BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC CMC

CÔNG TRÌNH THAM DỰ
HỘI NGHỊ SINH VIÊN NGHIÊN CỨU KHOA HỌC CẤP TRƯỜNG

NĂM HỌC 2024 – 2025

Tên công trình: Nghiên cứu và xây dựng mô-đun nhận diện hành vi của sinh viên trong giảng đường

Nhóm sinh viên thực hiện:

1. Đinh Nhật Thành, Mã SV BIT220143, Lớp: AI1

Đơn vị: Công Nghệ Thông Tin và Truyền Thông

Người hướng dẫn: Ths. Nguyễn Khánh Sơn

Hà Nội, 2025

TÓM TẮT CÔNG TRÌNH

Trong bối cảnh các hành vi bạo lực trong môi trường học đường ngày càng gia tăng, việc ứng dụng trí tuệ nhân tạo (AI) để giám sát và phát hiện hành vi nguy hiểm một cách kịp thời đang trở thành nhu cầu cấp thiết. Trường Đại học CMC là đơn vị đi đầu trong việc tích hợp công nghệ vào giáo dục để đảm bảo an toàn, hiệu quả và môi trường học tập tích cực.

Công trình này nhằm xây dựng một hệ thống nhận diện hành vi bạo lực (đấm, đá, đứng) trong lớp học bằng cách sử dụng dữ liệu khung xương người và mô hình học sâu (LSTM, GRU, DNN). Hệ thống chỉ sử dụng 33 điểm keypoint từ MediaPipe để trích xuất đặc trưng không gian và chuyển động học, từ đó giảm thiểu chi phí tính toán mà vẫn duy trì độ chính xác cao.

Nội dung nghiên cứu gồm ba phần:

1. **Đặt vấn đề** – trình bày bối cảnh, mục tiêu và tính cấp thiết của đề tài.
2. **Giải quyết vấn đề** – mô tả cách xây dựng dataset, trích xuất đặc trưng là tọa độ các điểm theo format khung xương của Mediapipe, huấn luyện và đánh giá mô hình.
3. **Kết luận** – đánh giá hiệu quả, rút ra hướng phát triển mở rộng như phát hiện nhiều người và nhiều hành vi phức tạp hơn.

Kết quả cho thấy hệ thống đạt độ chính xác cao, tuy nhiên hoạt động không ổn định cho thấy nhiều dấu hiệu data imbalance và overfitting, tuy nhiên cũng góp phần làm giàu bộ dữ liệu cho các công trình cùng tên về sau.

Chúng em cam đoan công trình nghiên cứu này là do bản thân thực hiện dưới sự hướng dẫn của giảng viên phụ trách

Giảng viên hướng dẫn Sinh viên thực hiện

MỤC LỤC

BẢNG PHÂN CÔNG NHIỆM VỤ NHÓM

Số	Tên Thành Viên	Nhiệm Vụ Được Giao
1	Đinh Nhật Thành	Thiết kế workflow hệ thống, Tìm và Tiền xử lý dữ liệu, phát triển mô hình, huấn luyện và đánh giá, phân tích hiệu suất, viết báo cáo.

DANH SÁCH CÁC TỪ VIẾT TẮT

Từ Viết Tắt	Ý Nghĩa
CNN	Convolutional Neural Network
DNN	Deep Neural Network
FPS	Frames Per Second
GRU	Gated Recurrent Unit
HOG	Histogram of Oriented Gradients
IoT	Internet of Things
LSTM	Long Short-Term Memory
ML	Machine Learning
MP-TFSS	MediaPipe-TensorFlow Skeleton Sequence
RNN	Recurrent Neural Network
ViT	Vision Transformer

1. Giới thiệu

Vấn đề được đề cập trong báo cáo này là phát hiện và phân loại hành vi bạo lực và không bạo lực một cách tự động trong môi trường học đường. Bạo lực học đường gây ra những mối đe dọa nghiêm trọng đến sự an toàn của học sinh và làm gián đoạn môi trường giáo dục. Các hệ thống giám sát truyền thống phụ thuộc phần lớn vào người điều khiển, khiến cho việc theo dõi liên tục và đáng tin cậy trở nên khó khăn và dễ xảy ra sai sót. Để giải quyết vấn đề này, em đã phát triển một hệ thống nhận dạng hành động dựa trên bộ xương sử dụng MediaPipe để trích xuất tư thế và mô hình học sâu dùng TensorFlow để phân loại hành động 3 cơ động là Đấm, Đá và Đứng trước khi mở rộng lên các hành động phức tạp hơn. Lựa chọn này ưu tiên hiệu quả tính toán, khả năng xử lý thời gian thực, dễ triển khai trong hạ tầng hiện có và khả năng mở rộng đến nhiều môi trường học đường khác nhau.

2. Tổng quan

2.1. Bối cảnh lịch sử

Nhận dạng hành động tự động đã là một lĩnh vực phát triển từ đầu những năm 2000, chủ yếu xuất phát từ nhu cầu giám sát an ninh tại các không gian công cộng như sân bay, nhà ga và trường học. Ban đầu, các phương pháp đơn giản hơn được sử dụng, trong đó cần chọn và trích xuất thủ công các đặc trưng quan trọng trực tiếp từ khung hình video. Những phương pháp này bao gồm các kỹ thuật như Histogram of Oriented Gradients (HOG). Với sự phát triển gần đây của năng lực tính toán, các phương pháp học sâu như Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), Long Short-Term Memory (LSTM) networks và Vision Transformers (ViTs) đã trở nên phổ biến nhờ khả năng tự động học các mẫu phức tạp từ dữ liệu.

2.2. Các giải pháp hiện có

Các phương pháp dựa trên CNN: Các cách tiếp cận như Two-Stream CNN phân tích dữ liệu không gian và thời gian từ video, giúp nhận diện hiệu quả các hành động phức tạp. Tuy nhiên, những phương pháp này đòi hỏi nhiều tài nguyên tính toán và thường không phù hợp để sử dụng trong thời gian thực.

LSTM và các mô hình chuỗi: Những phương pháp này phân tích hiệu quả dữ liệu tuần tự và động học theo thời gian, nhưng thường phụ thuộc vào quá trình tiền xử lý chính xác và chất lượng dữ liệu đầu vào.

Các kỹ thuật ước lượng tư thế: Những phương pháp như OpenPose và MediaPipe giúp trích xuất thông tin bộ xương người một cách hiệu quả, giảm đáng kể nhu cầu tính toán so với các phương pháp dựa trên video. Tuy vậy, để duy trì hiệu suất, việc trích xuất tư thế chính xác và ổn định là rất quan trọng.

2.3. Hạn chế và hướng tiếp cận của em

Các phương pháp trước đây nhìn chung tập trung vào xử lý video thô, dẫn đến chi phí tính toán cao và khó triển khai trong thời gian thực. Bên cạnh đó, các hệ thống dựa trên hình ảnh thường gặp khó khăn với vật cản, góc quay đa dạng và điều kiện ánh sáng không ổn định.

