Patient Monitoring using Activity Recognition

Group 1, Dec 5th, 2022

Aditya Bhat
NetID: ab2260

Prashant Kanth
NetID: ppk31

Parth Goel
NetID: pg514

Rishika Bhanushali
NetID: rb1182

ABSTRACT

Human Activity Recognition (HAR) is the process of predicting the activity of a person based on a trace of their movement using a camera. The main goal of the project is to recognize the patterns from the dataframes of the videos and correctly tagging them to the corresponding activity. In this project, the features are extracted by passing the frames to the key point detection model (KPDM), and the activities are classified into 6 *different classes by CNN-LSTM classification model. The combined model of CNN and LSTM is implemented to improve the HAR accuracy.* **Code:** https://github.com/parthgoe1/Patient-Activity-Recognition-System

I. INTRODUCTION

Hospitals and other medical centers have facilities of keeping patients under medical attention to monitor their progress. The nurse carefully monitors the patients and takes necessary action in case of any unusual activity or sense of emergency. However, sometimes patients cannot afford this service as it's expensive. If we have an automated system in place which tracks patient activity and informs concerned authorities to take immediate actions in case of emergency then many people would be benefitted. This will incur camera installation and setting up of other logistics but would be still cheaper than human monitoring. Also, There are many instances of complications or even deaths in rehab centers because of no or delayed action incase of events like collapse, heart attack or seizures. This can be mitigated by having a 24*7 surveillance in place that recognizes an abnormality in patient's behavior and alerts respective authorities. We introduce a novel model in this paper to monitor patient activity using the CNN-LSTM model.

II. RELATED WORK

Traditionally, most efforts in activity recognition are focused on improving accuracy by creating larger architectures which take as input several set of frames and produce predictions. These are aggregated over time and sampled either densely or randomly. Gowda et al. [1] proposed a smart frame selection method that improved over many state-of-the-art models. The authors press on the importance of smartly choosing the frames of the video by using an attention and relational network. The method includes two branches. The first computes a score δ_i for each frame, and the second computes a score γ_i of a pair of frames. Given n frames, top m frames are chosen using a final score which is multiplied both score δ_i and γ_i. Finally, a classifier is used for the final prediction.

The paper's main focus is to improve the inference efficiency of current action recognition backbones on trimmed videos and illustrate that an action model can accurately classify an action with a single pass over the video unlike the multi-clip sampling common with SOTA by learning to drop non-informative features.Dense sampling is considered the best solution for sampling video clips and is employed in almost all action recognition models. While this achieves the best performance, it is also highly inefficient. To solve this issue, SCSampler and MARL strategies were proposed. However, its hard selection mechanism can potentially disregard useful information, especially on videos with complex actions. In this paper authors have proposed a novel technique that removes the need for dense sampling via feature compression. This action recognition inference strategy is called Selective Feature Compression (SFC). This is proved to greatly increase model inference efficiency without compromising accuracy.

Some researchers have a different approach to look at the action recognition problem. In [2] authors believe that a core component in obtaining a detailed understanding of people in images and videos: human 2D pose estimation. They have followed a bottom up approach where they first detect the pose and then detect the person. The inference time is invariant to the number of people in the image. This gave better results as compared to the top-down ones.

The attention recognition framework has extended a lot on inferences of humans on mobile devices. The authors bazarevsky et al [3] have chosen a key point detection model which is taken from both worlds (heatmap and regression-based). The model produces heatmaps for all joints using an encoder-decoder network, which then uses a regressor to get the coordinates of all joints. The authors emphasize the need to augment the images by aligning the person so that the point between the hips is located at the center of the square image passed as input.

Sequence data are data points that are ordered in a meaningful manner such that previous data points or observations provide information about later data points or observations. Sequence data can be represented as observations of one or more characteristics of events over time. There are different types of sequence modeling tasks: one-to-one, one-to-many, many-to-one. Figure 1

HAR is a many-to-one task, where the network needs to

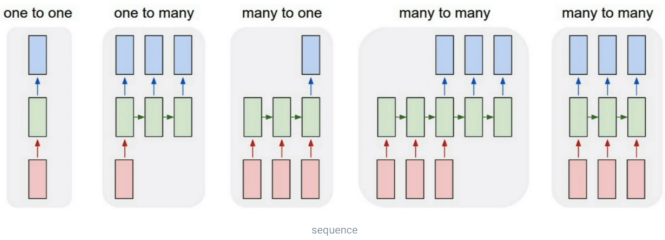

Fig. 1. Types of sequence modeling tasks [Source: calvinfeng.gitbook.io]

identify the activity in a video clip. A video clip is a sequence of video frames, therefore the input is a sequence of data. RNN (Recurrent Neural Network) is a deep learning algorithm that is specialized for sequential data processing. However, in practice RNNs are not good at capturing long-range dependencies. This is mainly due to the vanishing gradients problem. For our problem of Human Activity Recognition, we chose to use LSTM. LSTMs, Long Short Term Memory networks are a special kind of RNN, capable of learning long-term dependencies. All recurrent neural networks have the form of a chain of repeating modules of neural network. The repeating chain-like structure in LSTMs is shown in Figure 2.

