A POSE-BASED IMAGE SEARCHING USING COMPUTER VISION AND POST-ESTIMATE

Hang Wang[1] and Yu Sun[2]

[1]University of California—Berkeley, Berkeley, USA
[2]Cal Poly Pomona, USA

ABSTRACT

As the cost of human forces increases, people in some careers, like the artists, may find some difficulties when models are needed in the processes of making art. Obviously, one alternative solution is to find pictures online, however, when some specific poses are needed, they may also find some difficulties to describe them. This paper develops an application to search for pictures with the target pose described by the user's graphical input. In this project, one hundred images describing distinct actions and activities with various poses and view-angles are collected. Mediapipe is then used to analyze those images in a quantitative way. We also embedded a User Interface that allows the user to imitate the intended pose as well as the viewing angles by simply dragging around body joints of the figure. Different sets of feature points and matching algorithms are also tested to find out the best solution.

KEYWORDS

Computer vision, Pose detection, Image searching.

1. INTRODUCTION

The internet nowadays is ever powerful to connect everyone to any open resources that one may desire. Searching engines, for example, give their clients a way to access those resources by string matching. When we trying to search for an image, those images are categorized in a way so that we can easily find one, for example, by simply typing “a running man” and hit the search button. However, when someone is trying to search for a very detailed pose, the user must know how to describe that pose in text, such as “jumping with the left arm up and right arm down” or “sitting with both hands crossed on the leg”, which is annoying and not necessarily returning a promising result. The users oftentimes need to rephrase their words couple times before they give up or a satisfying result returns.

In this project, we define an innovative way of image-searching by the user's own graphical input. The clients of our application can expect a promising searching experience by simply modifying the provided default body-joint figure to imitate the pose in their mind. Such application is even more powerful when high-degree of accuracy is demanded such as “left arm curved 30 degrees”.

Our image-search engine is accomplished by linking the user input with preprocessed dataset using the matching algorithm.

The rest of this paper is structured as follows: Section 2 will detail the multiple challenges faced in this study and how they were overcome; Section 3 will describe the methodology and solution in greater detail; Section 4 details the experiments that were performed in this study, as well as a

David C. Wyld et al. (Eds): DMML, SEAS, ADCO, NLPI, SP, BDBS, CMCA, CSITEC - 2022
pp. 187-194, 2022. CS & IT - CSCP 2022 DOI: 10.5121/csit.2022.120716

thorough analysis of the results; Section 5 will list any related works that have also been done regarding volunteering; and lastly, Section 6 will conclude the study and state any future work that may be done.

2. CHALLENGES

2.1. Challenge 1: Aligning User Input for prediction in machine learning does not always translate perfectly. (2D input for user prediction does not translate perfectly to a 3D captured image turned 2D (aka a photo))

A three-dimensional space has much more information than a two-dimensional space. In this project, we are focusing on photographs, which are a 3D image flattened to 2D. The user input is therefore also treated as a 2D object, as controlling a 3D model is not only more complex but prone to error. When matching the user input to a predicted image, the process of estimating for a higher dimension has always been a challenge. Due to the complex nature of imaging, oftentimes user input does not perfectly correlate with an image as we are cutting one dimension down. Photographs, while 2D, are reflections of 3D space. We have to do this very precisely (understanding how a 3D object in different angles is displayed in a 2D image) to have a good matching result. To better match the user's input to the image there are a few options. We can either increase how much the user is going to actually put in, give them the ability to add more detail to their data, or limit their options, handling post-processing for them automatically. In this project, we give the users the ability to adjust the length of selected components to more flexibly demonstrate the depth of a three-dimensional image.

2.2. Challenge 2: Selecting a pattern matching model is a difficult process. (testing the models, comparing accuracy, comparing flexibility, etc.)

