO riduce significativamente gli errori di rilevamento sullo sfondo (falsi positivi) rispetto ad altri metodi che analizzano solo porzioni locali (patch) dell'immagine e che, non avendo accesso al contesto più ampio, tendono a confondere elementi di sfondo con oggetti di interesse [7].

2.2.1 YOLO Object Detection

YOLO Object Detection[2] rappresenta l'architettura di base di YOLO per affrontare il compito della *Object Detection* e fa parte delle one-stage detectors, ovvero reti che utilizzano una singola rete neurale per effettuare direttamente la previsione. L'obiettivo è l'identificazione simultanea di oggetti presenti in scena e la loro classificazione. L'output di questa attività è un insieme di *bounding box* che racchiudono gli oggetti nella scena, insieme ad etichette di classe e punteggi di confidenza per ogni box.

Il sistema divide l'immagine di input in una griglia $S \times S$. Se il centro di un oggetto ricade dentro una cella della griglia, quella cella sarà responsabile per il rilevamento di quell'oggetto. Ogni cella prevede B bounding box e i relativi punteggi di confidenza; inoltre, prevede le probabilità di classe condizionale C. Queste probabilità sono condizionate sulla cella della griglia che contiene un oggetto [7]. Come illustrato nella Figura 2.1, il processo elaborativo parte dalla suddivisione in griglia dell'input. Successivamente, il sistema genera in parallelo le regressioni delle bounding box e le mappe di probabilità per le classi. L'output finale è ottenuto dalla combinazione di queste componenti, restituendo le bounding box con la classificazione dell'oggetto identificato.

[2]Sito di YOLO Object Detection https://docs.ultralytics.com/tasks/detect/.

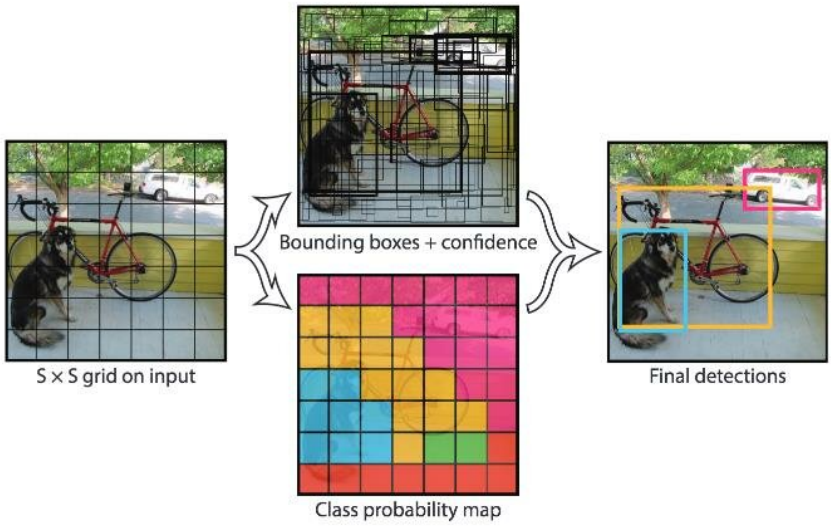

Figura 2.1: Il sistema di YOLO [7].

2.2.2 YOLO Pose

YOLO Pose[3] estende l'architettura di base di YOLO per affrontare il compito della *Multi-Person Pose Estimation*. L'obiettivo è l'identificazione simultanea di persone presenti in scena e la localizzazione di specifici punti di riferimento, definiti *keypoint*.

Il modello adotta lo standard del dataset COCO [8], che prevede 17 keypoint in punti strategici, come articolazioni o parti del corpo. A differenza degli approcci tradizionali basati su *Heatmaps* (mappe di calore), che stimano la probabilità di presenza di un giunto per ogni pixel dell'immagine e richiedono un elevato costo computazionale, YOLO utilizza un approccio basato sulla *regressione diretta*. Le coordinate dei keypoint vengono predette direttamente insieme alle bounding box, garantendo prestazioni migliori [9].

L'output generato per ogni persona rilevata è un vettore che unisce le informazioni di localizzazione della persona (Bounding Box) alle informazioni posturali. Ogni keypoint è descritto da una tripla *(x,y,conf)*, dove x,y sono le coordinate normalizzate e conf è il punteggio di visibilità/confidenza del punto. Il vettore risultante, composto

[3]Sito di YOLO Posehttps://docs.ultralytics.com/it/tasks/pose/.

da 6 parametri di rilevamento oggetto e 51 parametri di posa, è formalizzato come segue:

$$P_v = \{\underbrace{C_x, C_y, W, H, \text{box}_{\text{conf}}, \text{class}_{\text{conf}}}_{\text{Object Detection}}, \underbrace{K_x^1, K_y^1, K_{\text{conf}}^1, \ldots, K_x^{17}, K_y^{17}, K_{\text{conf}}^{17}}_{\text{Pose Estimation}}\} \tag{2.1}$$

Uno dei vantaggi di YOLO-Pose rispetto ad altri approcci è che non vi è alcun vincolo per i punti chiave che devono essere all'interno della bounding box, come mostrato in Figura 2.2. Pertanto, se i punti chiave si trovano all'esterno della bounding box a causa di qualche occlusione, possono comunque essere riconosciuti correttamente. Mentre, negli altri approcci, se il rilevamento della persona non è corretto, anche la stima della posa fallirà.

Figura 2.2: Esempio di YOLO Pose con keypoint al di fuori della bounding box [9].

2.3 Multi-Object Tracking (MOT)

Il Multi-Object Tracking (MOT) è un problema fondamentale nella computer vision che consiste nel localizzare molteplici oggetti di interesse e mantenere la loro identità (ID) coerente lungo tutta la sequenza video, nonostante occlusioni, movimenti rapidi o variazioni di aspetto [10].

L'approccio predominante, utilizzato anche in questo progetto, è il paradigma *Tracking-by-Detection*. Questo metodo suddivide il problema in due fasi distinte per ogni frame del video:

1. **Detection:** Un rilevatore identifica tutti i target presenti nel frame corrente.
2. **Data Association:** Un algoritmo di tracking associa le nuove rilevazioni alle traiettorie esistenti dai frame precedenti, basandosi su similarità spaziale o visiva.

Nello specifico, algoritmi moderni come **BoT-SORT** [11] o **ByteTrack** [12] (integrati con YOLO) eccellono nel gestire le associazioni anche quando la confidenza del rilevamento è bassa, riducendo la frammentazione delle tracce e garantendo che, se una persona viene temporaneamente occlusa, il sistema riesca a riassegnarle lo stesso ID quando torna visibile. Questo è cruciale per l'analisi comportamentale temporale: senza un tracking stabile, la rete LSTM non riceverebbe una sequenza coerente di movimenti della stessa persona.

2.4 Reti Neurali Ricorrenti

Le *Reti Neurali Ricorrenti* (RNN) [13] sono un tipo di architettura di rete neurale utilizzata per rilevare pattern in una sequenza di dati.

Tali dati possono essere scritti a mano, genomi, testo o serie temporali numeriche. Sono applicabili anche alle immagini, se queste vengono rispettivamente scomposte in una serie di patch e trattate come una sequenza. A un livello superiore, le RNN trovano applicazioni nella modellazione e generazione di testo del linguaggio, nel riconoscimento vocale, nella generazione di descrizioni di immagini o nel tagging di video [14].

A differenza di reti Feed-Forward neural network (MLP) dove le informazioni attraverso la rete senza cicli, le RNN hanno cicli e trasmettono le informazioni al loro interno. Ciò consente loro di estendere la funzionalità delle reti Feed-Forward per tenere conto anche degli input precedenti e non solo dell'input corrente.

Quando le reti neurali vengono allenate l'obiettivo è minimizzare l'errore. Per farlo, la rete calcola l'errore alla fine e poi torna indietro nel tempo (Backpropagation) per aggiornare i pesi (weights) che hanno causato quell'errore. In una RNN, poiché i dati sono sequenziali, la rete deve tornare indietro attraverso t passi temporali. Questo causa 2 problemi fondamentali di questo tipo di rete:

- Il Gradiente che Svanisce (Vanishing Gradients) [15].
- Il Gradiente che Esplode (Exploding Gradients) [15].

Il *gradiente che esplode* accade quando ci sono valori nei pesi grandi (> 1) nella moltiplicazione di matrici, questo fa sì che il gradiente aumenti con ogni strato (o passo temporale) fino a diventare spesso NAN (Not a Number). Quindi la rete aggiorna i pesi con valori pesantemente modificati e la struttura si rompe e diverge completamente.

Il *gradiente che svanisce* accade quando ci sono valori nei pesi piccoli (< 1) nella moltiplicazione di matrici, questo fa sì che il gradiente diminuisca con ogni strato (o passo temporale) e infine svanisca. Quindi la rete smette di imparare dalle prime parti della sequenza perché l'errore è praticamente zero. Di conseguenza, il modello tende a trascurare gli elementi iniziali della sequenza, compromettendo l'apprendimento di relazioni temporali a lungo raggio. Ciò porta le RNN ad avere una memoria a breve termine, non possono imparare da sequenze troppo lunghe. Per risolvere questo problema sono state create le LSTM.

2.4.1 LSTM

Le *memorie a lungo-breve termine* (Long-Short Term Memory, LSTM) [16] sono state create per risolvere il problema del *gradiente che svanisce* e del *gradiente che esplode*, ciò è reso possibile dal fatto che immagazzinano più informazioni al di fuori del flusso di rete neurale in strutture chiamate *celle a porte* (gate cell) e sono di 3 tipi:

- Input Gate.

- Forget Gate.
- Output Gate.

L'*input gate* I_t decide quali nuove informazioni sono abbastanza significative da essere memorizzate nello stato della cella. Utilizza lo stesso meccanismo del *forget gate*. Parallelamente, viene calcolato un *candidate state* (stato candidato) $\tilde{C}_t$. Questo vettore contiene le nuove informazioni grezze che potrebbero essere aggiunte allo stato della cella. A differenza delle porte che usano una sigmoide, qui si utilizza la funzione tangente iperbolica (tanh) per regolare i valori tra -1 e 1, permettendo alla rete di gestire dati positivi e negativi.

Il *forget gate* F_t decide quali informazioni dello stato della cella precedente non sono più rilevanti e devono essere scartate. Esamina lo stato nascosto precedente e l'input corrente, emettendo un numero compreso tra 1 (conservare) e 0 (scartare)

L'*output gate* O_t controlla quali parti dello stato della cella devono essere trasmesse allo stato nascosto successivo. Questa uscita filtrata viene utilizzata per le previsioni e trasmessa al passo temporale successivo nella sequenza.

I calcoli per queste porte e del candidato sono mostrati nelle Equazioni 2.2, 2.3, 2.4, 2.5.

$$O_t = \sigma(X_t W_{xo} + H_{t-1} W_{ho} + b_o) \tag{2.2}$$

$$I_t = \sigma(X_t W_{xi} + H_{t-1} W_{hi} + b_i) \tag{2.3}$$

$$F_t = \sigma(X_t W_{xf} + H_{t-1} W_{hf} + b_f) \tag{2.4}$$

$$\tilde{C}_t = \tanh(X_t W_{xc} + H_{t-1} W_{hc} + b_c) \tag{2.5}$$

Nelle Equazioni vengono usati $W_{xi}, W_{xf}, W_{x0}, W_{xc} \in \mathbb{R}^{d \times h}$ e $W_{hi}, W_{hf}, W_{h0}, W_{hc} \in \mathbb{R}^{h \times h}$ come matrici dei pesi, mentre $b_i, b_f, b_0, b_c \in \mathbb{R}^{1 \times h}$ sono le rispettive polarizzazioni.

