Problem Statement

- **What specific problem are you solving?**

  We're developing an AI-powered VAR system tailored for boxing that reviews and analyzes fight footage in real-time, identifying key actions like punches, blocks, and misses supporting fairer referee decisions and offering athletes accurate performance insights.

- **Why is this problem important in sports/media?**

  This issue matters because boxing lacks a standardized VAR system—unlike football or cricket—which has repeatedly led to controversial and biased decisions, like the infamous 2012 Pacquiao vs. Bradley fight. As a result, fans distrust outcomes and stakeholders face credibility challenges—highlighting an urgent need for accurate, AI-powered review tools in the sport .

Solution

- **Quick explanation of how your solution works**

  The solution captures body landmarks using MediaPipe, creates time-based sequences, and uses a trained LSTM model to classify motion gestures in real time, displaying the predicted label and confidence on screen.

- **Data sources and tools leveraged**

  Dataset: A CSV file based on the Olympic punch classification dataset, containing pose landmark coordinates and labels for different boxing motions.

  TensorFlow: Used to build and train the LSTM model for sequential motion classification.

  OpenCV: Handles real-time video capture (from webcam or file) and displays predictions on screen.

  MediaPipe: Extracts human body pose landmarks from video frames.

  NumPy, Pandas: Used for data loading, preprocessing, and numerical operations.

  Scikit-learn: Used for label encoding, train-test splitting, class weighting, and model evaluation metrics

Real World Application

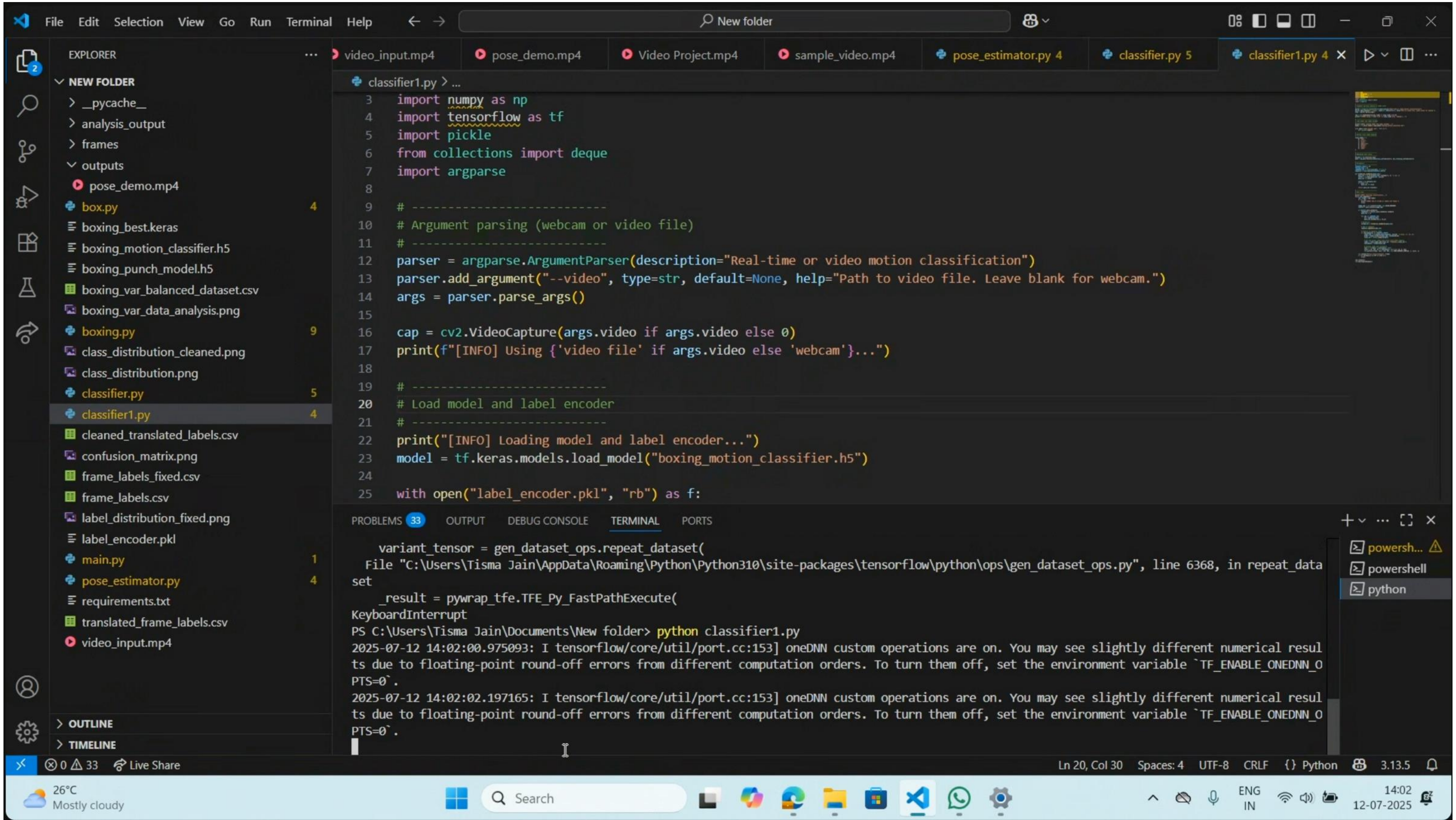
- **Where could this be applied in actual sports workflows?**

