RESEARCH ARTICLE OPEN ACCESS

Real-Time Sports Action Recognition Using deep Learning (CNN)

Uday Kiran. K[1], Bhanu Prasad Reddy. M[2], Lakshmi. V[3], Harini. K[4], Sumanth. K[5], Dr. Venkataramana. B[6]

[1]*Student, BTech CSE(AIML) 4th Year, Holy Mary Inst. of Tech. and Science, Hyderabad, TG, India,*
[2]*Student, BTech CSE(AIML) 4th Year, Holy Mary Inst. of Tech. and Science, Hyderabad, TG, India,*
[3]*Student, BTech CSE(AIML) 4th Year, Holy Mary Inst. of Tech. and Science, Hyderabad, TG, India,*
[4]*Student, BTech CSE(AIML) 4th Year, Holy Mary Inst. of Tech. and Science, Hyderabad, TG, India,*
[5]*Asst. prof, CSE(AIML), Holy Mary Inst. of Tech. and Science, Hyderabad, TG, India,*
[6]*Assoc. prof, CSE, Holy Mary Inst. of Tech. and Science, Hyderabad, TG, India,*

Abstract:
The "Real-Time Sports Action Recognition Using Deep Learning CNNs" project aims to create an intelligent sport recognition system in real-time based on high level Deep Learning techniques, especially Convolutional Neural Networks (CNNs). The goal is to analyze live sports video streams to confidently recognize the sport (football, basketball, cricket, tennis, etc.,) that is being played while relying on visual patterns and motion signals. This system can perform automated sport recognition and classification. The basic architecture employs a Python-based backend utilizing TensorFlow & OpenCV to process video frames and infer CNN models. It can recognize a wide range of sports actions including batting and bowling in cricket, kicking and goalkeeping in football, dribbling and shooting in basketball, serving and smashing in badminton or tennis, as well as running, cycling, swimming, etc.,

The base CNN model employs transfer learning techniques with models such as ResNet50 & MobileNetV2 to increase accuracy and performance. The framework also uses frame differencing and motion vector extraction techniques to better understand time to improve the classifiers within dynamic sport activity classifications. Then, the trained models are intended to operate within real-time with inference speeds that are improved through the other battery of processing introduced with the frames difference algorithm. Overall, the project is an amazing achievement towards putting intelligent video comprehension and automation into action in the sports sector. By applying deep learning, computer vision, and real-time system optimization algorithms, this solution has the potential to make a major difference in a few key areas.

Keywords: Real-time sport recognition, Deep learning, Convolutional Neural Networks (CNNs), ResNet50, TensorFlow, OpenCV, Computer vision.

Date of Submission: 23-01-2026 Date of acceptance: 05-02-2026

I. Introduction

The project focuses on recognizing sports actions (e.g., running, jumping, kicking, etc.,) from video feeds. It uses deep learning models (CNNs) to classify actions in real time. Removes the need for manual feature extraction by learning directly from visual data. Displays predictions instantly via a Streamlit web interface. Supports sports analytics, training, broadcasting, and performance tracking. With the power of Artificial Intelligence (AI) and Computer Vision (CV), we can now detect, classify, and analyze sports movements like running, jumping, kicking, etc., directly from live video streams or recorded footage. This breakthrough doesn't just make sports more interactive for viewers; it gives coaches, analysts, and players deeper insights into performance and decision making.

Video-based action recognition has become increasingly important in sports analytics as it enables automated understanding of complex player movements and dynamic visual environments. Existing studies discuss a wide range of sports action recognition techniques and highlight challenges such as motion variation, viewpoint changes, and background complexity [1]. Other research focuses on detecting key events and identifying important actors in multi-person video scenarios commonly found in sports footage. These works show that spatiotemporal modeling is effective for capturing player interactions and understanding complex sports events [6].

The core programming language is python which depicts dynamic with easy syntax & pretrained models in it. programming the system takes in real-time video input. The algorithms like TensorFlow, Keras, Pandas & Numpy are used for the Data visualization and Frame differencing. The core Deep learning technique is Convolution Neural Networks (CNNs) which learns from raw visual data to impove accuracy and flexibility. UCF101 & HMDB51 are the main datasets in which raw videos related to sports action exists. It pulls out key frames and uses them to pass through a CNN model trained on thousands of tagged sports action images. Utilizing strong pretrained models such as ResNet50 & MobileNetV2, we use transfer learning to improve performance despite small data. The output is shown immediately on a clean, interactive Streamlit web page, facilitating easy visualization and understanding of the identified actions. But the possibilities extend much wider than sport; Video auto-annotation, Smart training platforms, Motion detection, Intelligent surveillance.