Để giải quyết những vấn đề này, em đề xuất phương pháp MediaPipe-TensorFlow Skeleton Sequence (MP-TFSS), kết hợp việc trích xuất dữ liệu bộ xương hiệu quả với phân loại tuần tự bằng mô hình học sâu. Cụ thể, em sử dụng MediaPipe để trích xuất tư thế và mô hình học sâu dùng TensorFlow, trong đó mạng Deep Neural Network (DNN) thực hiện phân loại hành động từ dữ liệu tuần tự. Hướng tiếp cận này được thiết kế nhằm tăng hiệu quả tính toán (nhắm đến trên 15 FPS) và duy trì độ chính xác phân loại cao (mục tiêu 65–75%), đảm bảo một giải pháp thực tiễn, hiệu quả và có khả năng mở rộng phù hợp để triển khai trong môi trường học đường thực tế.

3. Phương pháp

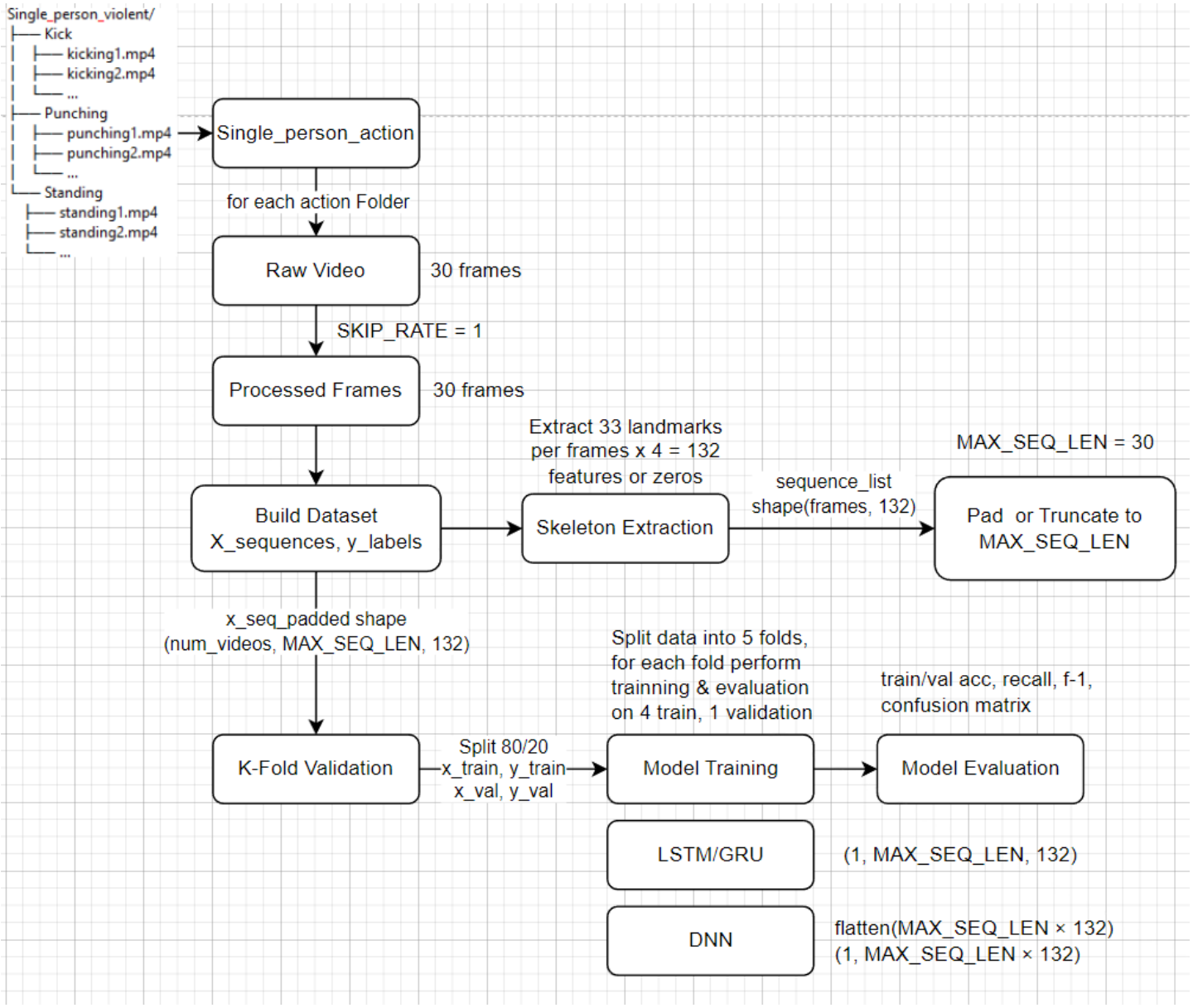
3.1. Ý tưởng chính

Phương pháp mà em đề xuất dựa trên ý tưởng sử dụng dữ liệu bộ xương, được trích xuất bằng MediaPipe, để nắm bắt các chuyển động quan trọng của cơ thể người. Các chuỗi bộ xương này mang đến một biểu diễn đơn giản nhưng vẫn chứa nhiều thông tin về hành động. Bằng cách xử lý dữ liệu này với mô hình học sâu dùng TensorFlow theo dạng chuỗi (DNN), em phân loại hiệu quả các hành động thành bạo lực hoặc không bạo lực [1] [3]. Cách tiếp cận này giúp giảm đáng kể yêu cầu tính toán so với việc phân tích trực tiếp các khung hình video thô, từ đó tăng tốc độ xử lý và cho phép phát hiện hành động theo thời gian thực.

3.2. Các bước triển khai

Tiền xử lý dữ liệu là một giai đoạn quan trọng để phân loại hành động chính xác và hiệu quả. Trong phần này, em điều chỉnh và mở rộng bộ dữ liệu từ bộ dữ liệu Tiền xử lý dữ liệu là một giai đoạn quan trọng để phân loại hành động chính xác và hiệu quả. Trong phần này, em điều chỉnh và mở rộng bộ dữ liệu từ bộ dữ liệu gốc Single Person Violent Activity [2] nhằm tăng tính đa dạng và cải thiện quá trình huấn luyện mô hình. Cụ thể, bộ dữ liệu tùy chỉnh của em bao gồm:

- 30 video cho hành động *đá (Kicking)*,
- 30 video cho hành động *đấm (Punching)*,
- 30 video cho hành động *đứng (Standing)*.



Em thiết lập *MAX_SEQ_LEN* = 5 để chuẩn hóa số lượng khung hình tối đa trong mỗi chuỗi và *SKIP_RATE* = 1 để đảm bảo rằng mỗi khung hình đều được sử dụng mà không bị bỏ qua. Cách tiếp cận này giúp cân bằng giữa nhu cầu thông tin thời gian đủ dày và giới hạn về hiệu quả tính toán. Quy trình tiền xử lý chi tiết được mô tả bên dưới:

Bước 1: Trích xuất frame từ video

Mỗi video trong bộ dữ liệu được đọc và xử lý một cách có hệ thống theo từng frame. Để tối ưu hiệu suất, các frame được trích xuất theo khoảng thời gian cố định được kiểm

soát bởi *SKIP_RATE*, giúp cân bằng giữa hiệu quả tính toán và chất lượng dữ liệu hành động thu được.

