UNIVERSITY OF SCIENCE AND TECHNOLOGY OF HANOI

DEPARTMENT OF INFORMATION AND COMMUNICATION TECHNOLOGY

Research and Development

GROUP PROJECT REPORT

Le Viet Hung	23BI14182
Pham Gia Bao	23BI14061
Dang Dinh Hoa	23BI14169
Tran Gia Khanh	23BI14218
Nguyen Gia Nam	23BI14328
Nguyen Duc Thanh	23BI14406

Supervisor: Nghiem Thi Phuong

Title:

Combination of computer vision and AI chatbot for automatic pickleball training

Hanoi, January 2026

Contents

1 Introduction

Pickleball is a rapidly growing sport globally due to its accessibility, moderate fitness requirements, and suitability for a wide range of people. However, the process of learning and perfecting the pickleball technique, especially for those practicing at home, still faces many limitations due to a lack of professional guidance and on-time feedback. Most players rely on watching videos or practicing on their own without objective tools to assess the accuracy of their movements, leading to low learning efficiency and the risk of developing long-term technical errors.

The development of artificial intelligence, particularly computer vision and deep learning, has opened up new opportunities in the field of smart sports coaching. Positivity recognition methods and time-series learning models have demonstrated the ability to analyze human movement with high accuracy.However, within the context of Pickleball, building a complete training system that combines movement analysis, shadowing practice, and user interaction remains an underexplored area of research.

This study proposes an AI-based at-home pickleball training platform to support players in self-learning and self-assessing their technique through video analysis.The focus of this thesis is the design and evaluation of a unified processing pipeline capable of extracting poses from video, classifying actions, supporting shadowing training, and feedback through AI chatbots.

1.1 Motivation

The primary motivation for this research stems from the gap between the growing need for pickleball training and the limited access to professional coaching. For beginners or recreational players, hiring a personal trainer isn't always feasible due to cost, time, and training space. Even with online instructional materials, learners still lack a mechanism for objective feedback to determine whether their movements are technically correct.

From a research perspective, although many studies have focused on action recognition or motion analysis in sports, most existing systems only go as far as action classification and do not prioritize a holistic training experience.In particular, shadowing—an effective motor learning method through imitation and synchronization of movements—is rarely systematically integrated with intelligent movement assessment and feedback.

Furthermore, deep learning models often operate on a mechanism where input is taken and output is True/False, making it difficult for average users to understand and improve the technique. This creates a need for a system that can not only accurately analyze data but also interpret the results in an intuitive and accessible way. These limitations are the driving force behind the research into an AI solution that is both technically sound and user-oriented for home-based pickleball training.

1.2 Solution

To address the aforementioned issues, this study proposes an AI-based Pickleball training solution with a unified pipeline architecture.The system's input data consists of raw videos

recorded in MP4 format, representing both the model movements and the movements performed by the trainee.These videos are divided into short segments and extracted into fixed-resolution frame sequences to ensure consistency during processing.

Our methodology is informed by previous research in tennis action recognition, specifically the work titled "Action Recognition in Tennis Using Deep Neural Networks" [2]. The authors of that study utilized LRCNN and Optical Flow+LSTM architectures on the THETIS dataset. While they achieved promising results, they concluded that performance could be further enhanced by incorporating pose-based keypoints.

The system uses the YOLOv11 model to perform pose extraction, converting image data into skeletal keypoints, thereby representing body movement over time. Following the pose extraction step, the data undergoes preprocessing including normalization, kinematic feature extraction, and data enhancement to improve the model's learning capabilities.

The pipeline was then divided into two functional branches. The first branch used an LSTM model to learn time-series features and perform action classification, allowing for the evaluation of the accuracy of the Pickleball technique.The second branch supports shadowing practice by comparing the practitioner's movements to a reference movement, helping users adjust their movements over time. Finally, the results from the two branches are integrated into an AI chatbot that acts as a virtual coach. This agent's role is to interpret analysis results, provide feedback, suggest technical improvements, and complete the intelligent training loop for home users.

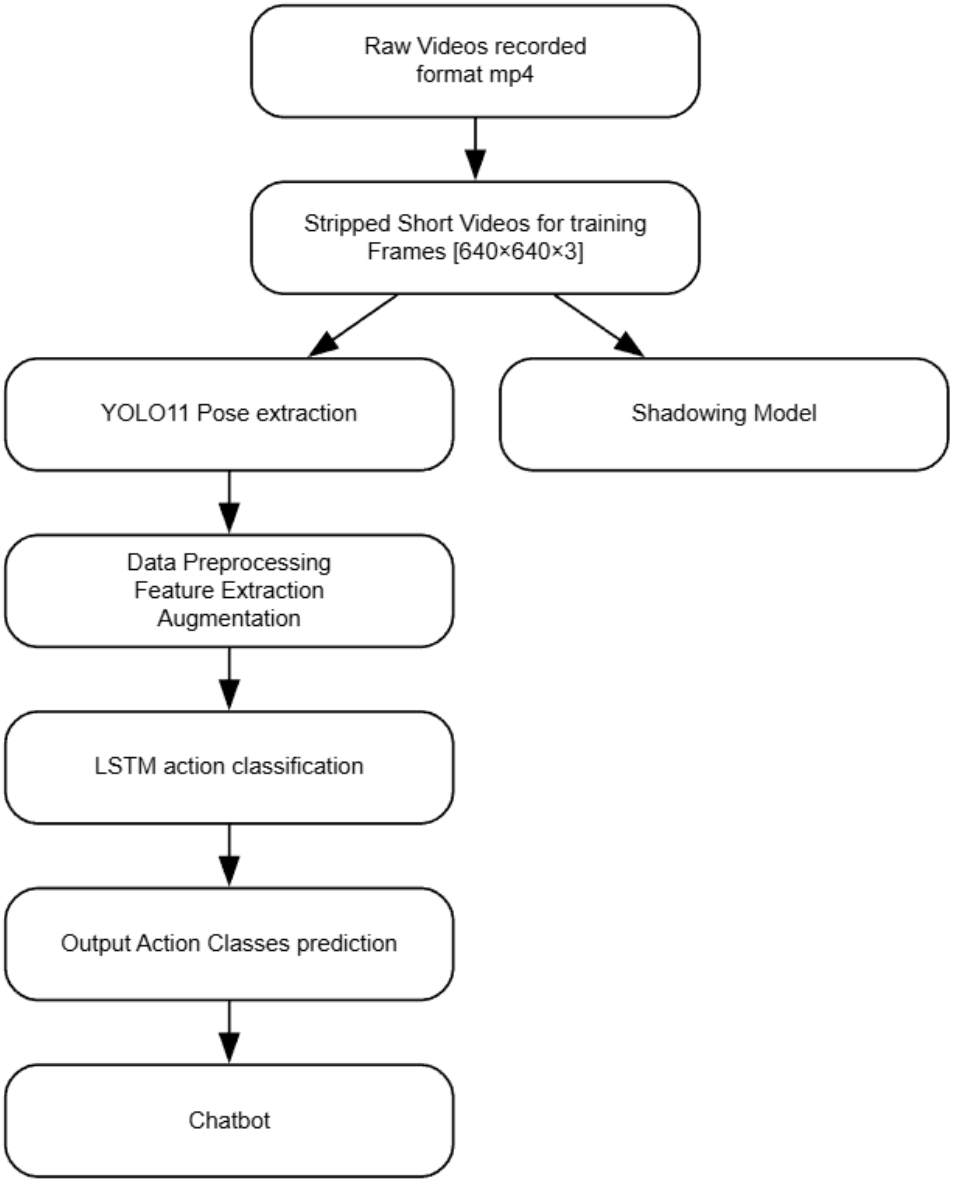


Figure 1: Project Pipeline

2 Dataset

2.1 Dataset for model prediction

For training and validation, we recorded six participants performing techniques from four distinct angles: front, left, right, and rear. For our testing set, we collected YouTube videos of professional players. These diverse, real-world camera angles allow us to evaluate the model's robustness and prevent issues like overfitting or data leakage. Next, the raw videos (usually 5–7 minutes for our recorded videos and 3–5 minutes for Youtube videos) are transformed into the proper data as we trimmed them down into many small clips, each clip ranges from 1 to 4 seconds (averaging 2 to 2.5 seconds) and is trimmed to ensure it only captures exactly one execution of a technique. In addition, we made sure to produce a balanced distribution of videos for the datasets.

Distribution of videos' durations in dataset:

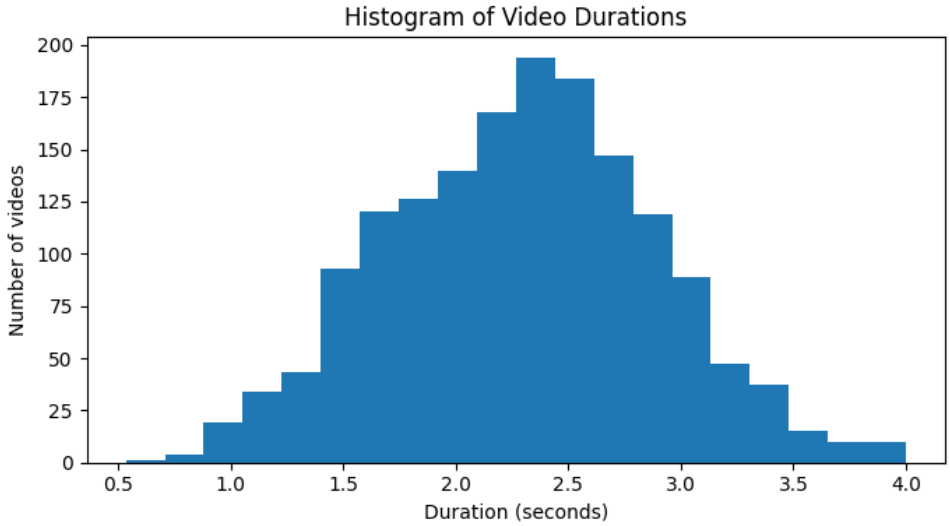


Figure 2: Histogram of Video Durations

2.1.1 Dataset1(2 techniques):

Dataset1 is composed of 2 technique classes. Each of the class has data videos recorded in 4 different angles and by 6 different players with a relatively balanced distribution.
Distribution:

- Classes: DriveForehand, DriveBackhand
- Total: 1340 videos
- Training set: 400 videos per class
- Validation set: 140 videos per class
- Test set: 130 videos per class

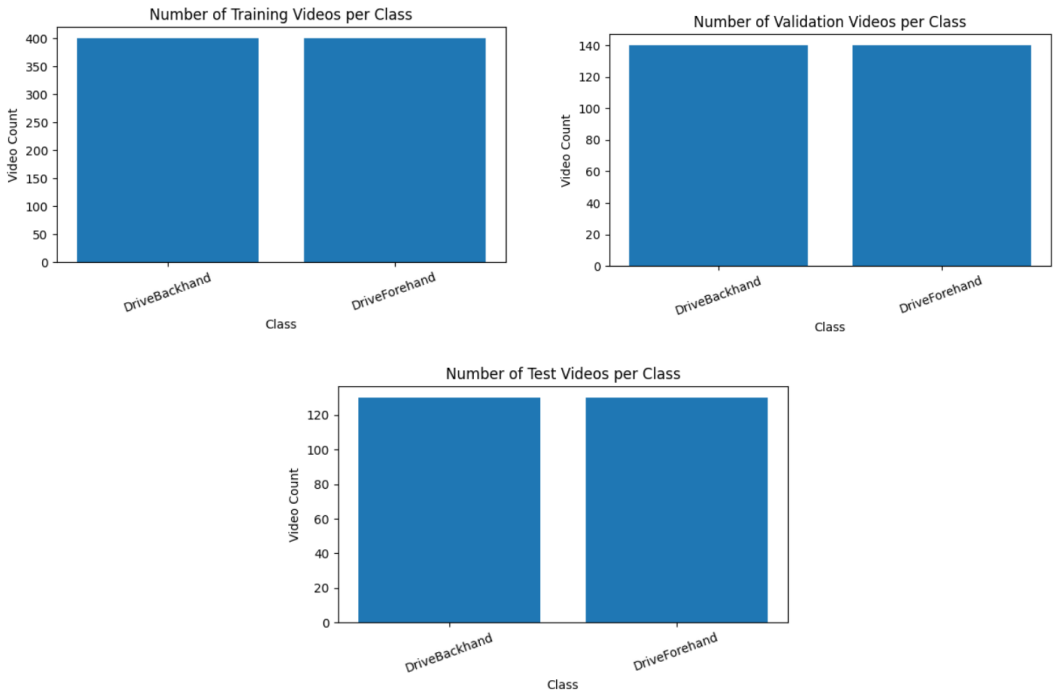


Figure 3: Distribution for Dataset 1

2.1.2 Dataset2(4 techniques):

Dataset2 is the expansion of Dataset1 as we added 2 more technique classes into it. The criterias of data quality and diversity are similar to Dataset2.