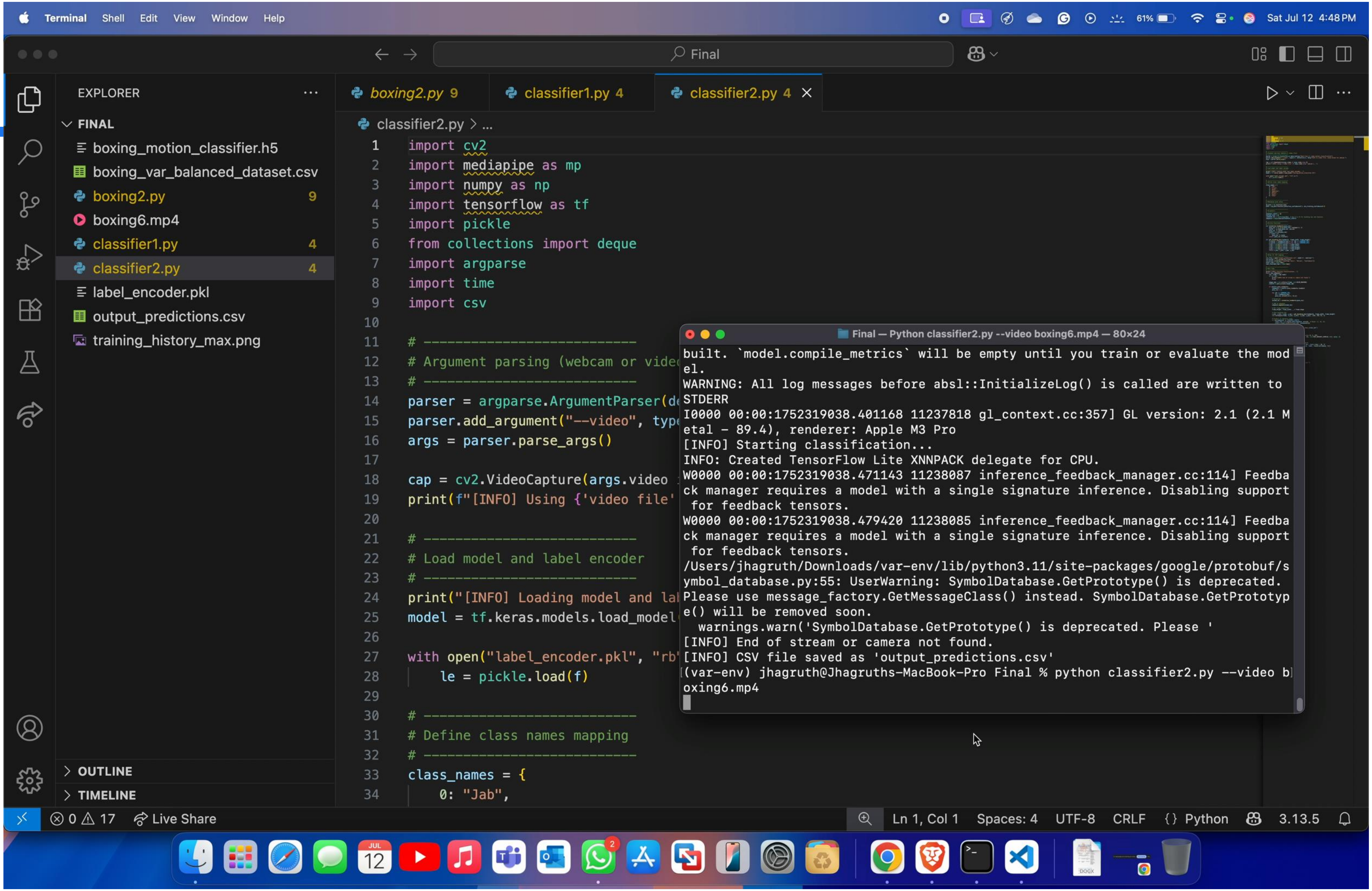
  This could be applied in ringside review systems, post-fight analysis, judging audits, or coaching sessions enhancing fairness, training, and storytelling in the sport. Additionally could be the basis of online spars.