Il nuovo stato della cella C_t viene ottenuto come combinazione lineare tra lo stato precedente C_{t-1} modulato dal *forget gate* e lo stato candidato $\tilde{C}_t$ modulato dall'*input gate*, tramite prodotto elemento per elemento mostrato nell'Equazione 2.6

$$C_t = F_t \odot C_{t-1} + I_t \odot \tilde{C}_t \tag{2.6}$$

2.4.2 Bidirectional LSTM (BiLSTM)

Per molte attività di elaborazione di sequenze, è utile analizzare sia il futuro che il passato di un dato punto della serie. Tuttavia, le *Reti Neurali Ricorrenti* convenzionali sono progettate per analizzare i dati in una sola direzione: il passato. Una soluzione parziale a questo inconveniente consiste nell'introdurre un ritardo tra gli input e i relativi target, fornendo così alla rete alcuni intervalli temporali di contesto futuro [17]. Sebbene una soluzione parziale sia stata quella di introdurre un ritardo tra gli input e i target per fornire un contesto futuro limitato, questo approccio riduce la rete a finestre temporali fisse, simili ai Percettrone multistrato (MLP) [18], perdendo i vantaggi della ricorrenza. Una soluzione superiore è rappresentata dalle *Reti Neurali Ricorrenti Bidirezionali* (BRNN), introdotte da Schuster e Baldi [19]. In questo modello, l'input viene presentato simultaneamente in due direzioni:

- Forward (dal passato al futuro).
- Backward (dal futuro al passato).

Queste due direzioni sono gestite da due reti ricorrenti separate, entrambe connesse allo stesso livello di output. Le BRNN hanno dimostrato miglioramenti significativi in compiti di apprendimento sequenziale, come la previsione della struttura delle proteine e l'elaborazione del parlato [20, 21].

Capitolo 3

Metodologia e Implementazione

3.1 Selezione e Addestramento dei Modelli

L'architettura del sistema proposto si basa sulla cooperazione di tre reti neurali distinte, ognuna deputata a un compito specifico all'interno della pipeline di rilevamento:

- **YOLO11-Pose:** Per l'estrazione delle feature biometriche (scheletro).
- **YOLO11-Detect:** Per il riconoscimento specifico di armi bianche (coltelli).
- **Bi-LSTM:** Per l'analisi temporale della sequenza di movimento.

3.1.1 Modello di Pose Estimation

Per il componente di stima della posa è stato selezionato il modello **YOLO11n-pose** (versione *nano*), pre-addestrato sul dataset *COCO*. Sebbene fosse possibile effettuare un fine-tuning, i test preliminari hanno dimostrato che il modello "out-of-the-box" possiede già una capacità di generalizzazione sufficiente per rilevare correttamente i 17 keypoint scheletrici anche in posizioni complesse o durante colluttazioni. Pertanto, il modello è stato integrato mantenendo i pesi originali, ottimizzando così i tempi di sviluppo senza compromettere l'accuratezza dell'estrazione delle feature.

3.1.2 Modello di Rilevamento Armi

Per il rilevamento delle minacce armate, è stato utilizzato il modello **YOLO11n** pre-addestrato su *ImageNet*. A differenza del modello di posa, i pesi originali non erano sufficienti per discriminare con alta precisione un oggetto piccolo e specifico come un coltello in contesti vari.

Costruzione del Dataset

Per l'addestramento è stato creato un dataset aggregando immagini provenienti da due fonti principali per garantire variabilità visiva:

1. **Roboflow Universe:** Un sotto-insieme curato in maniera mirata per la rilevazione di armi bianche [22][1].
2. **Open Images Dataset V7:** Sono state estratte le sole istanze appartenenti alla categoria "Knife"[2].

Il dataset risultante è stato suddiviso in set di Training, Validation e Test secondo le proporzioni riportate in Tabella 3.1.

Split	**Numero Immagini**	**Percentuale**
Training Set	2865	∼75%
Validation Set	258	∼7%
Test Set	704	∼18%
Totale	**3827**	**100%**

Tabella 3.1: Composizione del dataset per il rilevamento coltelli.

Configurazione del Training

Il processo di fine-tuning è stato eseguito utilizzando l'infrastruttura di Ultralytics. I parametri di addestramento sono stati impostati come segue per massimizzare la mAP (mean Average Precision) evitando l'overfitting:

[1]https://universe.roboflow.com/tesi-c7kla/knife-fwpuw-rrrcw.
[2]https://storage.googleapis.com/openimages/web/index.html.

- **Epoche:** Il training si è arrestato all'epoca 181. La durata del processo è stata gestita tramite il meccanismo di *Early Stopping*. Questa tecnica monitora costantemente le prestazioni sul dataset di validazione e interrompe l'addestramento qualora non si osservino miglioramenti significativi per un numero prefissato di epoche consecutive (patience). Ciò è fondamentale per prevenire l'*overfitting*, evitando che la rete "memorizzi" il rumore dei dati di training perdendo capacità di generalizzazione.
- **Image Size:** 640×640 pixel.
- **Batch Size:** Fissato a 16. Questo parametro definisce il numero di campioni processati simultaneamente dalla GPU prima di calcolare l'errore medio e aggiornare i pesi della rete.
- **Optimizer:** È stato adottato l'algoritmo SGD (*Stochastic Gradient Descent*) con momentum.

Il modello risultante (`best.pt`) ha dimostrato un notevole miglioramento nella confidenza di rilevamento rispetto al modello base, riducendo drasticamente i falsi negativi in condizioni di scarsa illuminazione o movimento mosso (motion blur).

3.1.3 Modello Bi-LSTM

Per il rilevamento delle azioni violente, è stato implementato un modello **Bi-LSTM**, progettato specificamente per l'analisi di sequenze video eterogenee.

Acquisizione dei Dataset

Per l'addestramento del modello è stato costruito un dataset video proveniente da quattro sorgenti differenti, bilanciando scene con azioni violente esplicite con azioni comuni per ridurre i falsi positivi:

- **A Dataset for Automatic Violence Detection in Videos**[3]**:** Contenente video violenti e non, realizzati simulando le azioni in ambiente controllato [23].

[3]https://github.com/airtlab/A-Dataset-for-Automatic-Violence-Detection-in-Videos.

- **Indoor Action Dataset**[4]**:** Un insieme di video contenente solamente azioni di vita quotidiana (sedersi, camminare, prendere oggetti) [24].
- **Real Life Violence Situations Dataset**[5]**:** Contiene 1000 video violenti e 1000 non-violenti, estratti da YouTube, che ritraggono situazioni reali in diverse condizioni ambientali [25].
- **Real-World Fighting 2000**[6]**:** Un dataset con 1000 video violenti e 1000 non violenti, provenienti da videocamere di sorveglianza reali [26].

Strategia di Suddivisione del Dataset

La ripartizione dei dati in set di Training, Validation e Test è stata eseguita adottando una strategia gerarchica per garantire sia la correttezza della valutazione che il bilanciamento dei campioni.

Nella fase preliminare del lavoro, è stata effettuata una selezione dei dati destinati alla valutazione delle performance, isolando un totale di 55 video dal dataset originale. Per garantire una valutazione imparziale e bilanciata, il campione è stato suddiviso equamente tra le tre classi principali individuate: video violenti, video non-violenti e scene di azioni violente caratterizzate dalla presenza di coltelli.

Questa strategia di campionamento, che prevede la separazione dei dati a livello di *video sorgente* piuttosto che di singoli frame, è di fondamentale importanza per contrastare il fenomeno del *Data Leakage* [27]. In contesti di analisi video, il rischio principale è che il modello possa memorizzare caratteristiche statiche o ambientali — come lo sfondo, le condizioni di illuminazione o l'abbigliamento dei soggetti — qualora frame appartenenti allo stesso video fossero presenti sia nel set di addestramento che in quello di test.

Garantendo che il **Test Set** sia composto da sequenze video interamente inedite, si obbliga il sistema a generalizzare le feature apprese, testando la sua capacità di riconoscere l'azione violenta in contesti mai visti in precedenza. Da questo sottoinsieme di 55 video sono state successivamente derivate 183 sequenze focalizzate sui

[4]https://github.com/DaniDeniz/IndoorActionDataset.
[5]https://www.kaggle.com/datasets/mohamedmustafa/real-life-violence-situations-dataset.
[6]https://www.kaggle.com/datasets/vulamnguyen/rwf2000.

soggetti (*person sequences*). Tali dati permettono di analizzare le dinamiche del movimento umano con precisione, riducendo le interferenze causate da elementi irrilevanti presenti nella scena e massimizzando l'efficacia del processo di inferenza.

Successivamente, per i video destinati all'addestramento, la suddivisione tra **Training** (80%) e **Validation** (20%), come si vede in Tabella 3.2, è stata effettuata a livello di **singola sequenza scheletrica estratta** (ovvero per ogni Person ID) e non a livello di video sorgente. Questa metodologia è stata necessaria per mitigare la varianza nella densità dei soggetti: un singolo video potrebbe contenere una rissa di gruppo con 20 persone, mentre un altro un singolo soggetto. Operando sui file delle singole persone, si è garantito un bilanciamento preciso, come riportato in Tabella 3.2.

Il **Validation set** è stato utilizzato per valutare il comportamento del modello durante l'addestramento, monitorandone la capacità di generalizzazione sui dati non visti. Le metriche di validation riportate nel corso del lavoro sono riferite alla configurazione iperparametrica finale del modello, mantenuta costante durante l'intero processo di training e descritta nella sezione dedicata all'architettura e alla configurazione dell'addestramento.

Split	**Numero Sequenze (Persone)**	**Percentuale**
Training Set	14.004	∼79%
Validation Set	3.540	∼20%
Test Set	183	∼1%
Totale	**17.727**	**100%**

Tabella 3.2: Composizione finale del dataset basata sulle sequenze scheletriche estratte.

Preprocessing e normalizzazione

Per l'addestramento della rete neurale, è stato necessario eseguire preliminarmente una fase di pre-elaborazione (preprocessing) dei video. Poiché le reti Bi-LSTM richiedono input numerici strutturati e non flussi video grezzi, i dati sono stati convertiti in tensori contenenti:

- **Sequenze di Keypoint:** Coordinate scheletriche separate per ogni ID tracciato.

- **Etichette (Labels):** Classificazione binaria (1 = Violento, 0 = Non Violento).

Nello specifico, ogni persona presente nel video è stata convertita in un file *.npz*. Tuttavia, i dati grezzi forniti dal modello di posa sono sensibili alla posizione del soggetto nell'inquadratura e alla sua distanza dalla telecamera. Per mitigare questo problema e garantire l'invarianza di scala, è stata implementata una funzione di normalizzazione relativa al torso.

L'algoritmo opera secondo i seguenti passaggi logici:

1. Definizione del Centro di Riferimento. Vengono identificati i keypoint corrispondenti alla spalla sinistra (P_{sx}) e alla spalla destra (P_{dx}). Se entrambi i punti sono rilevati con confidenza sufficiente, si calcola il centro del torso (C) come punto medio, come mostrato nell'Equazione 3.1:

$$C = \frac{P_{sx} + P_{dx}}{2} \tag{3.1}$$

Tutti i keypoint dello scheletro vengono traslati sottraendo C. In questo modo, l'origine del sistema di coordinate (0,0) coincide sempre con il centro del petto del soggetto.

2. Definizione del Fattore di Scala. Per gestire la variazione della distanza dalla telecamera, si calcola la larghezza delle spalle (W) come distanza Euclidea:

$$W = ||P_{sx} - P_{dx}||_2 \tag{3.2}$$

Per evitare instabilità numerica, viene imposto un valore minimo di sicurezza ($W_{min} = 0.01$).