Bước 2: Trích xuất dữ liệu bộ xương sử dụng MediaPipe

Với mỗi frame đã được trích xuất, em sử dụng MediaPipe Pose để phát hiện các điểm mốc trên cơ thể người, cung cấp 33 điểm chính cho mỗi người. Mỗi điểm được biểu diễn bằng bốn giá trị: x, y, z, và visibility. Các vector có 132 chiều (33 điểm × 4 giá trị mỗi điểm) này chứa đầy đủ thông tin về tư thế cần thiết cho nhận dạng hành động.

Bước 3: Xử lý điểm mốc bị thiếu:

Đôi khi MediaPipe có thể không phát hiện được điểm mốc do vật cản, điều kiện ánh sáng yếu hoặc phông nền phức tạp. Để xử lý tình huống này, các frame thiếu điểm mốc sẽ được biểu diễn bằng vector toàn số 0, đảm bảo mỗi chuỗi vẫn giữ nguyên số chiều một cách nhất quán.

Bước 4: Chuẩn hóa chuỗi dữ liệu

Các video có độ dài khác nhau nên dẫn đến số lượng frame không đồng đều giữa các chuỗi. Để xử lý sự khác biệt này, em chuẩn hóa các chuỗi về cùng độ dài, được xác định bởi *MAX_SEQ_LEN*. Những chuỗi ngắn hơn sẽ được đệm thêm bằng số 0, còn chuỗi dài hơn sẽ bị cắt bớt, nhằm đảm bảo đầu vào cho mô hình Deep Neural Network có kích thước đồng nhất.

Bước 5: Chia bộ dữ liệu:

Các chuỗi dữ liệu sau khi xử lý được chia thành tập huấn luyện và tập kiểm tra bằng phương pháp lấy mẫu phân tầng, nhằm duy trì phân bố đồng đều giữa các loại hành động. Bước này đảm bảo quá trình đánh giá mô hình phản ánh đúng hiệu suất trên toàn bộ các loại hành động. Phương pháp tiền xử lý chi tiết này đảm bảo dữ liệu đầu vào có chất lượng cao, từ đó cải thiện đáng kể độ chính xác và độ tin cậy của mô hình phân loại phía sau.

3.3. Ứng dụng mô hình

Các chuỗi dữ liệu bộ xương sau khi xử lý sẽ được đưa vào mô hình học sâu được xây dựng bằng TensorFlow. Kiến trúc mô hình này đặc biệt phù hợp với tính tuần tự của dữ liệu bộ xương. Các chuỗi bộ xương thể hiện hành động theo chiều thời gian, nên rất lý tưởng cho các phương pháp mô hình hóa theo chuỗi như Long Short-Term Memory (LSTM), Gated Recurrent Units (GRU), và các mô hình Deep Neural Network (DNN) nói chung.

Lý do lựa chọn LSTM, GRU và DNN:

- **Xử lý quan hệ phụ thuộc theo thời gian:** Các mô hình LSTM và GRU có khả năng nắm bắt tốt các mối quan hệ phụ thuộc theo thời gian và các mẫu phức tạp trong dữ liệu tuần tự, điều này rất quan trọng để nhận diện chính xác các hành động mang tính động.

- **Ghi nhớ thông tin ngữ cảnh:** LSTM và GRU có khả năng duy trì trạng thái ghi nhớ, cho phép biểu diễn và dự đoán hiệu quả các chuỗi hành động và chuyển động liên tục theo thời gian.

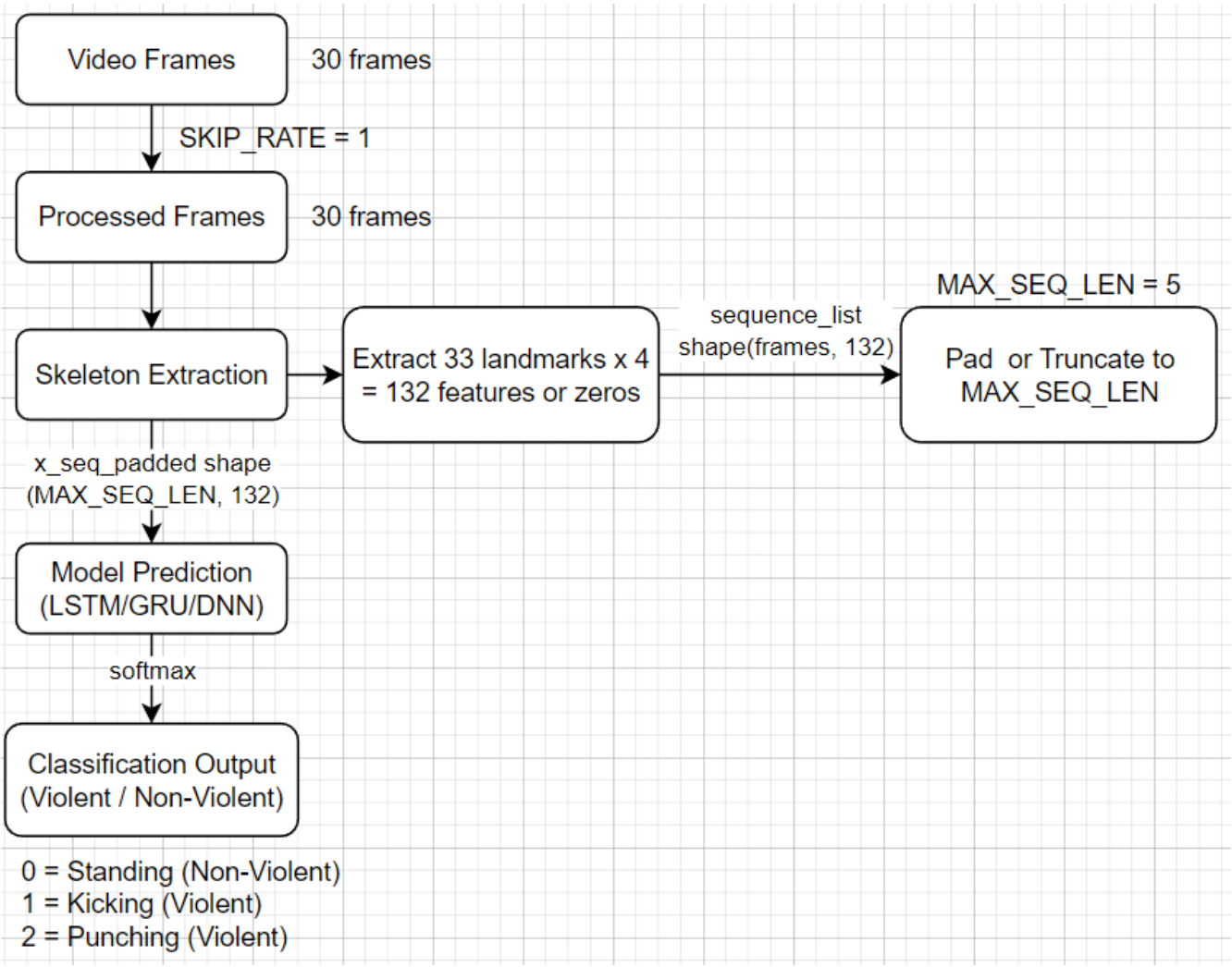
- **Giảm thiểu vấn đề mất dần gradient:** Cả LSTM và GRU đều được thiết kế đặc biệt để khắc phục vấn đề mất dần gradient thường gặp trong các mô hình RNN truyền thống, giúp quá trình huấn luyện ổn định và hiệu quả hơn.

