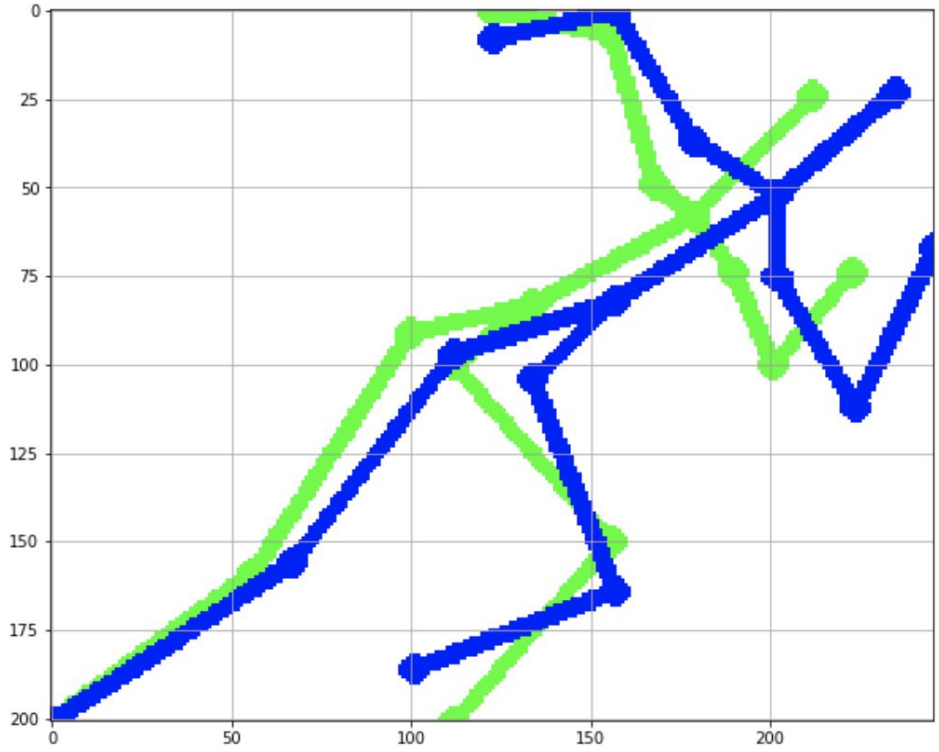


Pose Estimation and Matching

Matthew Avallone (mva271)
Siddharth Choudhary (sc7530)
Kshitija Patel (kap676)

Introduction

The goal of this project is to perform human pose estimation and matching. Given two images, a reference image and a trial image, human poses would be detected using keypoints and compared to determine the similarity between them. These key points can be plotted as a 2D stick-figure. The reference image would be overlayed on the trial image to find the similarity between the poses which are depicted in both of the images.

Motivation

Pose estimation and matching have a wide range of applications. Many individuals like dancers, athletes, yoga enthusiasts, etc. learn from following videos of experts in their field. This helps them learn various different poses, but it's difficult for them to assess the correctness of these poses. By correctness, this means how close the position of their body parts matches the true position.

Our application would be of utmost use to them to instantly assess the differences in their pose compared to that of the experts. Being able to see the overlay can help them spot differences instantaneously and learn how to correctly position themselves.

Data Used

For training the pose detection model, we have used MPII's (Max Planck Institute Informatik) Human Pose dataset.

The dataset includes around 25K images containing over 40K people with annotated body joints. The images were systematically collected using an established taxonomy of everyday human activities. Overall the dataset covers 410 human activities and each image is provided with an activity label. Each image was extracted from a YouTube video and provided with preceding and following un-annotated frames. Thus making the dataset robust, and perfect for our application. Sample images are shown in Figure 1.