3. Trasformazione Finale. Ogni keypoint P_i viene infine normalizzato applicando la traslazione e scalando per la larghezza delle spalle (Eq. 3.3):

$$P'_i = \frac{P_i - C}{W} \tag{3.3}$$

Il risultato P'_i rappresenta la posizione del giunto relativa al centro del corpo, espressa in unità di "larghezze di spalle".

4. Gestione dei Fallback. Nel caso in cui il modello di posa non riesca a rilevare le spalle (es. occlusione), l'algoritmo adotta una strategia di *fallback*: utilizza il centro dell'immagine come riferimento (0.5, 0.5) senza fattore di scala. Questo approccio permette di mantenere il flusso dei dati attivo, lasciando alla rete Bi-LSTM la gestione del rumore.

Architettura della Rete e Configurazione dell'Addestramento

L'implementazione del modello di classificazione temporale è stata realizzata utilizzando il framework Keras/TensorFlow [28]. L'architettura definita è di tipo sequenziale e progettata specificamente per gestire le sfide tipiche dei dati scheletrici, come la lunghezza variabile delle azioni e il rischio di overfitting.

Definizione dell'Architettura. La rete è composta da una serie di strati, ognuno con una funzione specifica nel processo di estrazione delle feature temporali:

1. **Masking Layer:** Il primo strato gestisce la lunghezza variabile delle sequenze video. Poiché l'input deve avere una dimensione fissa (150×51), le sequenze più brevi vengono riempite (padding) con un valore sentinella (-999.0). Questo strato "maschera" tali valori, istruendo i livelli successivi a ignorarli nei calcoli, evitando che il rumore del padding influenzi la predizione.

2. **Primo Strato Bi-LSTM:** Un livello LSTM Bidirezionale con 64 unità. L'impostazione `return_sequences=True` permette di passare l'intera sequenza di output (e non solo l'ultimo stato) al livello successivo, così che il livello successivo possa analizzare ogni istante della sequenza.

3. **Dropout (0.3):** Uno strato di regolarizzazione che disattiva casualmente il 30% dei neuroni durante l'addestramento per prevenire l'adattamento eccessivo ai dati di training.

4. **Secondo Strato Bi-LSTM:** Un livello LSTM Bidirezionale con 32 unità. Questo strato sintetizza le informazioni estratte dal precedente, producendo una rappresentazione temporale più compatta.

5. **Batch Normalization:** Normalizza gli input del livello successivo, rendendo l'addestramento più stabile e veloce.

6. **Dense Layer (Fully Connected):** Uno strato denso con 32 neuroni e funzione di attivazione *ReLU*, seguito da un ulteriore Dropout (0.2).

7. **Output Layer:** Un singolo neurone con funzione di attivazione *Sigmoide*, che restituisce una probabilità compresa tra 0 e 1 (dove 1 indica la classe "Violento").

Strategia di Addestramento. Il modello è stato compilato utilizzando l'ottimizzatore **Adam**, scelto per la sua efficacia durante l'addestramento, e la funzione di perdita **Binary Crossentropy**, standard per classificazione a due classi.

Per massimizzare le prestazioni e garantire la generalizzazione, sono state adottate diverse strategie durante il fitting:

- **Gestione dello Sbilanciamento:** Sono stati calcolati pesi correttivi (*Class Weights*) inversamente proporzionali alla frequenza delle classi nel training set, penalizzando maggiormente gli errori sulla classe minoritaria.

- **Early Stopping:** L'addestramento, configurato per un massimo di 100 epoche, viene interrotto anticipatamente se la *validation loss* non migliora per 10 epoche consecutive, con ripristino dei pesi migliori.

- **Learning Rate:** In presenza di plateau della validation loss, il tasso di apprendimento viene ridotto del 50% dopo 3 epoche senza miglioramenti.

La validation delle prestazioni è stata condotta adottando la seguente configurazione iperparametrica, mantenuta costante durante l'addestramento:

- *Batch Size:* 32

- *Numero massimo di epoche:* 100

- *Lunghezza massima delle sequenze (Max Frames):* 150

3.2 Architettura Software del Sistema

Il sistema sviluppato segue un'architettura modulare che separa nettamente la logica di acquisizione e visualizzazione (Frontend) dalla logica di elaborazione (Backend). Il flusso di lavoro è di tipo *pipeline* come mostrato in Figura 3.1. Per ogni frame acquisito dal flusso video, il sistema avvia due processi di inferenza distinti: una di tipo object detection con YOLO-Object e una di rilevamento di persone con YOLO-Pose. I dati strutturati relativi agli scheletri vengono successivamente processati dal modello temporale Bi-LSTM, il quale classifica l'azione svolta analizzando la dinamica del movimento. Nella fase conclusiva, i risultati dell'object detection e della classificazione comportamentale convergono in un modulo di decisione logica che determina lo stato di minaccia globale. Il frame viene quindi annotato graficamente e, unitamente a una lista strutturata dei metadati rilevati, viene inviato al frontend per la visualizzazione all'utente finale. L'implementazione è realizzata interamente in Python[7], sfruttando il paradigma della Programmazione Orientata agli Oggetti (OOP), consentendo di incapsulare le diverse responsabilità funzionali in moduli ben definiti e favorendo estendibilità, manutenibilità e riuso del codice.

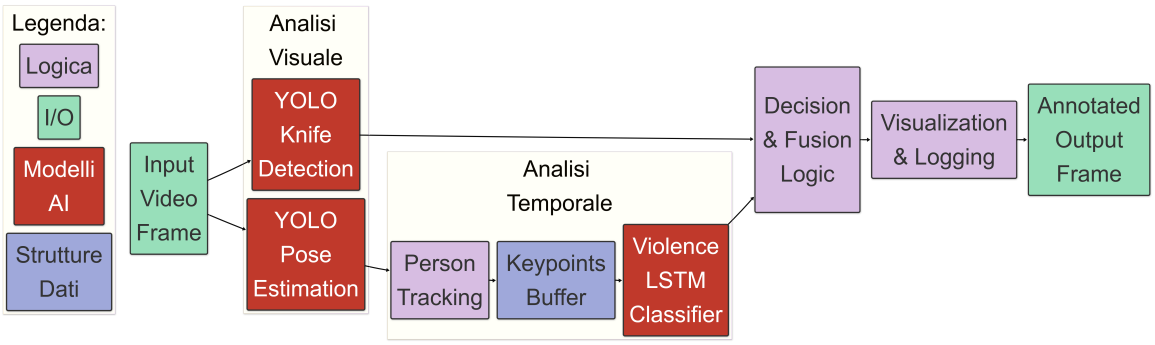

Figura 3.1: Pipeline del sistema.

[7]https://www.python.org/.

3.2.1 Acquisizione video e Interfaccia (Frontend)

L'interazione utente e del flusso video è affidata al modulo principale, sviluppato utilizzando la libreria **Tkinter**[8]. Questo componente agisce come *Controller*, gestendo il caricamento dei dati e l'aggiornamento della vista.

Gestione dello Stream Video

L'acquisizione dei flussi video, sia da sorgente live (webcam) che da file locale, è implementata tramite le API della libreria **OpenCV**[9]. È stata definita una classe dedicata, `VideoStream`, che astrae la complessità della lettura frame-by-frame e fornisce un'interfaccia unificata al sistema.

Le sue responsabilità principali includono:

- Gestione e calcolo degli FPS (Frames Per Second).
- Bufferizzazione dell'ultimo frame elaborato (`old_frame`) per supportare la funzionalità di *Frame Skipping*.
- Inviare il frame al Backend per l'elaborazione
- Riceve frame modifcato dal Backend e renderizzarlo

Calcolo FPS. Per il Calcolo degli FPS viene usato un algoritmo che utilizza il modulo time di python. Il calcolo si basa sulla misurazione del tempo trascorso (Δt) tra l'elaborazione del frame corrente e quella del frame precedente. Nello specifico, viene acquisito il timestamp attuale (t_{curr}) e sottratto al timestamp salvato nell'iterazione precedente (t_{prev}). Il valore degli FPS è ottenuto calcolando il reciproco di questo intervallo temporale, come espresso nell'Equazione 3.4. Per evitare errori di instabilità numerica, il calcolo viene eseguito solo se l'intervallo temporale è strettamente positivo. Questo calcolo viene eseguito ad ogni iterazione del ciclo di elaborazione del frame.

$$FPS = \frac{1}{\Delta t} = \frac{1}{t_{curr} - t_{prev}} \tag{3.4}$$

[8] https://docs.python.org/3/library/tkinter.html.
[9] https://opencv.org/.

Frame Skip. Una delle criticità principali nei sistemi di visione in tempo reale è la latenza computazionale: se il tempo di elaborazione di un singolo frame (T_{proc}) supera l'intervallo di tempo tra due frame consecutivi (T_{frame}), il sistema accumula un ritardo crescente (lag), compromettendo la sincronizzazione live.

Per mitigare questo problema e adattare il carico computazionale alle risorse hardware disponibili, è stata implementata una strategia di *Frame Skipping* adattivo. Il sistema non elabora l'intero flusso video, ma seleziona un sottoinsieme di frame a intervalli regolari definiti da un parametro N_{skip}.

Formalmente, dato un flusso video in ingresso $F = \{f_1, f_2, \ldots, f_t\}$ e una funzione di Inferenza AI (che include YOLO e Bi-LSTM), l'output visualizzato V_t al frame t è definito dalla seguente logica:

$$V_t = \begin{cases} \text{Inferenza AI}, & \text{se } t \mod N_{skip} \equiv 0 \\ \text{Rendering Frame Precedente}, & \text{altrimenti} \end{cases} \tag{3.5}$$

Nel caso in cui il frame corrente non venga processato (ramo *altrimenti*), il sistema visualizza l'ultimo frame elaborato disponibile (V_{t-1}). Questa scelta progettuale è necessaria per l'esperienza utente: se venisse mostrato il frame attuale non processato (f_t), questo risulterebbe privo di annotazioni grafiche (bounding box e label). L'alternanza rapida tra frame annotati e frame grezzi causerebbe uno sgradevole sfarfallio visivo (*flickering*). Mantenendo l'ultimo frame elaborato, si sacrifica leggermente la fluidità del video in favore della coerenza e stabilità delle informazioni visualizzate.

Interfaccia Grafica (UI)

L'interfaccia utente è progettata per fornire feedback in tempo reale. Il ciclo di aggiornamento non è bloccante e sfrutta il metodo `.after()` di Tkinter per creare un loop ricorsivo. L'interfaccia è divisa in 3 sezioni principali:

- Area di Visualizzazione Video
- Area di Visualizzazione Metadati
- Pannello di Controllo e Funzionalità

E quando non vengono aperti video risulta essere come nella Figura 3.2, mentre con un video aperto risulta essere come in Figura 3.3. Dei video dimostrativi possono essere visionati alla seguente playlist su YouTube[10].