Selecting a model is essential for every machine learning project, as selecting a model has a huge impact on the accuracy of the result. Each model has its own strengths and weaknesses. Depending on the problem, a model's accuracy and impact can vary wildly. The process of selecting a suitable model for our computer vision project is specifically tough as there are numerous features that we can collect from an image. Because the data and process of image matching is flexible in implementation, we can end up with a lot of different models that work depending on how things are done. Each choice we make can either be an important feature or just some noise depending on the project itself. The standard way of selecting a suitable model for training is to do comparisons among all selected models. Whichever returns the highest accuracy is the one chosen. Often when a project is much simpler, just picking the most commonly accurate models will suffice as well. In this project, to select a good model to match user input to the actual image in the database, we do the following: for every model we are testing them on a wide array of training data and then splitting our tests to have them undergo cross validation. This will give us the average accuracy and success for each model.

2.3. Challenge 3: Finding out which model parameters are important is a tiresome process

There are numerous models in Machine Learning that are developed well. Among those, most models are flexible in terms of the parameters they are taking in, allowing the developers to tune the models to fit their specific needs. A model's parameters often have a direct correlation to its success rate. Selecting the perfect maximum depth for Random Forest or the optimal kernel size for a SVC model is a very time consuming process. The number of features, defining what a pose is for our project, also needs to be considered as the format of our training data also impacts what

model parameters are important. The process of hyper-parameter tuning is an excellent solution to this problem as we test the models with a wide gamut of parameters so as to see what lends itself best to our problem. Some problem solutions, however, do not require modifying model parameters past the default options. In our project, we use hyper-parameter tuning on the model that initially returns the highest general accuracy without any modification.

3. METHODOLOGY/SOLUTION

3.1. Overview of the solution (whole system)

The two most important things are the preprocessed dataset and the matching algorithm. The preprocessed dataset contains images with different poses as well as skeleton drawings. The matching algorithm is a machine learning algorithm that matches the user's skeleton drawing with the skeleton drawings on the preprocessed images. Once we have these two components, the whole system is constructed in the following way: first, a website with a 2D geometric user interface that shapes like an individual person figure is provided for the user to imitate the pose by simply dragging things around. Once the users finish their drawings, they will click on the "Search" button so that a web API function is called to trigger a HTTP request that is sent to the web server where the pre-trained model is located. The web server then calls this model which is trained on the dataset with all preprocessed images. The model returns the most close image and returns to the web server. The web server then interprets the request and asks to query the database in our database server where we store and provide access to persistent data [our preprocessed images]. The database server receives the query and returns the image that is asked for to the web server. The web server then construct the response, send it back through HTTP response so that the client browser can render the page to show the related image.

