A SMART GENERALIZED FRAMEWORK FOR DUAL-VIDEO POSE ESTIMATION COMPARISON FOR SOCIETAL AND BEHAVIORAL ANALYSIS USING ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

Tyler Kaiyang Chen [1], Yen-Hao Wang [2]

[1] Northwood High School, 4515 Portola Parkway, Irvine, CA 92620
[2] Computer Science Department, California State Polytechnic University, Pomona, CA 91768

ABSTRACT

This paper presents the design, implementation, and evaluation of a Golf Swing Analyzer, a low-cost, accessible system that delivers real-time feedback on golf swing mechanics [1]. Our system leverages MediaPipe for pose estimation, and a rule-based machine learning model training on labeled golf swing images to assess the swing based on parameters like elbow stability and shoulder posture. The backend, built with Python Flask, processes user inputs and runs swing analysis while the frontend provides an intuitive interface for ease of use [2].

To validate our approach, we conducted an experiment with 20 diverse swing images, which highlighted issues such as image blur, incorrect camera angles, and background distractions that impacted prediction accuracy. Compared to existing methods using expensive motion capture systems, or deep neural network-based analysis, our approach is faster, more accessible, and does not require expensive equipment or large training datasets.
By improving accessibility and affordability, Perfect Pivot enables golfers of all skill levels to refine their swing, making golf coaching more inclusive and improving their technique more conveniently.

KEYWORDS

Golf Swing Analysis, Pose Estimation, Machine Learning, Real-Time Feedback

1. INTRODUCTION

Golf has been a sport that has grown in great interest over recent years. There has been a 30% increase in golfers since 2016. But at times it may seem like an exclusive sport only accessible to the privileged. This is somewhat true, all the club fees, the lesson fees, and the course fees pile up quickly compared to other sports. However, we believe that sports should have no barriers, and everyone should have access to participate in their desired sport and fit within their budget. We sith to offer people who are interested in learning golf a pathway into the sport without needing to spend hundreds of dollars on coaches without being ready to commit fully.

David C. Wyld et al. (Eds): SEC, DMA, AIAPP, CYBI, CoSIT, SIGL, NLPML – 2025
pp. 175-185, 2025. CS & IT - CSCP 2025 DOI: 10.5121/csit.2025.150813

We analyzed three methodologies addressing golf swing analysis and pose estimation. The first paper by Bourgain, et al. focused on golf swing mechanics using detailed kinematic analysis in controlled environments. While comprehensive, this method relies on specialized motion tracking equipment and lacks real-time feedback. Our system improves accessibility by offering near-instant analysis using a simple phone camera.

The second methodology compared Mediapipe and YOLO libraries for pose estimation [3]. While YOLO provides slightly higher accuracy, it requires higher computing resources. Mediapipe strikes a balance by offering better processing time without unnecessary multi-object detection overhead.
The third study by Liao, et al., presented an unsupervised neural net approach for self-training, which compares user swings with professional models. However, it requires extensive training data and computing power. Our rule-based approach offers faster and more accessible analysis for everyday users without the need for large datasets.

Perfect Pivot is an AI Golf Coach that provides golf instruction to users through uploading images [4]. We seek to offer users a simple yet effective system to understand and improve their golf swings. This app is free of charge and offers a detailed breakdown of mistakes and suggestions on how to fix it. Furthermore, we've included a list of common mistakes people tend to make in golf and a list of tips to improve your swing. We believe that by offering beginner-friendly instruction we can attract more eager golfers to the game without needing to spend large sums immediately. Our app will help users establish a strong foundation in their golf swing until they are ready to spend money for personalized improvements.

We began by dividing the images and their corresponding labels into two equal sets - one for training the model from scratch and the other for testing its inference capabilities. This approach ensures an objective evaluation of the model's accuracy while assessing its sensitivity to external factors such as image blur, improper camera angles, and background spectators as distractions.

2. CHALLENGES

In order to build the project, a few challenges have been identified as follows.

2.1. Good and Bad Swing

Q: How does your algorithm differentiate between a good and a bad swing?

A: Our model identifies several key features of a person in the image and uses them as points to calculate and determine whether this swing reflects a foundationally sound swing. Our model has data from a variety of different good and bad swings. We understand that good swings come in many different variations, however, our model shows that the key points of these swings all fall within a range that is mechanically and biologically sound. Since we take into account multiple keypoint relations, our model can use multiple groups of data to determine the output

2.2. Subtle Movements

Q: You can have a swing that looks horrible, but it works in real life.