- **Who benefits? (e.g., fans, analysts, broadcasters, athletes)**

  Athletes, judges, coaches, broadcasters, fans and even general public all gain from clearer decisions, richer analysis, and more engaging storytelling around every fight.

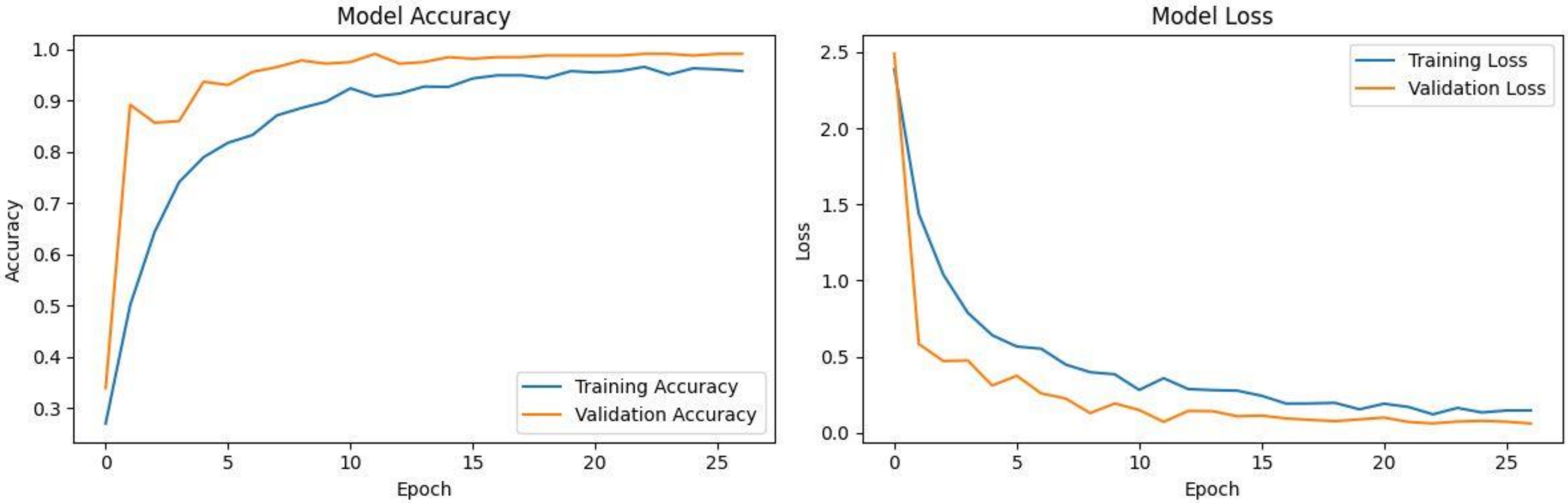
Demo


File Edit Selection View Go Run Terminal Help
New folder
EXPLORER
NEW FOLDER
__pycache__
analysis_output
frames
outputs
pose_demo.mp4
box.py
boxing_best.keras
boxing_motion_classifier.h5
boxing_punch_model.h5
boxing_var_balanced_dataset.csv
boxing_var_data_analysis.png
boxing.py
class_distribution_cleaned.png
class_distribution.png
classifier.py
classifier1.py
cleaned_translated_labels.csv
confusion_matrix.png
frame_labels_fixed.csv
frame_labels.csv
label_distribution_fixed.png
label_encoder.pkl
main.py
pose_estimator.py
requirements.txt
translated_frame_labels.csv
video_input.mp4
OUTLINE
TIMELINE
video_input.mp4
pose_demo.mp4
Video Project.mp4
sample_video.mp4
pose_estimator.py 4
classifier.py 5
classifier1.py 4
classifier1.py > ...
import numpy as np
import tensorflow as tf
import pickle
from collections import deque
import argparse
# ---------------------------
# Argument parsing (webcam or video file)
# ---------------------------
parser = argparse.ArgumentParser(description="Real-time or video motion classification")
parser.add_argument("--video", type=str, default=None, help="Path to video file. Leave blank for webcam.")
args = parser.parse_args()
cap = cv2.VideoCapture(args.video if args.video else 0)
print(f"[INFO] Using {'video file' if args.video else 'webcam'}...")
# ---------------------------
# Load model and label encoder
# ---------------------------
print("[INFO] Loading model and label encoder...")
model = tf.keras.models.load_model("boxing_motion_classifier.h5")
with open("label_encoder.pkl", "rb") as f:
PROBLEMS 33
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
variant_tensor = gen_dataset_ops.repeat_dataset(
File "C:\Users\Tisma Jain\AppData\Roaming\Python\Python310\site-packages\tensorflow\python\ops\gen_dataset_ops.py", line 6368, in repeat_data
set
_result = pywrap_tfe.TFE_Py_FastPathExecute(
KeyboardInterrupt
PS C:\Users\Tisma Jain\Documents\New folder> python classifier1.py
2025-07-12 14:02:00.975093: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly different numerical resul
ts due to floating-point round-off errors from different computation orders. To turn them off, set the environment variable `TF_ENABLE_ONEDNN_O
PTS=0`.
2025-07-12 14:02:02.197165: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly different numerical resul
ts due to floating-point round-off errors from different computation orders. To turn them off, set the environment variable `TF_ENABLE_ONEDNN_O
PTS=0`.
powersh...
powershell
python
Live Share
Ln 20, Col 30
Spaces: 4
UTF-8
CRLF
Python
3.13.5
26°C
Mostly cloudy
Search
ENG
IN
14:02
12-07-2025



Terminal Shell Edit View Window Help
Sat Jul 12 4:48 PM
Final
EXPLORER
FINAL
boxing_motion_classifier.h5
boxing_var_balanced_dataset.csv
boxing2.py 9
boxing6.mp4
classifier1.py 4
classifier2.py 4
label_encoder.pkl
output_predictions.csv
training_history_max.png
OUTLINE
TIMELINE
boxing2.py 9
classifier1.py 4
classifier2.py 4
classifier2.py > ...
import cv2
import mediapipe as mp
import numpy as np
import tensorflow as tf
import pickle
from collections import deque
import argparse
import time
import csv
# ---------------------------
# Argument parsing (webcam or vide
# ---------------------------
parser = argparse.ArgumentParser(de
parser.add_argument("--video", typ
args = parser.parse_args()
cap = cv2.VideoCapture(args.video
print(f"[INFO] Using {'video file'