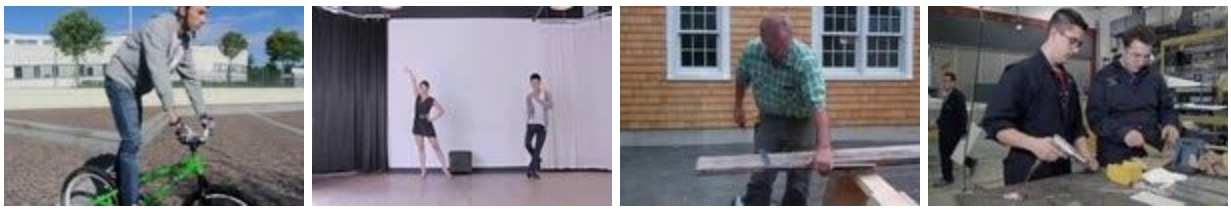


Figure 1: Sample Training Images from the MPII Dataset

For the purpose of testing our system, random images with similar looking poses were collected from the internet. We chose images of human poses that aligned with the types of activities described in our motivation. The test images only contain one person present in them. This simplifies the task of comparing poses, but also makes sense for our application. Sample test images are shown in Figure 2.

Figure 2: Sample Pair of Test Images

Approach

This project is broken down into two phases: pose estimation and pose matching. For pose estimation, a pre-trained neural network was used to predict sixteen key points in a given image, along with confidence scores associated with the coordinate locations. Fifteen key points represent locations on the human body, and one key point represents the background. The joints include head, neck, shoulders, elbows, wrists, hips, knees, ankles, and chest. A threshold of 10% was used to determine whether to include a key point in the pose or not. We chose a low threshold value in order to prevent excluding missing joints.

For pose matching, two distance metrics were used to measure the similarity between the pose vectors predicted by the model. There are many ways of finding the distance between two vectors, but for our case, it was imperative that the score be independent of vector magnitudes since these could vary depending on factors such as the person's age, height, and distance from the camera. The objective is to minimize the distance between vectors when the poses are very similar.

Before computing the distance, the pose vectors are normalized between (0,1). Two normalization techniques were tested. The first technique just divides all coordinates by

the maximum for each axis. The second technique is L2 normalization, which divides each pair of coordinates by their magnitude.

The cosine distance and a weighted distance were tested as possible distance metric candidates. The cosine distance computes the angle between vectors and subtracts one from it. The equation for cosine distance is shown in Equation 1.

$$Cosine\ Distance(pose1, pose2) = 1 - \frac{pose1 \cdot pose2}{||pose1||\ ||pose2||} \quad (1)$$

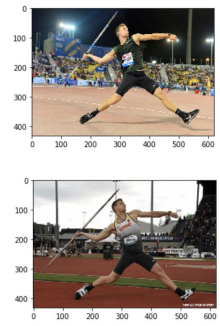
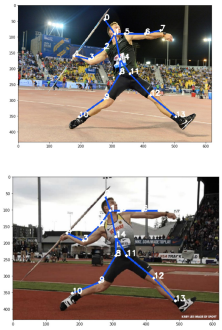
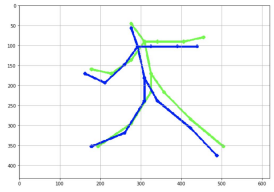
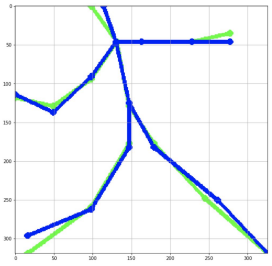
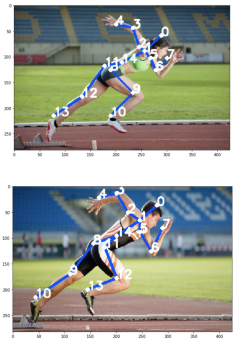
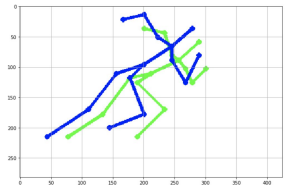
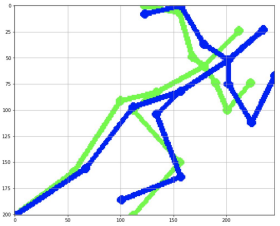
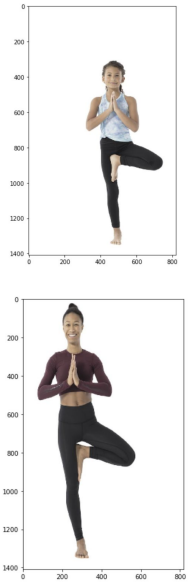
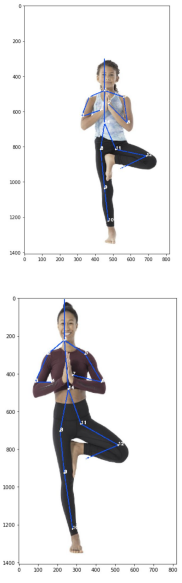
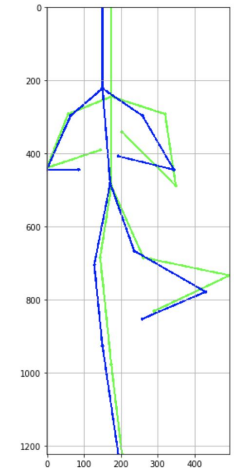
The weighted distance metric we implemented was developed by Google researchers in the paper "PersonLab". It improves upon the cosine distance by also incorporating the confidence scores outputted by the model. The confidence score is described as the probability of a joint location at position (x,y). We were to be able to weight the pose data so that low confidence joints have less effect on the distance metric than high confidence joints. The equation for the weighted distance metric is shown in Equation 2.

