Project 3: Robot Understanding of Human Behaviors Using Skeleton-Based Representations

Quinn Vo

***Abstract*— This paper will go over the implementations of several skeleton based representations and use Support Vector Machines (SVMs) to classify certain human behaviors from a subset of a well known dataset called the MSR Daily Activity 3D dataset. The dataset contains six activity categories including cheer up, toss paper, lie on sofa, walk, standup, and sit down. The library LIBSVM was used to learn a C-SVM model with a radial basis function (RBF) kernel on the training files. Then, the model created was used to predict the human behaviors on the testing files. The raw data had to be converted into a representation that would allow an SVM to accurately use features to predict human activity. Three such representations will be discussed in this paper: relative angles and distances (RAD), histogram of joint position differences (HJPD), and histogram of oriented displacements (HOD). Using a relative angles and distances (RAD) implementation, the SVM was able to predict with 62.5% accuracy. Using a histogram of joint position differences (HJPD) representation, the SVM was able to predict with 66.67% accuracy. Finally, using a histogram of oriented displacements (HOD) representation, the SVM was able to predict with 50% accuracy. This project was all done in Python without using Robot Operating System (ROS).**

I. INTRODUCTION

A. The dataset

The goal of this project is to create a model that is able to distinguish six different human behaviors. The data presented had many different files. Each file represented one of the six human behaviors listed below:

1) Cheer up
2) Toss paper
3) Lie on sofa
4) Walk
5) Stand up
6) Sit Down

Each file has a number of frames. Each frame represents a human skeleton that has the positions of 20 different joints in an XYZ axis. This is shown in Figure 1.

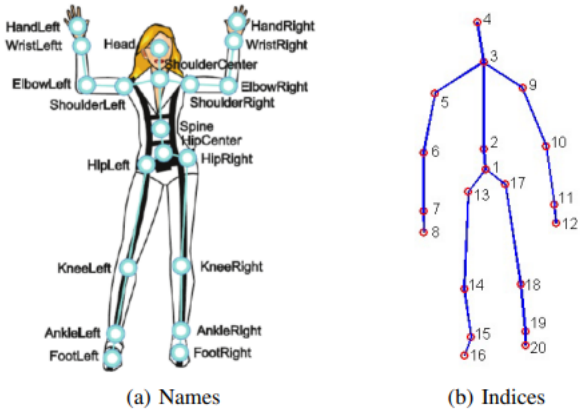

Fig. 1. Skeleton joint names and indices

The raw data had to be converted into a representation that would allow an SVM to accurately use features to predict human activity. Therefore, a discussion of the 3 different representations will follow.

1) Relative Angles and Distances (RAD)
2) Histogram of Joint Position Difference (HJPD)
3) Histogram of Oriented Displacements (HOD)

B. RAD representation

Using only selected joints, joint 1 was used as reference joint. Joints 4, 8, 12, 16, 20 were used to calculate relative distances and angles. A histogram was made for each distance from joint to reference joint, meaning that one frame in a file is one data point in a histogram that holds data for one of the distances from the joint to the reference joint. A histogram was also made for each angle for a pair of joints, using joint 1 as the reference joint. (5 for distances and 5 for angles). Outliers were removed; histograms are normalized; histograms are concatenated. 10 histograms made per instance (file). Figures 2 and 3 show what is going on with the RAD representation.

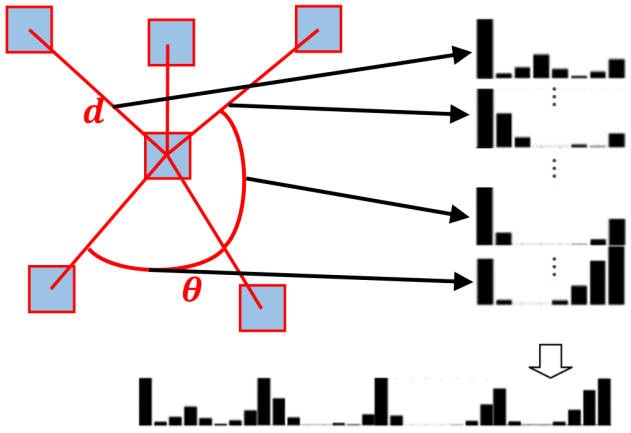

Fig. 2. RAD Visual

Algorithm 1: RAD representation using star skeletons

Input : Training set `Train` or testing set `Test`
Output : `rad_d1` or `rad_d1.t`