- **Tính linh hoạt và đơn giản của DNN:** Các mô hình DNN dạng feed-forward đơn giản phù hợp làm mô hình baseline, dù không có cơ chế xử lý thông tin theo thời gian, đôi khi vẫn mang lại hiệu quả do tốc độ huấn luyện nhanh hơn,

yêu cầu tính toán thấp hơn, và hoạt động tốt trên các chuỗi ngắn hoặc tập dữ liệu có độ phức tạp thời gian thấp.

4. Báo cáo chi tiết về quá trình xử lý chuỗi trong mô hình

Phần này cung cấp phân tích chi tiết về cách mỗi mô hình (LSTM, GRU và DNN) xử lý dữ liệu chuỗi bộ xương như đã được triển khai trong mã nguồn:



4.1. Long Short-Term Memory (LSTM)

Layer (type)	Output Shape	Param #
lstm (LSTM)	(None, 5, 64)	50,432
lstm_1 (LSTM)	(None, 32)	12,416
dense (Dense)	(None, 3)	99

Bước 1: Nhập chuỗi dữ liệu

Các chuỗi khung xương với kích thước *MAX_SEQ_LEN* × 132 được đưa vào lớp LSTM. Mỗi timestep đại diện cho các tọa độ bộ xương được trích xuất bằng MediaPipe.

Bước 2: Xử lý theo thời gian

Lớp LSTM xử lý dữ liệu chuỗi theo từng timestep, ghi nhớ quan hệ thời gian bằng cách duy trì thông tin trạng thái bên trong. Điều này cho phép mô hình ghi nhớ các frame trước đó khi dự đoán hoặc phân tích frame hiện tại, từ đó mô hình hóa hành động theo thời gian một cách hiệu quả.

Bước 3: Trích xuất đặc trưng và các lớp Dense

Các đầu ra từ lớp LSTM, đại diện cho các mẫu theo thời gian, được đưa vào các lớp dense (liên kết đầy đủ). Các lớp này tiếp tục phân tích sâu hơn các đặc trưng đã được LSTM trích xuất để tinh chỉnh kết quả dự đoán.

Bước 4: Lớp đầu ra

Cuối cùng, thông tin đã xử lý được sử dụng bởi lớp dense đầu ra để phân loại chuỗi hành động thành bạo lực hoặc không bạo lực.

4.2. Gated Recurrent Unit (GRU)

Layer (type)	Output Shape	Param #
gru (GRU)	(None, 5, 64)	38,016
gru_1 (GRU)	(None, 32)	9,408
dense_6 (Dense)	(None, 3)	99

Bước 1: Nhập chuỗi dữ liệu

Tương tự như LSTM, GRU nhận đầu vào là các chuỗi có kích thước *MAX_SEQ_LEN* × 132.

Bước 2: Xử lý theo thời gian hiệu quả

GRU xử lý dữ liệu chuỗi một cách hiệu quả nhờ cơ chế cổng đơn giản hơn so với LSTM. Nó duy trì và cập nhật trạng thái ẩn để nắm bắt quan hệ phụ thuộc theo thời gian, đồng thời tiêu tốn ít tài nguyên tính toán hơn.

Bước 3: Các lớp Dense để tinh chỉnh đặc trưng

Đầu ra từ các lớp GRU được đưa vào các lớp dense tiếp theo để xử lý sâu hơn. Các lớp này thực hiện tính toán bổ sung nhằm giúp mô hình phân biệt các mẫu phức tạp trong dữ liệu.

Bước 4: Phân loại cuối cùng

Một lớp dense cuối cùng thực hiện việc phân loại các chuỗi đã xử lý thành các lớp đã định (bạo lực hoặc không bạo lực).

4.3. Deep Neural Network (DNN)

Layer (type)	Output Shape	Param #
dense_27 (Dense)	(None, 128)	84,608
dropout_10 (Dropout)	(None, 128)	0
dense_28 (Dense)	(None, 64)	8,256
dropout_11 (Dropout)	(None, 64)	0
dense_29 (Dense)	(None, 3)	195

Bước 1: Làm phẳng chuỗi dữ liệu

Không giống như LSTM và GRU, DNN yêu cầu dữ liệu đầu vào được làm phẳng hoặc tổng hợp thành một vector một chiều. Bước này giúp giảm độ phức tạp thời gian, khiến mô hình hoạt động nhanh hơn nhưng có thể kém nhạy với các quan hệ phụ thuộc dài hạn theo thời gian.

Bước 2: Xử lý qua các lớp Dense

Các đặc trưng chuỗi đã làm phẳng được đưa qua nhiều lớp dense, cho phép DNN phát hiện các mẫu không gian và mối quan hệ phức tạp mà không cần mô hình hóa rõ ràng các yếu tố thời gian.

Bước 3: Kích hoạt và ánh xạ phi tuyến

Các lớp trung gian sử dụng các hàm kích hoạt như ReLU để đưa vào tính phi tuyến, điều này rất quan trọng trong việc biểu diễn chính xác các mẫu dữ liệu phức tạp.

Bước 4 Phân loại đầu ra:

Cuối cùng, các đặc trưng đã được xử lý từ các lớp dense được đưa vào lớp đầu ra, nơi thực hiện phân loại chuỗi thành hành động bạo lực hoặc không bạo lực dựa trên các mối quan hệ không gian đã học.

4.4. Tinh chỉnh và đánh giá mô hình

Quá trình đánh giá mô hình kết hợp giữa phương pháp chia tập huấn luyện - kiểm tra truyền thống với chiến lược kiểm định chéo K-fold nhằm đảm bảo đo lường hiệu suất một cách chính xác và tổng quát.

Quy trình huấn luyện: Mô hình được huấn luyện bằng TensorFlow, sử dụng các callback như early stopping và lưu checkpoint. Early stopping theo dõi hàm mất mát trên tập xác thực để tránh overfitting, trong khi checkpoint lưu lại trọng số tốt nhất dựa trên độ chính xác xác thực. Quá trình huấn luyện sử dụng các tham số cố định: *EPOCH* = 30, *BATCH_SIZE* = 8 và *PATIENCE* = 10.