Distribution:

- Classes: DriveForehand, DriveBackhand, Smash, Volley
- Total: 2680 videos
- Training set: 400 videos per class
- Validation set: 140 videos per class
- Test set: 130 videos per class

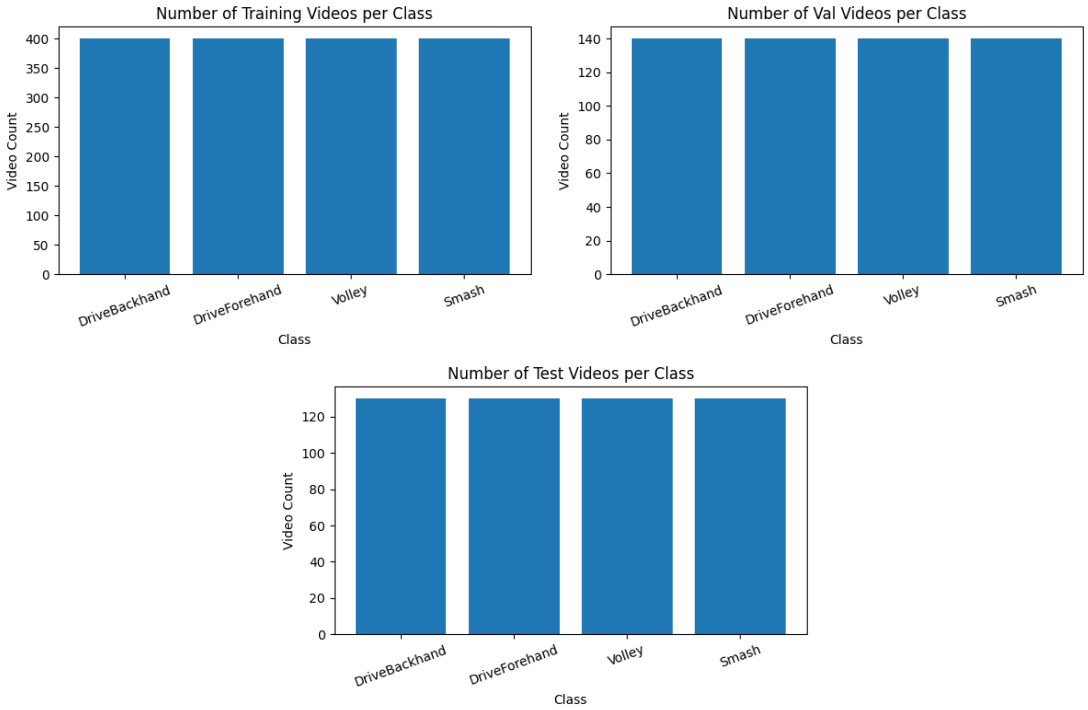


Figure 4: Distribution for Dataset 2

2.2 Data set using for model shadowing

Data source: The specific dataset used for this module is derived from the dataset2(4 techniques) that was used for the prediction model. In addition to these four techniques, we expanded the dataset by introducing a fifth motion: the Serve. Unlike the existing data, the footage for the Serve was specifically recorded from only a semi-professional Pickleball player of our team to ensure technical excellence. Following the same strict standard as Dataset 2, this new footage was carefully filtered and extracted to capture the most representative visual data. As a result, the final dataset provides a comprehensive collection of five essential Pickleball techniques, combining pre-existing data with high-quality, expert-guided recordings.

Data Selection Methodology: To create a high-quality learning resource, the final images were selected based on the following criteria:

- **Camera Angle:** We specifically chose frames captured from a front-facing camera. This perspective allows users to clearly observe the player's body alignment and racket position.
- **Expert Consultation:** The selection process was guided by the advice of semi-professional Pickleball players. Their expertise helped us identify the most technically correct moments for each movement.
- **Manual Extraction:** Instead of using random frames, we manually picked four keyframes for each technique.
- **Dataset Scale:** For each technique, we selected 4 representative images. This results in a specialized set of 20 definitive images that represent the gold standard for Pickleball training.

The Four Essential Phases of a Stroke: This dataset implements a "Discrete Motion

Decomposition" method, breaking down complex actions into four static phases to reduce the learning curve for users:

- **The Preparation Phase:** Establishes the initial stance, center of gravity, and racket readiness.
- **The Backswing Phase:** Captures the generation of potential energy via body coiling.
- **The Contact Point:** Defines the precise spatial-temporal impact moment, crucial for directional control.
- **The Follow-Through:** Illustrates the correct full motion, helps users to complete the action by holding the last correct phase. This can unconsciously create muscle memory for users in the future.

By practicing these four static milestones in order, users can slowly rebuild the entire dynamic movement with high precision.

3 Model, Chatbot and Algorithm

3.1 Model pipeline

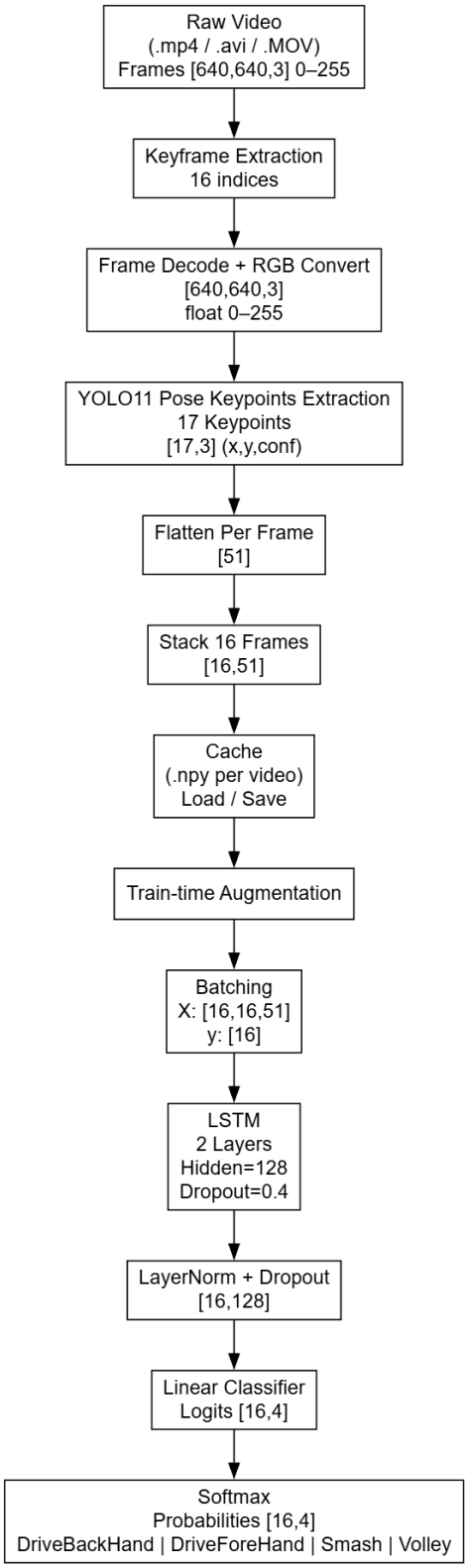


Figure 5: Model's Pipeline

3.1.1 Pose Extraction

Keyframe Extraction: The Keyframe Extraction step transforms a raw, high-frame-rate video into a condensed sequence of 16 representative frames using OpenCV library. This process is essential for bridging the gap between high-frequency visual data and the temporal reasoning required by LSTM classifier model. For example, our videos are filmed at 30 to 60 frames per second (FPS). In a 3-second Pickleball clip, the number would go up to 180 frames. However, the movement of a player during performing a technique does not change significantly between frame 1 and frame 2. By extracting a fixed set of 16 keyframes, the model removes "filler" frames that provide no new information, allowing the network to focus only on the progression of the action and reduce computation cost.

In addition, deep learning models, especially LSTMs and Transformers, require a consistent input shape. If for example one video is 2 seconds and another is 3 seconds, they can't be fed directly into the same network. Keyframe Extraction ensures that every video, regardless of its original length, is normalized into a 16-frame sequence. This allows the LSTM to learn a standardized temporal pattern for each type of pickleball techniques. [3]

Our method is to extract keyframes based on our targeted length $N = 16$. Mathematically:

$$\text{new\_indices} = \bigcup_{i=0}^{T-1} \text{repeat}(i, \lceil \alpha \rceil), \quad \alpha = \frac{N}{T} \tag{1}$$

So the effective output is: Output length $N = 16$. Indices are the original frames selected from the video or positions of frames in the video timeline that were initially chosen to represent the clip. In the very-short-video case, this is simply indices $= [0, 1, 2, \ldots, T - 1]$, meaning every available frame is used. When the number of available frames (T) is smaller than the target length (N), the model still needs a fixed-length sequence. The `new_indices` is used: a length-expanded version of indices, created by repeating (repeatedly sampling) each original frame index multiple times so that the final sequence reaches exactly N frames which can be used as a consistent input shape for the model.

Keypoint Extraction: Once the key frames have been extracted, human pose estimation model YOLOn-pose is applied to those key frames to extract the keypoints features, which can then be used for the classification of different actions. The keypoints features are the pixel coordinates of the human skeletal joints in the key frame. Human pose estimation schemes try to identify the relative placement or layout of different human joints and body parts in an image, which can then be used for action recognition. YOLOv11 Pose, a powerful model developed and released by Ultralytics trained on COCO keypoints dataset, which can detect 17 different keypoints on a person's body, including: Nose, Left Eye, Right Eye, Left Ear, Right Ear, Left Shoulder, Right Shoulder, Left Elbow, Right Elbow, Left Wrist, Right Wrist, Left Hip, Right Hip, Left Knee, Right Knee, Left Ankle, Right Ankle.

Architecture of YOLO: YOLOv11-Pose extends the YOLO (You Only Look Once) family of real-time object detectors by jointly performing person detection and keypoint localization in a single forward pass. The architecture follows a one-stage detection paradigm, consisting of a convolutional backbone for feature extraction, a neck for multi-scale feature aggregation, and a task-specific head that simultaneously predicts bounding boxes, object confidence, and body keypoints. For each detected person, the model outputs 17 anatomical keypoints following the COCO human pose format, where each keypoint is represented by its two-dimensional image coordinates and an associated confidence score.

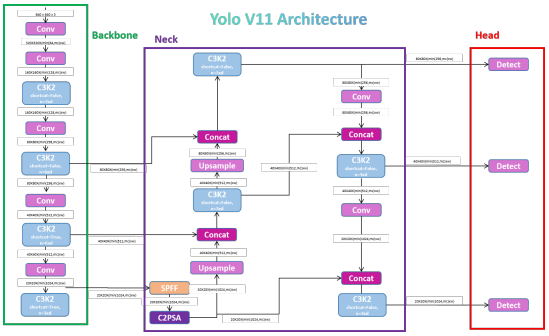


Figure 6: YOLOv11 Architecture [4]

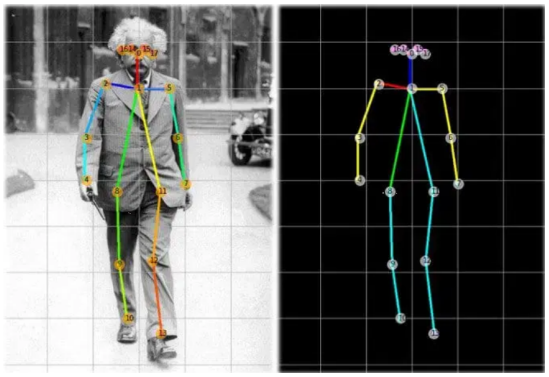


Figure 7: Example of image being applied with Keypoints extraction by YOLOv11 Pose

In our pipeline, only the first detected person in each frame is considered, which is appropriate given the single-player nature of the target application. The raw keypoint coordinates are normalized by the frame width and height to ensure scale invariance across different video resolutions. Each frame is therefore represented by a 51-dimensional feature vector consisting of 17 keypoints with three values each: normalized x-coordinate, normalized y-coordinate, and confidence score. These frame-level features are later aggregated into fixed-length temporal sequences for sequence modeling.