By combining deep learning, computer vision, and real-time video processing, this project demonstrates the ways in which AI can transform our understanding of physical motion-not only in sports, but in a myriad of real-worlduses**.**

II. Literature Review

Sports Action Recognition (SAR) has evolved from traditional handcrafted feature-based methods, such as optical flow and trajectory descriptors, to modern deep learning approaches capable of learning complex spatiotemporal patterns from video data. Early techniques were effective in controlled environments but suffered from poor robustness to background clutter, viewpoint variations, and rapid motion commonly found in sports videos. The adoption of Convolutional Neural Networks (CNNs) significantly improved spatial feature extraction by learning visual cues such as player posture, equipment, and scene context directly from frames; however, frame-based CNN models alone were limited in capturing temporal continuity. To address this shortcoming, researchers combined CNNs with recurrent architectures, particularly Long Short-Term Memory (LSTM) networks, which enhanced motion modeling across sequential frames but introduced additional computational overhead that constrained real-time deployment. More advanced spatiotemporal architectures, including 3D CNNs and Inflated 3D ConvNets, further improved recognition accuracy by jointly learning spatial and temporal features, though their high memory and processing requirements restricted their use in low-latency systems. Pose-based and skeleton-driven methods offered fine-grained action understanding by analyzing joint movements, yet their dependence on accurate keypoint detection limited performance in crowded or fast-paced sports scenes. Recent multimodal and transformer-based approaches have demonstrated strong modeling capabilities but remain computationally expensive for live video applications. Overall, existing studies reveal a clear trade-off between recognition accuracy and real-time efficiency, motivating the need for lightweight yet robust models. This project addresses this gap by leveraging efficient pretrained CNN architectures with streamlined temporal processing to achieve reliable sports action recognition under real-time constraints.

Several studies have explored vision-based sports action recognition to support performance analysis and automated sports intelligence. These works collectively highlight the growing role of deep learning in analyzing player movements, detecting key actions, and supporting notational analysis. While such systems demonstrate improved recognition accuracy, they commonly report challenges related to limited benchmark datasets, high computational requirements, and difficulties in handling fast-paced and visually complex sports environments, which restrict their applicability in real-time scenarios [1]. Temporal modeling has been widely investigated to improve action recognition by capturing sequential motion patterns. LSTM based architectures have shown advantages in modeling time-dependent features and improving robustness to noise when compared to traditional methods. However, these approaches often require large labeled datasets and exhibit high computational complexity, making them less suitable for deployment in real-time or resource-constrained environments, particularly when signal variability or sensor differences are present [2]. Dataset-driven and sport-specific studies further demonstrate that CNN-based models can achieve high accuracy when trained on well-structured data. Approaches utilizing key-frame selection and multimodal datasets have been effective in recognizing fine-grained sports actions, such as martial arts techniques and sport-specific movements. Despite their success, these systems typically depend on large training datasets, are computationally intensive, and may struggle to generalize across different camera angles, player physiques, or recording conditions [3]. Large-scale spatiotemporal architectures, including 3D convolutional models, have significantly advanced action recognition by jointly learning spatial and temporal features from video sequences. While these models achieve strong benchmark performance, their complex network structures demand substantial processing power and memory resources, which limits their practicality for real-time applications and edge-device deployment [4]. Fine-grained action recognition has also been studied through object-centric and interaction-based modeling, where human–object relationships are explicitly analyzed. Such approaches are particularly useful in sports involving equipment interactions, such as football and cricket. However, they rely heavily on accurate object detection and tracking, and their performance may degrade in noisy or visually cluttered environments, increasing computational cost and system complexity[5].

Multi-person action recognition has received attention in team sports scenarios, where identifying key actors and events is critical. Attention-based recurrent models improve event detection by focusing on relevant players within crowded scenes. Despite these advances, these systems typically require large annotated datasets and face scalability challenges in complex or overlapping environments [6]. Skeleton-based and RGB-D approaches have demonstrated effectiveness in capturing detailed motion dynamics, particularly in structured sports such as basketball. By incorporating depth and joint-level information, these models improve recognition accuracy on benchmark datasets. Nevertheless, their reliance on specialized sensors, high computational requirements, and sensitivity to dataset quality limit their use in real-time and unconstrained sports settings [7]. Transfer learning has been widely adopted to address data scarcity in sport-specific recognition tasks. CNN models pretrained on large image datasets have shown promising results when fine-tuned for sports activities such as hockey and baseball. While transfer learning improves performance with limited data, challenges remain in handling occlusions, rapid movements, and maintaining real-time inference speeds [8][9].