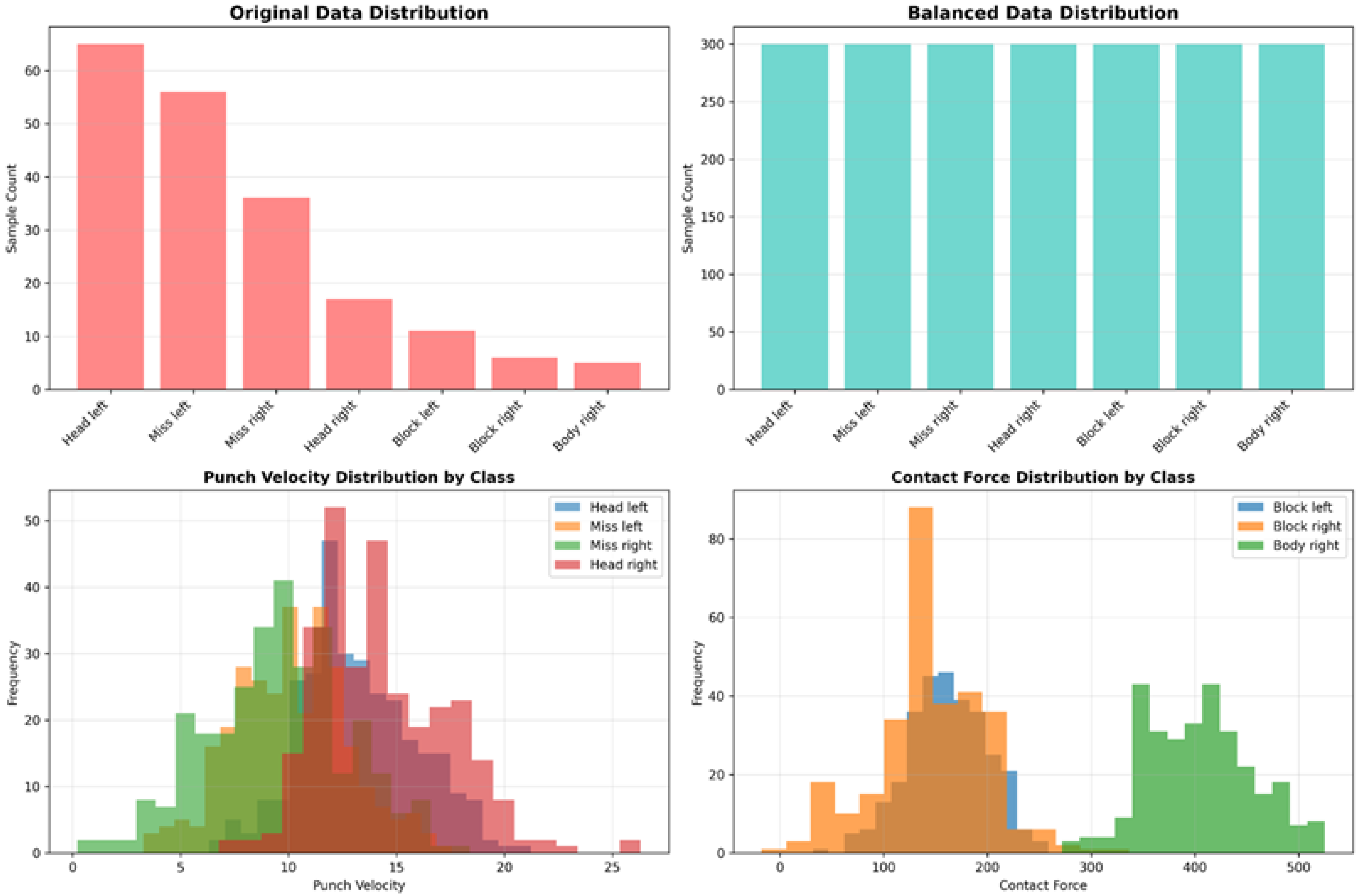
Load model and label encoder
----
print("[INFO] Loading model and lal
model = tf.keras.models.load_model
with open("label_encoder.pkl", "rb'
le = pickle.load(f)
# ---------------------------
# Define class names mapping
# ---------------------------
class_names = {
0: "Jab",
Final — Python classifier2.py --video boxing6.mp4 — 80×24
built. `model.compile_metrics` will be empty until you train or evaluate the model.
WARNING: All log messages before absl::InitializeLog() is called are written to STDERR
I0000 00:00:1752319038.401168 11237818 gl_context.cc:357] GL version: 2.1 (2.1 Metal - 89.4), renderer: Apple M3 Pro
[INFO] Starting classification...
INFO: Created TensorFlow Lite XNNPACK delegate for CPU.
W0000 00:00:1752319038.471143 11238087 inference_feedback_manager.cc:114] Feedback manager requires a model with a single signature inference. Disabling support for feedback tensors.
W0000 00:00:1752319038.479420 11238085 inference_feedback_manager.cc:114] Feedback manager requires a model with a single signature inference. Disabling support for feedback tensors.
/Users/jhagruth/Downloads/var-env/lib/python3.11/site-packages/google/protobuf/symbol_database.py:55: UserWarning: SymbolDatabase.GetPrototype() is deprecated. Please use message_factory.GetMessageClass() instead. SymbolDatabase.GetPrototype() will be removed soon.
warnings.warn('SymbolDatabase.GetPrototype() is deprecated. Please '
[INFO] End of stream or camera not found.
[INFO] CSV file saved as 'output_predictions.csv'
(var-env) jhagruth@Jhagruths-MacBook-Pro Final % python classifier2.py --video boxing6.mp4
0 17 Live Share
Ln 1, Col 1 Spaces: 4 UTF-8 CRLF {} Python 3.13.5