Các chỉ số đánh giá được sử dụng:

- **Accuracy (Độ chính xác):** Tỷ lệ dự đoán đúng trên tổng số dự đoán.
- **Precision (Độ chính xác dương):** Tỷ lệ dự đoán đúng trong số các trường hợp được dự đoán là dương tính.
- **Recall (Khả năng phát hiện):** Tỷ lệ trường hợp dương tính thực sự được dự đoán đúng.
- **F1-score:** Trung bình điều hòa giữa precision và recall.

Triển khai kiểm định chéo K-Fold: Để tăng độ tin cậy trong đánh giá mô hình, mã nguồn sử dụng kiểm định chéo 5-fold:

- Bộ dữ liệu được chia thành 5 phần bằng nhau bằng KFold từ thư viện sklearn.

- Ở mỗi fold:
  - Một mô hình mới được khởi tạo bằng hàm tùy chỉnh model_fn().
  - Tiến hành huấn luyện trên 4 fold, fold còn lại dùng để xác thực.
  - Trọng số tốt nhất được lưu bằng ModelCheckpoint.
  - Sau khi huấn luyện xong, trọng số tốt nhất sẽ được tải lại để đánh giá.
  - Tiến hành dự đoán và tính toán các chỉ số đánh giá như ma trận nhầm lẫn và báo cáo phân loại.
- Kết quả từ cả 5 fold được tổng hợp và tính trung bình để tạo thành đánh giá cuối cùng toàn diện.

Phương pháp này đảm bảo mô hình được huấn luyện và xác thực trên toàn bộ các phần của bộ dữ liệu, giúp giảm thiên lệch và cải thiện khả năng tổng quát hóa. Việc sử dụng EarlyStopping và ModelCheckpoint trong từng fold giúp tăng độ ổn định của mô hình và tránh hiện tượng overfitting.

5. Thí nghiệm và đánh giá kết quả K-Fold

Phần này trình bày kết quả từ các thí nghiệm sử dụng ba mô hình khác nhau—LSTM, GRU và DNN—kèm theo các đánh giá trực quan.

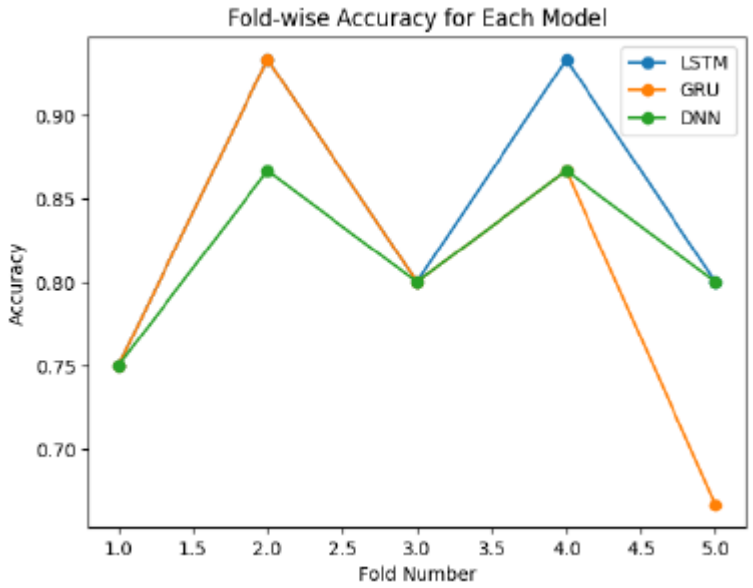


Figure 4: So sánh độ chính xác theo từng fold giữa các mô hình

Đánh giá: Tất cả các mô hình đều đạt độ chính xác cao nhất ở Fold 2, do đó Fold 2 sẽ được sử dụng cho phần trực quan hóa. GRU thể hiện mức độ dao động cao hơn giữa các fold, trong khi LSTM duy trì độ chính xác cao và ổn định ở nhiều fold.

5.1. Đường cong huấn luyện

5.1.1. LSTM - Fold 2

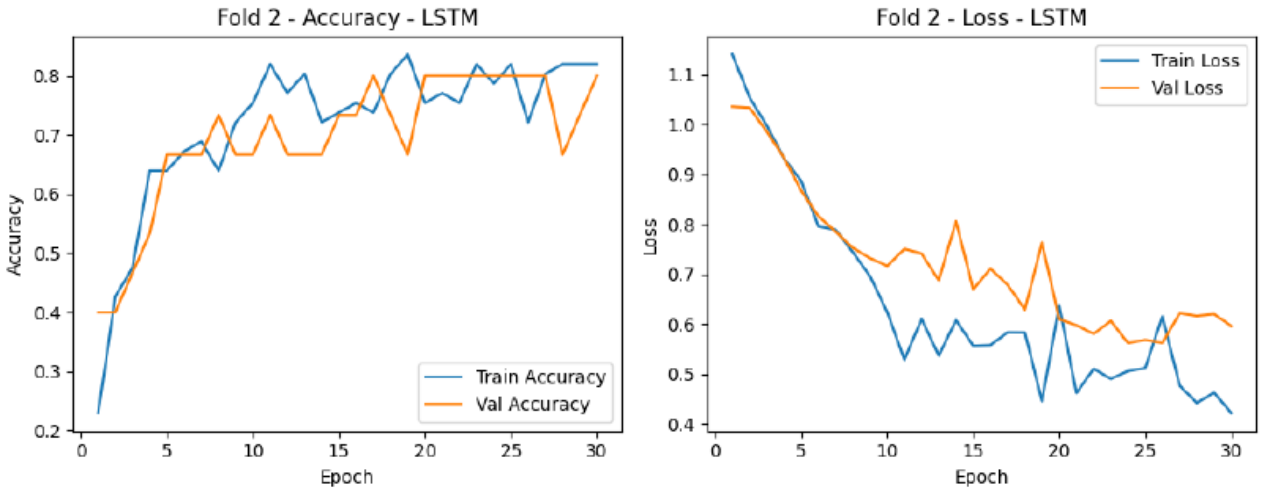


Figure 5: Độ chính xác và hàm mất mát của LSTM trên Fold 2 qua các epoch

Đánh giá: Mô hình LSTM cho thấy sự cải thiện ổn định với độ chính xác huấn luyện cuối cùng đạt 0.8197 và độ chính xác xác thực là 0.8000. Các đường cong về độ chính xác và mất mát cho thấy khả năng học ổn định. Mặc dù có một vài dao động nhẹ ở phần cuối của đường cong mất mát xác thực, điều này chỉ ra hiện tượng overfitting nhẹ nhưng khả năng tổng quát vẫn ở mức chấp nhận được.