3.1.2 Extracted Feature Cache

After extracting keypoint features from raw video frames using the YOLOv11-Pose model, the next step in the pipeline is to prepare these features for efficient training and robust generalization. Rather than repeatedly performing expensive pose extraction during

model training, all keypoint sequences are first precomputed and stored on the directory of our google drive. Caching features ensures that the training loop can quickly load fixed sequences of keypoint vectors without redundant recomputation, which significantly speeds up training and simplifies reproducibility. The cached features are fixed-length sequences of shape (seq_length, feature_dim), where feature_dim equals 51 (17 keypoints $\times$ 3 values: x, y, confidence), and seq_length is the number of key frames equals 16 used to represent each action clip as presented above.

3.1.3 Data Augmentation

To improve the robustness of the downstream classifier, multiple data augmentations are applied to the cached sequences of keypoints in training set. These augmentations are designed to simulate real-world noise and variability that the model may encounter. Data augmentation in sequential models serves the dual purpose of regularization and domain generalization, similar to how image augmentations improve robustness in vision models. In the context of skeletal pose data, augmentations help the action recognition model learn to focus on meaningful motion patterns rather than weak dependencies on specific keypoints or exact temporal alignments.

1. **Random keypoint masking:** randomly zeros out portions of the keypoint sequence. The individual keypoints are masked with a probability, simulating missing detections. encouraging the model to learn action cues from incomplete or occluded skeletal structures.

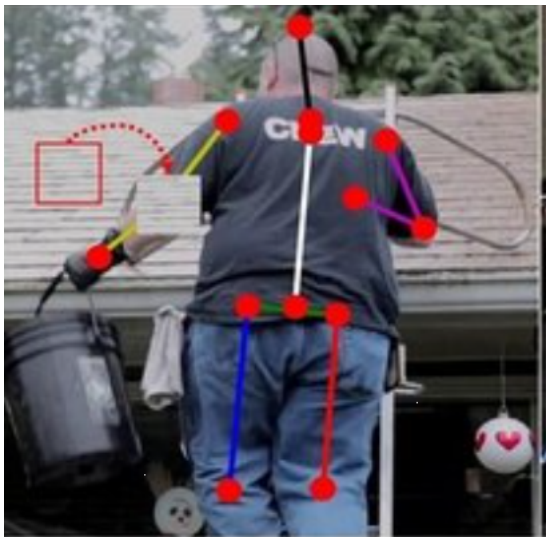


Figure 8: Random keypoint masking [9]

2. **Random temporal shifts:** sequences are shifted by a fraction of sequence length L, mathematically equivalent to applying a permutation π on the time index such that $\pi(t) = (t + k) \pmod{L}$, for some random integer shift k

3. **Add feature noise:** adding small isotropic Gaussian noise to all features.

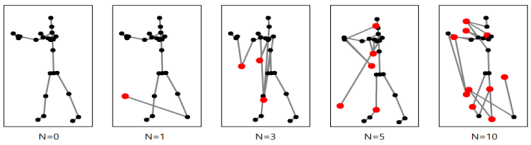


Figure 9: Feature noise [10]

4. **Frame dropout:** randomly zeros complete time slices to simulate abrupt data loss or detection failure, forcing the model to rely on context from neighboring frames.

Figure 10: Example of frame droupout

3.1.4 LSTM action classifier

For this temporal action recognition problem, where the class depends not only on individual poses but on how poses evolve over time, we chose LSTM model, which is specifically designed to process sequential data and can capture long-range temporal dependencies, such as preparation, swing, and follow-through phases in pickleball strokes.

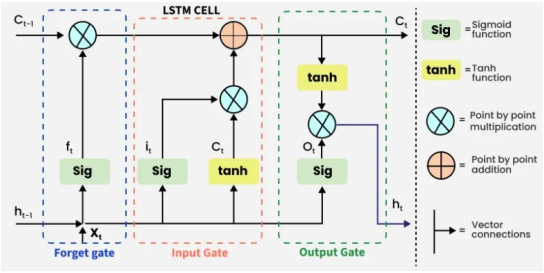


Figure 11: LSTM model architecture [11]

LSTM model architecture : At the core is a many-to-one architecture LSTM with a hidden size which we can alter for finetuning. The LSTM processes the input sequence in temporal order and outputs hidden states that summarize motion dynamics. Instead of using outputs at every timestep, the model uses only the final hidden state of the LSTM, which represents a compact summary of the entire action sequence. This final hidden state is passed through Dropout for regularization, followed by Layer Normalization to stabilize training and improve convergence. A fully connected linear layer then maps the normalized hidden representation to the final class logits. The model is trained using the AdamW optimizer, which combines adaptive learning rates with weight decay for better generalization. The loss function is cross-entropy with label smoothing, helping reduce overconfidence and improve stability on small or noisy datasets.

3.2 Chat bot

3.2.1 Chatbot System Pipeline

The pickleball training chatbot system follows a comprehensive pipeline to analyze player technique from uploaded videos and provide personalized coaching feedback. The complete pipeline consists of the following stages:

1. **Video Upload and Processing:** User uploads a 3–5 second video file (MP4 or WebM format, max 50MB), each video is assigned a unique session ID for tracking.
2. **Frame Extraction:** Extracts keyframes based on multiple criteria: sharpness score, motion score, pose detection score. Output: typically extracts 10 keyframes.
3. **Pose Estimation (MediaPipe Pose Estimation):** Extracts 33 body landmarks from each keyframe, including shoulders, elbows, wrists, hips, knees, ankles, head, nose, eyes. Output: JSON files containing landmark coordinates (x, y, z) and visibility scores.
4. **Vision LLM Analysis:** Using Hugging Face BLIP for descriptive analysis of visual elements that MediaPipe cannot capture (stance, balance, arm extension, body rotation, hand position, arm structure). Each keyframe is analyzed with skill-specific prompts; outputs are saved as JSON.
5. **Data Combination:** Merges MediaPipe pose landmarks with Vision LLM insights into a unified structure containing both quantitative and qualitative information.
6. **Phase Detection:** Segments the swing into Ready, Backswing, Contact, Follow Through using pose landmarks (wrist velocity, elbow angles, shoulder rotation, hand position); outputs phase labels per frame.
7. **Rule-Based Issue Evaluation:** Compares detected phases against predefined technical rules to produce issue codes, descriptions, severity levels, and coaching tips.
8. **LLM-Based Feedback Generation:** Converts structured technical feedback into natural, encouraging coaching language for display to the user.

3.2.2 Phase Detection

The system segments pickleball swings into four distinct phases. Phase detection algorithms differ slightly between forehand and backhand skills.

1. **Ready Phase:** Initial preparation position before the swing begins; minimal wrist velocity (absolute value < 0.002); stable stance, neutral arm position.
2. **Backswing Phase:** Backward motion preparing for the forward swing; wrist velocity is negative (< -0.002); shoulder rotation away from target with weight transfer preparation.
3. **Contact Phase:** Peak moment of the swing; elbow angle $> 160°$; wrist velocity absolute value > 0.01; maximum arm extension and acceleration.
4. **Follow Through Phase:** Completion after the peak; any frame not classified above, typically with positive wrist velocity and smooth deceleration/rotation completion.

3.2.3 Evaluation Process

Phase completeness check. Verify that all required phases appear. Critical issues: missing CONTACT (no swing peak), missing BACKSWING, or missing FOLLOW THROUGH.

Quantitative analysis (MediaPipe).

- **Forehand:** Arm extension (FH01) elbow angle $< 150°$; acceleration (FH04) contact velocity $\leq$ backswing; follow-through length (FH05) fewer than 2 positive-velocity frames.
- **Backhand:** Hand separation (BH01) wrist distance > 0.06; torso rotation (BH02) < 0.15; elbow structure (BH03) either elbow $< 100°$; contact motion (BH05) wrist velocity < 0.01.

Qualitative analysis (Vision LLM).

- Stance (FH06/BH06): detects "closed" stance $\rightarrow$ stance needs adjustment.
- Balance (FH07/BH07): detects "unstable" balance $\rightarrow$ balance instability.
- Body rotation (FH08/BH08): detects "minimal/insufficient" rotation $\rightarrow$ insufficient rotation.
- Backhand specifics: hand position (BH01) confirms separated hands; arm structure (BH03) detects collapsed arms.

Combined evaluation logic.

- **Priority:** High-severity first (phase completeness, arm extension, hand position), then medium (acceleration, rotation, stance, balance), then low (follow-through length).
- **Integration:** MediaPipe quantitative data is primary; Vision LLM qualitative data confirms/refines; agreement boosts confidence; conflicts resolved conservatively.

3.2.4 Evaluation Rules Summary

Drive Forehand Rules

1. FH00: No CONTACT phase $\rightarrow$ No swing peak detected.
2. FH01: Elbow angle $< 150°$ at contact or Vision LLM sees bent/partial extension $\rightarrow$ Arm too bent.
3. FH02: No BACKSWING phase $\rightarrow$ No backswing detected.
4. FH03: No FOLLOW THROUGH phase $\rightarrow$ No follow-through.
5. FH04: Contact velocity $\leq$ backswing velocity $\rightarrow$ Low acceleration.
6. FH05: Fewer than 2 positive-velocity follow-through frames $\rightarrow$ Short follow-through.
7. FH06: Vision LLM detects closed stance $\rightarrow$ Stance needs adjustment.
8. FH07: Vision LLM detects unstable balance $\rightarrow$ Balance instability.
9. FH08: Vision LLM detects minimal rotation $\rightarrow$ Insufficient body rotation.
10. FH99: All checks pass $\rightarrow$ Good shadow drive forehand.

Drive Two-Handed Backhand Rules

1. BH00: No CONTACT phase $\rightarrow$ No clear contact phase detected.

2. BH01: Wrist distance > 0.06 or Vision LLM detects separated hands $\rightarrow$ Hands separated.
3. BH02: Shoulder rotation < 0.15 or Vision LLM detects insufficient rotation $\rightarrow$ Insufficient torso rotation.
4. BH03: Either elbow angle $< 100^\circ$ or Vision LLM detects collapsed arms $\rightarrow$ Elbows collapsed.
5. BH04: Fewer than two FOLLOW THROUGH frames $\rightarrow$ Short or incomplete follow-through.
6. BH05: Contact wrist velocity < 0.01 $\rightarrow$ Weak contact motion.
7. BH06: Vision LLM detects closed stance $\rightarrow$ Stance needs adjustment.
8. BH07: Vision LLM detects unstable balance $\rightarrow$ Balance instability.
9. BH08: Vision LLM detects insufficient rotation $\rightarrow$ Insufficient body rotation.
10. BH99: All checks pass $\rightarrow$ Good shadow two-handed backhand.