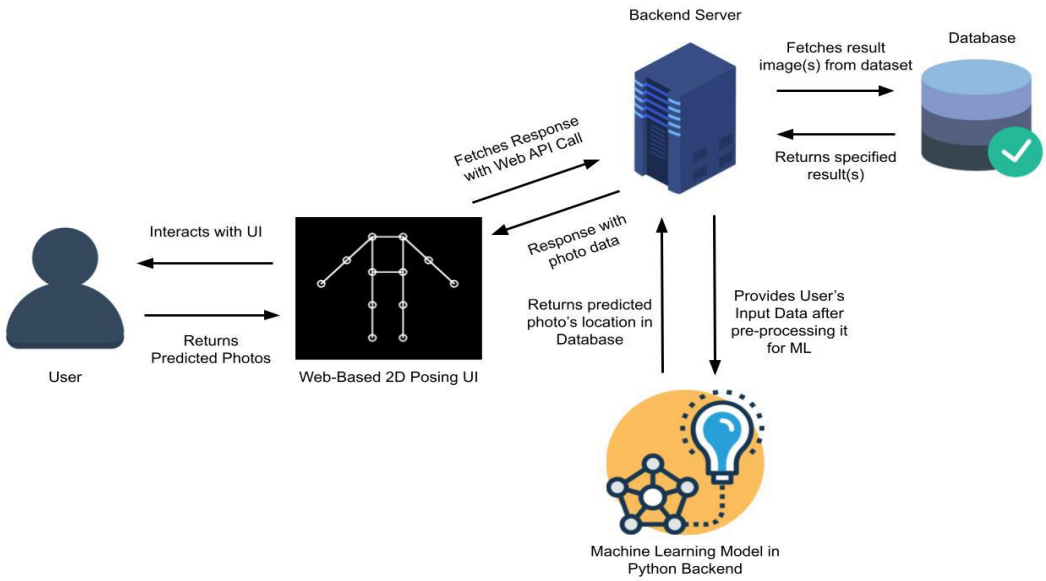

Figure 1. System work flow

3.2. The implementation

When designing our user interface, we created two object classes called Hinge and Joint, representing what they actually mean in the human body. With only one Hinge object originally assigned to the end position of the Joint object instead of both side, we are skipping the problematic case where we have to delete the duplicated Hinge objects at the exactly same location when connecting two Joints together. In our case, when trying to connect two body joints, the Hinge object actually handles the connection through the "add_connection" function. The way of how we achieving this makes much more sense and provides more modularity when we do the experiment. With these two object classes, the full body figure is then created where we assign the default positions of every parts by hard-code, which gives us the following UI:

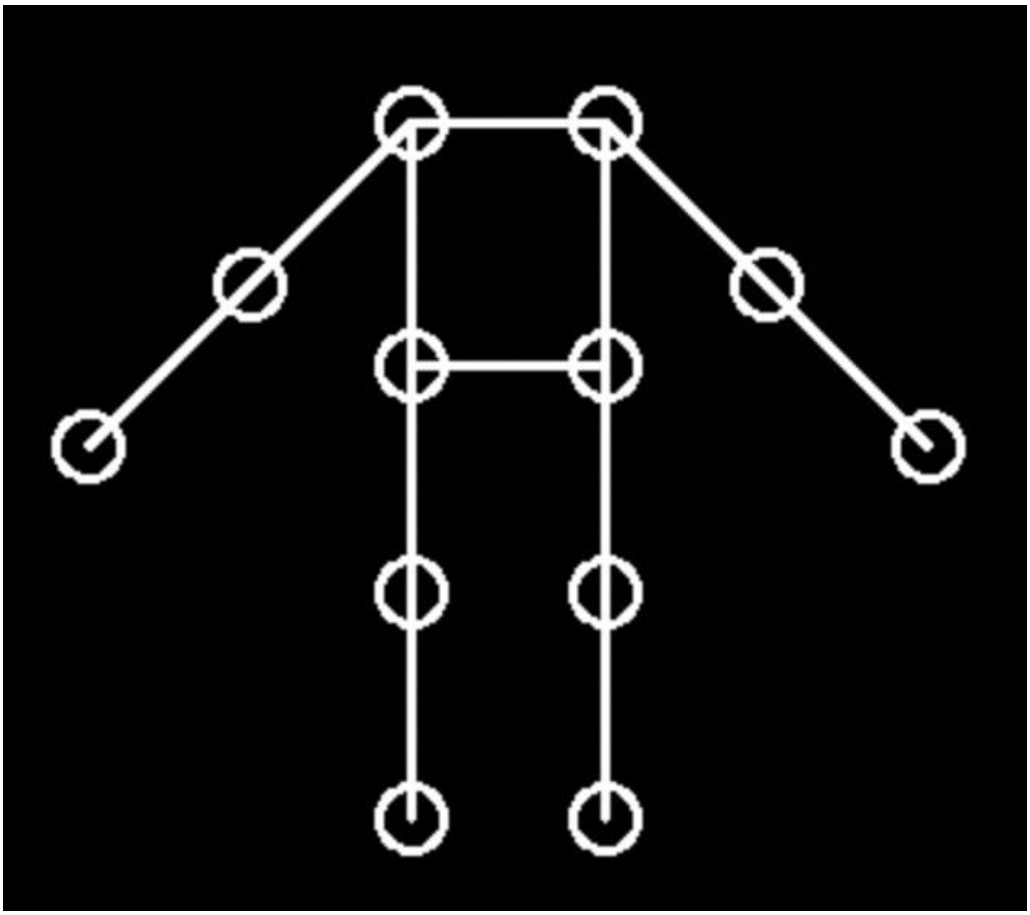

Figure 2. Example user interface

To make our model and dataset work on the website, we use the Python Flask server to organize everything. It takes in the HTTP requests with the drawings, cleans the data, passes it to machine learning, gets responses and then pulls out the correct image from the database.