5.1.2. GRU - Fold 2

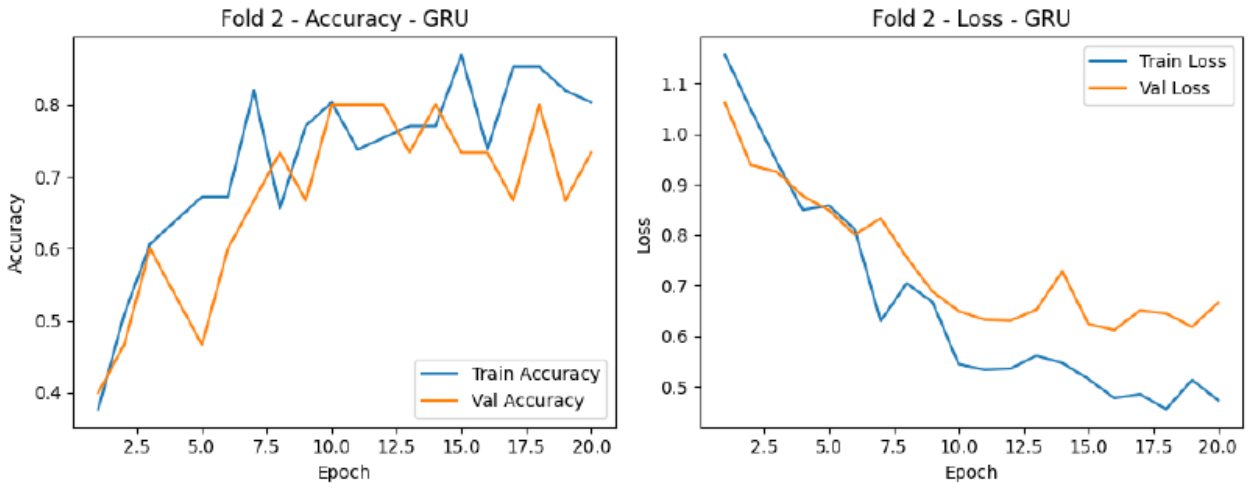


Figure 6: Độ chính xác và hàm mất mát của GRU trên Fold 2 qua các epoch

Đánh giá: GRU đạt độ chính xác huấn luyện cuối cùng là 0.8033 và độ chính xác xác thực là 0.7333. Mô hình học khá hiệu quả trong các epoch đầu, tuy nhiên đường cong xác thực dao động nhiều hơn, cho thấy độ nhạy với dữ liệu huấn luyện và khả năng cần thêm điều chỉnh hoặc tăng cường kỹ thuật regularization.

5.1.3. DNN - Fold 2

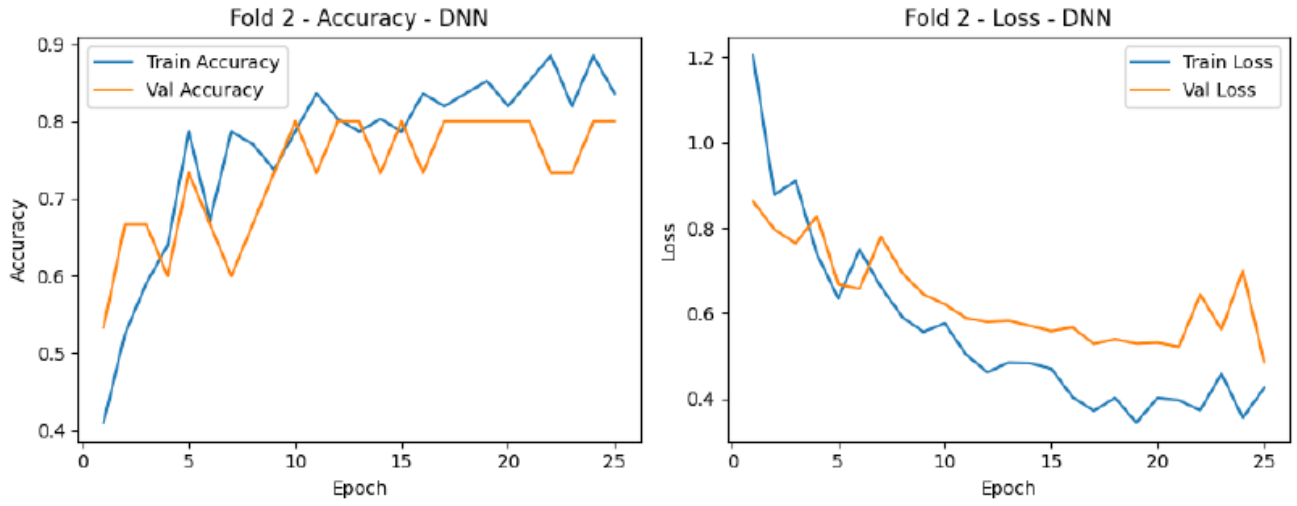


Figure 7: Độ chính xác và hàm mất mát của DNN trên Fold 2 qua các epoch

Đánh giá: DNN thể hiện sự hội tụ tốt nhất trong số các mô hình, với độ chính xác huấn luyện đạt 0.8361 và độ chính xác xác thực là 0.8000. Độ chênh lệch giữa tập huấn luyện và xác thực là tối thiểu, các đường cong mượt mà, phản ánh khả năng tổng quát tốt và hiện tượng overfitting gần như không đáng kể.

5.2. Độ chính xác theo fold và so sánh thống kê

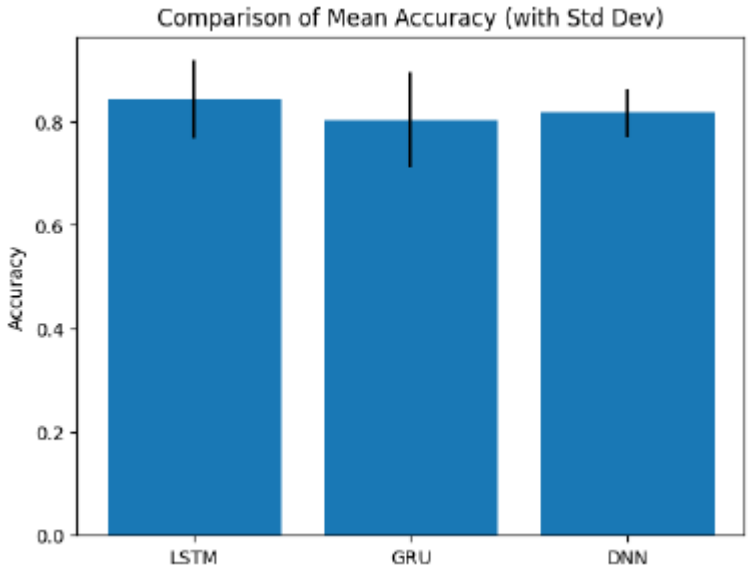


Figure 8: So sánh độ chính xác trung bình với độ lệch chuẩn

Đánh giá: LSTM có độ chính xác trung bình cao nhất (0.8433), tiếp theo là DNN (0.8167). GRU cũng cho kết quả tốt nhưng có độ lệch chuẩn cao hơn, cho thấy hiệu suất không ổn định giữa các fold.