Model Accuracy
Accuracy
Epoch
Training Accuracy
Validation Accuracy
Model Loss
Loss
Epoch
Training Loss
Validation Loss

punch_velocity	hand_acceleration	deceleration_rate	peak_velocity	velocity_profile_smoothness	horizontal_angle	vertical_angle	punch_distance	hand_trajectory_x	hand_trajectory_y	hand_trajectory_z	target_precision
13.741785382528082	17.585207096486446	16.61922134525173	17.04605971281605	0.8265846625276664	33.126904344406555	9.737638446522174	0.7267434729152908	0.5295788421097571	1.8042560043585965	0.262926584575003	0.8534270246429744
12.466256938155166	14.8268672131323	17.05636228025797	11.558312700057956	0.8708863595004755	19.322639008961794	1.0154418533047087	0.6696861235869124	0.7107699869993116	1.767136828118997	0.29074813740894073	0.8698896304410711
14.531314555985496	22.06872008571247	14.819974696049165	16.007065795784047	0.8861636025047634	29.839041963159005	6.084186816525242	0.8038036566465969	0.5946260941335072	1.9064643655814006	0.09042039167282043	0.9821902504375224
13.24030069266144	18.783165816539668	15.012783641606152	13.530825733249706	0.7084629257949585	31.63483741787713	3.9718564504196916	0.5697722730778381	0.5758071432500986	1.7904050856814537	0.4508948720968424	0.9174577812831839
12.749128412719102	16.489573037651404	11.123341422334668	14.137125949612054	0.7437696286273895	38.788739445081454	2.2417272972985907	0.804993440501754	0.4825120061495644	1.7177938483794324	0.3650813773895736	0.7769135683566045
10.063295824431696	20.361253811227357	17.89648894751851	12.358635363296578	0.9463376129244322	38.30224741549198	7.4661804799834695	0.8396792982653948	0.5631917825995694	1.674626383564251	0.22883884562995813	0.8184189715034562
13.03523436032551	14.262783663864036	15.432952314627954	14.770634759457673	0.7616142563798867	36.22980084756422	5.174626155337999	0.5357029702169377	0.6536681040522425	1.8060784526368234	0.38664409945402217	1.0053802052034904
11.320170335526416	21.2668517909021	15.160700047738658	11.844510444141388	0.7784696290740032	40.4367819914774	2.8089001048485898	0.6716458589581975	0.606835775985572	1.6848399652394184	0.47151552714602607	0.9633919022318012
15.667277872966556	15.876991603143658	16.109548570365572	15.549268106858673	0.7573069528421916	34.5237971515056	-4.723802020207218	0.547561235866571	0.562114777291026	1.625221681803515	0.4305929043145308	0.7569858622039367
13.202479669337581	16.13190144053822	14.479694374106812	13.013998130682335	0.7910635243055788	41.7968167761682	6.071046457895142	0.5807090404739346	0.7349399813149876	1.780729952087661	0.36502896950711683	0.9629628841923612
12.742489912481796	19.785471076310742	12.954448291916318	18.18477455137092	0.7493982618500298	25.286491097698143	8.474332620500203	0.7291662693962936	0.6936179725578232	1.812834550926428	0.2990202581722468	0.8102745628514169
14.644149058005048	17.520184410109717	14.952459480243277	11.994941270724382	0.848148686400761	32.690730888638896	5.968155681014268	0.5672769056447677	0.6779019771361758	1.9032738913002578	0.2912991881234514	0.9401711722098942
10.602168346115755	18.451181359428624	15.854389939442898	17.752341678431772	0.9450423838186051	30.384770754700774	2.3047559859549263	0.6991919171506507	0.4019650189469036	1.9331458765854355	0.39435520965770293	0.8530824347895295
16.37787993880631	18.347023902878576	17.948242960159565	14.135036962820218	1.0560747924881988	49.042726739545635	4.25310755456278	0.7471570950954356	0.6968063924377721	1.886863155753235	0.22280612315359163	0.9686051459998439
11.215332706658266	14.822359434333144	14.84330225681707	15.910284641002477	0.7514273953664456	39.03237212414275	3.409227144882678	0.5707127167737656	0.5839454460068163	1.6464757677580626	0.2557080555722254	0.7802122107411151
12.180206021298083	15.133378678198723	10.98388419935607	14.406927271734446	0.7743649254715697	23.621970323218605	3.0602813472724204	0.5418451996385605	0.8530712452608846	1.838163975694945	0.2993621886946706	1.0479944138890027
11.902629882833976	15.277309013875207	13.558071673579168	15.510782451651512	0.9000917187624381	27.179558041611593	5.297996916287677	0.725138712337179	0.3495892078317942	1.8043360192379936	0.24699009928433224	0.957059866859316
6.320888749681776	15.60931423358857	16.44268031795135	13.593909227914015	0.8871145873371309	30.168118506273437	5.2597693624187	0.634432276460792	0.775167309248971	1.7754420843301213	0.3270082129660162	0.8588123033877533
13.089036395271414	20.3125955816609	11.303534385550396	16.287508086413858	0.8838496407494414	31.67769668879359	6.898345598318855	0.8770692857804396	0.6272799382587743	1.7748220586300336	0.26325112803678047	0.8150155630535209