For the matching algorithms that we are comparing, we used SVM, GaussianNB, Random Forest(maximum depth=2), Random Forest(no maximum depth), and CNNs.

4. EXPERIMENT

4.1. Experiment: Find the best matching algorithm

To find out which machine learning algorithms works the best as our matching algorithm, our group conducted couple experiments with each candidate models testing on different cross validations. The Mediapipe's pose landmark model converts images to skeleton drawings with 32 feature points as shown in the following figure. We would also like to see if our search engine is scalable, so we tested our models on different size of datasets.

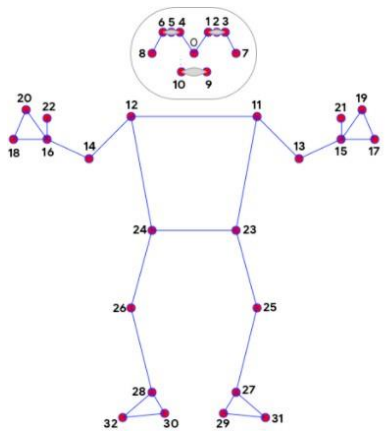

Figure 3. Default pose landmark model

First, we choose 10 images with 5 of them represent running and another 5 of them represent yoga then duplicated the dataset by 4 times in order to do the cross-validation. We used 3-fold cross-validation, SVM model gave the result of being 79.85% accurate, however, GaussianNB model, Random Forest model with maximum depth set to 2, and Random Forest model with unlimited maximum depth all gave 100% accuracy.

Next, we tested those models again on 50 images containing much more distinct activities and poses including sitting, jumping, yoga, running, etc. Again, for cross-validation, we choose to duplicate the dataset by 4 times. We used 3-fold cross-validation, SVM model gave the result of being 72.28% accurate, GaussianNB model and Random Forest model with unlimited maximum depth both gave 100% accuracy, and Random Forest model with maximum depth set to 2 gave us an accuracy of 67.86%.

Finally, with all 100 images, we duplicated the dataset by 8 times so that we can do a 3-fold cross-validation, a 5-fold cross-validation, and a 7-fold cross-validation. For 3-fold cross-validation, SVM model gave the result of being 93.97% accurate, GaussianNB model and Random Forest model with unlimited maximum depth both gave 100% accuracy, and Random Forest model with maximum depth set to 2 gave us an accuracy of 71.66%; For 5-fold cross-validation, SVM model gave the result of being 94.63% accurate, GaussianNB model and Random Forest model with unlimited maximum depth both gave 100% accuracy, and Random Forest model with maximum depth set to 2 gave us an accuracy of 73.45%; For 7-fold cross-validation, SVM model gave the result of being 97.32% accurate, GaussianNB model and Random Forest model with unlimited maximum depth both gave 100% accuracy, and Random Forest model with maximum depth set to 2 gave us an accuracy of 76.34%.

4.2. Analysis

As shown in the experiments, SVM model is the only which returns a good result while not being overfitted. However, we would ask our self a question, does overfitting really matters in our matching problem? The answer is no, since we only care about how accurately a matching model can be to link a user input to one of the images in our database. We finally choose the SVM model as our matching algorithm in our application.

5. RELATED WORK

In the paper "Image Matching Algorithm based on Feature-point and DAISY Descriptor", their team is focusing on an image matching algorithm where they combined SURF algorithm which is based on partial features and DAISY descriptor which is based on the principal direction. Their proposed algorithm, while almost maintaining the same computing speed, improves the SURF algorithm's ability on image rotation. In our research, while trying to increase the confidence level of matching the image where people are standing on a surface with slope or in different rotation to the user inputs, we can also use the idea of how to improve the accuracy of matching images in rotating condition. The main difference between our focus and their group's focus is that we are matching feature points to the images while they are trying to match images to images. One of the strengths of our research is our project's unique approach to matching, centered around matching to a particular image just given some raw data.