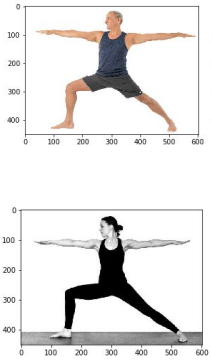
$$Weighted\ Distance(pose1, pose2) = \frac{1}{\sum_{i=1}^{n} P(pose1(x_i, y_i))} \cdot \sum_{i=1}^{n} P(pose1(x_i, y_i)) \cdot ||\ pose1(x_i, y_i) - pose2(x_i, y_i)|| \quad (2)$$

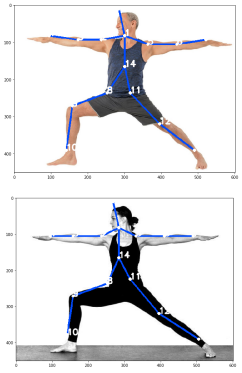
Multiple techniques were tested to improve the accuracy of the pose matching. The first approach uses the affine transformation matrix to match point 1 (neck) and point 14 (the abdomen). The idea behind this was to align the centers of the bodies in order to reduce the effect of translation on the similarity scores. The affine transformation is computed via least squares.

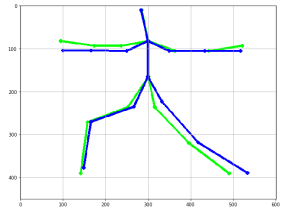
The second approach involves cropping and scaling the pose vectors to align them on top of each other. Overlaying the poses on top of each other visually makes it easier to judge their similarity. The minimum values for x and y coordinates were found and subtracted off of all coordinates for each pose. Afterwards, each axis of the poses was scaled by the maximum value of the test pose divided by the maximum value of the source pose.

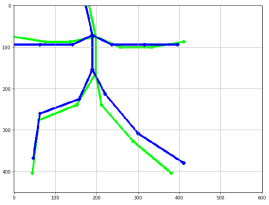
Results

Input Images	Pose Estimation of Input Images	Poses Overlaid	Poses Overlaid after Resizing, and Similarity Score
			 Cosine Distance: 0.043522486675600325 Weighted Distance: 0.11009960679613295
			 Cosine Distance: 0.05666593282493859 Weighted Distance: 0.26906564819132067
			 Cosine Distance: 0.06117845805971822 Weighted Distance: 0.17730516149360648