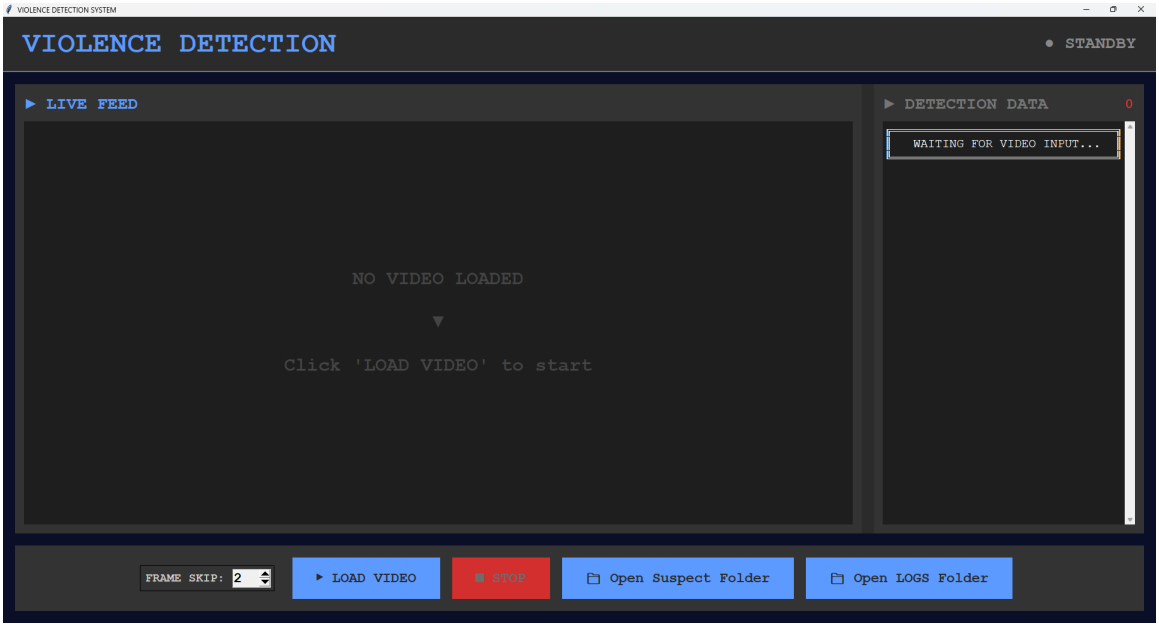

Figura 3.2: Interfaccia grafica senza video caricati.

[10]URL: `https://www.youtube.com/playlist?list=PLo1f0U_Wr1t2QE8hyGbKqB8WCAeb5DOUH`

Figura 3.3: Interfaccia grafica con video caricati.

Pannello di Controllo e Funzionalità. La sezione inferiore dell'interfaccia ospita i comandi necessari per la gestione del flusso operativo del sistema. I controlli disponibili sono:

- **Load Video:** Permette all'utente di selezionare e caricare la sorgente video da analizzare tramite una finestra di dialogo di sistema. Il sistema supporta i formati video standard gestiti da OpenCV.
- **Stop:** Interrompe immediatamente il processo di inferenza corrente, rilascia le risorse video e riporta il sistema in stato di *Standby*, azzerando i contatori delle rilevazioni.
- **Open Suspect Folder:** Fornisce un accesso rapido alla directory locale in cui il sistema salva automaticamente i volti dei soggetti classificati come "Sospetti" o "Violenti".
- **Open LOGS Folder:** Apre la cartella contenente i file di registro testuali, utili per l'analisi forense e il debug delle sessioni precedenti.

- **Frame Skip:** Un selettore numerico (Spinbox) che consente di modificare dinamicamente il parametro N_{skip}. Questa funzionalità permette all'operatore di bilanciare il carico computazionale in base all'hardware in uso, scegliendo quanti frame saltare tra una elaborazione e l'altra.

Area di Visualizzazione Video. Il pannello sinistro dell'interfaccia è dedicato al rendering del flusso video elaborato. Questa area mostra l'output del sistema in tempo reale, arricchito dalla sovrapposizione grafica dei risultati dell'inferenza. Nello specifico, per ogni frame visualizzato vengono renderizzati:

- Le *Bounding Box* colorate che identificano persone (Verde/Rosso) e armi (Blu).
- Le etichette testuali (Labels) contenenti lo stato del soggetto, l'ID di tracciamento e gli score di confidenza.

Area di Visualizzazione Metadati. Quando un frame viene elaborato, oltre a restituire il frame disegnato, vengono anche inviati tutti i dati utili relativi a quel frame. Questi dati vengono mostrati sulla destra della finestra e sono così formati: La visualizzazione dei dati analitici sull'interfaccia grafica segue uno standard di formattazione specifico, progettato per fornire all'operatore un riscontro immediato sullo stato di ogni soggetto. La stringa identificativa associata a ciascuna *bounding box* è strutturata secondo il seguente pattern:

`<Tipo> <ID> | <Stato> (<Score>) | Confidence: <Conf>`

dove i campi rappresentano:

- `<Tipo>`: Classificazione del soggetto ("Persona" o "Sospetto CON COLTELLO").
- `<ID>`: Identificativo univoco assegnato dal tracker (es. "1", "4").
- `<Stato>`: Esito della classificazione Bi-LSTM ("VIOLENTA" o "Non-Violenta").
- `<Score>`: Punteggio di confidenza della predizione di azioni violente (valore tra 0.0 e 1.0).

- `<Conf>`: Confidenza media del rilevamento dei keypoint scheletrici.

Seguono esempi di varie combinazioni di output generato dal sistema:

```
Sospetto CON COLTELLO 4 | VIOLENTA (0.85) | Confidence:  0.92

Sospetto CON COLTELLO 4 | Non-Violenta (0.85) | Confidence:  0.92

Persona 4 | VIOLENTA (0.85) | Confidence:  0.92

Persona 4 | Non-Violenta (0.85) | Confidence:  0.92
```

3.2.2 Sistema di Elaborazione (Backend)

Il cuore computazionale del sistema è nella classe `ViolenceDetectionSystem`. Questo modulo riceve in input i frame grezzi provenienti dal frontend e restituisce in output il frame annotato graficamente, unitamente ai metadati dell'analisi (stato di violenza, posizione bounding box, confidenza).

Il flusso di elaborazione per ogni singolo frame segue una pipeline sequenziale, strutturata in sei fasi logiche:

1. **Preservazione del Dato:** Duplicazione del frame per operazioni forensi.
2. **Object Detection:** Scansione del frame per la ricerca di armi (coltelli).
3. **Pose Estimation & Tracking:** Rilevamento delle persone e assegnazione degli ID temporali.
4. **Logica di Fusione e Predizione Bi-LSTM:** Analisi spaziale (sovrapposizioni) e temporale (movimento) per determinare la minaccia.
5. **Rendering Grafico:** Disegno delle bounding box e delle etichette informative sul frame.
6. **Output Formatting:** Aggregazione dei risultati per l'invio all'interfaccia grafica.

Preservazione del Dato

La prima operazione eseguita consiste nella duplicazione del frame in input (`frame.copy()`). Questa operazione è principale per garantire l'integrità del dato in caso di rilevamento positivo: il sistema necessita di un'immagine "pulita", senza annotazioni grafiche sovraimpresse, da cui estrarre e salvare i volti dei soggetti sospetti. Lavorare direttamente sul frame originale impedirebbe questa operazione, poiché le successive fasi di rendering andrebbero a "sporcare" i pixel del volto con rettangoli e testo.

Rilevamento Armi (YOLO Object Detection)

Il frame originale viene sottoposto all'inferenza del modello YOLO11, specificamente addestrato per il riconoscimento della classe "Knife". L'algoritmo scansiona l'intera immagine e restituisce una lista di oggetti rilevati, ognuno caratterizzato dalle coordinate spaziali della bounding box, dalla classe di appartenenza e dal punteggio di confidenza. Questi dati vengono memorizzati temporaneamente per essere successivamente confrontati con la posizione delle persone.

Stima della Posa e Tracciamento (YOLO Pose)

Successivamente, il sistema procede all'individuazione delle figure umane e all'estrazione dello scheletro (17 keypoint). A differenza di una semplice inferenza statica (metodo `predict`), è stato scelto di utilizzare il metodo `track`. Questa modalità integra un algoritmo di tracciamento (Tracker) che permette di assegnare a ogni persona un identificativo univoco (ID) e di mantenerlo consistente attraverso i frame successivi, anche durante movimenti rapidi.

La configurazione del tracciamento prevede il passaggio di parametri specifici per ottimizzare il compromesso tra velocità e precisione:

- **Frame**: Rappresenta l'immagine di input attuale su cui il modello deve eseguire l'inferenza.
- **Source**: Specifica l'algoritmo di tracciamento utilizzato per mantenere l'identità dei soggetti tra i frame. Nel sistema è stato implementato *BoT-SORT*, selezionato per la sua robustezza nel gestire movimenti rapidi e occlusioni, preferendolo all'alternativa *ByteTrack*.

- **Persist**: Parametro booleano essenziale per l'analisi video. Se impostato a *True*, garantisce la persistenza degli ID assegnati, permettendo al sistema di "ricordare" le persone e mantenere lo stesso identificativo tra un frame e l'altro.
- **Verbose**: Controlla l'output nel terminale. È impostato a *False* per evitare di stampare i dettagli di rilevamento standard di YOLO per ogni frame, mantenendo i log del sistema puliti e leggibili.
- **Image size (imgsz)**: Ridimensiona l'immagine di input prima dell'elaborazione. Questa riduzione è strategica per massimizzare la velocità di elaborazione su hardware con risorse limitate, accettando un lieve compromesso sulla risoluzione.
- **Half**: Abilita i calcoli a "mezza precisione" (*Half Precision*), passando da 32-bit a 16-bit (FP16). Questo ottimizza l'uso della memoria GPU e accelera notevolmente l'elaborazione senza perdite significative di accuratezza.

Logica di Fusione e Predizione Bi-LSTM

In questa sezione viene effettuata la parte più importante per il corretto funzionamento dell'intero sistema. Si inizia con la fusione delle coordinate dei coltelli trovati con quelli delle persone e poi la predizione con il modello Bi-LSTM.

Sovrapposizione Armi con Persone. Vengono iterati i risultati del traking Di YOLO pose, così da poter lavorare uno ad uno sulle singole persone rilevate. Per ogni persona viene controllato, iterando i risultati dell'Object Detection, se le bounding box degli oggetti trovati si sovrappongo a quella della persona di cui si sta controllando. Questo ci permette di capire in maniera semplice se una persona ha un coltello, perché se lo tenesse in mano comunque la sua bounding box si troverebbe all'interno di quella della persona. In questa prima fase viene decisa la classificazione della persona, quindi se non si sovrappongono viene classificata solamente come *Persona*, invece se si sovrappongo viene classificata come *Sospetto con coltello*.

Predizione azioni violente. Per predire le azioni violente tramite Bi-LSTM non bastava far elaborare i dati dal modello e vedere i risultati, ma è stato necessario

effettuare dei passaggi intermedi per elaborare quei dati per far in modo che il modello riuscisse ad utilizzare per poter effettuare una corretta predizione.

Per prima cosa si verifica che la somma della confidenza dei keypoint e la somma delle coordinate non siano 0, questo per evitare di dare al modello dei dati nel caso in cui venga identificata una persona ma non vengono trovati i keypoint o che non sono validi.

Dopodiché le coordinate dei keypoint vengono normalizzate al torso, come già spiegato nelle Formule 3.1, 3.2, 3.3, ciò deve essere fatto perché quando il modello è stato allenato è stata utilizzata questa configurazione. Infatti i dati che vogliamo predire devono essere processati allo stesso modo di come sono stati processati per allenarlo. Una volta che le coordinate sono state normalizzate, vengono concatenate con la confidenza degli stessi, risultando in questo formato $(17, 3)$ $[x_{\text{rel}}, y_{\text{rel}}, conf]$. Successivamente, tali valori vengono appiattiti fino a ottenere un vettore di dimensione (51), ottenuto come 17 $(punti)$ $\times$ 3 $(coordinate + confidenza)$, nella forma $[x_1, y_1, c_1, x_2, y_2, c_2, \ldots, x_{17}, y_{17}, c_{17}]$.