Pose-based systems further enhance action recognition by analyzing skeletal keypoints to capture fine-grained movement patterns. These approaches have achieved strong performance in recognizing actions such as cricket strokes and tennis swings. However, their effectiveness is highly dependent on accurate pose estimation, and performance may degrade in complex backgrounds or real-time settings due to increased computational overhead [10][11]. Hybrid and multi-stream deep learning architectures have been proposed to improve robustness by combining spatial, temporal, and motion-based features. These models achieve high recognition accuracy across multiple datasets by leveraging complementary information streams. Despite their effectiveness, such architectures often suffer from increased model complexity, higher processing costs, and limited adaptability to real-time applications, especially on lightweight devices [12][13][14][15].

III. Methodology

The methodology for developing a Real-Time Sports Action Recognition System is a **Deep Learning–based Spatio-Temporal Action Recognition approach**, combining both **Convolutional Neural Networks (CNNs)** and **temporal sequence modeling** (using **3D CNNs** or **ConvLSTM**). follows a carefully designed workflow that integrates deep learning, computer vision, and real-time video analytics. The goal is to automatically identify and classify various sports actions as they occur in live video streams.

This approach emphasizes modularity and scalability, dividing the system into multiple stages: data acquisition, preprocessing, feature extraction, model training, classification, and real-time deployment. Each module contributes to specific aspects of accuracy, efficiency, and speed, ensuring that the final system performs smoothly in practical, real-world conditions.

Modern deep learning models, including 3D Convolutional Neural Networks (3D CNNs), Convolutional LSTM networks (ConvLSTM), and Vision Transformers (ViT), play a crucial role in enhancing both spatial and temporal understanding of human actions. These architectures allow the system to not only recognize objects but also interpret movement patterns across time.

3.1 Data Acquisition and Dataset Preparation

A reliable action recognition system depends on a diverse and well-labelled dataset of sports videos. Using videos from multiple sports helps the model generalize across different camera angles, lighting conditions, player styles, and environments. Well-balanced data also improves performance on unseen actions.

The primary datasets used in this work are UCF101 and HMDB51. The UCF101 dataset contains 13,320 realistic video clips across 101 action categories, including sports actions such as basketball dunk, cricket bowling, and soccer juggling. The HMDB51 dataset includes around 7,000 videos covering 51 categories and offers a good mix of sports and everyday human actions, which supports general motion learning.

All videos are clearly labelled based on their action type and split into training (70%), validation (15%), and testing (15%) sets to ensure balanced learning and fair evaluation.

3.2 Data Preprocessing

Since video data is exclusively large and computationally heavy, preprocessing becomes a vital step to optimize performance and accuracy. The system performs a series of preprocessing operations:

(i) **Frame Extraction**: Frames are sampled at fixed intervals (for example, 15–30 frames per second) to provide consistent temporal input.

(ii) **Frame Resizing and Normalization**: All frames are resized (commonly to 224×224 pixels) and normalized to the range for compatibility with deep learning networks.

(iii) **Optical Flow Estimation**: Motion vectors are computed between consecutive frames to capture dynamic patterns like player movement or ball trajectory.

(iv) **Data Augmentation**: Random transformations such as flipping, rotation, cropping, and brightness adjustments enrich the dataset and reduce overfitting.

(v) **Label Encoding**: Sports categories are converted into numerical labels so that they can be processed by neural networks.

This stage dramatically improves model robustness, ensuring the system can handle real-world variations and noisy inputs.

3.3 Feature Extraction using Deep Learning Models

The heart of the system lies in its ability to learn meaningful features from video data. Different deep learning architectures are integrated to capture both spatial (appearance-based) and temporal (motion-based) information:

(i) **2D CNNs:** Extract spatial features from individual frames, identifying patterns like player position, field texture, and equipment. Pretrained models such as ResNet50 and MobileNetV2 are fine-tuned on sports videos for efficient feature learning.