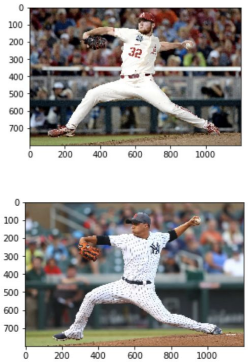


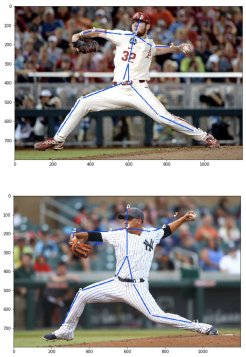


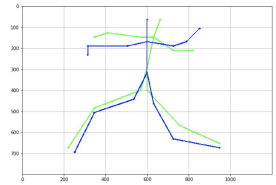


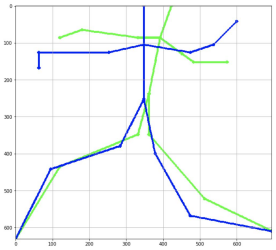


Cosine Distance: 1.0
Weighted Distance: 0.31266940250355413

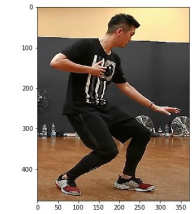


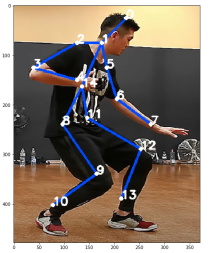


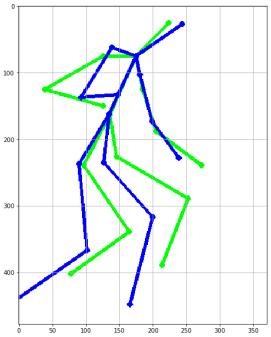


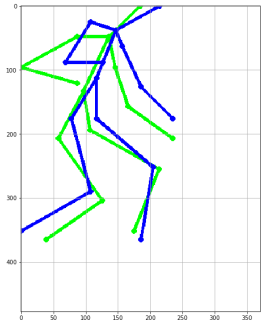


Cosine Distance: 0.017753955840534297
Weighted Distance: 0.14740922877551646

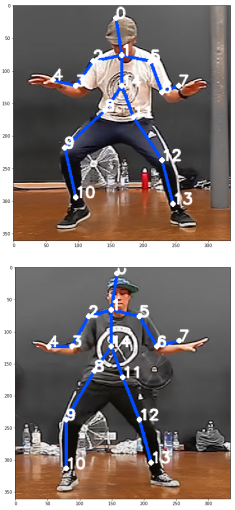


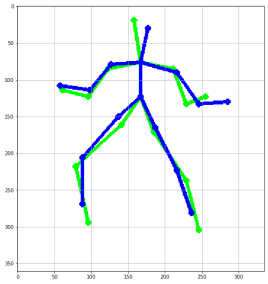


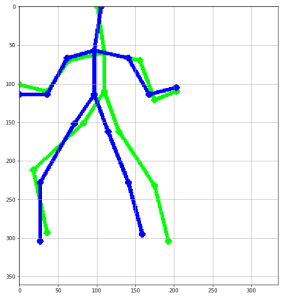




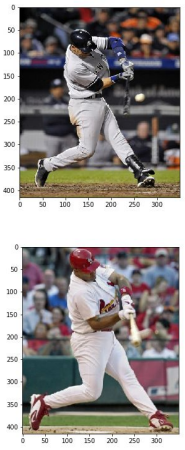
Cosine Distance: 0.5000000000000001
Weighted Distance: 0.10783574155084759

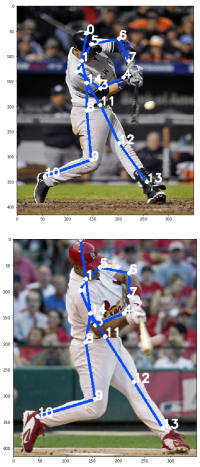




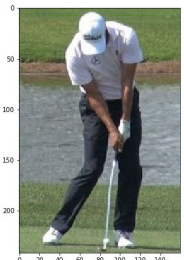


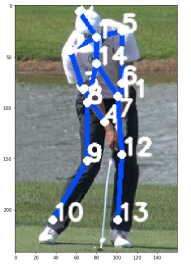
Cosine Distance: 0.5000000000000001
Weighted Distance: 0.12435626214251407