Ora bisogna creare una coda, per poter salvare i vari keypoint frame-per-frame e creare una sequenza. Ogni persona ha una coda personale, nel caso non l'avesse viene creata con il suo ID e con una lunghezza massima di 150 frame. 150 frame perché in un video a 30 fps risultano essere 5 secondi, più che sufficienti per capire se un'azione è violenta o no. Una volta creata la coda vengono aggiunte a quest'ultima le coordinate con la confidenza appiattite. Poi si verifica che l'attuale coda sia maggiore di 30, corrispondente a 1 secondo, valore minimo che serve per poter avviare un predizione con risultati corretti. Questo valore può variare in base a quanti frame erano stati decisi di saltare. Infatti se venisse deciso di elaborare a frame alterni ci ritroveremmo che per un secondo di video vengono elaborati solamente 15 frame invece che 30. Per evitare che la predizione, in questo caso, inizi dopo 2 secondi di video viene usata questa Formula

$$|\mathcal{S}_i| \geq \frac{30}{k}. \tag{3.6}$$

dove $|\mathcal{S}_i|$ indica il numero di frame validi nella sequenza associata all'individuo i e k indica che viene preso un frame ogni k frame disponibili.

Se tutte queste condizioni vengono soddisfatte si arriva a effettuare la predizione.

Ma prima vengono presa l'attuale sequenza e gli viene aggiunto un padding, dato che facciamo iniziare la predizione prima di aver elaborato 150 frame ma il modello è allenato con 150. Quindi viene creato un padding, in particolare, viene creato un array tridimensionale di dimensione $(1, MAX_{FRAMES}, 51)$ con MAX_{FRAMES} che corrisponde a 150.

- **1** Rappresenta il batch size, cioè il numero di sequenze elaborate contemporaneamente.
- **51** Corrisponde al numero di feature per frame, derivanti dai 17 keypoint della persona (x,y,conf).
- MAX_{FRAMES} Indica il numero massimo di frame considerati per una sequenza, 150 in questo caso.

L'array viene inizializzato con un valore di mascheramento come quello deciso durante l'allenamento, per questo modello -999.0.

Successivamente viene sovrascritta la prima parte della sequenza con i dati della sequenza in possesso, mentre i restanti rimangono mascherati. In questo modo, la rete Bi-LSTM può elaborare sequenze di lunghezza variabile senza problemi, ignorando i valori di padding.

Infine i dati con il padding vengono convertiti in **tensore** e viene passato alla rete Bi-LSTM che può iniziare la predizione. Come risultato ci darà la predizione per la sequenza corrente.

Decisione della predizione di azioni violente. Una volta che si ha il valore della predizione dell'azione violenta si verifica che questo valore sia superiore ad una soglia minima stabilità per poterlo considerare come un'azione violenta. Il valore scelto è 0.7 perché rimane superiore a 0.5 e quindi dovrebbe già essere abbastanza sicuro che sia violenta. Però per evitare che sia troppo al limite è stato impostato 0.7.

Se la persona su cui si sta lavorando risulta essere *sospetta* (quindi che le bounding box della persona e del coltello si sovrappongo) o *violenta*, viene stampato un log avvertendo di questo rilevamento e viene salvato il volto del sospettato.

Infine tutti i dati che sono stati trovati fino a questo momento vengono messi insieme creando la stringa come mostrato in precedenza.

Salvataggio Log e volti sospetti. Come funzioni di contorno è stato sviluppato un sistema di Log e di salvataggio dei volti dei sospetti, così da poter avere sempre sotto traccia cose sospette che succedono.

I Log vengono salvati in un file diverso per ogni giorno e al loro interno ci sono delle stringhe contenente l'ora in cui è stata rilevata l'azione violenta o la presenza di un sospettato e tutti i relativi dati, quindi:

- **Prefisso:** Se possiede o meno il coltello.
- **ID:** L'Id univo della persona.
- **Stato:** Se l'azione è violenta o no.
- **Confidenza:** La confidenza della persona.

La funzione di estrazione del volto opera in due modalità distinte per garantire robustezza:

1. **Modalità Keypoint-based:** Se i punti chiave del volto (naso, occhi, orecchie) sono rilevati con successo, il sistema calcola un bounding box minimo che li racchiude. Successivamente, questo box viene espanso con margini asimmetrici: 50% in larghezza, 150% verso l'alto (per includere la fronte e i capelli) e 100% verso il basso (per il mento).
2. **Modalità Fallback:** Qualora i keypoints non siano disponibili (come il volto parzialmente occluso o girato), l'algoritmo stima la posizione del volto ritagliando il terzo superiore del bounding box relativo all'intera persona.

Infine, viene effettuato un controllo sui limiti dell'immagine per evitare errori di segmentazione prima del salvataggio su disco.

Rendering Grafico. Ora che tramite i passaggi precedenti sono stati raccolti tutti i dati necessari, si può procedere a modificare il frame aggiungendo le box e le label. YOLO offre un metodo *plot* per poter disegnare in automatico sia le box che le label, ma non è stato utilizzato a causa della sua lentezza e personalizzazione. Infatti facendo dei test e misurano quanto tempo impiegava a disegnare si è scoperto che è

molto più lento rispetto a disegnarlo "a mano", poi non si poteva controllare il come venisse disegnato.

Quindi vengono prese le coordiante della box della persona e degli oggetti e viene creato un rettangolo, che può essere di 3 colori:

1. **Blu** Solo per i coltelli.
2. **Verde** Se la persona non è violenta e non ha coltelli.
3. **Rosso** Se la persona è violenta o ha un coltello.

Output Formatting. L'ultimo passo è la creazione di una lista contenente l'unione delle strighe create dalla *Logica di Fusione e Predizione Bi-LSTM*, ovvero quelle relative alle persone e di quelle degli oggetti. Il risultato finale è una lista con stringhe di persone e di oggetti che verrà passata al frontend e visualizzata nell'interfaccia grafica, insieme al nuovo frame disegnato.

Capitolo 4

Metriche di valutazione

4.1 Metriche di Valutazione

Per sviluppare un sistema di machine learning è fondamentale sapere come si comporta, per questo esistono le metriche di valutazione. Esse ci aiutano a misurare l'efficacia dei nostri modelli. In questo caso andremo ad utilizzare le classification metrics, ovvero andremo a misurare le prestazioni dei modelli di classificazione per validare l'efficacia dei modelli di Machine Learning sviluppati e misurarne le prestazioni in uno scenario reale, è fondamentale adottare metriche di valutazione. Poiché il sistema proposto opera nell'ambito della classificazione, la valutazione si basa sull'analisi della corrispondenza tra le predizioni del modello e le etichette reali (Ground Truth).

4.1.1 Precision, Recall e F1-Score

Precision

La Precisione è la frazione di elementi *Veri Positivi* (TP) divisa per il numero totale di unità previste positivamente. In particolare, i Veri Positivi sono gli elementi che sono stati etichettati come positivi dal modello e sono effettivamente positivi, mentre i *Falsi Positivi* (FP) sono gli elementi che sono stati etichettati come positivi dal modello, ma sono effettivamente negativi. La precisione ci dice quanto possiamo fidarci del modello quando prevede che un'unità sia positiva [29].

$$Precision = \frac{TP}{TP + FP} \tag{4.1}$$

Recall

La Recall è la frazione di elementi *Veri Positivi* (TP) divisa per il numero totale di unità classificate positivamente. In particolare, i *Falsi Negativi* (FN) sono gli elementi che sono stati etichettati come negativi dal modello, ma che in realtà sono positivi. La Recall misura la capacità del modello di trovare tutte le unità positive nel set di dati [29].

$$Recall = \frac{TP}{TP + FN} \tag{4.2}$$

F1-Score

L'F1-score rappresenta la media armonica tra *Precision* e *Recall*. La media armonica è scelta perché, rispetto alla media aritmetica, penalizza maggiormente i valori estremi, infatti in caso di uno dei due valori molto basso, il valore F1 scenderebbe drasticamente [30].

$$F1 - score = \frac{2(Precision \cdot Recall)}{Precision + Recall} \tag{4.3}$$

4.1.2 Mean Average Precision

La *Mean Average Precision* (mAP) è una metrica completa ampiamente utilizzata per valutare le prestazioni dei modelli di visione artificiale, in particolare in attività come il rilevamento di oggetti e la segmentazione delle istanze. A differenza della semplice accuratezza, che verifica semplicemente se un'immagine è classificata correttamente,

mAP valuta la capacità di un modello di trovare gli oggetti e la precisione con cui posiziona il riquadro di delimitazione attorno ad essi.

- **Intersezione su Unione (IoU):** Misura la sovrapposizione tra il riquadro previsto e l'annotazione di verità. È un rapporto compreso tra 0 e 1. Una previsione è spesso considerata un *vero positivo* solo se l'IoU supera una soglia specifica, come 0,5 o 0,75.
- **Precision**
- **Recall**

Nella pratica, la mAP viene spesso riportata secondo diverse soglie di IoU, tra cui:

1. **mAP@50:** considera un rilevamento corretto se l'IoU è almeno 0,50. È una metrica blanda.
2. **mAP@50-95:** è la media di mAP calcolata a soglie IoU da 0,50 a 0,95 con incrementi di 0,05. Questa metrica premia i modelli che raggiungono un'elevata accuratezza di localizzazione.

4.1.3 Analisi ROC e AUC

ROC

La curva **ROC** è uno strumento grafico per analizzare la capacità diagnostica di un classificatore binario al variare della soglia di discriminazione. Storicamente derivata dalla teoria dei segnali sviluppata per i radar durante la Seconda Guerra Mondiale, questa curva illustra il *trade-off* tra due metriche antagoniste:

- **Asse Y:** Il *Vero Positivo Rate* (TPR), coincidente con la Recall. Rappresenta la sensibilità del sistema.
- **Asse X:** Il *Falso Positivo Rate* (FPR), definito come la frazione di negativi erroneamente etichettati come positivi (Falsi Allarmi).

La curva viene generata calcolando le coppie (FPR, TPR) per ogni possibile soglia di confidenza del modello. Un classificatore ideale tende verso l'angolo in alto a sinistra del grafico (TPR=1, FPR=0), massimizzando i veri positivi e minimizzando i falsi allarmi [31].

AUC (Area Under the Curve)

L'Area Sotto la Curva (**AUC**) è una metrica scalare che sintetizza l'intera curva ROC in un singolo valore numerico, permettendo di confrontare diversi modelli indipendentemente dalla soglia scelta. Il valore dell'AUC varia nell'intervallo $[0, 1]$:

- AUC = 0.5: Indica un classificatore puramente casuale (nessuna capacità discriminante).
- AUC = 1.0: Indica un classificatore perfetto (come descritto di seguito).

L'interpretazione probabilistica dell'AUC è particolarmente intuitiva: essa rappresenta la probabilità che il classificatore assegni un punteggio di confidenza più alto a un'istanza positiva scelta casualmente rispetto a un'istanza negativa scelta casualmente. In termini pratici, l'AUC misura il grado di *separabilità* tra le classi Violento e Non-Violento [31].

4.1.4 Matrice di confusione

La *matrice di confusione* è uno strumento utilizzato per valutare le prestazioni di un modello di classificazione nel machine learning, in quanto consente di confrontare le etichette previste dal modello con quelle effettive (*ground truth*) su un insieme di dati. In particolare, essa è una tabella che riporta, per ciascuna classe, il numero di istanze correttamente e erroneamente classificate, mettendo in relazione le classi reali con quelle predette dal modello. Gli elementi lungo la diagonale principale rappresentano le predizioni corrette, mentre i valori al di fuori della diagonale corrispondono agli errori di classificazione.