5.3. Ma trận nhầm lẫn và phân tích theo lớp

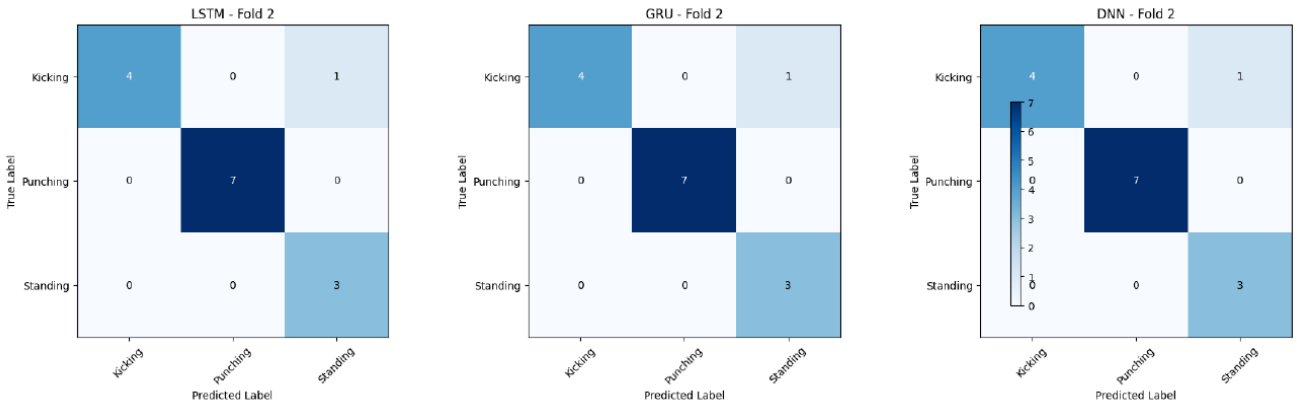


Figure 9: Ma trận nhầm lẫn của LSTM, GRU và DNN trên Fold 2

Đánh giá: Cả ba mô hình đều phân loại tốt nhất ở lớp ”Punching”. Mô hình DNN phân loại chính xác toàn bộ các mẫu thuộc lớp ”Kicking” và ”Punching”, nhưng kém hơn một chút khi nhận diện ”Standing”. LSTM và GRU đều mắc lỗi phân loại một mẫu ”Kicking” và một mẫu ”Standing”.

5.4. Đánh giá Demo

Demo được thiết lập trong môi trường nội bộ, ghi lại video thời gian thực với tốc độ trung bình hơn 15 FPS. Đã tiến hành thử nghiệm nhiều kịch bản bao gồm hành vi bạo lực (ví dụ: pushing, punching) và không bạo lực (ví dụ: walking) nhằm đánh giá độ chính xác và tốc độ phát hiện trong điều kiện thực tế.

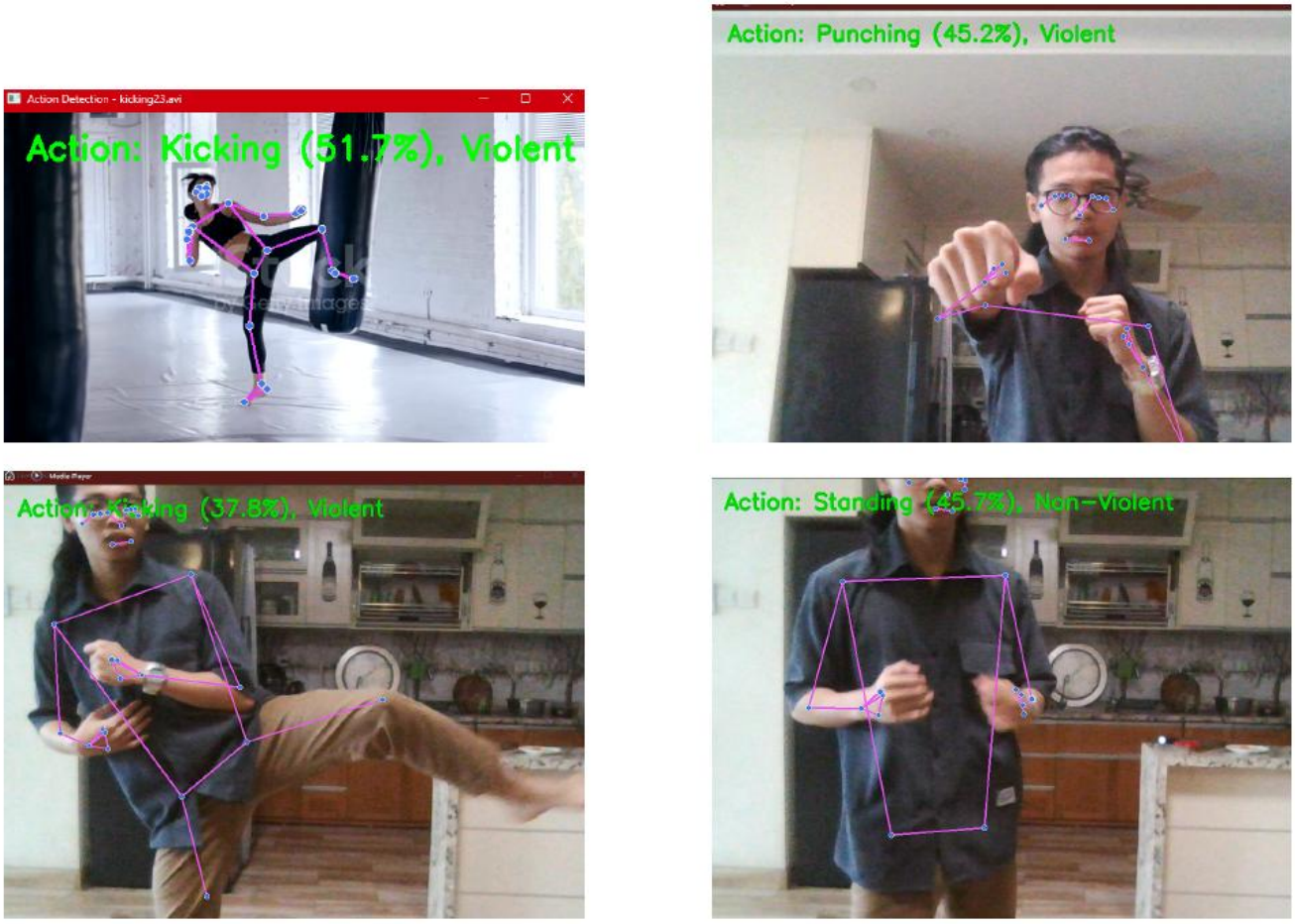


Figure 10: Ảnh chụp từ Demo thời gian thực, thể hiện kết quả phát hiện hành vi bạo lực (hàng trên) và không bạo lực (góc dưới bên phải).

Trong quá trình Demo:

- Hệ thống duy trì tốc độ trung bình 8 FPS.