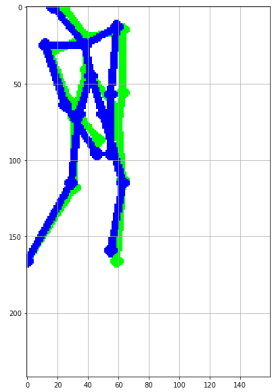




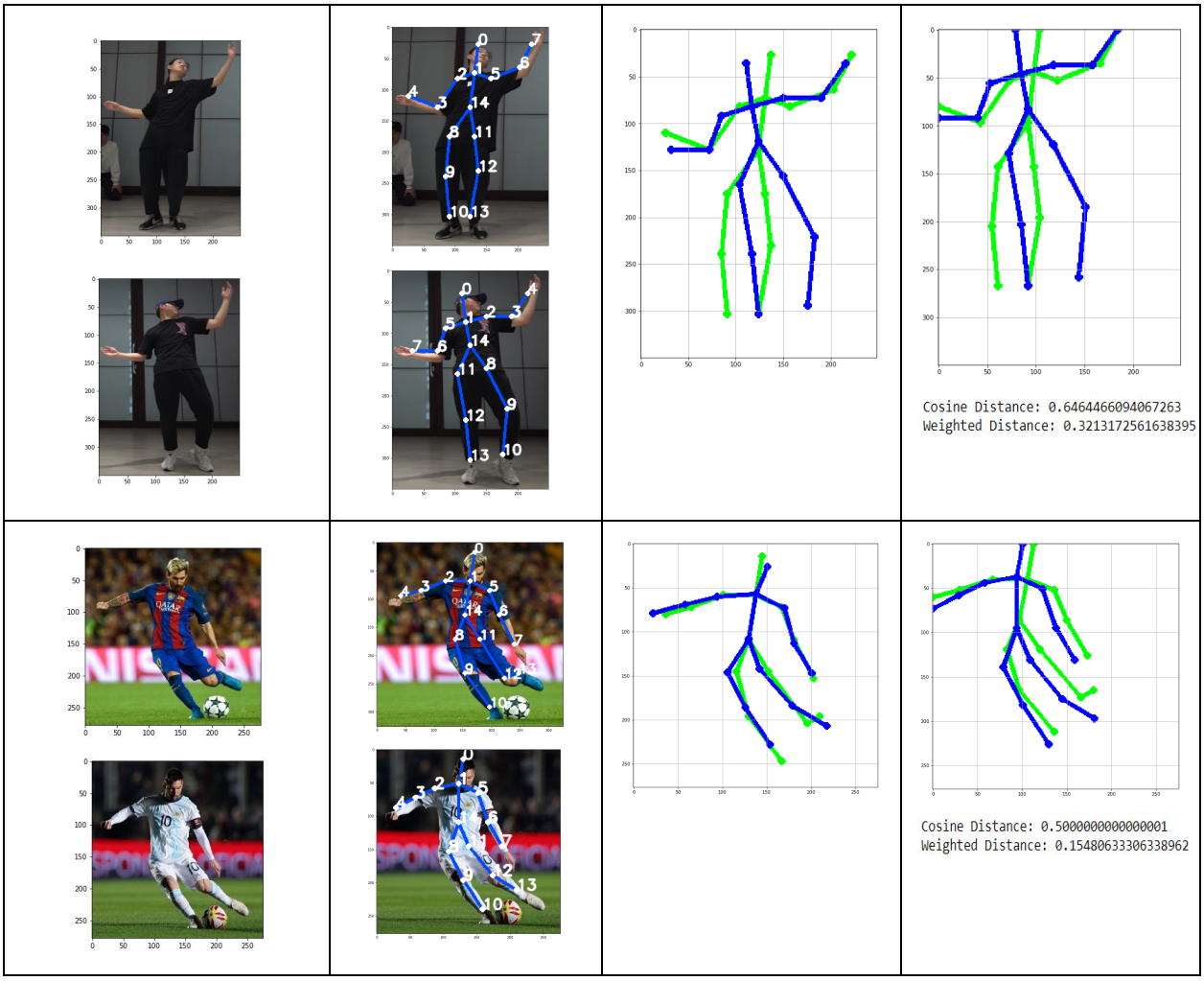
Cosine Distance: 0.5000000000000001
Weighted Distance: 0.16794195866508724







Cosine Distance: 0.3675444679663241
Weighted Distance: 0.21227507411793375



Distance Metrics

The weighted distance metric outperformed the cosine distance in accurately detecting a pose match vs. non-match. This makes sense since the weighted distance takes into account the confidence scores associated with each key point location.

Normalization

We found that simply dividing all of the coordinates in each axis by their respective maximum produced better results than using L2 normalization. With L2 normalization, the distance scores tended to be lower using both metrics, even on non-matching test poses.

Affine Transformation

The affine transformation to map the neck to the abdomen between the two poses did not improve the results for pose matching. This transformation distorted the pose, such that it no longer resembled the original estimation.