3.3 Shadowing Algorithm

3.3.1 Overal progam pipeline of edagogical Framework and Discrete Motion Decomposition

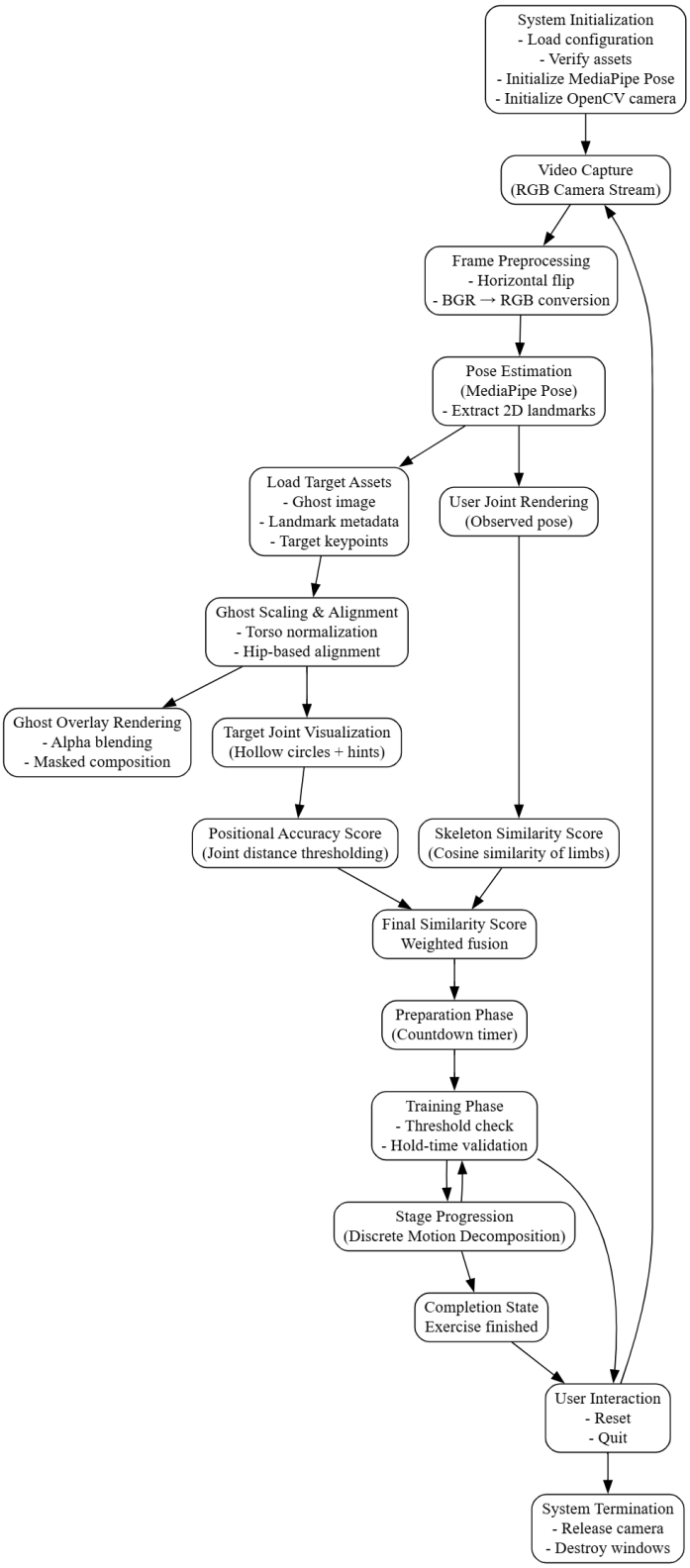


Figure 12: Overall program pipeline.

The development of the Shadowing Trainer module is predicated on specialized visual data designed to facilitate the acquisition of complex skills in Pickleball, specifically the Serve, Drive Forehand, Drive Backhand, Smash, and Volley techniques. To mitigate the cognitive load associated with real-time execution of flow motion linkages, this system implements a **Discrete Motion Decomposition** methodology.

This framework deconstructs complex kinetic chains into four distinct, static keyframes stored within the input repository. By isolating these critical temporal junctures, the system effectively reduces the steep learning curve inherent in sports training, enabling users to concentrate on postural precision and form.

3.4 The Data Processing Pipeline (Asset Builder)

To support real-time Pickleball training and motion assessment with low latency, the system adopts an offline biometric asset construction pipeline. This pipeline is designed following the **Extract–Transform–Load (ETL)** paradigm and consists of three main stages: asset initialization and image preprocessing, feature extraction and biometric normalization, and structured asset persistence.

3.4.1 Asset Initialization and Image Preprocessing

The first stage is responsible for preparing the data environment and standardizing the geometric properties of the input images. Prior to processing, the system removes the existing asset directory (assets/) if it exists and recreates it from scratch. This clean-slate policy ensures data consistency and prevents stale or outdated assets from affecting subsequent training sessions. Next, the reference images of Pickleball techniques are geometrically standardized to align with real-world usage conditions. Since webcam-based systems typically display mirrored images, each input frame is horizontally flipped to ensure spatial consistency between the instructor's reference motion and the user's egocentric viewpoint.

$$I_{\text{processed}} = \text{flip}(I_{\text{raw}}, \text{axis} = 1) \quad [8]$$

This preprocessing step reduces cognitive load for the user and improves the accuracy of posture imitation.

3.4.2 Feature Extraction and Biometric Normalization

After preprocessing, the system extracts biomechanical features from the standardized images. MediaPipe Pose is configured in Static Image Mode with a high-accuracy model setting(Complexity = 2) , which is well suited for offline processing where accuracy is prioritized over inference speed.

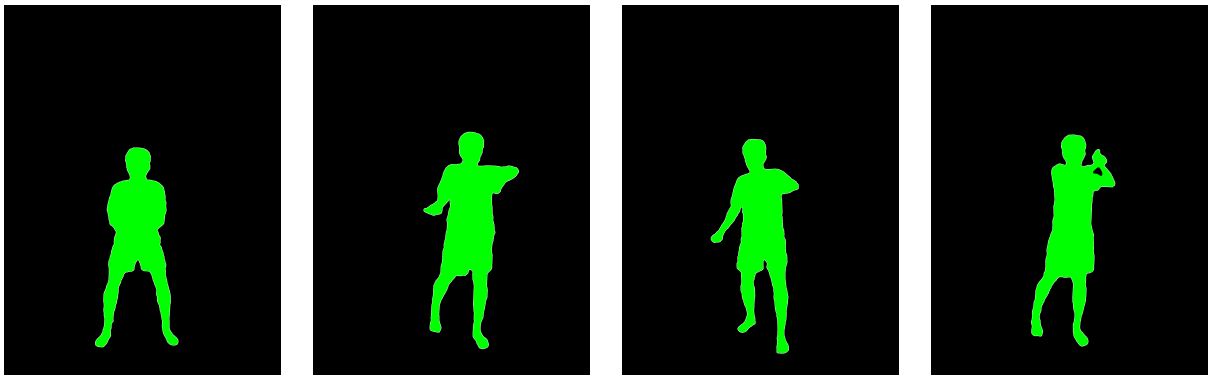
Two primary types of features are extracted: Skeletal Landmarks: A set of 33 human joint landmarks represented as normalized 3D coordinates (x, y, z), capturing the spatial configuration of the instructor's posture.

Segmentation Masks: A probabilistic mask separating the human body from the background, which is subsequently binarized using a threshold $\tau = 0.5$

3.4.3 Structured Asset Persistence

In the final stage, all processed data are stored as structured biometric assets. For each reference frame, three types of files are generated:

1. Silhouette Image (ghost.png): A background-free human silhouette used as a visual guidance overlay during training.



(a) The Preparation Phase (b) The Backswing Phase (c) The Contact Point (d) The Follow-Through

Figure 13: Sequential phases of the Pickleball stroke represented by Ghost silhouettes

2. Biometric Metadata (meta.npy): Reference biometric parameters, including upper-body scale and hip-center coordinates.

3. Target Landmarks (target.npy): Ground-truth skeletal landmark coordinates used for motion evaluation.

Storing these assets in advance significantly reduces computational overhead at runtime, as expensive pose inference and segmentation are performed offline. During real-time training, the system only needs to load the precomputed assets and apply lightweight geometric transformations, ensuring smooth and responsive performance.

3.5 Real-Time Shadowing Training Pipeline

This chapter presents the real-time training pipeline of the Shadowing-based Pickleball Trainer after all reference assets and biomechanical metadata have been constructed. The system is designed to guide users through motion imitation by dynamically aligning a reference motion ("Ghost") with the user's body and continuously evaluating pose correctness. To ensure clarity and structural coherence, the pipeline is organized into two sequential stages: **Dynamic Geometric Alignment and Pose Evaluation with Real-Time Feedback**.

3.5.1 Stage I – Dynamic Geometric Alignment (Ghost Adaptation)

The first stage aims to normalize geometric discrepancies between the reference motion model and the user's physical body. Since users may vary significantly in height and body proportions, directly overlaying the reference motion would lead to spatial inconsistencies and inaccurate feedback. To address this, the system performs dynamic geometric adaptation to align the reference "Ghost" with the user in real time.

Using pose landmarks extracted from the webcam stream, the system estimates the user's torso height H_{user} , which serves as a scale-invariant anatomical measurement. A scaling factor α is then computed as the ratio between the user's torso height and the reference torso height H_{ref} stored in the asset metadata. This factor is applied to resize the Ghost image accordingly.

Following scaling, spatial alignment is achieved by computing a translation vector based on the hip joint, which functions as a stable biomechanical anchor across frames. The resized Ghost is shifted such that its hip position coincides with the user's hip location in the camera frame. Finally, alpha blending is applied to overlay the Ghost onto the live video stream with partial transparency, producing a semi-transparent "shadow" effect that enables intuitive visual imitation.

This stage outputs both the augmented video frame and the transformation parameters (scale and translation), which are reused in subsequent evaluation steps. By enforcing geometric consistency, Stage I establishes a unified coordinate space in which meaningful pose comparison can be performed.

3.5.2 Stage II - Pose Evaluation and Real-Time Feedback

Once geometric alignment has been established, the system proceeds to evaluate the user's performance by analyzing both spatial accuracy and skeletal posture. This stage operates entirely within the normalized coordinate system produced in Stage I.

First, the reference pose keypoints are transformed using the previously computed scale and translation parameters, mapping them into the user's coordinate space. For positional accuracy assessment, the Euclidean distance between each transformed reference joint and the corresponding user joint is computed. A joint is considered correctly positioned if this distance falls below a predefined tolerance threshold set to 10% of the video frame width, allowing for minor execution variations without penalizing the user excessively. The proportion of correctly aligned joints constitutes the positional accuracy score S_{acc} .

In parallel, the system evaluates skeletal posture using cosine similarity between corresponding limb vectors. For each major skeletal segment (e.g., shoulder-to-elbow, elbow-to-wrist), direction vectors are constructed from joint pairs. The cosine of the angle between the user vector and the reference vector is computed, and the final skeletal similarity score S_{skel} is obtained by averaging across all selected segments. This metric emphasizes angular correctness and is invariant to absolute limb length. To generate intuitive feedback, the system renders augmented reality cues directly onto the video stream. Correct joints are highlighted in green, incorrect joints in red, and guiding lines are displayed to indicate the direction of necessary adjustment. A composite performance

score is then calculated as a weighted combination of positional and skeletal metrics:

$$S = 0.4 \cdot S_{\text{acc}} + 0.6 \cdot S_{\text{skel}}$$

This score reflects the system's prioritization of biomechanical posture over exact spatial matching.The overall training flow is orchestrated by a finite-state machine within the main controller function. The system advances to the next training stage only when the composite score exceeds a predefined threshold (0.85) and remains stable for a minimum duration of 0.5 seconds. This temporal constraint prevents accidental transitions caused by transient pose matches and ensures that users achieve consistent correctness before progression.

Through this two-stage pipeline, the Shadowing Trainer delivers a robust and interpretable real-time training experience that combines geometric normalization, biomechanical evaluation, and visually guided feedback within a unified framework.

4 Evaluation

4.1 Evaluation Strategy for the Pose Extraction model

Since YOLOv11-Pose is used as a pretrained feature extractor and no ground-truth keypoint annotations are available for the dataset, traditional supervised pose estimation metrics such as mAP cannot be directly computed. Instead, the quality of the pose extraction process is evaluated using proxy metrics that assess detection reliability, confidence, and temporal consistency. These metrics are chosen to reflect how suitable the extracted keypoints are for downstream temporal action recognition rather than absolute pose accuracy. We applied these metrics on Dataset2 to cover all of our data videos in order to achieve overall performance of YOLOv11-Pose.