Q: There might be some subtle movements that are hard to capture on camera, how do you solve this problem?

A: These questions are out of the scope of this project. This project tries to apply machine learning to perform analysis on golf swings, and there could be outliers that make it difficult to automate this process. We focus on the more conventional swings targeting the general audiences rather than world class golf players which would benefit more from private instructors. Furthermore, we believe that there is no "perfect" golf swing, but there is an ideal golf swing that has been developed scientifically over the decades [9]. Our goal is not to promote a specific type of swing, rather help golfers develop a foundation they can build off of.

2.3. No Result

Q: The user uploaded a picture, but not getting the analysis back. What are some reason why this is happening?

A: This can happen for a few reasons:

1. Blurry picture/motion blur: Our model needs to identify key body components to perform analysis, therefore we might have trouble identifying these components from a blurry picture. We recommend taking this picture at a higher shutter speed on continuous burst which will require a third-party app, or use the Video feature in the Camera app and set frames per second to 120 if possible.
2. Incorrect camera angle: Our model only takes in the front-on view on the golf swing, we cannot perform analysis from a picture taken from behind the golfer. We suggest standing directly in front of the user when recording this video
3. Background Color: If the person in a picture is wearing similar colors to the background, our model will have trouble identifying the location and features of the person
4. Chaotic Background: In rare instances, our model will fail to identify the user if there are other people immediately next to them (people in the background are fine). We suggest recording in a place where the user is able to stand out from the rest of the picture.

3. SOLUTION

The user is first met with the home page, where it shows the menu tabs of other pages (About, Analyze Swing), and some instructions on how to use this application. The user can then navigate to the Analyze Swing page, select a golf swing image from their phone's photo album, upload the image to our backend server that performs the analysis on this image, which will return the predicted result back to the user and shown onto the screen.

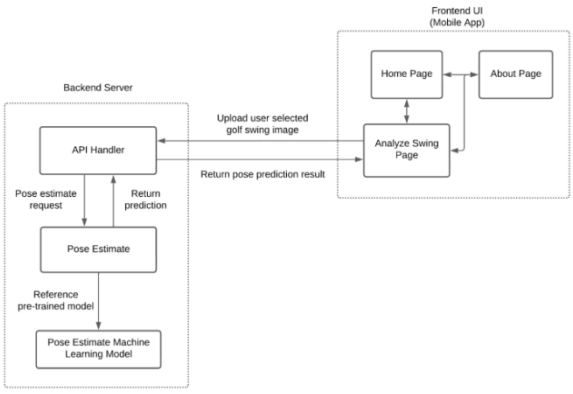


Figure 1. Overview of the solution

Frontend UI, Analyze Swing Page

Purpose: the user can navigate to this page, choose a golf swing pic, upload, and show the predicted result.

The swing analysis page allows users to upload an image file from their photo album into our servers and perform a swing analysis [10]. The swing will then be analyzed by our algorithms to provide swing feedback.

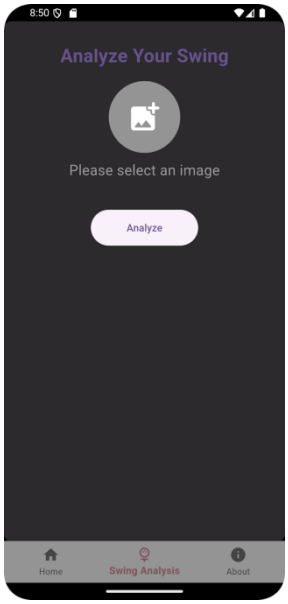


Figure 2. Screenshot of the analyze page

```
Future<void> _pickImage() async {
  final ImagePicker _picker = ImagePicker();
  final XFile? _pickedFile =
      await _picker.pickImage(source: ImageSource.gallery);

  if (_pickedFile != null) {
    setState(() {
      _image = File(_pickedFile.path);
    });
  }
}
```

```
Future<void> _uploadImage() async {
  print("uploading image");
  setState(() {...});

  Map<String, String> headers = {...};

  http.MultipartRequest request = http.MultipartRequest('POST',
      Uri.parse('$_url/golf-swing-assessment')); //post request to URL/analyze
  request.headers.addAll(headers);

  request.files.add(
    await http.MultipartFile.fromPath(
      'image_data',
      _image!.path,
      contentType: MediaType('application', 'jpeg'),
    ),
  );
```

Figure 3. Screenshot of code 1

Initially, we identified that with Future <void> this function can run without processing an output. We start off by calling the ImagePicker() constructor, then the user can pick the image from their photostream. Once we check to see if an image is selected, we will then update the _image variable with the file path of the picture.