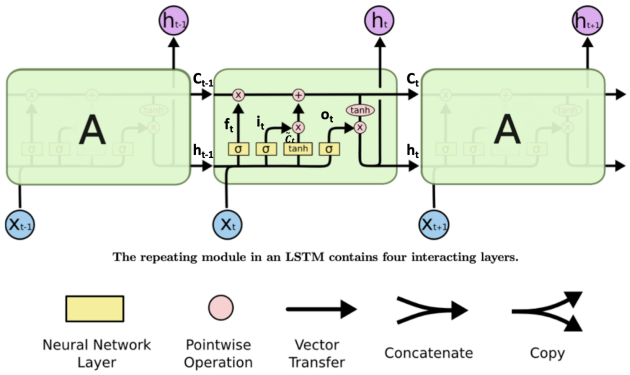

Fig. 2. LSTM architecture [Source: colah.github.io]

The key to LSTMs is the cell state, the horizontal line running through the top of the diagram. The LSTM has the ability to remove or add information to the cell state, carefully regulated by structures called gates.

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \quad (1)$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (2)$$

$$\tilde{C}_t = tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \quad (3)$$

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \quad (4)$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \quad (5)$$

$$h_t = o_t * tanh(C_t) \quad (6)$$

In the past Deep Neural Networks (DNNs), Convolutional Neural Networks (CNNs), and Long-Short Term Memory

Fig. 3. Categories: Normal (row1),Abnormal (row2)

Recurrent Neural Networks have been individually applied for large vocabulary continuous speech recognition tasks and have achieved tremendous results. However, all these neural networks are limited in their modeling capabilities and the authors [4] of this paper believe that speech recognition performance can be improved by combining these networks in a unified framework.

One issue with LSTMs is that the temporal modeling is done on the input feature x_t. However, higher-level modeling of x_t can help to disentangle underlying factors of variation within the input, which should then make it easier to learn temporal structure between successive time steps. For example, it had been shown that CNNs learn speaker-adapted/discriminatively trained features, and thus remove variation in the input. Thus, it could be beneficial to proceed LSTMs with a few fully connected CNN layers. Doing this the authors show that it provides a 4-6% relative improvement in WER over an LSTM, the strongest of the three individual models.

III. Dataset

A. *Data Gathering*

Since it was difficult to get real data of a rehab center and then train the model for some specific events, we generated synthetic data using Unity3D. We divided the training data into two broad categories; Activities that are considered normal and those which are abnormal. Please refer to Figure 3.
Normal activities- Sitting, Standing, Walking ,Sleeping
Abnormal activities- Falling, pain, coughing

B. *Data Preprocessing*

We divide the simulations into frames for keypoint detection model. Further, the output from keypoint models are normalized to pass it on to activity detection model. As shown in Figure 4

IV. Method

We used the 33 keypoints from mediapipe as the first set of features. Since each point is 3 dimensional(x,y,z), we had a total of 99 features. We computed additional 19 features

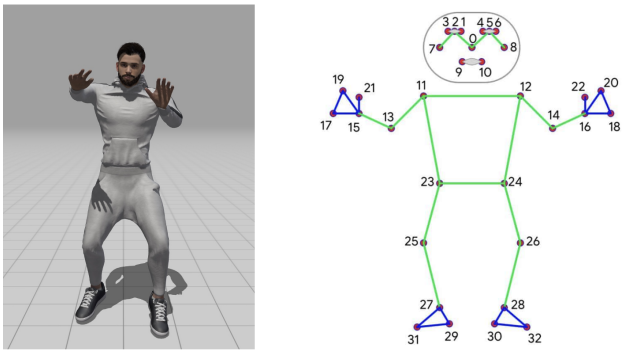

Fig. 4. Frames to Keypoints

which are distances between selected keypoints. They are: (refer figure 5)