4.1.5 Sfarfallio

Lo *sfarfallio* [32] un fenomeno indesiderato nei sistemi di analisi video, in cui le predizioni del modello variano in modo rapido e incoerente tra frame consecutivi, nonostante la scena osservata potrebbe essere rimasta invariata. Questo comportamento è tipicamente dovuto a incertezze nella fase di rilevamento o classificazione e può essere accentuato dalla presenza di rumore, occlusioni parziali o variazioni improvvise di

illuminazione. Lo sfarfallio compromette la stabilità temporale delle predizioni, riducendo l'affidabilità complessiva del sistema e rendendo meno interpretabili i risultati, specialmente in applicazioni che richiedono continuità e coerenza nel tempo.

4.2 Risultati

In questa sezione vengono presentati e analizzati i risultati sperimentali ottenuti dai tre moduli principali che compongono l'architettura proposta: il rilevatore di oggetti (YOLO-Object), il modulo di stima della posa (YOLO-Pose) e il classificatore sequenziale (Bi-LSTM).

Le prestazioni sono state misurate utilizzando metriche standard del settore, quali Precision, Recall, F1-Score e mAP, integrate da un'analisi specifica sulla stabilità temporale delle predizioni per ridurre fenomeni di sfarfallio.

4.2.1 YOLO-Object

Precision, Recall, F1-Score

Il modello è stato valutato su un dataset di test composto da 704 immagini contenenti 779 istanze di oggetti.

Class	**Precision (P)**	**Recall (R)**	**F1-Score**
All	0.890	0.825	0.856

Tabella 4.1: Risultati delle prestazioni del modello YOLO sul dataset di test.

I risultati mostrati nella Tabella 4.2.1 mostrano una solida capacità di generalizzazione. Nello specifico, il modello ha ottenuto una Precision di 0.89 e una Recall di 0.825. Combinando queste due metriche, si ottiene un F1-score di 0.856, il quale indica un buon bilanciamento tra la capacità di evitare falsi positivi e quella di rilevare la maggior parte degli oggetti di interesse.

$$F1 - score = \frac{2(P \cdot R)}{P + R} = \frac{2(0.89 \cdot 0.825)}{0.89 + 0.825} = 0.856 \tag{4.4}$$

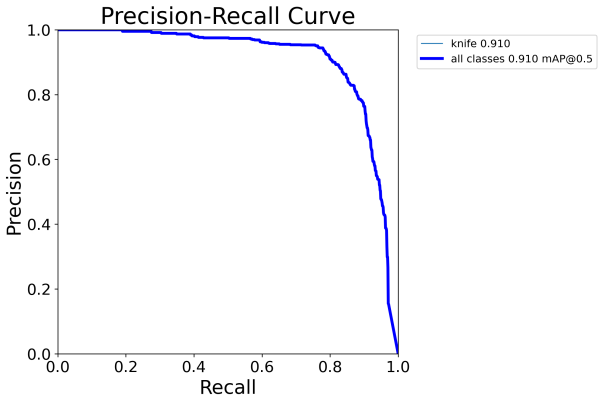

Figura 4.1: Grafico YOLO-Object Precision e Recall.

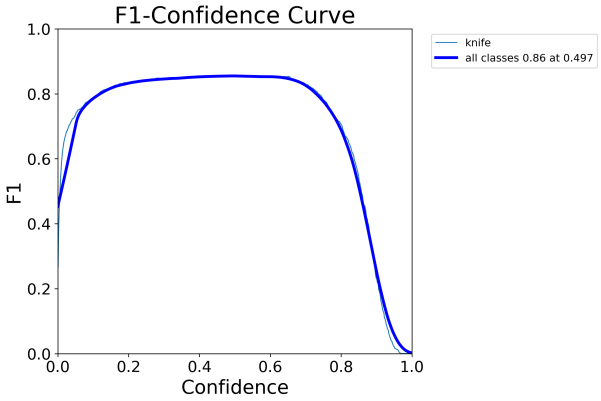

Figura 4.2: Grafico YOLO-Object F1-score.

Mean Average Precision

I risultati mostrati in Tabella 4.2.1, mostrano che il modello ha raggiunto un mAP@0.5 di 0.91, dimostrando che con una soglia di sovrapposizione (IoU) meno restrittiva, il rilevamento è estremamente accurato. Il valore di mAP@0.5:0.95 pari a 0.612 suggerisce che, mentre il modello localizza correttamente l'oggetto quasi sempre, le bounding box predette potrebbero avere lievi imperfezioni nell'allineamento perfetto con l'oggetto reale quando si richiedono soglie di precisione molto elevate.

Class	mAP@0.5	mAP@0.5:0.95
All	0.91	0.612

Tabella 4.2: Risultati delle metriche COCO del modello YOLO sul dataset di test.

Matrice di confusione

La Figura 4.3 illustra la matrice di confusione ottenuta dal modello di rilevamento coltelli (Knife Detector). A differenza dei classificatori standard, nell'ambito dell'Object Detection la matrice presenta una peculiarità strutturale nel quarto quadrante (incrocio tra *True Background* e *Predicted Background*).

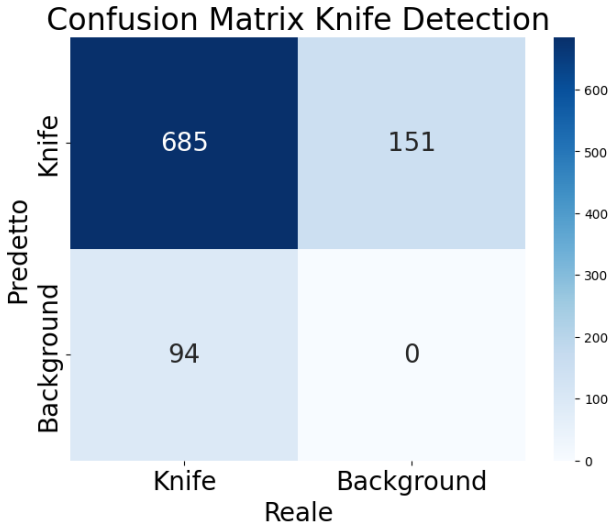

Figura 4.3: Matrice di confusione YOLO-Objects.

- **Vero Negativo (Assenti/Nulli):** Il quadrante in basso a destra appare vuoto o non valorizzato. Questo è un comportamento atteso e corretto per i rilevatori a singolo stadio come YOLO. Poiché il Background non è una classe target attiva ma rappresenta l'assenza di *Bounding Box*, il modello non conteggia esplicitamente quante volte ha correttamente ignorato lo sfondo.

4.2.2 YOLO-Pose

Precision, Recall, F1-Score

Le prestazioni del modello sono state valutate sul set COCO *val2017*. Poiché le annotazioni del set di test ufficiale non sono pubblicamente accessibili, l'utilizzo del set di validazione *val2017* rappresenta lo standard per il benchmarking e il confronto delle prestazioni per questo dataset [33].

Class	**Precision (P)**	**Recall (R)**	**F1-Score**
All	0.849	0.760	0.802

Tabella 4.3: Risultati delle prestazioni del modello YOLO sul dataset di validazione.

Analizzando i dati riportati nella Tabella 4.2.2, si riscontra una solida affidabilità predittiva. Il modello ha raggiunto una Precision di 0.849, confermando un'elevata accuratezza nel tracciamento dei keypoint quando una persona viene rilevata, riducendo al minimo i falsi positivi. La Recall si attesta a 0.760; sebbene questo valore sia leggermente inferiore alla precisione, indica comunque una buona capacità di individuare i soggetti nella maggior parte degli scenari. La combinazione di queste metriche restituisce un F1-score di 0.802, un risultato che mostra l'efficacia del modello nel fornire scheletri biometrici stabili e accurati.

$$F1 - score = \frac{2(P \cdot R)}{P + R} = \frac{2(0.849 \cdot 0.760)}{0.849 + 0.760} = 0.802 \tag{4.5}$$

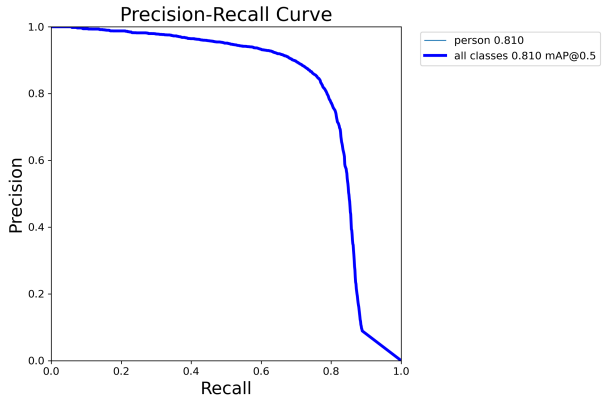

Figura 4.4: Grafico YOLO-Pose per Precision e Recall sullo scheletro.

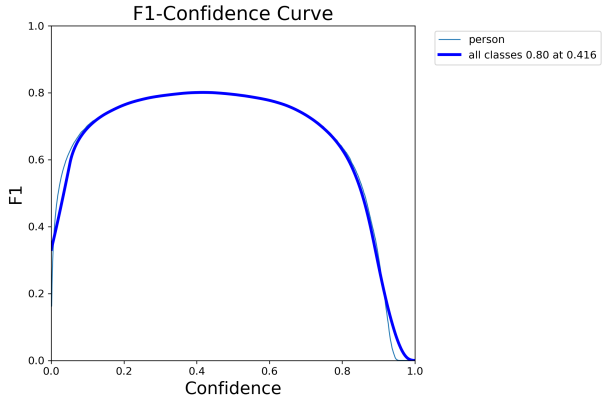

Figura 4.5: Grafico YOLO-pose per F1-score sullo scheletro.

Mean Average Precision

I risultati riportati nella Tabella 4.4 evidenziano le prestazioni del modello nello specifico task di stima della posa (Pose Estimation). Il valore di **mAP@0.5** pari a **0.81** indica una solida capacità del modello nell'identificare correttamente la struttura scheletrica dei soggetti: in oltre l'80% dei casi, la predizione dei keypoint ha una sovrapposizione significativa con il ground truth, confermando l'affidabilità del sistema nel rilevamento generale delle azioni.

Il valore di **mAP@0.5:0.95**, che si attesta a **0.511**, riflette invece la maggiore complessità intrinseca del task di allineamento fine. A differenza del semplice ri-

levamento di oggetti, la stima della posa richiede che ogni singolo punto articolare coincida perfettamente con la realtà.

Metrica	**mAP@0.5**	**mAP@0.5:0.95**
Pose (Keypoints)	0.810	0.511

Tabella 4.4: Metriche mAP relative alla stima dei keypoint (Pose) sul dataset di test.

Matrice di confusione

La Figura 4.6 mostra le prestazioni del modello YOLO11-Pose dedicato all'individuazione della figura umana. Anche in questo caso, la matrice riflette la natura di un task di *Object Detection*, presentando il quarto quadrante (Vero Negativo) non valorizzato.