4.1.1 Keypoint Confidence Analysis

The first evaluation metric analyzes the confidence scores predicted by YOLOv11-Pose [5] for each keypoint. For a given frame t and keypoint j, the model outputs a confidence value $c_{t,j} \in [0, 1]$ representing the model's estimated reliability of that keypoint. The mean confidence for each keypoint across all frames is computed as:

$$\bar{c}_j = \frac{1}{N} \sum_{t=1}^{N} c_{t,j} \tag{2}$$

where N is the total number of extracted frames. In addition, a global confidence distribution is analyzed by aggregating confidence values over all keypoints and frames. This metric is chosen because confidence scores directly reflect the model's internal uncertainty. High average confidence indicates stable and reliable keypoint detection, while low confidence often corresponds to occlusion, motion blur, or extreme body poses. Since downstream models rely on these keypoints as input features, confidence analysis provides an important indication of feature quality and noise levels. [5]

4.1.2 Detection Success Rate

To evaluate how consistently the pose extractor detects a valid human pose, a detection success rate is computed at the frame level [6]. For each frame, the average confidence across all keypoints is calculated as:

$$\bar{c}_t = \frac{1}{17} \sum_{j=1}^{17} c_{t,j} \tag{3}$$

A frame is considered successfully detected if $\bar{c}_t$ exceeds a predefined threshold τ. The detection success rate is then defined as:

$$\text{Detection Rate} = \frac{\sum_{t=1}^{N} \mathbb{I}(\bar{c}_t > \tau)}{N} \tag{4}$$

where $\mathbb{I}$ is the indicator function. Additionally, the ratio of frames containing zero feature vectors is reported to quantify complete detection failures. This metric is particularly important for video-based modeling, as missing or failed detections can disrupt temporal sequences and degrade the performance of sequence models such as LSTMs. A high detection success rate indicates that the pose extractor is robust across different frames and video conditions. [6]

4.1.3 Temporal Jitter Metric

To assess the temporal stability of the extracted keypoints, a temporal jitter metric is computed [7]. For each joint j, the Euclidean displacement between consecutive frames is calculated as:

$$d_{t,j} = \|\mathbf{p}_{t+1,j} - \mathbf{p}_{t,j}\|_2 \tag{5}$$

where

$$\mathbf{p}_{t,j} = (x_{t,j}, y_{t,j}) \tag{6}$$

denotes the normalized position of keypoint j at frame t. The mean temporal jitter for a video sequence is then obtained by averaging $d_{t,j}$ over all joints and time steps. This metric captures abrupt frame-to-frame variations in keypoint positions that are not caused by actual human motion but by detection noise. Low temporal jitter indicates smooth and consistent pose trajectories, which are crucial for learning meaningful motion patterns in action recognition tasks. Therefore, this metric directly reflects the suitability of the extracted keypoints for temporal modeling. [7]

4.2 Evaluation results of Pose Extraction model:

4.2.1 Training set:

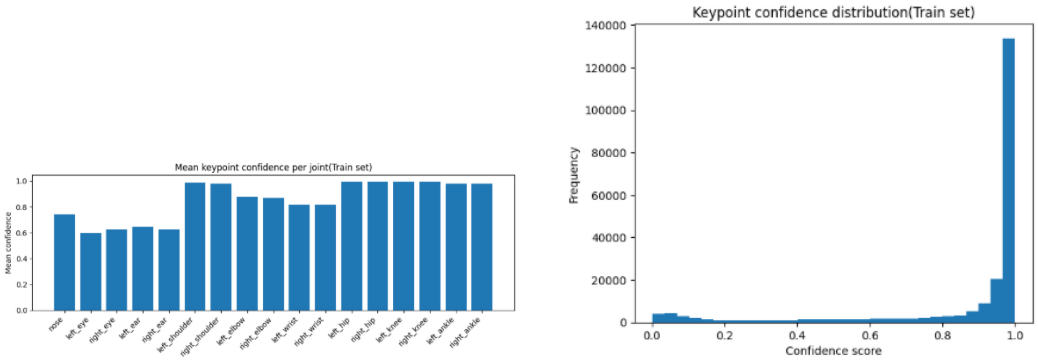


Figure 14: Evaluation on Training set

4.2.2 Validation set:

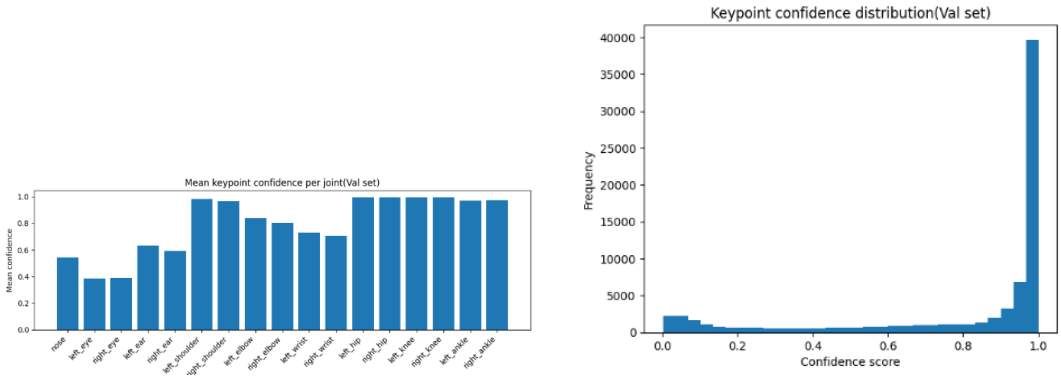


Figure 15: Evaluation on Valuation set

4.2.3 Test set:

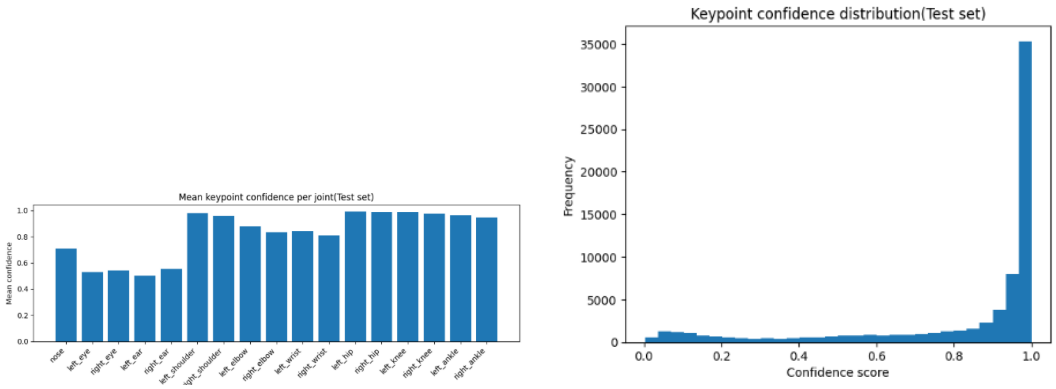


Figure 16: Evaluation on Test set

Table 1: Comparison of Metrics across Dataset Splits

Metric	Train Set	Validation Set	Test Set
Detection success rate	1.000	1.000	1.000
Zero-feature frame ratio	0.000	0.000	0.000
Mean temporal jitter	0.0065	0.0060	0.0206
Std temporal jitter	0.0033	0.0021	0.0312

4.2.4 Comments on Pose Extraction model Performance

The evaluation results on Dataset2 demonstrate that the YOLOv11-Pose model provides highly reliable pose extraction across all dataset splits. A detection success rate of 100% and a zero-feature frame ratio of 0 on the training, validation, and test sets indicate that the model consistently detects a human pose in every selected keyframe, with no complete detection failures.

Temporal stability analysis further shows that the extracted keypoints exhibit smooth and consistent motion patterns, particularly on the training and validation sets, where the mean temporal jitter remains low and tightly distributed. Although the test set displays a higher mean and variance in temporal jitter, this behavior is expected due to increased motion diversity, viewpoint variation, and more challenging recording conditions due to the diverse, real-world camera angles properties of the videos gathered on Youtube.

Importantly, the jitter values remain within a reasonable range, indicating that the pose sequences preserve coherent structure. This confirms the robustness of the pose extractor under the recording conditions of the dataset and validates its suitability as a fixed feature extraction module. Overall, these results suggest that YOLOv11-Pose produces stable, high-quality keypoint features that are well-suited for downstream sequence modeling and action recognition for our project.

4.3 Evaluation Strategy for the LSTM Action Classifier

Experiments To experiment with multiple versions of the model and get a comparison result table , we tuned our hyperparameters on the LSTM model by performing a uniform

grid search on a logarithmic scale for each hyperparameter. These include: learning rate, batch size, number of LSTM hidden units, layers, and dropout rate.

4.3.1 Results on Dataset1

Table 2: Performance comparison of LSTM hyperparameter configurations on Dataset1

LR	Batch	Hidden	Dropout	Layers	Acc	Train (%)	Val (%)	Test (%)
0.001	16	128	0.3	1	0.70	95.50	91.79	70.38
0.0005	32	256	0.4	2	0.69	95.00	91.43	68.85
0.0001	16	256	0.5	2	0.65	95.50	90.00	65.00
0.0003	32	128	0.5	2	0.55	93.38	80.36	55.38
0.001	16	64	0.4	1	0.61	97.00	92.50	61.50
0.001	16	128	0.6	1	0.65	96.75	92.86	64.62
0.001	32	256	0.6	2	**0.74**	**97.12**	**91.43**	**74.23**
0.001	32	128	0.4	2	0.70	97.00	92.68	69.62
0.001	16	256	0.2	1	0.66	97.00	91.79	66.15
0.001	16	128	0.3	2	0.69	96.25	92.86	69.23

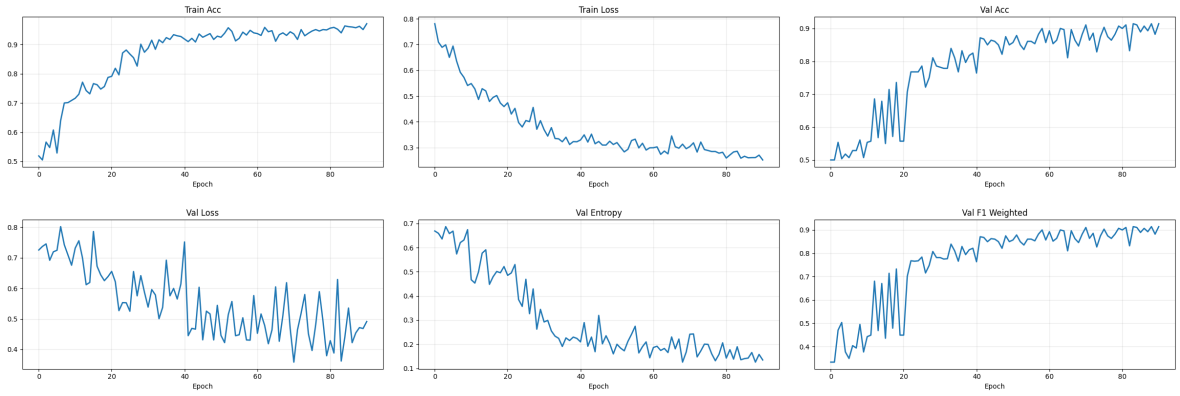


Figure 17: Training and validation performance metrics of the best-performing LSTM model on Dataset1.

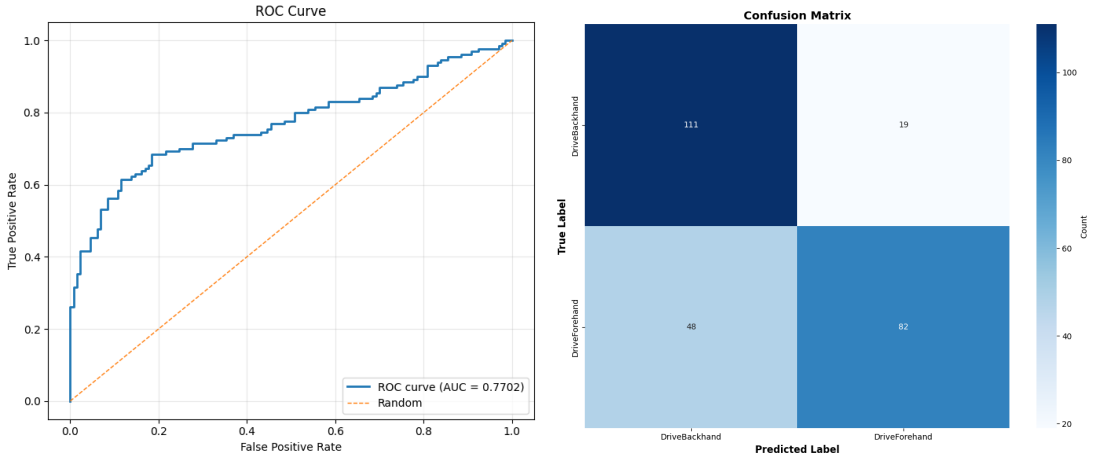


Figure 18: ROC curve and confusion matrix of the best-performing LSTM model on Dataset1.