Cropping and Resizing

Cropping and resizing the pose vectors before computing the distance metric tended to produce better results. The pose vectors became aligned on top of each other, which made it easier to see their similarity

Difficulties and Challenges

Efficient segmentation - found MPII dataset which does not need segmentation

The initial idea was to use segmentation techniques like semantic segmentation to efficiently edit out a pose from the image with the help of background subtraction. The MPII dataset essentially does the same work and estimates the pose of the human body present in the image, but uses keypoint detection instead. Through experiments, testing and further research, it was concluded that keypoint detection for pose matching, although gives good results, often produces incorrect outputs. Semantic Segmentation technique for pose detection and matching may be a better method but is beyond the scope of this project.

Differences in the images - size of person, translation, scaling

Handling of invariance in the tested images proved to be more challenging than the pose estimation itself. While in some cases the model was successfully able to determine the pose presented in the image even when certain parts of the body were hidden or not clearly visible, in other cases the model failed to detect the presented pose. It was also observed that if two poses that were similar or identical were different in terms of the size of the person, the matching of the two shapes would be incorrect which was not the expected output. In order to deal with the invariance in terms of size and angle of the poses, affine transformations seemed to be the best option in order to match the poses as closely as possible.

Affine transformation gave incorrect results

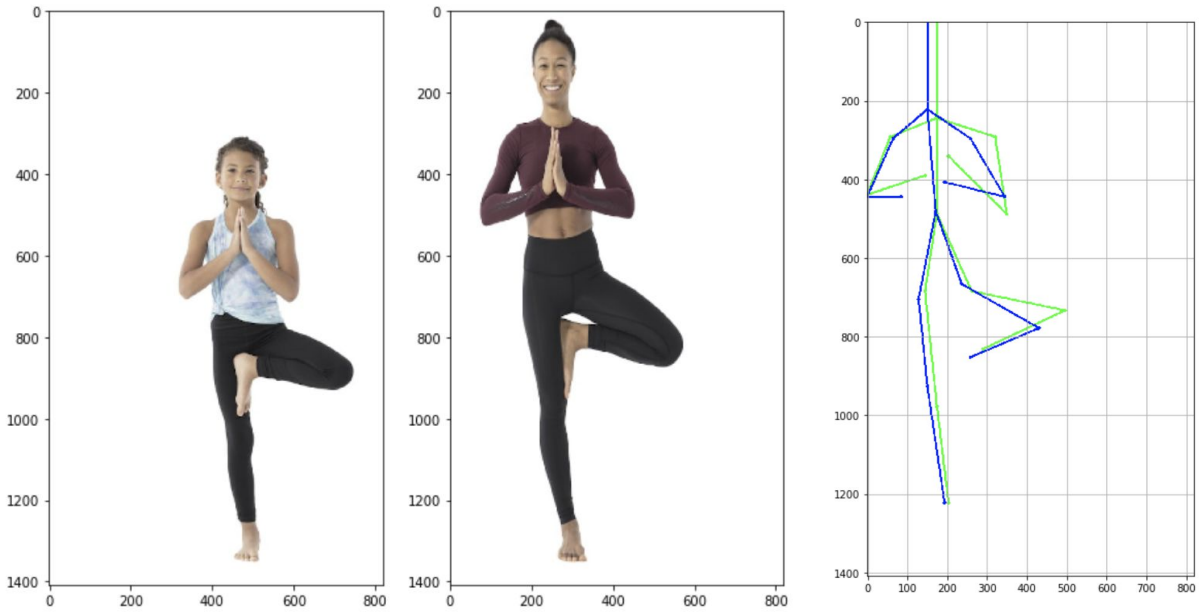
When the project encountered invariance due to size of a person, the angle at which the person stood and the angle of image capture, affine transformations were introduced into the project in order to scale, rotate and translate the poses to precisely match two similar poses. In more than half of the images that were tested, affine transformation worked well when it came to centering the two poses, but there were some cases where the affine transformation gave incorrect and even worse outputs than the expected or original output. Thus the project is limited to the images in which the parts of the human body are clearly visible and not differ by a lot in terms of the affine system.

Strengths and Weaknesses

In terms of software engineering and performance, the **application is fast** and it only needs to pretrain the model once on the MPII dataset.

The system is **able to handle translation**, and even if the two people are not centered, or are standing at different distances from the camera, the pose matching and overlaying takes place correctly.