(ii) **3D CNNs:** Extend 2D convolutions into the time dimension, allowing the network to analyse frame sequences and understand motion directly. Prominent examples include C3D and I3D, which effectively learn both spatial and temporal cues simultaneously.

(iii) **ConvLSTM Networks:** Integrate convolutional layers with recurrent LSTM units, preserving spatial information while modelling time-dependent motion. This architecture excels in capturing smooth sequences like running, jumping, or swinging.

(iv) **Vision Transformers (ViT) and Timesformer:** Transformer-based architectures divide frames into small patches and use self-attention to capture long-range dependencies. These models often outperform CNNs in recognizing complex actions that involve multiple interacting elements.

3.4 Model Architecture and Training Procedure

The proposed architecture blends both spatial and temporal modeling to achieve a holistic understanding of sports actions.

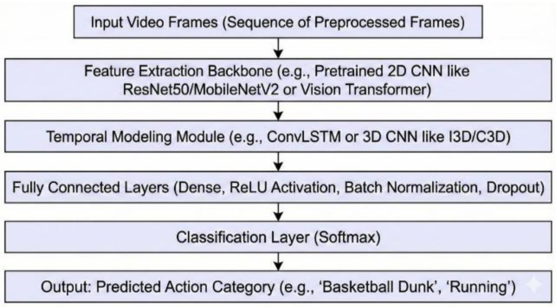
Input Video Frames (Sequence of Preprocessed Frames)
Feature Extraction Backbone (e.g., Pretrained 2D CNN like ResNet50/MobileNetV2 or Vision Transformer)
Temporal Modeling Module (e.g., ConvLSTM or 3D CNN like I3D/C3D)
Fully Connected Layers (Dense, ReLU Activation, Batch Normalization, Dropout)
Classification Layer (Softmax)
Output: Predicted Action Category (e.g., 'Basketball Dunk', 'Running')

Fig 1: Proposed System Architecture

This figure explains a sequence of preprocessed video frames as input and extracts spatial features using a pretrained CNN backbone such as ResNet50 or MobileNetV2. These features are then passed to a temporal modeling module like ConvLSTM or a 3D CNN to capture motion information across frames. The learned spatiotemporal features are refined through fully connected layers. Finally, a softmax classifier predicts the corresponding sports action category.

The proposed system uses a pretrained CNN or Vision Transformer as the feature extraction backbone, where individual video frames are processed to generate rich, high-dimensional feature embeddings. These frame-level features are then passed to a temporal modeling module such as ConvLSTM or a 3D CNN, which learns how motion and actions evolve across consecutive frames. After temporal aggregation, the extracted features are flattened and fed into fully connected layers with ReLU activation, while batch normalization and dropout are applied to improve training stability and reduce overfitting. A final softmax classification layer maps the learned features into discrete sports action categories such as basketball dunk, running, or tennis serve. The model is trained using the Adam optimizer with a learning rate of 1×10^{-4} and dynamic decay, categorical cross-entropy as the loss function, and is trained for 25–50 epochs with a batch size of 16–32, while accuracy and F1-score are used as the primary performance evaluation metrics.

Training is performed on GPU-enabled systems using TensorFlow or PyTorch, enabling fast and efficient processing of high-dimensional video data.

3.5 Real-Time Inference and Deployment

Once trained, the model is integrated into a real-time interface that processes live video streams. This may be implemented through a Streamlit dashboard or an OpenCV-based application.

3.5.1 Workflow for Live Recognition:

In the live recognition workflow, the system continuously captures video frames from a webcam or IP camera feed and preprocesses each frame in real time by resizing and normalizing it to match the model's input requirements. These frames are then passed through the trained deep learning model for inference, where the current sports action is predicted. The predicted action label along with its confidence score is displayed directly on the live video stream, enabling real-time monitoring and analysis.

3.5.2 Deployment Options:

The system supports flexible deployment across multiple platforms. It can be deployed as a local or cloud-based web application using Streamlit or Flask on platforms such as Streamlit Cloud or Cloudflare. For portable and low-latency applications, the model can run on edge devices like Jetson Nano or Raspberry Pi 5. Additionally, the system can be scaled on cloud infrastructure such as AWS, Google Cloud, or Azure to support large-scale sports analytics and multi-stream processing.