4.3.2 Results on Dataset2

Table 3: Performance comparison of LSTM hyperparameter configurations on Dataset2

LR	Batch	Hidden	Dropout	Layers	Acc	Train (%)	Val (%)	Test (%)
0.001	16	128	0.3	1	0.33	89.44	83.57	33.27
0.001	32	256	0.3	1	0.33	91.44	83.75	33.08
0.001	16	256	0.3	2	0.32	88.94	83.39	31.73
0.001	16	128	0.4	2	**0.41**	**89.19**	**85.36**	**40.96**
0.001	16	64	0.4	1	0.37	78.84	70.36	36.54
0.001	16	128	0.6	1	0.35	86.94	80.71	35.19
0.001	32	256	0.6	2	0.39	85.88	80.00	38.65
0.001	32	128	0.4	2	0.36	90.81	85.36	36.35
0.001	16	256	0.2	1	0.35	91.38	85.18	34.62
0.001	16	128	0.3	2	0.36	90.12	85.36	35.77

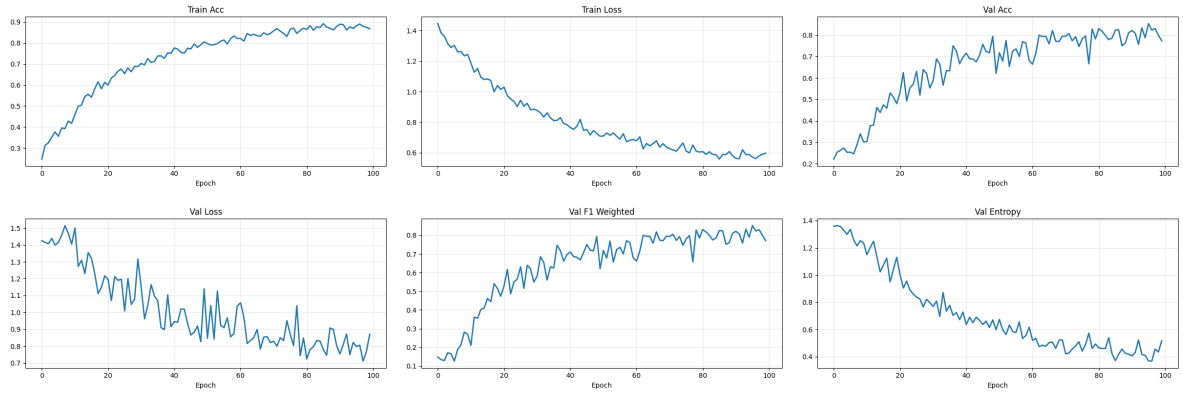


Figure 19: Training and validation performance of the best-performing LSTM model on Dataset2.

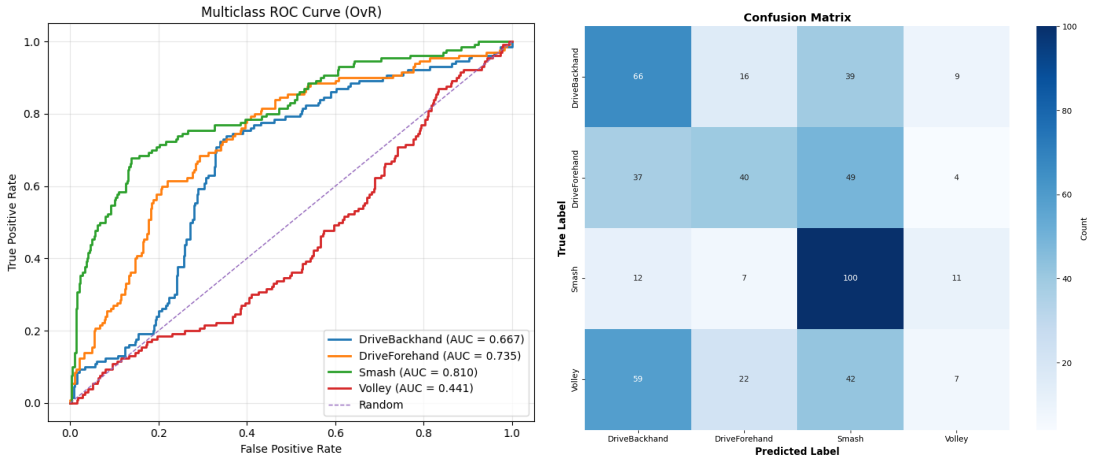


Figure 20: ROC curve and confusion matrix of the best-performing LSTM model on Dataset1.

4.3.3 Comments on Action Classifier model Performance

Based on the experimental results obtained on Dataset1, the hyperparameter tuning process demonstrates that model performance is sensitive to the choice of learning rate, LSTM capacity, and regularization strength. In general, configurations using moderate learning rates (approximately 10^{-4} to 10^{-3}), smaller batch sizes, and shallow LSTM architectures exhibit better generalization performance. In contrast, overly large hidden sizes or excessive dropout rates tend to degrade validation and test accuracy, likely due to underfitting or unstable optimization.

From these experiments, we conclude that the best-performing configuration uses a learning rate of 0.001, batch size 16, 128 LSTM hidden units, a single LSTM layer, and a dropout rate of 0.3. This model achieves the highest test accuracy (74.23%) among all evaluated configurations. Additional evaluation metrics for this model are presented to provide a more comprehensive view of its overall performance.

For Dataset2 with four technique classes, the model's test accuracy dropped to approximately 41% as our best performance, even though training and validation accuracy remained high. This shows that the model does not generalize well when the task becomes more complex. The test set uses real YouTube footage, which differs from the controlled recording conditions used for training, introducing variations in lighting, camera distance, background, and player style. This reason contributes to the higher misclassification and explains why the model performs well on training and validation data recorded by ourselves but poorly on unseen test data from online videos.

5 Website

5.1 System Design Overview

5.1.1 System Architecture

System Architecture:

- **Frontend:** PHP with HTML5, CSS3, JavaScript for user interface
- **Backend:** PHP 7.4+ with MySQL database for data management, Python for Video Analysing.
- **Web Services:** RESTful APIs for communication between frontend and processing services

System Architecture Diagram:

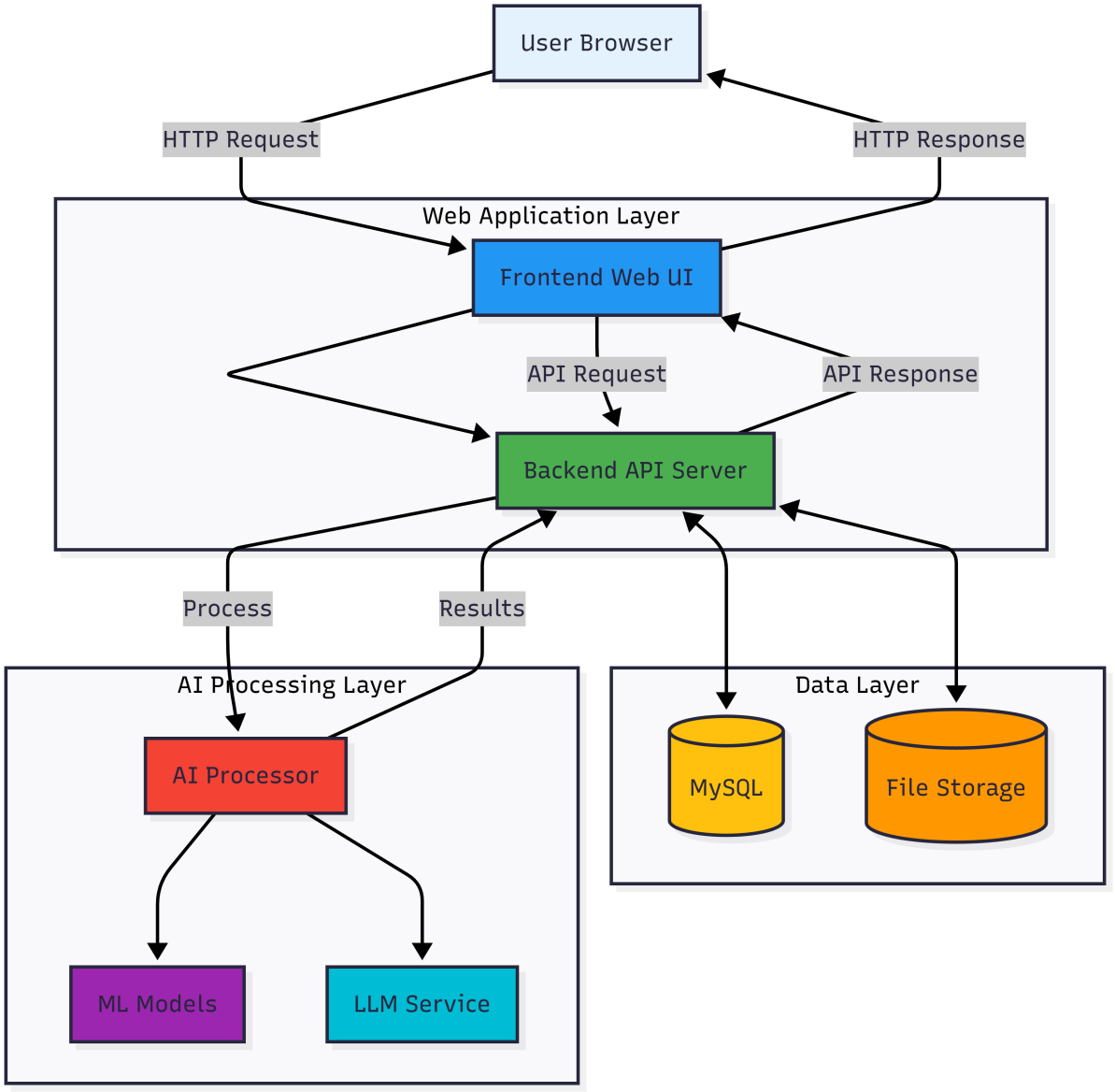


Figure 21: System Architecture Diagram of the overall website.

5.1.2 Entity Relationship Diagram (ERD)

Figure 22: Entity Relationship Diagram (ERD) of the MySQL database.

5.1.3 Use Case Diagram

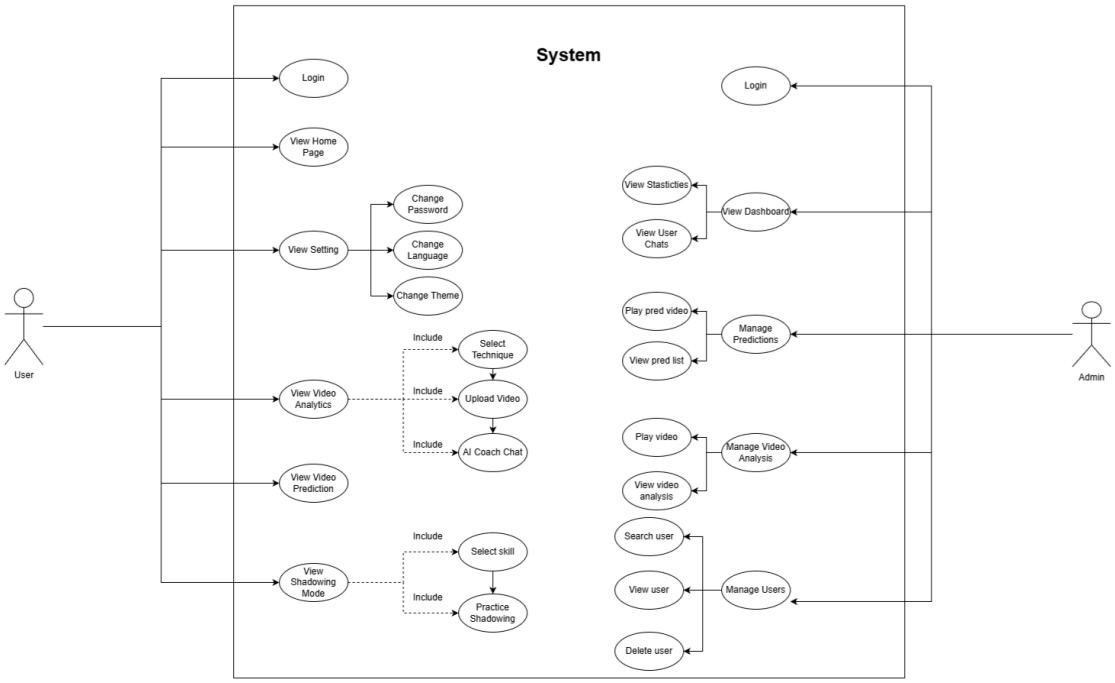


Figure 23: Use Case Diagram of the main user interactions.