1: **for** *each instance in* `Train` *or* `Test` **do**
2: **for** *frame* $t = 1, ..., T$ **do**
3: Select joints that form a star skeleton (Figure 3);
4: Compute and store distances between body extremities to body center $(d_1^t, ..., d_5^t)$;
5: Compute and store angles between two adjacent body extremities $(\theta_1^t, ..., \theta_5^t)$;
6: **end**
7: Compute a histogram of N bins for each $\boldsymbol{d}_i = \{d_i^t\}_{t=1}^T$, $i = 1, ..., 5$;
8: Compute a histogram of M bins for each $\boldsymbol{\theta}_i = \{\theta_i^t\}_{t=1}^T$, $i = 1, ..., 5$;
9: Normalize the histograms by dividing T to compensate for different number of frames in a data instance;
10: Concatenate all normalized histograms into a one-dimensional vector of length $5(M + N)$;
11: Convert the feature vector as a single line in the `rad_d1` or `rad_d1.t` file.
12: **end**
13: **return** *`rad_d1` or `rad_d1.t`*

Fig. 3. RAD Algorithm

C. HJPD Representation

Using Joint 1 as the reference joint, joints 2-20 were used to calculate relative distances from the reference joint. The same as RAD implementation, but HJPD implementation ignores pairwise angles. It is also different from the RAD implementation in that it calculates the distances using all the joints as opposed to selecting a handful. This means that 19 histograms were made per instance. Outliers were removed; histograms are normalized; histograms are concatenated.

D. HOD Representation

Each joint was projected into 2D cartesian plans. Therefore, each joint had 3 projections (XY, XZ, YZ). This is shown in the visual below. For each pair of points in each projection, the angle was calculated and placed in one of the bins. The distance's magitude was used as the histogram's real-value count. Histograms were divided by number of data points it contained. Finally, they were concatenated.

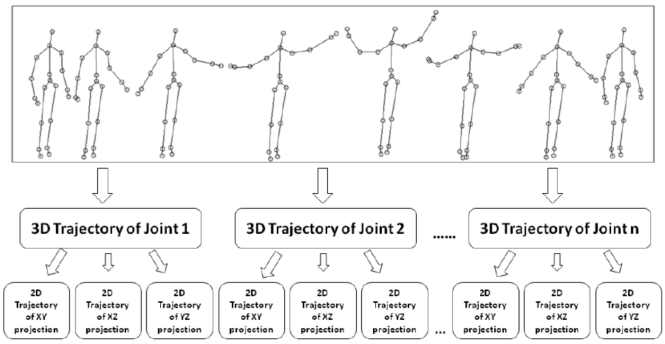

Fig. 4. HOD Visual

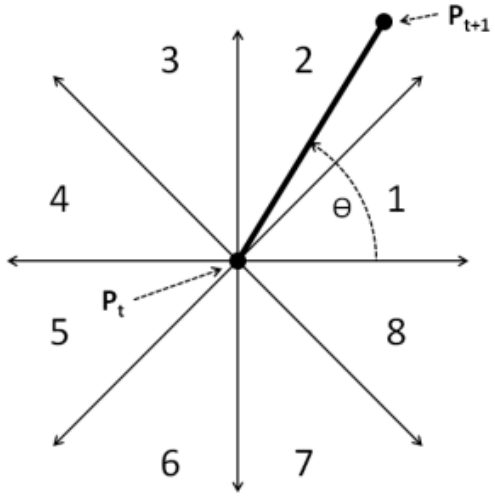

Fig. 5. 2D Projection and Bin Placement

E. Support Vector Machine

Support vector machines are one of the ways used in supervised learning for classification. In the case of this project, each of the instances had to be classified as one of the six human activities. A support vector machine (SVM) recognizes patterns using a separating hyperplane. SVM training algorithm builds a model that outputs an optimal hyperplane which assigns new examples into one category or the other. The library LIBSVM was used to learn a C-SVM model with a radial basis function (RBF) kernel on the training files. Then, the model created was used to predict the human behaviors on the testing files.

II. DESIGN

Using LIBSVM's default hyperparameters for the SVM, graphs showing the changes of accuracy according to differ-

ent numbers of bins were created in order to determine the best number of bins for each representation. Each of these graphs will be shown in the results section. After a number of bins were selected for each representation using the results of the graphs, hyperparameter selection had to be done on the SVM. LIBSVM had a built in script called grid.py that was able to determine the best hyperparameters to input for C (cost parameter) and gamma parameter. The results of the output of grid.py will also be shown under the results section.

III. RESULTS

A. RAD results

Using the RAD representation, it was clear that choosing 12 or 18 bins would provide the most accuracy as shown in the figure below. Ultimately, 18 bins was chosen and hyperparameter selection was performed using the grid.py script. The hyperparameters were found to be $C = 0.03125$ and $\gamma = 0.5$.