3.6 Evaluation Metrics

To ensure robust and fair performance evaluation, several quantitative metrics are used:

(i) **Accuracy:** Measures overall correctness of the model's predictions.
(ii) **Precision, Recall, and F1-Score:** Evaluate class-wise performance and balance between false positives and false negatives.
(iii) **Confusion Matrix:** Provides a visual breakdown of misclassifications across action categories.
(iv) **Frame Rate (FPS):** Indicates how efficiently the system processes frames per second.
(v) **Latency:** Average time taken per frame prediction, targeted to remain below 100 milliseconds for real-time usability.

4. Implementation

The implementation of the Real-Time Sports Action Recognition system is done in a step-by-step manner. Each part of the system plays a specific role, from reading videos to predicting the action. The entire system is built using Python, OpenCV, and deep learning models like ResNet50, MobileNetV2, and LSTM. The following sections explain how the system is put together and how each algorithm works.

4.1 Implementation Setup

The system is developed in Python using libraries such as OpenCV for video frame processing, TensorFlow for deep learning model development, and NumPy and Pandas for efficient data handling. Streamlit is used to build the user interface, while GPU acceleration is employed to speed up model training. The codebase is organized into modular components to ensure easy maintenance and scalability.

4.2 Module 1 - Video Dataset Loading

This part reads the sports videos from different folders.
Each folder name represents one action class (like cricket, football, basketball).

Algorithm - Video Loading

The dataset folder is first read to identify the available action classes based on the folder names. All video file paths are then collected and each file is associated with its corresponding class label. The dataset is subsequently split into training, validation, and testing sets to ensure balanced learning and fair evaluation. This process prepares the data in an organized manner, allowing the model to learn effectively from a wide range of video examples.

4.3 Module 2 - Video Pre-processing

Before giving a video to the model, the video is converted into frames and cleaned.
This helps the model understand the content correctly.

Algorithm - Convert Video to Frames

Each video is opened using OpenCV and a fixed number of frames are read from it. Every frame is resized to a standard resolution of 224×224 pixels and the pixel values are normalized for model compatibility. If a video contains fewer frames than required, the last frame is repeated to maintain consistency. This process ensures that all videos are converted into a uniform input format suitable for deep learning models.

Data Augmentation

Data augmentation is applied to improve the robustness of the model by introducing small variations in the video frames. During this process, frames undergo operations such as horizontal flipping, slight adjustments in brightness or contrast, small rotations, and random cropping. These transformations increase data diversity and help the model generalize better, leading to improved performance on real-world videos.

4.4 Module 3 - Feature Extraction Using Pretrained Models

Models like ResNet50 and MobileNetV2 are used to extract important features from each frame.
These models are already trained on large image datasets, so they understand basic visual patterns.

Algorithm - Extract Features

A pretrained model such as ResNet50 & MobileNetV2 is loaded and its top classification layer is removed to use it as a feature extractor. Each video frame is passed through the model to obtain a feature vector that captures important visual information. The feature vectors from all frames are then combined into a sequential representation, forming a compact and meaningful description of the video content for further temporal analysis.

4.5 Module 4 - Temporal Modeling and Action Classification

After extracting frame features, the model needs to understand movement across frames.
This is done using LSTM model.

Algorithm - Learn Motion Features

- Send the sequence of frame features into the LSTM model.
- The model learns how the action changes over time.

After processing all frames, the model generates one final feature vector for the whole video.

Final Action Prediction

The extracted feature sequence is passed through fully connected dense layers to refine the learned representation. A softmax function is then applied to generate probability scores for each action class. The class with the highest probability is selected as the predicted action, and its corresponding confidence score is returned as the final output.

4.6 Module 5 - Training the Complete Model

Training teaches the model to recognize actions correctly.
Videos are passed repeatedly through the system during training.

Algorithm - Model Training

Set up CNN, LSTM, and final classifier layers.
Pick settings like batch size, learning rate, and number of epochs.