Again, we identify that this function will run without returning outputs. We tell the user that the image is uploading and update the interface. We then create a Map to represent the header requests as a key-value pair. We then prepare to send a post-request to the server at $_url/golf-swing-assesment. Finally, we access the uploaded image path and attach the image to our request to send to the server.

API Handler

See the Pythonanywhere server, the flask_app.py file.

It “redirects” the user’s request and has our server to perform different tasks depending on what the user requests.

It also sets up any temporary files, path, and variables for our next component, Pose estimate.

```
@app.route('/test/<message>')
def test_method(message):
  print("the user entered:", message)
  return jsonify(message)


@app.route('/upload', methods=['POST'])
def upload_image():
  if "image" not in request.files:
    return jsonify({"error": "no file has been uploaded"}), 400

  file = request.files["image"]

  if file.filename == "":
    # generally happens when the user failed to upload/server failed to
    # receive the file due to unstable internet connection
    return jsonify({"error": "no selected file"}), 400
imgPath = os.path.join("golf_image_data/temp", (str)(file.filename))
file.save(imgPath)

try:
  predictionResult = predictImage(imgPath)
  # optional, we can either keep or delete the user uploaded image
  os.remove(imgPath)
  return jsonify({
      "prediction": list(predictionResult["prediction"]),
      # "avg_good_swing_data": predictionResult["avg_good_swing_data"],
      "curr_swing_data" : list(predictionResult["curr_swing_data"])
  })

except Exception as e:
  return jsonify({
      "Error": "Server failed to predict the uploaded image",
      "error_statement": str(e)
  }), 500
```

Figure 4. Screenshot of code 2

The first app route notifies us that if a user requests to do something, then we will use whatever the user sent as their URL request and print it on the server. The user will see the same message in JSON format [5].

The second app route notifies us if a user seconds in a POST request, which in our case means uploading an image. We return an error if we don’t find a file in the request. If the user uploads a file, our server will receive the file. We then check if the file is empty, if so we return another error telling the user the request failed.

We save the uploaded file into the golf_image_data/temp directory, then call the predictImage() function to analyze the image using its image path. We return predictImage’s result in JSON format, we will then delete the temporarily saved image to save space. We will also let the user know that an internal error occurs if the image can not be processed.

Pose Estimate

Pythonanywhere pose_estimate.py

Open the pre-trained model, identify pose landmarks (elbow angle, shoulder tilt, etc), pass these data into our model, make a prediction, and return the result.

We use Mediapipe to identify pose landmark data from a picture and feed it into our pre-trained model which would then predict the quality of the swing based on the data relative to a good swing [8]. Then we will return whether the swing is a good swing or a bad swing.

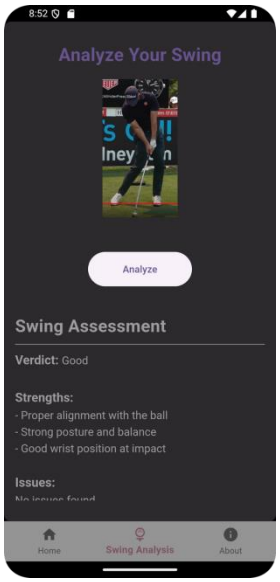


Figure 5. Screenshot of assessment page

```
def identify_pose_landmarks(folder_name: str, img_filenames: list[str],
                            print_output: bool):
    # Utility functions for drawing landmarks and connections.
    # mp_drawing = mp.solutions.mediapipe.python.solutions.drawing_utils
    mp_drawing = mp.solutions.mediapipe.solutions.drawing_utils
    # Drawing styles for landmarks.
    mp_drawing_styles = mp.solutions.mediapipe.solutions.drawing_styles
    # MediaPipe's pose estimation solution.
    mp_pose = mp.solutions.mediapipe.solutions.pose
def predictImage(imgPath):
    with open('pose-estimate-model.pkl', 'rb') as f:
        model = pickle.load(f)

    # open the image & obtain the skeleton coordinates
    temp_filenames = os.listdir(path + "temp/")
    temp_pose_landmarks = identify_pose_landmarks("/temp/", temp_filenames,
                                                  False)

    # figure out what is the shoulder tilt, elbow angle, wrist distance, etc of
    # this particular image
    temp_processed_landmarks = create_training_data(temp_pose_landmarks)

    # run our model with the processed landmarks & make prediction
    scaler = StandardScaler()
    temp_processed_landmarks_scaled = scaler.fit_transform(
        temp_processed_landmarks)
    predictions = model.predict(temp_processed_landmarks_scaled)
```

Figure 6. Screenshot of code 3

In the function identify_landmark_pose(), we take in three parameters; what the image folder is called, the files you want to analyze, and whether to display results. Then we use the built-in

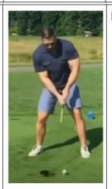
commands from the Mediapipe library to draw and style points and lines of the skeleton, then we detect the poses [7].