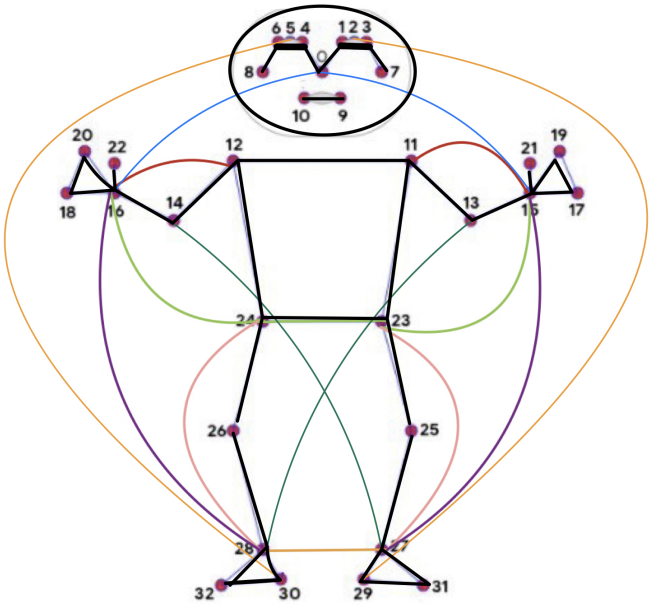

Fig. 5. Keypoints Distances

15-16	15-11	12-16	15-23	16-24
23-24	16-28	15-27	23-27	24-28
28-27	5-30	2-29	14-27	13-28
0-15	0-16	26-25	31-32	

These points are selected based on the estimated position and distance between the skeletal points for various activities. Therefore we gather a total of $99 + 19$, i.e 118 normalized feature points, and are passed as input to our model.

In terms of model input, as we know that LSTMs take 3-dimensional data in which the dimensions are [samples, time steps, features], CNN-LSTMs take 4-dimensional data in which the dimensions are [samples, n_{steps}, n_{length}, features], where n_{steps} means the number of steps we want to divide our timestep in, and n_{length} means the length of the timestep after dividing it into steps. So basically n_{steps} * n_{length} = time steps. So for the training purpose of our model we have taken $52,128$ samples, our time step $= 50$ divided into two steps so $n_{steps} = 2$ and $n_{length} = 25$.

For CNN-LSTM architecture, let's say TDCK represents Time Distributed Conv1D layer with K dimensions, TDCDK represents Time Distributed Conv1D Dropout layer with K dimensions, TDMP represents Time Distributed MaxPooling1D layer with K dimensions, TDFK represents Time Distributed Flatten layer with K dimensions, LSTMD represents LSTM dropout with K units and DEK represnts Dense layer with K dimensions. So our models architecture is:

TDC23x64-TDCD21x64-TDMP10x64-TDF640-LSTMD100-LSTMD100-LSTMD100-LSTMD100-DE100-DE6

Model architecture figure is given in Figure 6

V. Experiments

We experimented with different features and different types of architecture.

A. *Different Features Experimented:*

- 2 dimensional feature, 33 keypoints on both x and y axis. Making a total of 66 features.
- Included the z axis as well to get the depth affect, giving $33 \times 3 = 99$ features.
- Pose Normalized Keypoints. μ : hip center, σ : farthest keypoint from hip center. Features: distances between 17 pairs of keypoints (Scale Invariant and motion sensitive features)
- Finally, added 19 distances between different keypoints to these 99 features giving a total of 118 features.

B. *Different Architectures Experimented:*

- ***LSTM:*** 2 layers, hidden state dimensions: 128, learning rate: 2e-4 decayed by 0.5 every 50 epochs for 120 epochs, Dropout (0.5). Linear layer with Softmax Activation as Output for 6 classes. Please refer figure 7 for training curve.
  Limitations:
  - Inconsistent validation results for standing/walking, sleeping/falling.
  - Low generalization
- ***LSTM:*** 3 hidden layers with 500, 100 and 50 'units', Dropout (0.5), dense layer with 64 hidden state dimensions. Linear Layer with Softmax Activation as Output for 6 classes. Please refer figure 8 for training curve.
  Limitations:
  - More transition inconsistency.
  - More misclassifications.

Apart from these, we also experimented with bi-directional LSTM and GRUs. However, we didn't find much improvement.