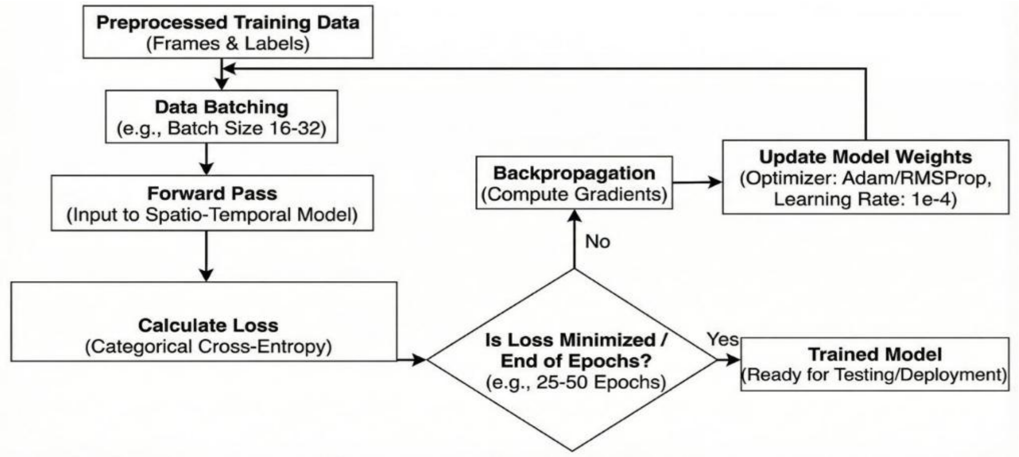
Preprocessed Training Data
(Frames & Labels)
Data Batching
(e.g., Batch Size 16-32)
Forward Pass
(Input to Spatio-Temporal Model)
Calculate Loss
(Categorical Cross-Entropy)
Is Loss Minimized / End of Epochs?
(e.g., 25-50 Epochs)
No
Yes
Backpropagation
(Compute Gradients)
Update Model Weights
(Optimizer: Adam/RMSProp, Learning Rate: 1e-4)
Trained Model
(Ready for Testing/Deployment)

Fig-2: Model Training Procedure

The figure illustrates the model training workflow where preprocessed video frames and labels are grouped into batches and passed through the spatio-temporal network during the forward pass. The loss is computed using categorical cross-entropy and gradients are calculated through backpropagation

For each epoch:

Convert videos to frames → Extract features → Run through temporal model → Compare prediction and true label →Update weights → Save the best-performing model.

This continues until the model reaches good accuracy.

4.7 Module 6 - Real-Time Action Recognition

In real-time mode, the system reads frames from a webcam or video file and predicts the action continuously.

Algorithm - Live Prediction

Load the trained model → Start the webcam or open a video file → Keep collecting frames into a buffer.

When the buffer is full:

Preprocess frames → Extract features → Predict action → Show the action label on the frame.

Continue until the user stops the system. This creates a real-time action detection experience.

IV. Results

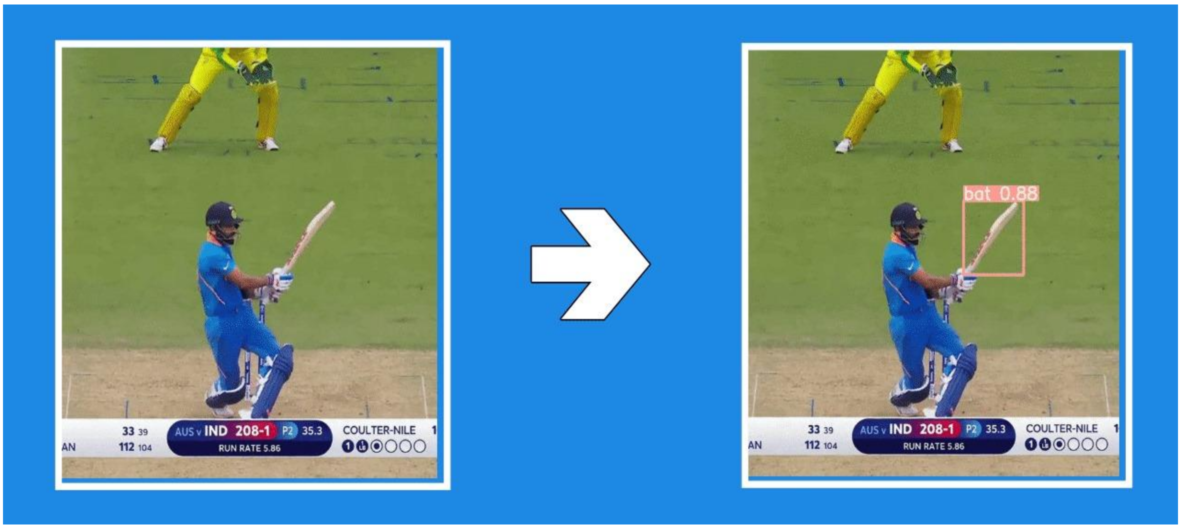

Fig-3: Shows the model detecting a cricket batting action and labelling it in real time with a bounding box

This figure shows the conversion of a raw cricket video frame into an annotated output where the bat is detected using a bounding box. The confidence score highlights the accuracy of object detection. This step supports better understanding of player interaction with sports equipment. It also improves action recognition reliability.