8.053199377739304	22.48813293446755	16.635914140885145	13.88883065817909	0.877996862631982	25.996087621612986	12.337255938850479	0.6629221181975228	0.6164092191907339	1.8225766623898692	0.33848073853893706	0.9223884024279132
11.66053825177468	23.007064575868178	14.351021621590988	10.993714093763788	0.8254256935914055	32.81821144201864	-3.0906599288247154	0.6445705133482188	0.5653598204686983	1.8196206364813419	0.44791648759562763	1.0126565029547758
11.643281014799129	16.885677402012625	11.481220763205446	12.44436662481825	0.7389424154534171	49.01816354738898	7.807035179442384	0.7715550949941 59	0.7082508096064852	1.6370948228782771	0.2580383786976181	0.9489374561227918
8.608427066190242	19.81802985403692	11.798926618759292	17.509588363968728	0.6418070592116277	48.571650946320304	5.633052401607856	0.6403286888129608	0.5182621369786575	1.789913611435207	0.2969892238060121	1.0103301882016522
13.14007433578469	20.948072951836544	19.16368611115644	16.02874013003626	0.6659125768668355	24.763384266114386	3.1255442669129634	0.6526091050210834	0.6776488530703684	1.6774256186846535	0.31494134115816624	0.8244617067646678
10.932582355530581	23.437345673990787	16.769379838638688	12.875066448211465	0.913240773905552	42.78043559701384	6.865429886651589	0.4929775280109544	0.4909294236262697	1.7252481364448495	0.29404532567199626	0.9620672097550678
14.845709514939994	16.45186581534788	15.240301942352458	13.075449422589916	0.8065503772567684	32.52662301225089	5.6664013149011385	0.6021251378336523	0.7883634188360281	1.6605392697780497	0.28505026846709125	0.8560268941725825
9.98964808311983	15.696607304687614	14.913287781533047	14.468429465073042	1.0050500492814076	27.013167674089672	7.952967195429752	0.6286011155774491	0.592580443552135	1.8174819492166603	0.2101822382750161	0.938240974618405
10.968028273789034	16.83689532019205	15.714663476812262	14.668913579974049	0.9158544272672831	51.0816363101308	4.469158317517852	0.5701702755461546	0.3931021157978209	1.676906996005808	0.29734984217009625	1.0794557863517789
12.082204799207863	18.440141059299968	18.016272416270894	12.366128658025527	0.8868673308872901	31.853289501381113	5.086234468804452	0.7778451862607298	0.6286648602029855	1.7546436548156148	0.19121150872160644	0.9746253566027221
16.413810073085852	17.802749216781038	13.612001183267004	17.76231413888118	0.7051986099583756	17.409552347039345	6.320043350159997	0.5997945775647389	0.446815077430393	1.8208356447299354	0.3195040571016959	0.8435921369263272
14.10680689966886	21.987457590397295	15.491302924253675	15.418007515177024	0.8410264305712774	46.52093772359578	2.970823093822191	0.8300940432910816	0.5939763074033485	1.6069224897881953	0.31024835319286315	0.831894834252512
11.882056543698507	15.954047256006604	12.495949975262711	13.43779941422809	1.0297686526849523	40.12674289013608	3.2864630306516096	0.7072582781356159	0.8099033154879003	1.842463368291277	0.30477042959361395	0.8353063222294427
16.818857925251777	19.308971009022095	15.0950086954205	14.240062653436523	0.9113517972730416	26.817659478412757	4.227870387729965	0.4831415926122411	0.6598834683907918	1.814719593970274	0.2613450830160501	1.0573986763290033
12.436114822521386	21.518187057778093	16.35900038648608	13.258771335827788	0.9271698710648344	12.211659034959396	8.446297101116311	0.47602862212219005	0.5456338587895302	1.6380330105373857	0.1964254819423275	1.016082678737852
11.852394635448242	17.410950452745066	14.820996851464841	13.925555526980316	0.9227629543636979	35.415567086458395	7.197920231646737	0.6419283419891418	0.6117952785474136	1.5501799315467892	0.3733062139761985	0.9346488475897993
15.26381483239496	21.56109091668114	16.59682555507296	11.713990174539033	1.0133431532301096	25.829236855850496	5.9079063957171	0.5745724149980642	0.5903792480110511	1.782876241030348	0.3257085772360218	0.9421920754235285
10.902595630776243	14.030730619777504	19.105037900340925	16.01963417868527	0.7811849654822276	53.01948644275447	7.945296460974783	0.6175168616491378	0.22508914276896402	1.9790942572590742	0.18883420265891232	0.7354601253043108
10.025929658574157	20.82231356396465	12.543781516154226	13.550733700005951	0.9050052099024551	27.253244361090918	5.31612651843788	0.5165974505081796	0.5097948534073179	1.7819781934142538	0.17256050131746645	0.9440474737960399
10.81627343939033	18.836982486671406	12.911632366850608	18.290298255546777	0.7312401580970487	37.47856568181622	6.901330642796296	0.6913799097451498	0.5722068511683437	1.7370179300266573	0.3035049177583288	0.8852997998371511