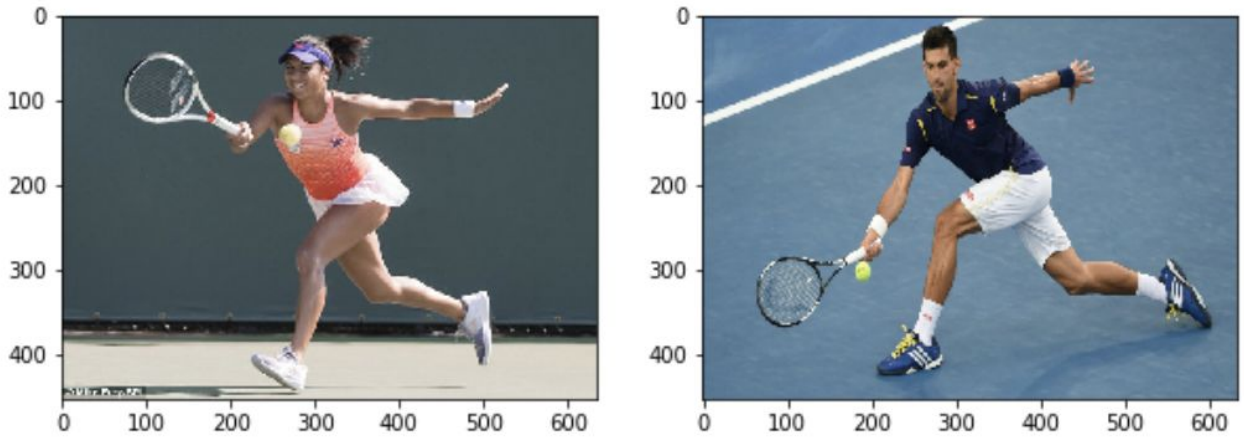
The system is also **able to handle invariance in the two body types**. If the height or overall dimensions of the two humans in the two images vary considerably (i.e, a 10 year old tries to learn a pose from an image of a 25 year old), the application has the ability to scale the images to match each other and is able to overlay them correctly. This can be seen in the following example -

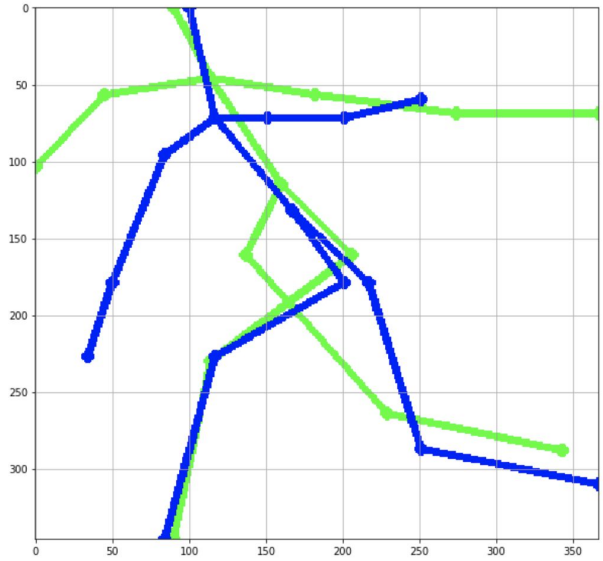


The main weakness is that the application is **unable to handle invariance in the two poses**. An identical pose which is either the mirror image of the reference image or is rotated by a certain angle are not overlaid correctly. Thus a ballerina doing the correct pose but facing the opposite direction as the reference will not be able to assess the correctness of her pose using this application. Hence the user will have to perform preprocessing on the input images and ensure the required rotation, flipping, etc are performed before the images are provided as input.

The **angle at which the images of the poses are captured also matters** a lot, and the application cannot deal with invariance in that. This can be seen in the following example-