Bowled Cover drive Defence

Pull shot Reverse sweep

Fig-4: Displays correct classification of various cricket shots such as bowled, cover drive, defence, pull shot, and reverse sweep.

This figure demonstrates the model's ability to recognize and classify various cricket shots such as bowled, cover drive, defence, pull shot, and reverse sweep. Each frame represents a distinct batting action. The results show effective differentiation between visually similar movements. This confirms robust spatiotemporal learning.

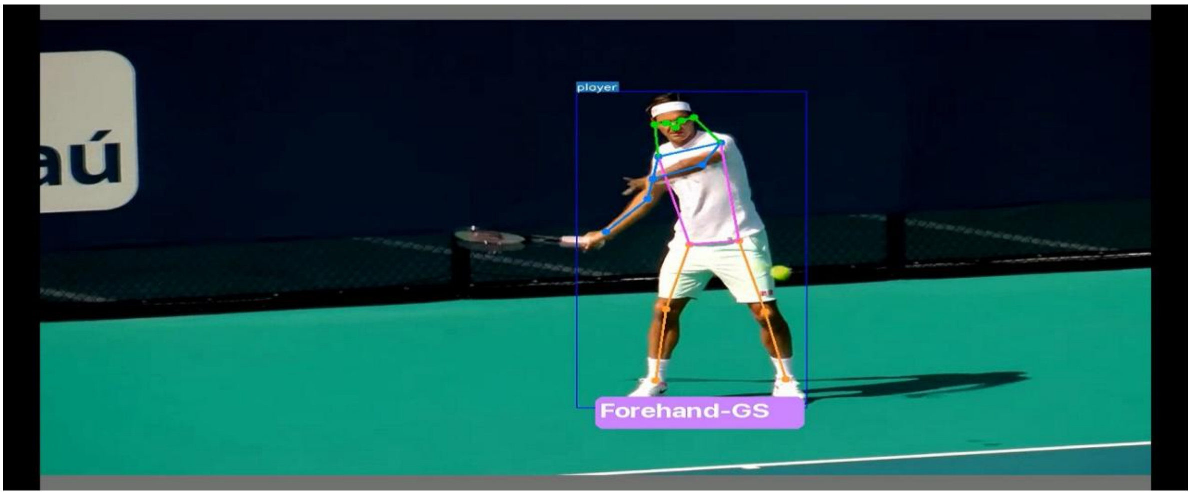

Fig-5: Shows pose-based tennis action recognition where the system detects the player and labels the stroke as Forehand-GS.

This figure presents pose-based tennis action recognition where the player is detected and skeletal keypoints are tracked. The system accurately classifies the action as a forehand groundstroke. Joint-level motion analysis helps capture fine-grained movement details. This improves recognition accuracy in complex sports actions.

V. Conclusion

This project set out to solve the challenge of recognizing sports actions in real-time, and the results were a success. I utilized a mix of deep learning techniques, pairing spatial models like ResNet50 with temporal ones like LSTMs to "watch" the video and understand not just what is in the frame, but how it is moving.

One of the biggest wins was using pretrained models. This allowed me to get high accuracy without needing a supercomputer to train the system from scratch. Through rigorous testing, I found the system held up well even when the camera moved or the background got cluttered.

VI. Future Enhancement

The real-time recognition system created in this project offers a strong base for sports analytics, yet it can be expanded in several meaningful ways. A major upgrade would involve shifting to advanced video processing models (like Transformers) that handle long sequences of movement better than standard networks. This would significantly boost performance in complex scenarios where players move rapidly.

Another key improvement would be the addition of body tracking. By analyzing how a player's joints move, the system could detect fine details in their technique, making it a powerful tool for training and injury prevention. Beyond just seeing physical actions, the system could evolve to understand game context identifying moments like goals, fouls, or specific strategies.

For practical use, the model needs to be optimized to work on everyday devices like mobile phones or smart cameras. This involves compressing the data so it doesn't require high-end hardware. Furthermore, mixing video data with audio or depth sensors would improve accuracy in crowded or noisy environments. Eventually, this system could serve as an automated coaching assistant, a support tool for referees, or a new feature for fitness apps and TV broadcasting.