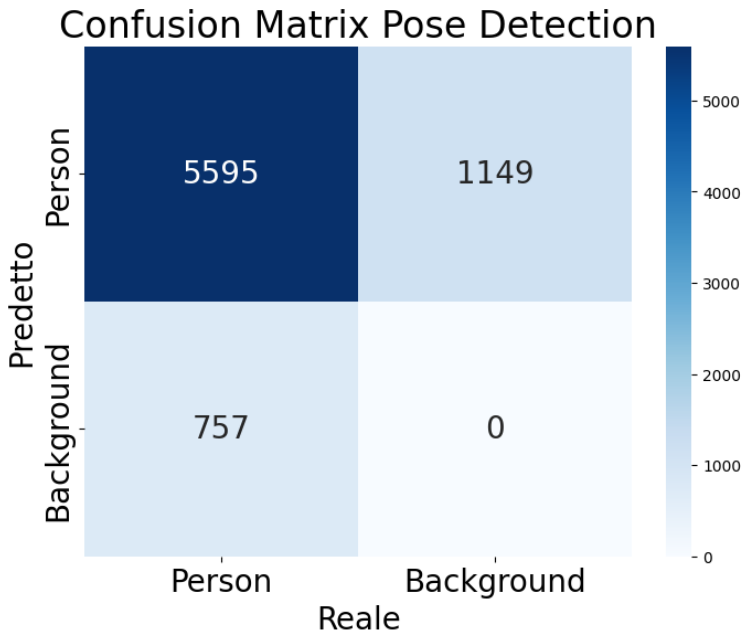

Figura 4.6: Matrice di confusione YOLO-Pose.

L'analisi quantitativa dei risultati evidenzia:

- **Vero Negativo:** Analogamente al rilevatore di coltelli, questo quadrante non è popolato poiché il modello non produce bounding box per il background correttamente ignorato.

4.2.3 Bi-LSTM

Per valutare le prestazioni del modello *Bi-LSTM* a livello di video, è stato utilizzato il **Test Set**, suddiviso in video violenti e non violenti. Un video viene classificato come violento se il modello rileva almeno un'azione violenta per **almeno 30 frame consecutivi**; in caso contrario, il video viene considerato non violento.

La soglia minima di 30 frame è stata determinata empiricamente attraverso una serie di esperimenti condotti sul **Validation Set**, come si può vedere dalla Figura 4.7, variando il numero di frame consecutivi da 10 a 40. È emerso che soglie troppo basse (ad esempio 10 frame) consentono di identificare correttamente quasi tutti i video violenti, ma producono un numero elevato di falsi positivi sui video non violenti.

Incrementando progressivamente la soglia, si osserva che intorno ai **30 frame** la *recall* per le classi violento e non violento risulta bilanciata. Al contrario, superata tale soglia, il comportamento si inverte: il modello tende a non riconoscere più correttamente i video con azioni violenti, mentre migliora la classificazione dei video non violenti. Per questo motivo, il valore di 30 frame è stato scelto come compromesso ottimale tra sensibilità e robustezza della classificazione.

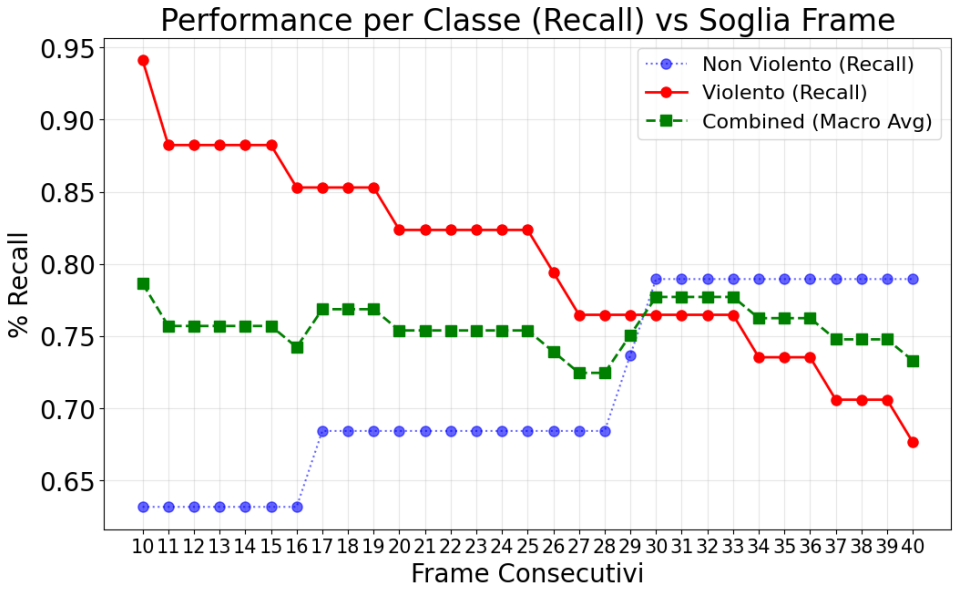

Figura 4.7: *Recall* in funzione del numero di frame consecutivi per la classificazione a livello di video. Il valore di 30 frame rappresenta un compromesso ottimale tra le classi violento e non violento.

Precision, Recall, F1-Score

Classe	Precision	Recall	F1-Score
Non Violento	0.65	0.79	0.71
Violento	0.87	0.76	0.81

Tabella 4.5: Metriche di classificazione del modello Bi-LSTM (Soglia sequenza: 30 frame).

I risultati riportati in Tabella 4.5 evidenziano le prestazioni del classificatore temporale Bi-LSTM. L'accuratezza complessiva del modello si attesta allo *0.77*, dimostrando una buona capacità generale nel distinguere sequenze dinamiche complesse.

Analizzando nello specifico la classe di interesse *Violento*, si osserva una *Precision molto elevata (0.87)*. Questo dato indica che il sistema genera pochissimi falsi positivi: quando viene segnalata un'azione violenta, è altamente probabile che l'evento stia realmente accadendo. La *Recall di 0.76* per la classe violenta indica che il modello è in grado di identificare correttamente circa tre quarti degli eventi aggressivi presenti

nel dataset di test. La combinazione di queste metriche restituisce un *F1-Score di 0.81* per la classe violenta, confermando l'efficacia del modello nel task principale.

ROC-AUC

La Figura 4.8 illustra l'andamento della curva ROC ottenuta applicando la soglia di persistenza temporale ottima di $N = 30$ frame. L'area sottesa alla curva, quantificata nel valore **AUC (Area Under the Curve) pari a 0.6749**, fornisce una misura sintetica della capacità discriminante globale del classificatore.

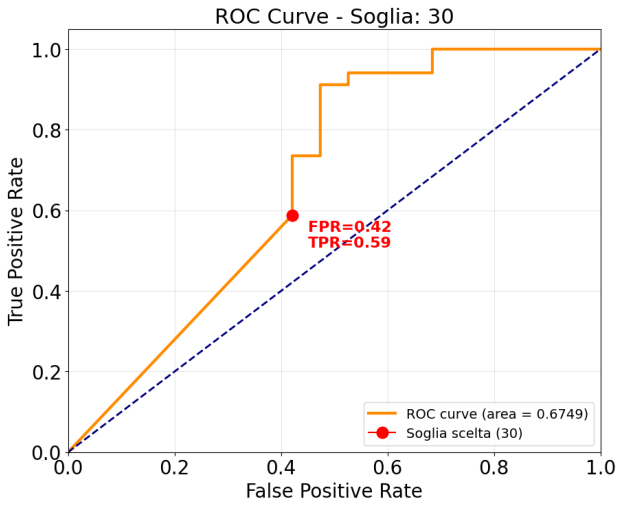

Figura 4.8: Curva ROC calcolata con soglia di persistenza temporale a 30 frame.

Da questo risultato si possono trarre delle considerazioni:

- **Capacità Predittiva:** Il punteggio risulta nettamente superiore alla soglia di casualità (0.5), confermando che il modello ha appreso efficacemente caratteristiche cinematiche distintive per distinguere azioni violente da quelle neutre.
- **Trade-off:** La forma della curva, che si mantiene sopra la diagonale, mostra l'esistenza di un intervallo operativo in cui il modello bilancia efficacemente Vero Positivo Rate e Falso Positivo Rate, supportando la scelta della soglia operativa adottata.

Matrice di confusione

La Figura 4.9 mostra le prestazioni del modello Bi-LSTM dedicato alla predizione del tipo di azione. In questo caso, a differenza delle precedenti, si può notare come tutti i quadranti sono utilizzati, implicando che ci sono Falsi Positivi, Falsi Negativi, Veri Positivi, e Veri Negativi.

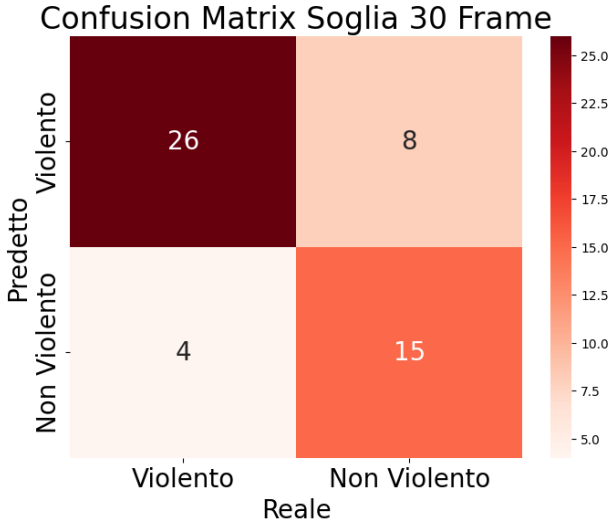

Figura 4.9: Matrice di confusione Bi-LSTM per video.

L'analisi quantitativa dei risultati evidenzia:

- **Vero Positivo:** Il primo quadrante infine mostra come su 34 video violenti, riesce a fare una giusta predizione su 26 di essi, classificandoli violenti.
- **Falso Positivo:** Il secondo quadrante mostra come su 34 video violenti, sbaglia la predizione su 8 di essi, classificandoli non-violenti.
- **Falso Negativo:** Il terzo quadrante mostra come su 19 video non-violenti, sbaglia la predizione su 4 di essi, classificandoli violenti.
- **Vero Negativo:** Il quarto quadrante mostra come su 19 video non-violenti, riesce a fare una giusta predizione su 15 di essi

Questo ci mostra come con la soglia impostata a 30 frame il sistema riesce a fare una buona distinzione. I video non-violenti ne identifica tali il 79% mentre il restante

21% non ci riesce. I video violenti riesce a indentificare correttamente il 76% e il restante 24% non ci riesce.

Sfarfallio

Un aspetto critico per l'usabilità di un sistema di sorveglianza in tempo reale è la coerenza delle segnalazioni nel tempo. L'analisi ha restituito un **Flicker Rate di 0.0202**.

Questo valore, estremamente contenuto, indica un'elevata stabilità delle predizioni del modello Bi-LSTM. Significa che, una volta identificata un'azione, il modello tende a mantenere la classificazione in modo solido, evitando oscillazioni rapide e spurie tra le due classi.

Capitolo 5

Conclusioni

Il presente lavoro di tesi ha portato alla progettazione e all'implementazione di un sistema di *Computer Vision* specializzato nel monitoraggio della sicurezza e nella rilevazione di situazioni violente. L'attenzione si è focalizzata in particolare sul riconoscimento di dinamiche violente e sull'individuazione tempestiva di armi, scenari in cui la velocità di risposta rappresenta un fattore determinante per l'efficacia dell'intervento.

L'obiettivo principale della è stato il superare i limiti dell'analisi statica delle immagini, orientandosi verso un'analisi temporale delle immagini, una comprensione delle azioni nel tempo. L'architettura sviluppata ha dimostrato che l'integrazione di diverse metodologie di *Deep Learning* permette di ottenere una solidità superiore rispetto ai singoli approcci isolati. Nello specifico, il sistema ha beneficiato delle seguenti soluzioni:

- **Rilevamento degli oggetti:** L'utilizzo di **YOLO11** per l'*Object Detection* ha garantito un'elevata precisione nel riconoscimento degli oggetti, mantenendo le prestazioni necessarie per l'elaborazione in tempo reale.
- **Stima della posa e rappresentazione del movimento:** L'integrazione della *Pose Estimation* tramite **YOLO-Pose** si è rivelata fondamentale per isolare il movimento umano dal *background*. Questo ha permesso alla rete **Bi-LSTM** (*Bidirectional Long Short-Term Memory*) di lavorare su dati strutturati (i *keypoints*), riuscendo a distinguere con successo tra azioni simili ma con valenza diversa, come un movimento brusco accidentale rispetto a un'aggressione fisica.