In the function predictImage, we load a pre-trained machine-learning model from a pickle file and find images to analyze in the temp folder [6]. It then calls the identify_landmark_pose() function to map out the key points in the picture. We then access the function create_training_data() to extract key data from these plotted points to standardize them using StandardScaler. Finally, we analyze the swing to determine whether it is good or bad using the pre-trained model.

4. EXPERIMENT

We tested the model's accuracy using 20 different image samples with a variety of different swing types. It is essential to test out the accuracy as it will highlight the overall capabilities and potential improvements for the model in the future.
To evaluate our model's accuracy, we compiled a dataset of golf swing images sourced from online videos and images. Each image was manually classified as either a good/bad swing, then processed using the PyTorch machine learning library where we applied the train_test_split() function to divide the dataset into training and testing subsets.

The training dataset was used to develop and train the model, enabling it to learn the distinguishing features of an effective golf swing. The testing dataset, being kept separated from the training set, was used to evaluate the model's accuracy by comparing its predictions against the expected classifications. By ensuring the same images don't appear in both sets and using varied inputs across different runs, we are able to test the model's robustness across different swing types and body postures while maintaining the integrity of the evaluation.

Image Number	Input	Expected Output	Actual Output
1		Good	Good
2		Good	Good
3		Bad	Bad
4		Good	Good
5		Good	Good

6		Bad	Bad
7		Bad	Bad
8		Good	Good
9		Good	Good
10		Good	Good
11		Good	Goodd
12		Good	Good
13		Good	Good
14		Good	Good

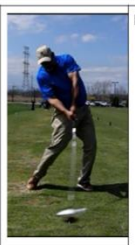
15		Good	Good
16		Good	Good
17		Bad	Good
18		Bad	Bad
19		Good	Good
20		Good	Good

Figure 7. Figure of experiment

5. RELATED WORK

The research paper "Golf Swing Biomechanics: A Systematic Review and Methodological Recommendations for Kinematics" by Bourgain, et al. systematically reviews golf swing biomechanics with an emphasis on movement kinematics such as joint angular kinematics, crunch factor, kinematic sequence, and more that influence golf swing efficiency [11].

Our system prioritizes elbow stability and shoulder posture as primary indicators of a good or bad golf swing, whereas many studies in the review discuss wrist flexion/extension, and pelvis-torso rotation (X-factor). The reviewed studies also mostly consist of retrospective data analysis conducted in controlled environments with precise motion capture equipment. This differs from our system that aims to provide real-time analysis and allow users to receive instant feedback on their swing mechanics. The low-cost phone camera also makes it more practical and scalable for general users.

We reviewed three papers comparing MediaPipe and YOLO for pose estimation [12]. The first study (Zhang, 2024) used both YOLO and MediaPipe for motion tracking, second study (Do, 2024) compared YOLO (v5-v8) and MediaPipe for hand gesture recognition, and third benchmarked YOLOv7 and Mediapipe with focus on accuracy and speed [14].

The first paper utilized YOLOv5 for a wider-range object detection and MediaPipe for more fine-grained keypoint tracking. This approach lines up with the second study showing YOLO having higher accuracy with more computing requirements, while MediaPipe runs efficiently on a CPU with tradeoff for slightly lower accuracy.

Instead of combining YOLO and MediaPipe, we found MediaPipe alone is sufficient for pose estimation without YOLO's multi-object detection overhead [13]. And unlike Study 2, which focuses on hand gesture recognition, we emphasize full-body pose tracking for real-time analysis. Based on these requirements, we chose MediaPipe over YOLO because it's faster, lightweight, with only a slight tradeoff in accuracy. It is also more practical for consumer devices that often do not come with powerful computing hardware.

The research paper "AI Golf: Golf Swing Analysis Tool for Self-Training" by Liao, et al. proposes a neural network based golf swing analysis system [15]. It focuses on self-training by comparing a user's swing to professional players' swings by using motion synchronization, discrepancy detection, and motion manipulation to help users gradually adjust their swings.