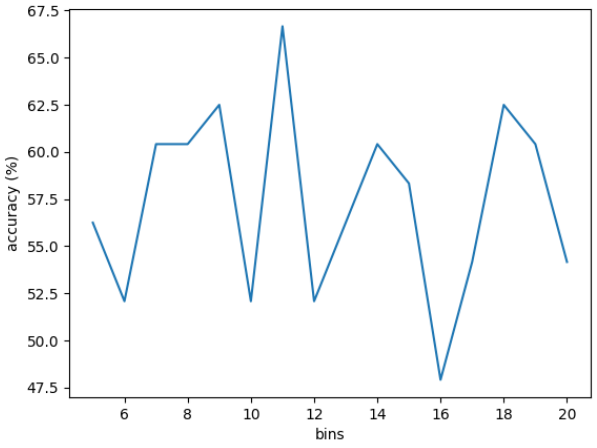

Fig. 6. RAD Sensitivity with Number of Bins

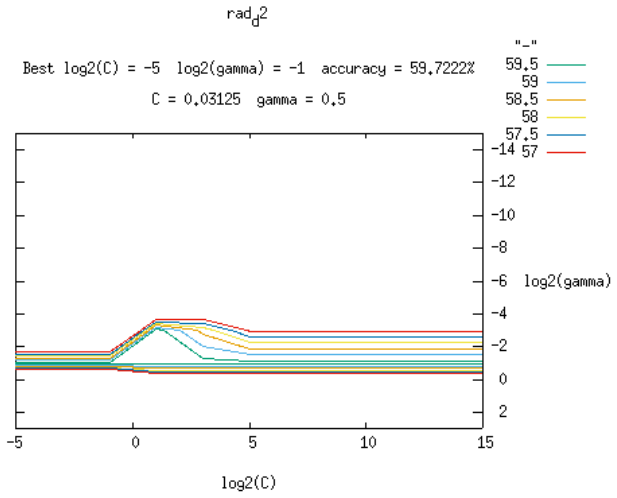

Fig. 7. RAD Hyperparameter Selection grid.py

The model was able to predict with 62.5% accuracy with the confusion matrix shown below.

```
Confusion Matrix:
[[8. 0. 0. 0. 0. 0.]
 [0. 7. 0. 2. 2. 0.]
 [0. 0. 3. 0. 0. 0.]
 [0. 0. 3. 5. 1. 0.]
 [0. 0. 2. 1. 5. 6.]
 [0. 1. 0. 0. 0. 2.]]

Accuracy: 62.5%
```

Fig. 8. RAD Results

B. HJPD results

Using the HJPD representation, it was clear that choosing 7 or 18 bins would provide the most accuracy as shown in the figure below. Ultimately, 18 bins was chosen and hyperparameter selection was performed using the grid.py script. The best hyperparameters were found to be $C = 2.0$ and $\gamma = 0.5$.