5.1.4 Sequence Diagram

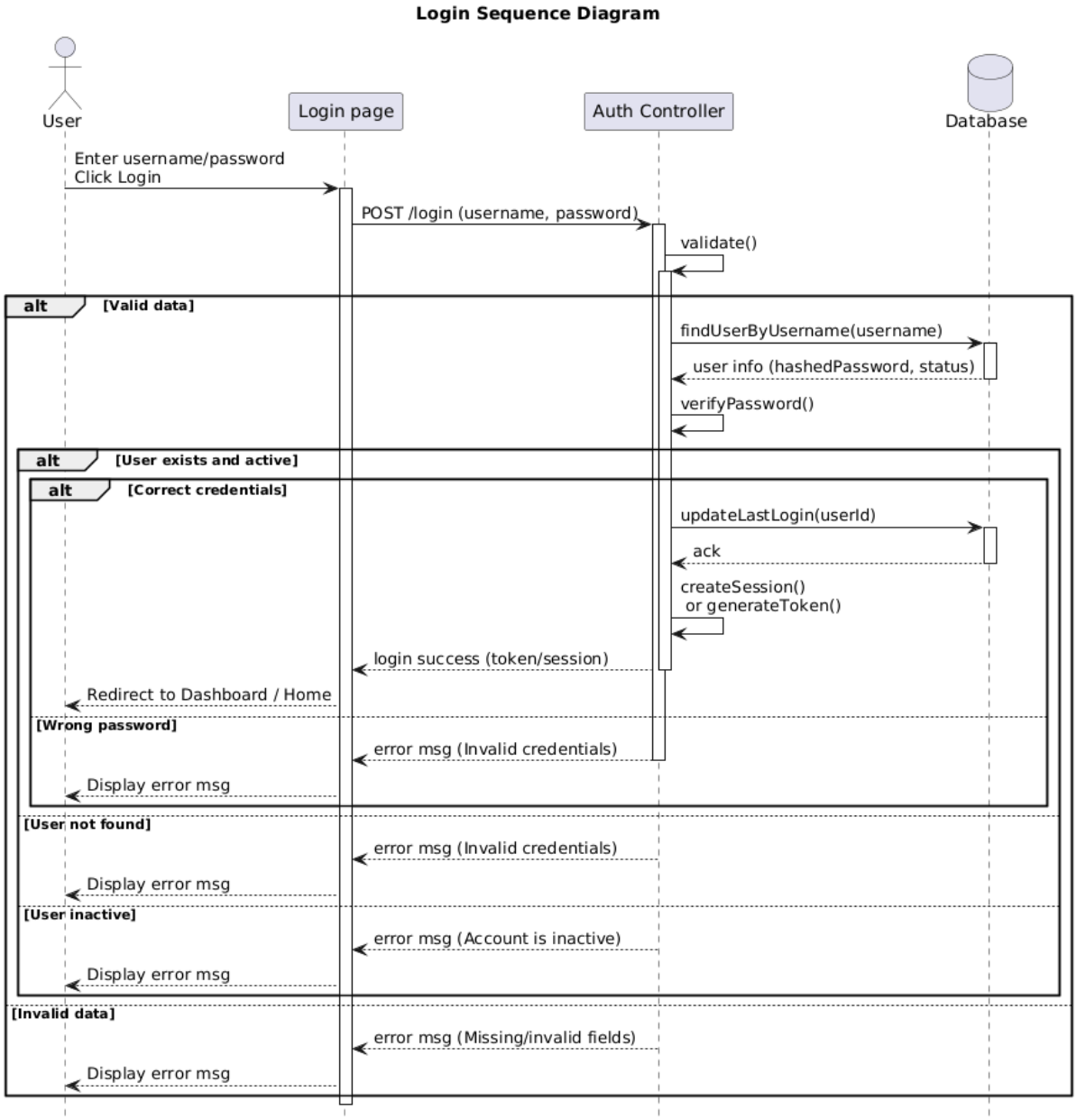
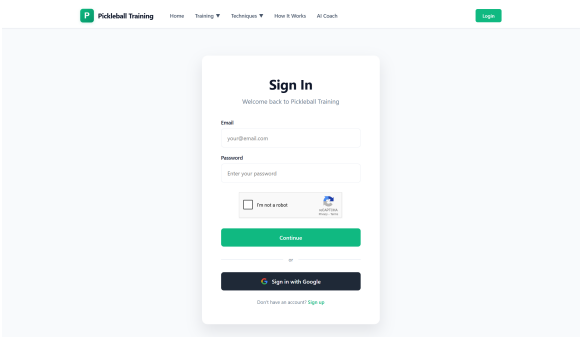


Figure 24: Login Sequence Diagram

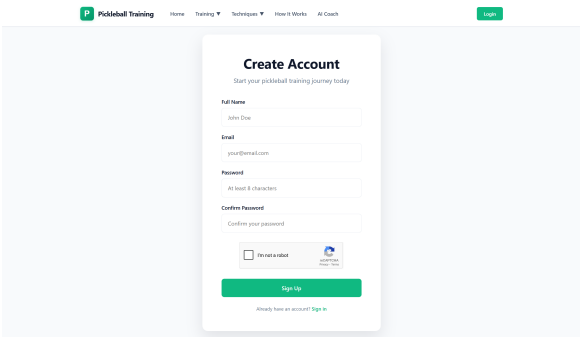
5.2 Authentication & User Management

Functionality	Technical Implementation
User Registration	Email verification, bcrypt password hashing
User Login	Email/password or Google OAuth 2.0
Profile Management	View and update user information
Password Change	Email verification required
Settings Management	Theme and language preferences

Table 4: Authentication & User Management Features



(a) Login Page



(b) Registration Page

Figure 25: Authentication Pages

User Workflow:

1. Register account with email verification
2. Login with email/password or Google OAuth
3. Access profile to view and update information
4. Configure settings (theme, language preferences)

5.3 Video Analysis Module and ChatBot Feedback

Website Functionality	User Experience & Features
Video Upload	Drag-and-drop or file selection (MP4/MOV/AVI)
Technique Selection	Dropdown menu to select training technique
Results Display	View detected techniques, scores, and feedback
AI Coach Chat	Interactive chat interface for questions
Analysis History	View list of previous analyses

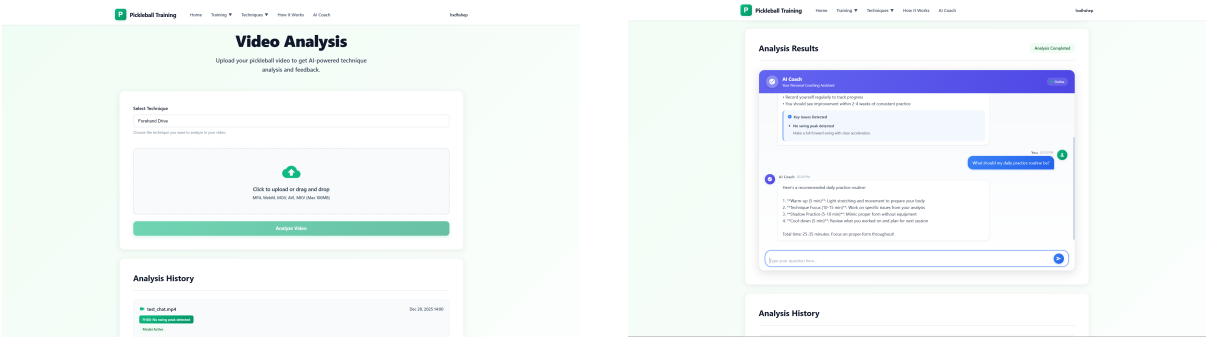
Table 5: Video Analysis Module Features

User Workflow:

1. Select technique from dropdown menu
2. Upload video file via drag-and-drop or file browser
3. Submit for analysis → View results with detected techniques and feedback
4. Access analysis history to review past analyses

System Workflow:

1. Extract key frames from user video based on four main scores: sharpness score, motion score, pose detection score, body parts score.
2. Using MediaPipe for pose estimation with each key frame. Then, this data will be combine with the description by LLM Model of Hugging Face (BLIP).
3. Combined Data will be processed to split into phase motion of each skill. For example, drive forehand will have four phases: Ready, Backswing, Contact and Follow Through.
4. Phases detected with necessary data like: elbow angle, wrist velocity,... will be the essential things for the system which will give feedback for users.



(a) Before Analysis - Upload Interface (b) After Analysis - Results Display

Figure 26: Video Analysis Module Interface

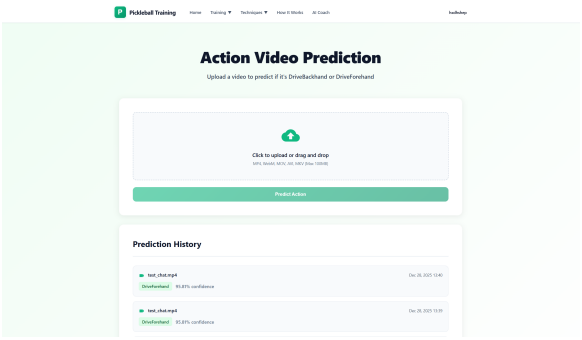
5.4 Action Prediction Module

Website Functionality	**User Experience & Features**
Video Upload	Upload video file for action prediction
Prediction Display	View predicted action and confidence score
Probability Breakdown	Show probability distribution for action classes
Prediction History	View list of previous predictions

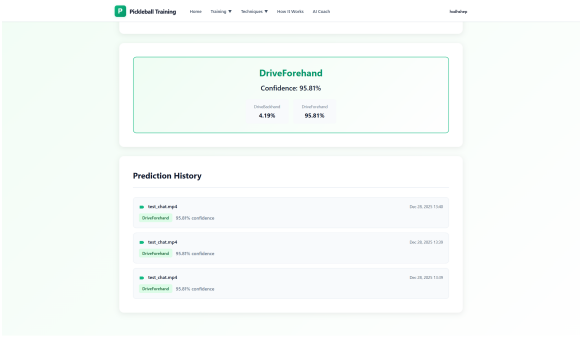
Table 6: Action Prediction Module Features

User Workflow:

1. Upload video file through web interface
2. Submit for prediction → View predicted action and confidence score
3. Access prediction history to compare previous results



(a) Before Prediction - Upload Interface



(b) After Prediction - Results Display

Figure 27: Action Prediction Module Interface

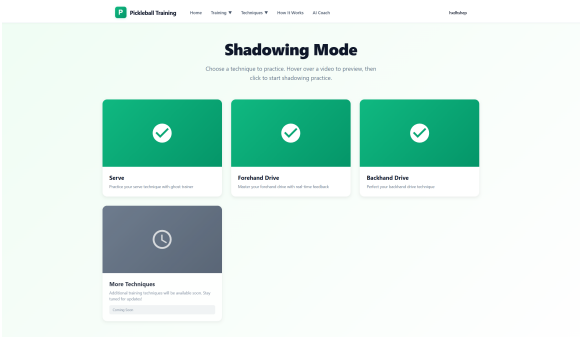
5.5 Shadowing Mode (Ghost Trainer)