In the paper "Classification of yoga pose using machine learning techniques", their team is focusing on using pose detection techniques to identify the posture and thus the accuracy of yoga poses. They used four machine learning algorithms to classify the yoga asana for Sun salutations set of postures as well as a real-time skeleton drawing using pose estimate technique. In our project, instead of real-time skeleton pose drawing with video-capturing, we asked the user to move the body joints we provided as the user interface. Instead of some limited set of yoga poses, we allowed users to search for a large number of poses in their daily routine.

In the paper "A fingerprint recognition algorithm using phase-based image matching for low-quality fingerprints", their team is focusing on finding a fingerprint matching algorithm for low-quality fingerprints. The fact that the fingerprint condition, whether environmental or personal causes, can highly affect the recognition process is the most changeling problem. They suggested a phase-based image matching algorithm where they use the phase components in 2D discrete Fourier transforms of fingerprint images to try to achieve a better performance. In our project, while trying to match the user input to the image, we can also use the same idea to handle the case where some images didn't catch the whole body. The main difference between our focus and their group's focus is that we are matching posing features to the image while they are matching sub-optimal fingerprints to the complete fingerprints. One of the strengths of our research is that we are extracting and modifying the important feature points while matching to the dataset.

6. CONCLUSION AND FUTURE WORK

Our final application is able to return the target image on the right in the dataset given the graphical input on the right as shown in the following example.

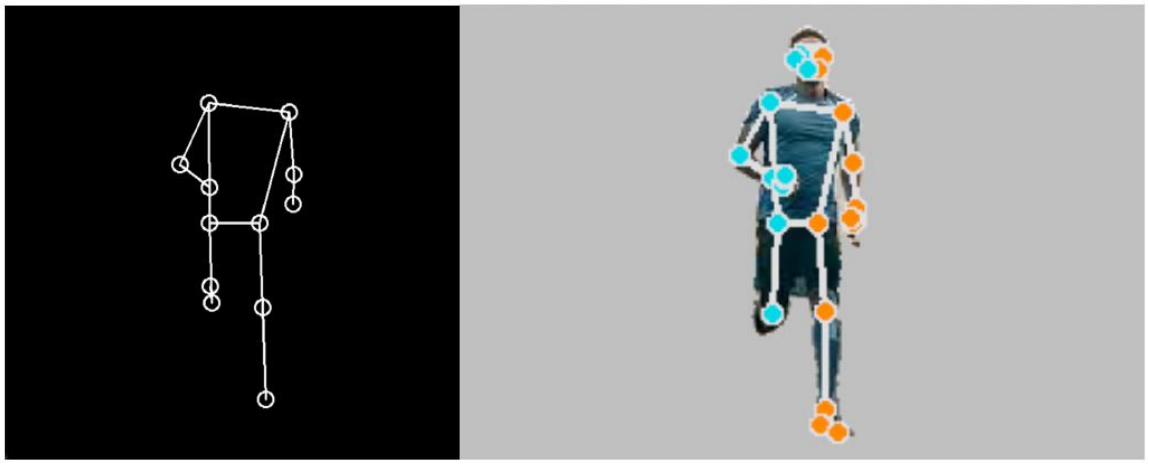

Figure 4. Example result

However, every project has limitations. In our project, some images that represent different actions/poses are probably matched to the same or very similar 2D body-joint data. When the user gives his body-joint inputs in our UI, he might, as a consequence, get many images that he doesn't expect. Another potential issue is that a lot of the practicability depends on a huge variety of images in the dataset which we may lack. A dataset that contains many different poses and actions is highly preferred while also being highly time-consuming. As a small research team, we may not have the ability to collect such a huge valid dataset.