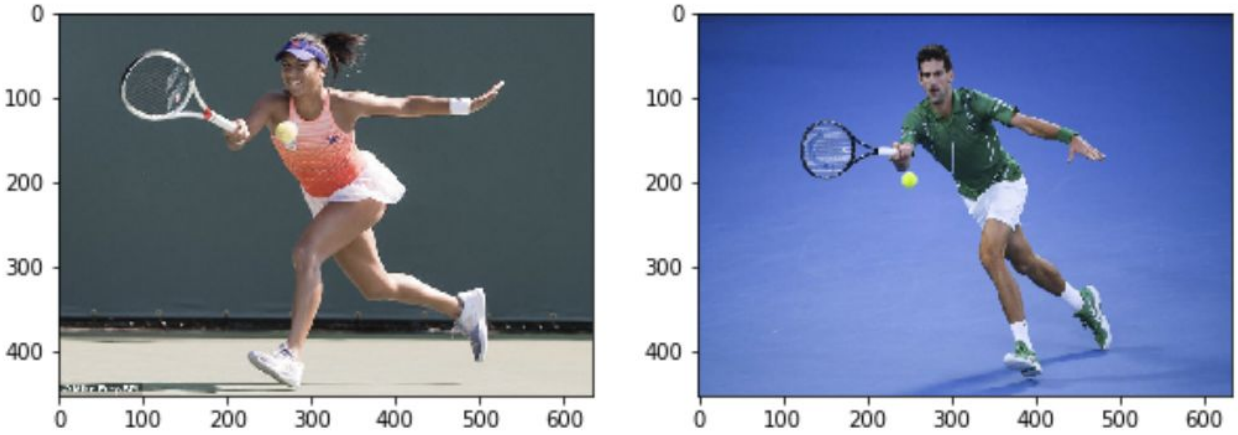
Same pose captured at different angles

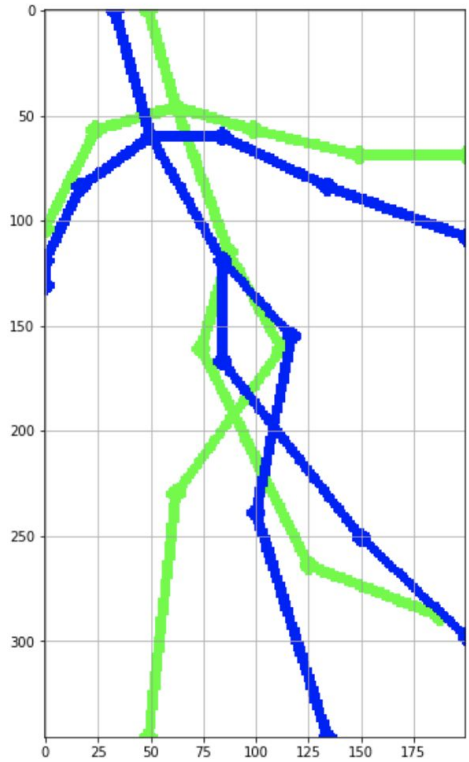




Here the hip position is estimated differently, as the angle at which we see the pose changes. Hence, the two images - test and reference - need to have similar angles.

Djokovic's image captured from a different angle





The application is **restricted to one person being captured in every frame**. The MPII data set is a multi-person dataset, and thus poses of multiple people in one frame can be correctly estimated. But the pose matching algorithm implemented after that is restricted to being able to overlay the pose of one person correctly. Hence this can also be considered as a shortcoming of the system.

Using a model trained on the MPII dataset is **unable to estimate complicated poses**, as such poses are not included in the dataset. This can be seen in the example below -

As we can see, the model is unable to predict the posture.

Future Scope

Training a better model

For this project, the MPII dataset was used which presents several thousand poses that were captured to create the data model. Since any model that is trained is limited to the dataset that is used, carefully expanding the dataset with a variety and abundance of poses will eventually lead to better pose estimation by the said model. If the dataset is collected methodically, it may even get rid of the invariance that was encountered due to complicated poses or hidden parts of the body.

Extend for pose matching of multiple people

The model developed in the project is limited to the pose of a single person per image. This can be further extended to multiple people present in the same image. Instead of using two different images to match the pose present, a single image with multiple people can also be used to match the poses of the people present in the image with each other. The application here can be for helping to choreograph a routine by checking for differences in the pose of each person.

Resolve weaknesses, create a user base with proper front end, and extend pose matching to video

In order to further develop the project, it is essential to first resolve as many weaknesses as we can. There were several difficulties and challenges that were encountered during the development and testing phases of the project as mentioned, that must be reduced as much as possible in order for it to be suitable for users. Putting ourselves in the minds of the users, it is a very tedious job to capture an image and then input it into the model for pose estimation is not a product that someone would use. Thus, extending the scope to video, preferably as a real time application, in order to imitate someone in the process of learning something seems to be the best choice.