- Phát hiện chính xác các hành vi bạo lực trong khoảng 4 trên 6 các trường hợp quan sát khi quay trực tiếp được nhưng sẽ mất từ 1-3s để mô hình nhận diện đúng.

- Việc che khuất một phần đôi khi dẫn đến phân loại sai hành vi không bạo lực, cho thấy sự cần thiết phải cải thiện khả năng theo dõi tư thế trong điều kiện ánh sáng đa dạng.

Tổng thể, Demo cho thấy quy trình phát hiện dựa trên khung xương hoạt động hiệu quả trong điều kiện thực tế và có thể làm nền tảng cho các hệ thống mở rộng hỗ trợ nhiều người trong tương lai.

6. Kết luận và hướng phát triển

Báo cáo này trình bày hệ thống **phát hiện hành vi bạo lực và không bạo lực** trong môi trường học đường, sử dụng MediaPipe Pose để trích xuất dữ liệu khung xương và các mô hình **LSTM**, **GRU** và **DNN** để phân loại. Hướng tiếp cận này giúp giảm đáng kể chi phí tính toán nhờ tập trung vào thông tin tư thế thay vì dữ liệu video thô, đồng thời kết quả thực nghiệm cho thấy độ chính xác cao qua nhiều chiến lược huấn luyện và xác thực. So với các phương pháp trước đây dựa vào dữ liệu điểm ảnh thô, phương pháp này mang lại hiệu suất thời gian thực tốt hơn trong khi vẫn duy trì độ chính xác phân loại ổn định.

Phát hiện và đóng góp chính: Chuẩn hóa và mở rộng bộ dữ liệu hành vi gồm xây dựng tập dữ liệu gồm ba lớp hành động (đấm, đá, đứng) từ Single Person Violent Activity, với 30 video cho mỗi lớp, đảm bảo phân bố đồng đều và độ đa dạng chuyển động.

Hướng phát triển trong tương lai:

- **Phát hiện nhiều người:** Mục tiêu tiếp theo là mở rộng từ nhận dạng hành động đơn lẻ sang xử lý nhiều người cùng lúc trong môi trường học đường. Điều này sẽ bao gồm tích hợp các module phát hiện người có khả năng theo dõi nhiều học sinh đồng thời.

- **Đa dạng hóa bộ dữ liệu:** Dự định mở rộng bộ dữ liệu bằng cách thu thập thêm nhiều video thể hiện các hành vi bạo lực đa dạng (ví dụ: shoving, xô xát nhóm) cũng như các hoạt động hàng ngày không bạo lực. Một bộ dữ liệu lớn và đa dạng hơn sẽ giúp mô hình tổng quát hóa tốt hơn.

- **Cải tiến mô hình:** Dự kiến nghiên cứu thêm các mô hình sử dụng cơ chế attention và tinh chỉnh các mạng hiện có nhằm nâng cao độ chính xác và rút ngắn thời gian huấn luyện. Các kỹ thuật như điều chỉnh siêu tham số, tăng cường dữ liệu và học chuyển tiếp cũng có thể được áp dụng để tối ưu hiệu suất.

Tổng thể, các cải tiến này sẽ đưa hệ thống trở thành một giải pháp toàn diện hơn, có khả năng hoạt động thời gian thực để phát hiện và ứng phó với các hành vi bạo lực trong môi trường học đường.

7. Phụ Lục

Code Block 1: Trích xuất chuỗi pose từ video

```
def get_pose_sequence_from_video(video_path, skip_rate=5,
visualize=False):
      """
      Đọc video, chuẩn hóa khung hình, phát hiện pose,
      trả về mảng (num_frames, 132) gồm (x,y,z,visibility)*33 điểm.
      """
      cap = cv.VideoCapture(video_path)
      sequence =

      with mp_pose.Pose(min_detection_confidence=0.5,
                        min_tracking_confidence=0.5) as model:
      idx = 0
      while True:
            ret, frame = cap.read()
            if not ret: break

            frame = ensure_vertical_orientation(frame)
            frame = resize_frame(frame)

            if idx % skip_rate == 0:
                  rgb = cv.cvtColor(frame, cv.COLOR_BGR2RGB)
                  res = model.process(rgb)

                  if res.pose_landmarks:
                  keypoints =
                  for lm in res.pose_landmarks.landmark:
                        keypoints += lm.x, lm.y, lm.z, lm.visibility
                  else:
                  keypoints = 0*132
                  sequence.append(keypoints)

                  if visualize:
                  cv.imshow("Pose", frame)
                  if cv.waitKey(10) & 0xFF == ord('q'):
                        break
            idx += 1
```

```
    cap.release()
    return np.array(sequence)
```

Mục đích: Chuyển từng khung thành tập vector pose để huấn luyện mạng.

Tham số

1. video_path (str): đường dẫn file video
2. skip_rate (int): chỉ lấy mỗi skip_rate khung
3. visualize (bool): có show video khi chạy không

Đầu ra:

- np.ndarray shape (T, 132) với T = số khung đã sampling

Các bước thực hiện

1. Mở video bằng cv.VideoCapture
2. Với mỗi khung: xoay, resize, chuyển sang RGB
3. Dùng model.process lấy pose_landmarks
4. Nếu có landmarks, nối list x,y,z,visibility của 33 điểm → 132 chiều, Ngược lại điền 132 số 0
5. Append vào sequence
6. (Nếu visualize) hiển thị khung, bấm ‘q’ để thoát
7. Trả về mảng Null

Code Block 2: Pad hoặc cắt chuỗi về độ dài cố định

```
def pad_or_truncate_sequence(seq, max_len=MAX_SEQ_LEN):
    """
    Nếu seq dài > max_len → cắt; nếu ngắn → pad zeros về max_len.
    """
    length, num_feat = seq.shape
    if length > max_len:
    return seq[:max_len, :]
    else:
    padded = np.zeros((max_len, num_feat))
    padded[:length, :] = seq
    return padded
```

Mục đích: Đồng nhất mọi sequence về độ dài MAX_SEQ_LEN.

Biến đầu vào

- seq: mảng (T,132)

Đầu ra

- Mảng (MAX_SEQ_LEN,132)

Các bước:

1. So sánh length và max_len
2. Cắt hoặc pad chuỗi

Tài Liệu Tham Khảo

[1] T. D. Nguyen. (2021). Student Behavior Recognition System for the Classroom Environment Based on Skeleton Pose Estimation and Person Detection. Online. Available:https://www.researchgate.net/publication/353746430_Student_Behavior_Recognition_System_for_the_Classroom_Environment_Based_on_Skeleton_Pose_Estimation_and_Person_Detection

[2] nizdarlaila. Pose Estimation Using MediaPipe. Online. Available:https://www.kaggle.com/code/nizdarlaila/pose-estimation-using-mediapipe Accessed: April 7, 2025.

[3] nam157. Human Activity Recognition. Online. Available: https://nam157.github.io/human_activity_recognition-/ Accessed: April 7, 2025.