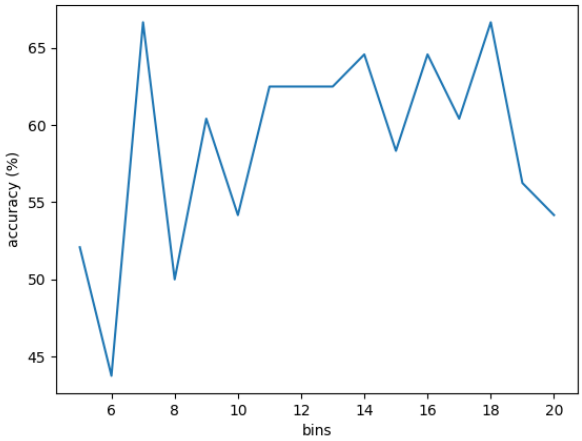

Fig. 9. HJPD Sensitivity with Number of Bins

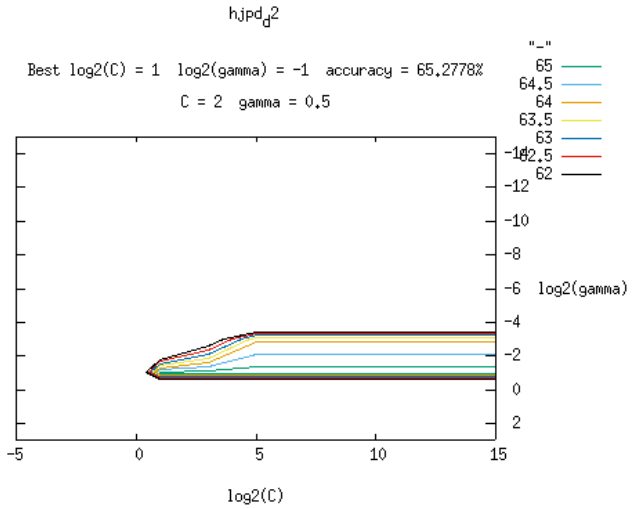

Fig. 10. HJPD Hyperparameter Selection grid.py

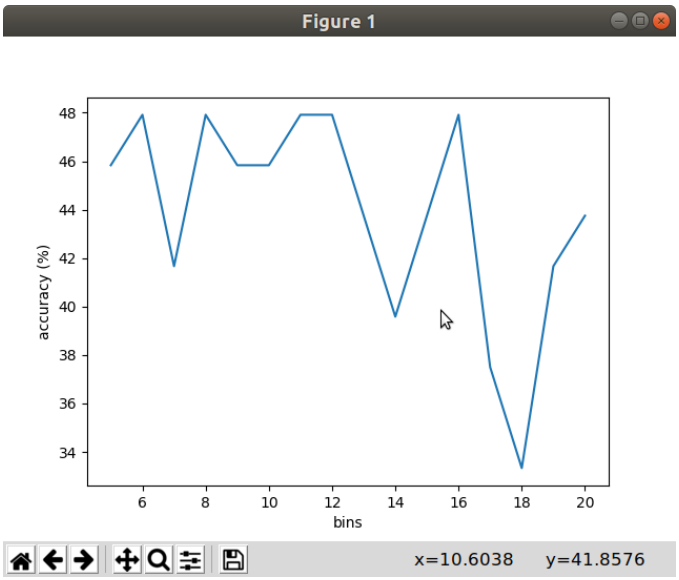

Fig. 12. HOD Sensitivity with Number of Bins

The model was able to predict with 66.67% accuracy with the confusion matrix shown below.

```
Confusion Matrix:
[[8. 1. 0. 0. 0. 0.]
 [0. 7. 1. 2. 1. 0.]
 [0. 0. 3. 0. 1. 0.]
 [0. 0. 1. 5. 1. 0.]
 [0. 0. 2. 0. 3. 2.]
 [0. 0. 1. 1. 2. 6.]]

Accuracy: 66.66666666666666%
```

Fig. 11. HJPD Results

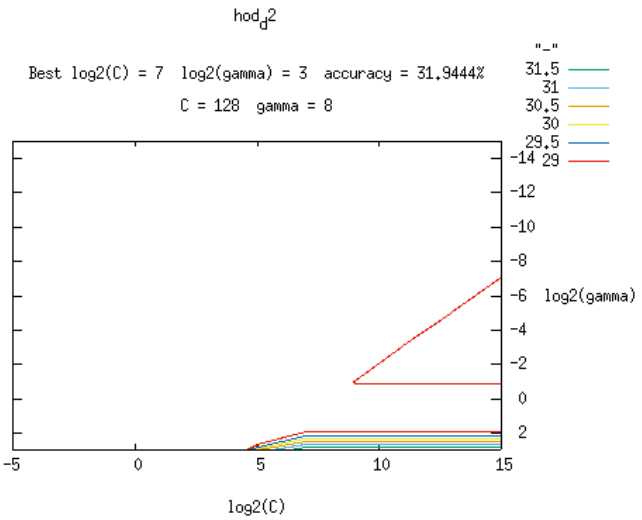

Fig. 13. HOD Hyperparameter Selection grid.py

C. HOD results

Using the HOD representation, it was clear that choosing 12 or 16 bins would provide the most accuracy as shown in the figure below. Ultimately, 12 bins was chosen and hyperparameter selection was performed using the grid.py script. The best hyperparameters were found to be $C = 128.0$ and $\gamma = 8$. However, more testing was performed and the gamma value ultimately used was $\gamma = 32$.

The model was able to predict with 50.0% accuracy with the confusion matrix shown below.

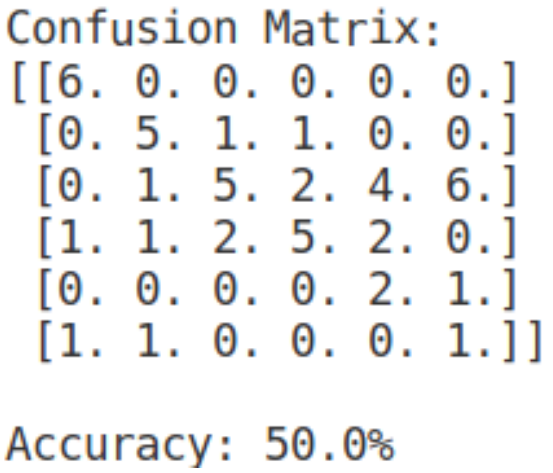

Fig. 14. HOD Results

IV. CONCLUSIONS

The significance of the findings are that we were semi-accurately able to predict certain human behaviors. From the confusion matrices, cheer up had the best classification results in all three representations. In the future, work could go into looking at better ways to extract features from the raw data set. In the real world, 60% accuracy is not something that is reliable. If something needed to be deployed and used in real practical scenarios, we would want that accuracy to be above 99%.