REFERENCES

[1]. N. A. Rahmad, M. A. As'ari, N. F. Ghazali, N. Shahar, and N. A. J. Sufri, "A survey of video-based action recognition in sports," Indonesian Journal of Electrical Engineering and Computer Science, vol. 11, no. 3, pp. 987–993, 2018. https://ijeecs.iaescore.com/index.php/IJEECS/article/view/11232/9141

[2]. M. Ghislieri, G. L. Cerone, M. Knaflitz, and V. Agostini, "Long short-term memory (LSTM) recurrent neural network for muscle activity detection," Journal of NeuroEngineering and Rehabilitation, vol. 18, art. 153, 2021. https://jneuroengrehab.biomedcentral.com/articles/10.1186/s12984-021-00945-w#Sec1

[3]. J. Lee and H. Jung, "TUHAD: Taekwondo Unit Technique Human Action Dataset with key frame-based CNN action recognition," Sensors, vol. 20, no. 17, art. 4871, 2020. https://www.mdpi.com/1424-8220/20/17/4871

[4]. J. Carreira and A. Zisserman, "Quo Vadis, Action Recognition? A new model and the Kinetics dataset," in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), 2017, pp. 4724–4733. https://arxiv.org/abs/1705.07750

[5]. C. Dai, X. Liu, and J. Lai, "Human action recognition using two-stream attention based LSTM networks," Applied Soft Computing, vol. 86, art. 105820, Jan. 2020. https://ieeexplore.ieee.org/document/8777182

[6]. V. Ramanathan et al., "Detecting events and key actors in multi-person videos," in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), 2016, pp. 3043–3053. https://arxiv.org/abs/1511.02917

[7]. C. Ma, J. Fan, J. Yao, and T. Zhang, "NPU RGBD dataset and a feature-enhanced LSTM-DGCN method for action recognition of basketball players," Applied Sciences, vol. 11, no. 10, art. 4426, 2021. https://www.mdpi.com/2076-3417/11/10/4426

[8]. J. Xiong et al., "Object-level trajectories based fine-grained action recognition in visual IoT applications," IEEE Access, vol. 7, pp. 103629–103638, 2019. https://www.sciencedirect.com/science/article/pii/S2405959520300114?via%3Dihub#sec2

[9]. S.-W. Sun, Y.-L. Chen, C.-C. Lin, and Y.-C. Liao, "Baseball player behavior classification system using long short-term memory with multimodal features," Sensors, vol. 19, no. 6, art. 1425, 2019. Baseball Player Behavior Classification System Using Long Short-Term Memory with Multimodal Features

[10]. T. Moodley and D. van der Haar, "CASRM: Cricket automation and stroke recognition model using OpenPose," in Digital Human Modeling and Applications in Health, Safety, Ergonomics and Risk Management. Posture, Motion and Health (HCII 2020), Lecture Notes in Computer Science, vol. 12198, Springer, 2020, pp. 67–78. https://link.springer.com/chapter/10.1007/978-3-030-49904-4_5

[11]. Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh, "Realtime multi-person 2D pose estimation using Part Affinity Fields," in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), 2017, pp. 7291–7299. https://arxiv.org/abs/1611.08050

[12]. K. Xia, J. Huang, and H. Wang, "LSTM-CNN architecture for human activity recognition," IEEE Access, vol. 8, pp. 56855–56866, 2020. https://ieeexplore.ieee.org/abstract/document/9043535

[13]. K. Peppas, A. C. Tsolakis, S. Krinidis, and D. Tzovaras, "Real-time physical activity recognition on smart mobile devices using

convolutional neural networks," Applied Sciences, vol. 10, no. 23, art. 8482, 2020. https://www.mdpi.com/2076-3417/10/23/8482

[14]. Z. Tu et al., "Multi-stream CNN: Learning representations based on human-related regions for action recognition," Pattern Recognition, vol. 79, pp. 32–43, 2018. https://www.sciencedirect.com/science/article/abs/pii/S0031320318300359?via%3Dihub

[15]. E. P. Ijjina and C. K. Mohan, "Hybrid deep neural network model for human action recognition," Applied Soft Computing, vol. 46, pp. 936–952, 2016. https://www.sciencedirect.com/science/article/abs/pii/S1568494615005268?via%3Dihub