The system consists of three core modules:

Motion Synchronizer: aligns two motion sequences with different speed and timings

Motion Discrepancy Detection: Identify the differences in motion between player and professional

Motion Manipulator: Generates intermediate swing motions, allowing users to transition gradually to an ideal swing form.

One key difference between our approach is the Rule-based vs Neural Net-based golf swing prediction. Our system performs statistical analysis of the player's vs professional's swings, whereas this paper relies on unsupervised neural network models, requiring large training datasets and high computing requirements.

This also leads to our system to be more accessible to the general public, and faster to iterate and improve our rule-based model as it could be deployed instantly without prior training.

6. CONCLUSIONS

Despite the progress made in developing this project, there still remain several limitations. One significant constraint is the reliance on a static image captured at the moment of ball impact, restricting the depth of motion analysis. Video analysis could provide a more comprehensive evaluation to the entire swing motion sequence. Additionally, the current model does not account for the player height or club type, which can significantly influence the swing mechanics and potentially reduce the accuracy of the analysis across diverse user profiles.

With more time and resources in the future, improvements could include expanding the training dataset with a broader range of swing data, which helps increase the analysis accuracy. Another key improvement would be shifting from a rule-based system to an unsupervised learning neural

network. This would allow the model to adapt more flexibly to new data and new key points without manual optimizations. The same model could also be extended to other sports with sufficient training data such as basketball, baseball, tennis, and more. If we were to restart the project, integrating video analysis and prioritizing the unsupervised neural nets would likely lead to a more robust system, and better user experience.

REFERENCES

[1] Han, Jee-Hoon. "The Relationship between Trust, Satisfaction and Perceived Performance of Golf Device Data-Focused on the Golf Swing Analyzer." Journal of the Korean Applied Science and Technology 38.1 (2021): 196-207.

[2] Mufid, Mohammad Robihul, et al. "Design an mvc model using python for flask framework development." 2019 International Electronics Symposium (IES). IEEE, 2019.

[3] Debalaxmi, Debashree, Dinesh Kumar Vishwakarma, and Virender Ranga. "Analyzing yoga pose recognition: A comparison of MediaPipe and YOLO keypoint detection with ensemble techniques." 2024 3rd International Conference on Applied Artificial Intelligence and Computing (ICAAIC). IEEE, 2024.

[4] Liao, Chen-Chieh, et al. "Ai coach: A motor skill training system using motion discrepancy detection." Proceedings of the augmented humans international conference 2023. 2023.

[5] Lv, Teng, Ping Yan, and Weimin He. "Survey on JSON data modelling." Journal of Physics: Conference Series. Vol. 1069. No. 1. IOP Publishing, 2018.

[6] Vartak, Manasi, et al. "ModelDB: a system for machine learning model management." Proceedings of the Workshop on Human-In-the-Loop Data Analytics. 2016.

[7] Ilham, Amil Ahmad, and Ingrid Nurtanio. "Dynamic Sign Language Recognition Using Mediapipe Library and Modified LSTM Method." International Journal on Advanced Science, Engineering & Information Technology 13.6 (2023).

[8] Bookstein, Fred L. "Size and shape spaces for landmark data in two dimensions." Statistical science 1.2 (1986): 181-222.

[9] Glazier, P., and Keith Davids. "Is there such a thing as a 'perfect'golf swing." International Society of Biomechanics in Sports' Coaches Information Service (2005).

[10] Nesbit, Steven M., and Monika Serrano. "Work and power analysis of the golf swing." Journal of sports science & medicine 4.4 (2005): 520.

[11] Bourgain, Maxime, et al. "Golf swing biomechanics: A systematic review and methodological recommendations for kinematics." Sports 10.6 (2022): 91.

[12] Zhang, Weijia, et al. "Combined MediaPipe and YOLOv5 range of motion assessment system for spinal diseases and frozen shoulder." Scientific Reports 14.1 (2024): 15879.

[13] Do, Van-Dinh, et al. "TQU-HG dataset and comparative study for hand gesture recognition of RGB-based images using deep learning." Indonesian Journal of Electrical Engineering and Computer Science 34.3 (2024): 1603-1617.

[14] Wang, Chien-Yao, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors." Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2023.

[15] Liao, Chen-Chieh, Dong-Hyun Hwang, and Hideki Koike. "Ai golf: Golf swing analysis tool for self-training." IEEE Access 10 (2022): 106286-106295.