In terms of how the way of our data-processing defines, we may expect that when turning a 3D image to a 2D body-joint data, different images may share the same or very similar data while the images themselves may vary widely. To address this issue in the future, a 3D body-joint data processing is preferred. However, as the accuracy of a 3D body-joint data processing highly depends on the pose-detection model we have, a substitute solution may be to add an extra field in our input data that will let the user select the body's facing position. In terms of how we can approach generating a larger dataset while maintaining its accuracy, having more people to work on it is obviously preferred. As the current size we have, one way of doing this is to set a separate program to automatically collect images from the web.

REFERENCES

[1] Li Li, (2014) Image Matching Algorithm based on Feature-point and DAISY Descriptor, Journal of Multimedia, Volume 9, NO. 6

[2] J. Palanimeera & K. Ponmozhi, (2021) Classification of yoga pose using machine learning techniques, Materials Today: Proceedings, Volume 37, Part 2, Pages 2930-2933

[3] K. Ito, A. Morita, T. Aoki, T. Higuchi, H. Nakajima & K. Kobayashi, (2005) A fingerprint recognition algorithm using phase-based image matching for low-quality fingerprints, IEEE International Conference on Image Processing 2005, pp. II-33

[4] F. Lv and R. Nevatia, (2007) Single View Human Action Recognition using Key Pose Matching and Viterbi Path Searching, 2007 IEEE Conference on Computer Vision and Pattern Recognition, pp. 1-8.

[5] J. Kilner, J. Guillemaut and A. Hilton, (2009) 3D action matching with key-pose detection, 2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops, pp. 1-8.

[6] R. M. Haralick, H. Joo, C. Lee, X. Zhuang, V. G. Vaidya and M. B. Kim, (1989) Pose estimation from corresponding point data, IEEE Transactions on Systems, Man, and Cybernetics, vol. 19, no. 6, pp. 1426-1446.

[7] Y. Yang and D. Ramanan, (2011) Articulated pose estimation with flexible mixtures-of-parts, CVPR 2011, pp. 1385-1392.

[8] V. Belagiannis and A. Zisserman, (2017) Recurrent Human Pose Estimation, 2017 12th IEEE International Conference on Automatic Face & Gesture Recognition, pp. 468-475.
[9] X. Zhu and D. Ramanan, (2012) Face detection, pose estimation, and landmark localization in the wild, 2012 IEEE Conference on Computer Vision and Pattern Recognition, pp. 2879-2886.
[10] Z. Cao, G. Hidalgo, T. Simon, S. -E. Wei and Y. Sheikh, (2021) OpenPose: Realtime Multi- Person 2D Pose Estimation Using Part Affinity Fields, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 43, no. 1, pp. 172-186.
[11] J. Shotton et al., (2013) Efficient Human Pose Estimation from Single Depth Images, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 35, no. 12, pp. 2821-2840.
[12] G. Borgefors, (1988) Hierarchical chamfer matching: a parametric edge matching algorithm, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 10, no. 6, pp. 849-865.
[13] S. Gold and A. Rangarajan, (1996) A graduated assignment algorithm for graph matching, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 18, no. 4, pp. 377-388.
[14] Bin Luo and E. R. Hancock, (2001) Structural graph matching using the EM algorithm and singular value decomposition, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 23, no. 10, pp. 1120-1136.
[15] L. P. Cordella, P. Foggia, C. Sansone, F. Tortorella and M. Vento, (1998) Graph matching: a fast algorithm and its evaluation, Proceedings. Fourteenth International Conference on Pattern Recognition, pp. 1582-1584 vol.2.
[16] S. Umeyama, (1988) An eigendecomposition approach to weighted graph matching problems, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 10, no. 5, pp. 695-703.
[17] D.K. Isenor, S.G. Zaky, (1986) Fingerprint identification using graph matching, Pattern Recognition,Volume 19, Issue 2, pp. 113-122.
[18] Gerard Sanromà, René Alquézar, Francesc Serratosa, (2012) A new graph matching method for point-set correspondence using the EM algorithm and Softassign, Computer Vision and Image Understanding, Volume 116, Issue 2, pp. 292-304.