motion_log

Timestamp_Minutes	Motion_Type	Confidence
0.0089	Block	0.652
0.0178	Block	0.812
0.0267	Block	0.85
0.0356	Block	0.825
0.0444	Block	0.886
0.0533	Block	0.746
0.0622	Block	0.803
0.0711	Uppercut	0.368
0.08	Block	0.734
0.0889	Block	0.779
0.0978	Block	0.756
0.1067	Block	0.608
0.1156	Block	0.623
0.1244	Block	0.661
0.1333	Duck	0.906
0.1422	Duck	0.899
0.1511	Duck	0.476
0.16	Cross	0.453
0.1689	Cross	0.533
0.1778	Uppercut	0.256




Person 1: Idle (0.91)
Person 2: Uppercut (0.30)
gettyimages
685549039

What's Next?

- **How could your project be scaled or extended?**

  Our project is to be scaled by refining the model with more diverse data, adding real-time capabilities, and building modular tools that can be demoed to sports orgs, broadcasters, or tech partners for immediate feedback and piloting.

- **Ideas for future features, platforms, or integrations**

  Future extensions include real-time multi-camera analysis, integration with broadcast platforms for automated replays, and mobile tools for referees or fans to access instant, contextual insights during matches

- **Potential impact beyond the hackathon**

  Our project can be scaled by expanding to other combat sports like MMA or wrestling, integrating with live broadcast systems for real-time analysis, or building athlete-specific analytics dashboards. It can also be extended to training environments for performance feedback and coaching insights.