- **Analisi temporale delle azioni:** L'approccio bidirezionale della rete ricorrente ha consentito al modello di analizzare le sequenze video sfruttando sia il contesto precedente che quello successivo a ogni frame, riducendo significativamente il tasso di falsi positivi nelle transizioni di movimento più ambigue.

Dall'analisi effettuta sono emersi risultati incoraggianti che confermano la validità dell'approccio proposto:

- **Affidabilità nel Rilevamento Oggetti:** Il modulo dedicato all'individuazione di armi ha dimostrato un'elevata *Precision* (0.89), fattore cruciale per un sistema di sorveglianza dove un falso allarme relativo a un'arma potrebbe generare risposte di emergenza ingiustificate. Il bilanciamento espresso dall'F1-score (0.856) conferma che il modello YOLO11 è in grado di minimizzare i falsi negativi senza compromettere la specificità.

- **Ottimizzazione Temporale e Trade-off:** L'integrazione del modello Bi-LSTM per la classificazione delle azioni basata su scheletri ha beneficiato in modo determinante della logica di persistenza temporale. L'analisi sperimentale ha evidenziato come la **soglia di 30 frame** rappresenti il punto di equilibrio ottimale (*Trade-off*) tra sensibilità e specificità. Tale configurazione permette al sistema di ignorare micro-movimenti o artefatti visivi di breve durata, riducendo drasticamente le segnalazioni spurie e garantendo che solo azioni con una reale consistenza temporale vengano classificate come violente.

- **Stabilità e Continuità del Segnale:** Un risultato particolarmente significativo per l'impiego in contesti reali è il **Flicker Rate** estremamente contenuto (0.0202). Questo valore indica che l'output del sistema è coerente e stabile: una volta rilevata una situazione di pericolo, la segnalazione rimane costante per tutta la durata dell'evento, evitando l'effetto "intermittenza" che renderebbe difficile la gestione da parte di un operatore umano.

- **Analisi della Discriminazione:** Tuttavia, il valore di **ROC-AUC** pari a 0.6749 evidenzia la presenza di un margine di miglioramento nella separazione netta tra le classi. Questo risultato suggerisce che alcune azioni non aggressive,

se caratterizzate da dinamiche cinematiche rapide o convulse, condividono proprietà geometriche con le azioni violente, rendendo la distinzione basata sulla sola posa ancora parzialmente ambigua in casi limite.

Sviluppi futuri. Il sistema resta migliorabile da diversi punti di vista, anche avendo ottenuto buoni risultati.

Il modello **Bi-LSTM**, pur avendo dimostrato buone capacità nella rilevazione delle sequenze, rappresenta l'elemento con il più ampio potenziale di ottimizzazione. Uno sviluppo futuro naturale consisterebbe nell'esplorazione di architetture diverse, in grado di gestire le dipendenze a lungo termine e le relazioni temporali con una maggiore capacità di astrazione rispetto alle reti ricorrenti classiche.

Uno sviluppo ulteriore potrebbe focalizzarsi sull'uso delle informazioni contestuali, non limitandosi all'azione di un singolo individuo, ma prendendo in considerazione anche la dinamica della scena, le interazioni fra persone e la relazione spaziale con gli oggetti. Tutte queste informazioni messe insieme consentirebbero di migliorare la distinzione delle azioni violente da quelle non violente che, pur condividendo una cinematica simile, possono generare confusione nel sistema.

Un ulteriore aspetto particolarmente importante è l'adattamento del sistema verso un'impostazione orientata all'anticipazione. In quest'ottica, l'obiettivo del sistema non sarebbe più limitato al riconoscimento di un atto violento già in corso, ma all'individuazione precoce di segnali precursori e pattern comportamentali anomali che precedono l'evento critico. Implementare una logica predittiva di questo tipo consentirebbe di attivare misure di prevenzione e contenimento in tempo reale, trasformando il sistema da uno strumento di monitoraggio a una soluzione di sicurezza proattiva, capace di cogliere le dinamiche di escalation prima che sfocino in episodi di violenza conclamata.

Bibliografia

[1] Mahmoud Elnady and Hossam E. Abdelmunim. A novel yolo lstm approach for enhanced human action recognition in video sequences. *Scientific Reports*, 15 (1):17036, May 2025. ISSN 2045-2322. doi: 10.1038/s41598-025-01898-z. URL https://doi.org/10.1038/s41598-025-01898-z.

[2] Surbhi Kapoor, Akashdeep Sharma, and Amandeep Verma. Diving deep into human action recognition in aerial videos: A survey. *Journal of Visual Communication and Image Representation*, 104:104298, 2024. ISSN 1047-3203. doi: https://doi.org/10.1016/j.jvcir.2024.104298. URL https://www.sciencedirect.com/science/article/pii/S1047320324002542.

[3] Oscar D. Lara and Miguel A. Labrador. A survey on human activity recognition using wearable sensors. *IEEE Communications Surveys Tutorials*, 15(3):1192–1209, 2013. doi: 10.1109/SURV.2012.110112.00192.

[4] Wei Wang, Alex Liu, Muhammad Shahzad, Kang Ling, and Sanglu Lu. Device-free human activity recognition using commercial wifi devices. *IEEE Journal on Selected Areas in Communications*, PP:1–1, 03 2017. doi: 10.1109/JSAC.2017.2679658.

[5] Arya Sarkar, Avinandan Banerjee, Pawan Kumar Singh, and Ram Sarkar. 3d human action recognition: Through the eyes of researchers. *Expert Systems with Applications*, 193:116424, 2022. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2021.116424. URL https://www.sciencedirect.com/science/article/pii/S0957417421017115.

[6] Zhengxia Zou, Zhenwei Shi, Yuhong Guo, and Jieping Ye. Object detection in 20 years: A survey. *CoRR*, abs/1905.05055, 2019. URL http://arxiv.org/abs/1905.05055.

[7] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection, 2016. URL https://arxiv.org/abs/1506.02640.

[8] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context, 2015.

[9] Debapriya Maji, Soyeb Nagori, Manu Mathew, and Deepak Poddar. Yolo-pose: Enhancing yolo for multi person pose estimation using object keypoint similarity loss, 2022. URL https://arxiv.org/abs/2204.06806.

[10] Ruopeng Gao, Ji Qi, and Limin Wang. Multiple object tracking as id prediction, 2025. URL https://arxiv.org/abs/2403.16848.

[11] Nir Aharon, Roy Orfaig, and Ben-Zion Bobrovsky. Bot-sort: Robust associations multi-pedestrian tracking, 2022. URL https://arxiv.org/abs/2206.14651.

[12] Yifu Zhang, Peize Sun, Yi Jiang, Dongdong Yu, Fucheng Weng, Zehuan Yuan, Ping Luo, Wenyu Liu, and Xinggang Wang. Bytetrack: Multi-object tracking by associating every detection box, 2022. URL https://arxiv.org/abs/2110.06864.

[13] M I Jordan. Serial order: a parallel distributed processing approach. technical report, june 1985-march 1986. Technical report, California Univ., San Diego, La Jolla (USA). Inst. for Cognitive Science, 05 1986. URL https://www.osti.gov/biblio/6910294.

[14] Robin M. Schmidt. Recurrent neural networks (rnns): A gentle introduction and overview, 2019. URL https://arxiv.org/abs/1912.05911.

[15] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, 1994.

[16] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9:1735–1780, 11 1997. doi: 10.1162/neco.1997.9.8.1735.

[17] Alex Graves and Jürgen Schmidhuber. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural Networks*, 18(5):602–610, 2005. ISSN 0893-6080. doi: https://doi.org/10.1016/j.neunet.2005.06.042. URL https://www.sciencedirect.com/science/article/pii/S0893608005001206. IJCNN 2005.

[18] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.

[19] M. Schuster and K.K. Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, 1997. doi: 10.1109/78.650093.

[20] Pierre Baldi, Søren Brunak, Paolo Frasconi, Gianluca Pollastri, and Giovanni Soda. *Bidirectional Dynamics for Protein Secondary Structure Prediction*, pages 80–104. Springer Berlin Heidelberg, Berlin, Heidelberg, 2001. ISBN 978-3-540-44565-4. doi: 10.1007/3-540-44565-X_5. URL https://doi.org/10.1007/3-540-44565-X_5.

[21] Michael Schuster. *On Supervised Learning from Sequential Data with Applications for Speech Recognition*. PhD thesis, Nara Institute of Science and Technology, 1999.

[22] Filippo Notari. Knife dataset. https://universe.roboflow.com/tesi-c7kla/knife-fwpuw-rrrcw, oct 2025. URL https://universe.roboflow.com/tesi-c7kla/knife-fwpuw-rrrcw.

[23] Miriana Bianculli, Nicola Falcionelli, Paolo Sernani, Selene Tomassini, Paolo Contardo, Mara Lombardi, and Aldo Franco Dragoni. A dataset for automatic violence detection in videos. *Data in Brief*, 33:106587, 2020. doi: 10.1016/j.dib.2020.106587.

[24] Daniel Deniz, Eduardo Ros, Eva M Ortigosa, and Francisco Barranco. Optimized edge-cloud system for activity monitoring using knowledge distillation. *Electronics*, 13(23):4786, 2024.

[25] Mohamed Soliman, Mohamed Hussein Kamal, Mina Abd El-Massih Nashed, Youssef Mohamed Mostafa, Bassel S. Chawky, and Dina Khattab. Violence recognition from videos using deep learning techniques. *2019 Ninth International Conference on Intelligent Computing and Information Systems (ICICIS)*, pages 80–85, 2019. URL https://api.semanticscholar.org/CorpusID:212706293.

[26] Ming Cheng, Kunjing Cai, and Ming Li. Rwf-2000: An open large scale video database for violence detection, 2020. URL https://arxiv.org/abs/1911.05913.

[27] Shachar Kaufman, Saharon Rosset, Claudia Perlich, and Ori Stitelman. Leakage in data mining: Formulation, detection, and avoidance. *ACM Trans. Knowl. Discov. Data*, 6(4), December 2012. ISSN 1556-4681. doi: 10.1145/2382577.2382579. URL https://doi.org/10.1145/2382577.2382579.

[28] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL https://www.tensorflow.org/. Software available from tensorflow.org.

[29] Margherita Grandini, Enrico Bagli, and Giorgio Visani. Metrics for multi-class classification: an overview, 2020. URL https://arxiv.org/abs/2008.05756.

[30] Yutaka Sasaki. The truth of the f-measure. *Teach Tutor Mater*, 01 2007.

[31] Google Developers. Machine learning crash course: Classification: Roc curve and auc. https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc, 2024. Accessed: 2024-05-20.

[32] Ondrej Miksik, Daniel Munoz, J. Andrew Bagnell, and Martial Hebert. Efficient temporal consistency for streaming video scene analysis. In *2013 IEEE International Conference on Robotics and Automation*, pages 133–139, 2013. doi: 10.1109/ICRA.2013.6630567.

[33] Ultralytics. Coco dataset - key features. https://docs.ultralytics.com/datasets/detect/coco/#key-features, 2024. Accessed: 2026-01-27.