Website Functionality	**User Experience & Features**
Technique Selection	Grid layout with available techniques
Practice Interface	Split-screen with camera feed and ghost overlay
Real-time Scoring	Live score percentage for pose matching
Stage Progression	4-stage training system with progress tracking
Controls	Start, Reset, Next Pose buttons

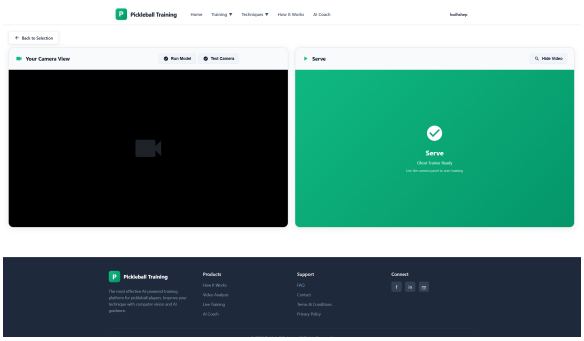
Table 7: Shadowing Mode Features

User Workflow:

1. Select technique from selection grid
2. Initialize camera and pose detection model
3. Start ghost trainer → Match pose to ghost overlay
4. Complete 4 stages (85%+ similarity held for 500ms per stage)
5. Use controls to reset stage or switch to next technique



(a) Technique Selection



(b) Practice Interface

Figure 28: Shadowing Mode Interface

5.6 Admin Management Module

Functionality	Features
User Management	View, edit, delete user accounts
Video Management	List, view details, delete videos
Statistics Dashboard	Charts and aggregated data visualization

Table 8: Admin Management Module Features

Admin Workflow:

1. Login to admin dashboard
2. Manage users (view, edit, delete accounts)
3. Manage videos (list, view details, delete)
4. View statistics and charts

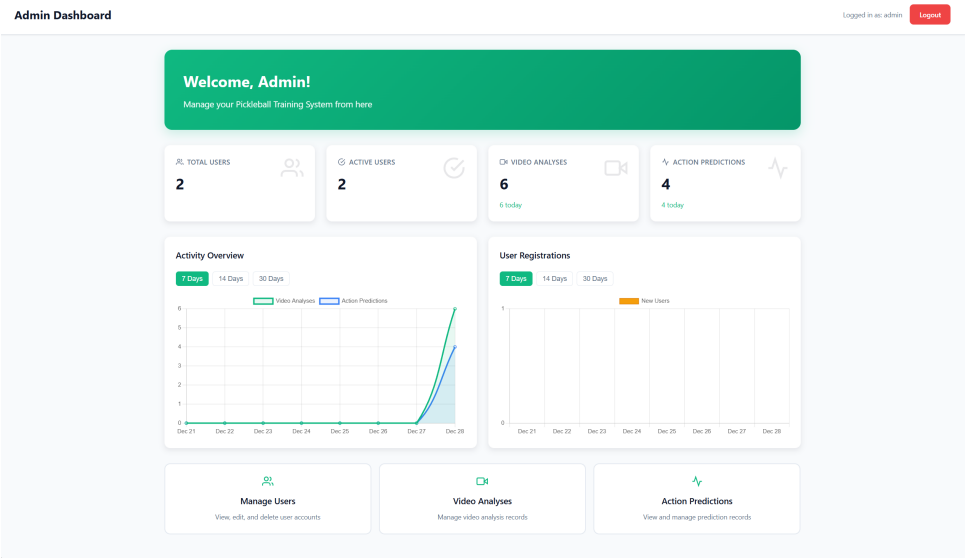


Figure 29: Admin Dashboard with Statistics and Charts

Key Technologies: PHP, MySQL, Chart.js

5.7 Security and Performance

Security Measures:

- Password hashing with bcrypt and salt
- Secure session management with PHP sessions
- SQL injection prevention via PDO prepared statements
- XSS prevention with output sanitization
- File upload validation and type checking
- CSRF protection with tokens
- Google reCAPTCHA for spam prevention

Performance Optimizations:

- Database indexing on frequently queried columns
- Client-side pose detection (MediaPipe in browser)
- Async processing for video analysis
- Optimized SQL queries with proper JOINs
- Asset compression and lazy loading
- Efficient frame capture intervals for live detection

6 Conclusion

6.1 Conclusion

Based on the experimental results, the model achieved promising performance in the two-technique classification task, while its performance dropped significantly when extended to four techniques. This outcome highlights the similar challenge reported by the authors of "Action Recognition in Tennis Using Deep Neural Networks": the critical role of both data quantity and data quality in action recognition tasks. In terms of data quality, our training and validation datasets were collected from only two semi-professional pickleball players within our team, whereas the remaining participants were amateurs. As a result, some techniques were executed incorrectly or inconsistently, which introduced noise into the dataset and negatively impacted the model's learning process and overall performance. This limitation becomes more pronounced as the number of target techniques increases.

To support skill analysis and feedback generation, the Chatbot section combines MediaPipe for human keypoint extraction with descriptive motion information produced by the Vision–Language Model BLIP. This combination provides both numerical data, such as joint coordinates, and high-level movement descriptions. Together, these inputs are used by a Large Language Model to produce meaningful and understandable feedback for users.

Based on these components, the Shadowing Mode proposes a simple and practical learning system for real-time pickleball training. The system consists of an offline preparation stage and a real-time training stage. Motion data are prepared in advance during the offline stage, which helps reduce computational load and ensures stable and repeatable reference movements during training. In real-time use, the system first aligns the reference "ghost" motion with the user through geometric matching, and then evaluates the user's pose by comparing body positions and skeleton structure. By using the hips as the body center and the torso for scaling, the system can adapt to different body sizes. Augmented reality feedback is applied to provide clear and intuitive guidance, making the system fast, accurate, and suitable for home-based practice.

6.2 Future Work:

In future work, several directions will be explored to improve both the accuracy and practicality of the model. A key priority is to collect a much larger and more diverse dataset, with more recordings from professional players and additional camera angles beyond the current front, left, right, and rear views. This will help reduce noise, improve robustness,

and allow the model to recognize a wider range of pickleball techniques. As the dataset grows, more advanced model architectures, such as attention-based and Transformer-based methods, will be investigated to better capture long-term motion patterns and focus on important frames and joints.

At the same time, the current rule-based evaluation system remains relatively simple and lacks sufficient detail to deliver highly accurate or personalized advice. Therefore, future work will focus on analyzing a wider range of pickleball skills in order to develop a more robust and comprehensive rule-based evaluation framework. Furthermore, training a Vision–Language Model specifically for detailed descriptive analysis—such as stance, balance, and body alignment—is an essential next step. To achieve this, a larger and more diverse dataset containing annotated images and videos will be required, enabling the model to produce precise and meaningful descriptions rather than relying primarily on prompt-based inference.

Finally, time-based motion analysis will be introduced to better understand continuous movement instead of evaluating only individual poses. Models such as recurrent neural networks or Transformers can be used to assess motion smoothness and consistency over time. The system may also integrate multi-view analysis and adapt scoring rules to individual users, making the training experience more personalized. These improvements would allow the system to support more complex sports movements and potentially extend to rehabilitation and other motion-based training scenarios.

Bibliography

[1] THETIS-dataset, *THETIS: A Real-world Dataset for Tennis Gesture Recognition*, : https://github.com/THETIS-dataset/dataset

[2] Stanford University, *CS230: Deep Learning - Project Report*, 2018. : https://cs230.stanford.edu/files_winter_2018/projects/6945761.pdf

[3] MDPI, *Applied Sciences: Human Action Recognition Research*, 2022. : https://www.mdpi.com/2076-3417/12/11/5455

[4] Analytics Vidhya, *YOLOv11 Model Building: A Comprehensive Guide*, 2025. : https://www.analyticsvidhya.com/blog/2025/01/yolov11-model-building/

[5] Ultralytics Documentation, *Pose Estimation Tasks and Models* : https://docs.ultralytics.com/tasks/pose

[6] LearnOpenCV, *Object Keypoint Similarity (OKS) Explained* : https://learnopencv.com/object-keypoint-similarity

[7] ResearchGate, *Detect-and-Track: Efficient Pose Estimation in Videos* : https://www.researchgate.net/publication/322076361_Detect-and-Track_Efficient_Pose_Estimation_in_Videos

[8] Bradski, G., and Kaehler, A., *Learning OpenCV*, O'Reilly Media, 2008.

[9] ResearchGate, *Keypoint masking to simulate the hard training samples* : https://www.researchgate.net/figure/Keypoint-masking-to-simulate-the-hard-training-samples-a-is-a-common-case-in-human_fig4_324055335

[10] Jongmin Yu, Yongsang Yoon, and Moongu Jeon, *Predictively Encoded Graph Convolutional Network for Noise-Robust Skeleton-based Action Recognition*, arXiv preprint, 2020.: https://arxiv.org/pdf/2003.07514

[11] GeeksforGeeks, *Introduction to Long Short Term Memory (LSTM)* : https://www.geeksforgeeks.org/deep-learning/deep-learning-introduction-to-long-short-term-memory/

[12] Shorten, C., and Khoshgoftaar, T. M., *A survey on image data augmentation for deep learning*, Journal of Big Data, vol. 6, no. 1, 2019.

[13] Sewell, W., and Komogortsev, O., *Real-time eye gaze tracking with an unmodified web camera*, CHI '10 Extended Abstracts on Human Factors in Computing Systems, 2010.

[14] Toshev, A., and Szegedy, C., *DeepPose: Human pose estimation via deep neural networks*, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014.

[15] Sun, K., Lan, C., Xing, J., Zeng, W., Liu, D., and Wang, J., *Human pose estimation using global and local normalization*, Proceedings of the IEEE International Conference on Computer Vision, 2017.

[16] Zhang, J., et al., *Whole-Body 3D Pose Estimation Based on Body Mass Distribution*, Sensors, vol. 12, no. 2, 2025.

[17] Mok, T. C., and Chung, A., *Affine Medical Image Registration With Coarse-To-Fine Vision Transformer*, Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022.

[18] Li, R., Dong, X., Cai, Z., and Hu, S., *Our AffineAlign Operation: Estimating affine transformation matrices between human poses*, ResearchGate, 2018.

[19] Vantigodi, S., and Babu, R. V., *Tracking using Human Pose Matching with Deep Association Metric*, Proceedings of the Florida Artificial Intelligence Research Society Conference, 2015.

[20] Wang, J., et al., *Pose embedding: A deep architecture for learning to match human poses*, arXiv preprint arXiv:1507.00302, 2015.

[21] Albert, J. A., et al., *Analysis of Kinect-Based Human Motion Capture Accuracy Using Skeletal Cosine Similarity Metrics*, Sensors, vol. 25, no. 4, 2025.

[22] Sidorov, G., et al., *Soft similarity and soft cosine measure: Similarity of features in vector space model*, Computación y Sistemas, vol. 18, no. 3, 2014.

[23] El-Hasnony, I. M., et al., *A Score-Fusion Method Based on the Sine Cosine Algorithm for Enhanced Multimodal Biometric Authentication*, Sensors, vol. 26, no. 1, 2025.

[24] Kumar, A., and Zhang, D., *Weighted Sum Fusion in Biometric Recognition*, ResearchGate, 2018.

[25] Deisenroth, M., Faisal, A., and Ong, C., *Mathematics for Machine Learning*, Cambridge University Press, 2020.

[26] Bishop, C., *Pattern Recognition and Machine Learning*, Springer, 2006.

[27] Hastie, T., Tibshirani, R., and Friedman, J., *The Elements of Statistical Learning*, Springer, 2009.

[28] Shalev-Shwartz, S., and Ben-David, S., *Understanding Machine Learning: From Theory to Algorithms*, Cambridge University Press, 2014.

[29] Jurafsky, D., and Martin, J., *Speech and Language Processing*, 3rd ed., 2024.

[30] Manning, C., Raghavan, P., and Schütze, H., *Introduction to Information Retrieval*, Cambridge University Press, 2008.

[31] Boyd, S., and Vandenberghe, L., *Convex Optimization*, Cambridge University Press, 2018.