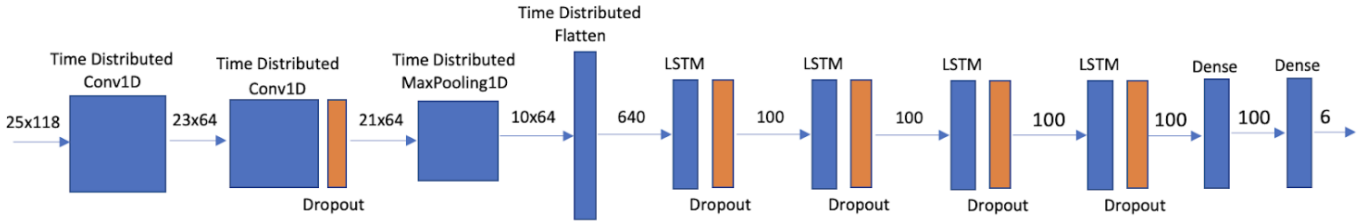

Fig. 6. Architecture of CNN-LSTM

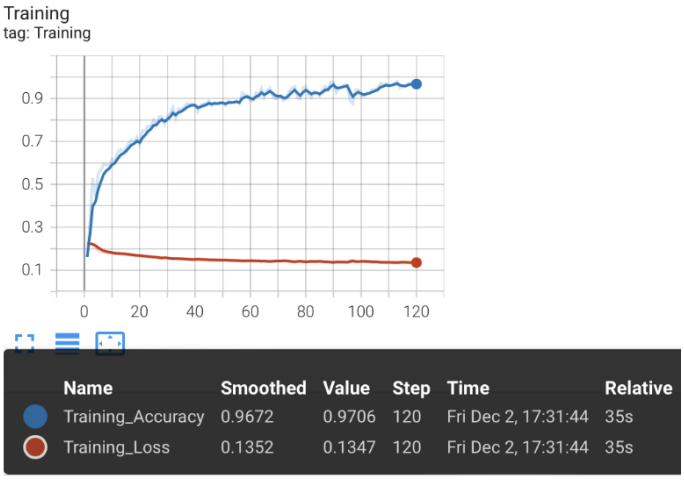

Fig. 7. Architecture A: Training Curve

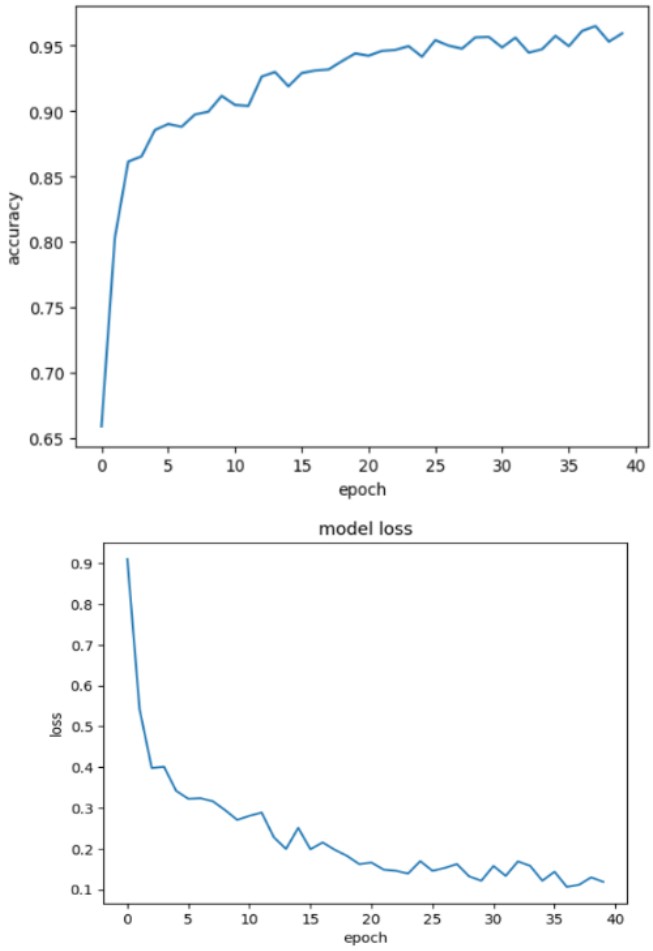

Fig. 8. Architecture B: Training Curve

VI. Results

In this section, we present classification results (Figure 13, 14, 11, 12) by our network as well as few misclassifications (Figure 9, 10) that occur due to similar pose keypoint features.

VII. Conclusion and Future Work

We were successfully able to predict Abnormal activities i.e, pain, cough, and falling but faced minor inconsistencies in walking and standing. Considering the skeletal points, both walking and standing have similar features. We also observed that a lot of false positives occur during activity transition. We plan on adding further normal and abnormal activities so as to imitate more human actions. We believe the false positive classification of standing and walking could be minimized by introducing a person detection model (yolov5 or mask-rcnn) and pass only the bbox of the person to the keypoint detection model. Therefore the problem boils down to static-type and movement-based activities. This way standing and walking will altogether be a different types and can almost never be misclassified.

References

[1] S. N. Gowda, M. Rohrbach, and L. Sevilla-Lara, "Smart frame selection for action recognition," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, pp. 1451–1459, 2021.

[2] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh, "Realtime multi-person 2d pose estimation using part affinity fields," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7291–7299, 2017.

[3] V. Bazarevsky, I. Grishchenko, K. Raveendran, T. Zhu, F. Zhang, and M. Grundmann, "Blazepose: On-device real-time body pose tracking," *arXiv preprint arXiv:2006.10204*, 2020.

[4] T. N. Sainath, O. Vinyals, A. Senior, and H. Sak, "Convolutional, long short-term memory, fully connected deep neural networks," in *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 4580–4584, IEEE, 2015.

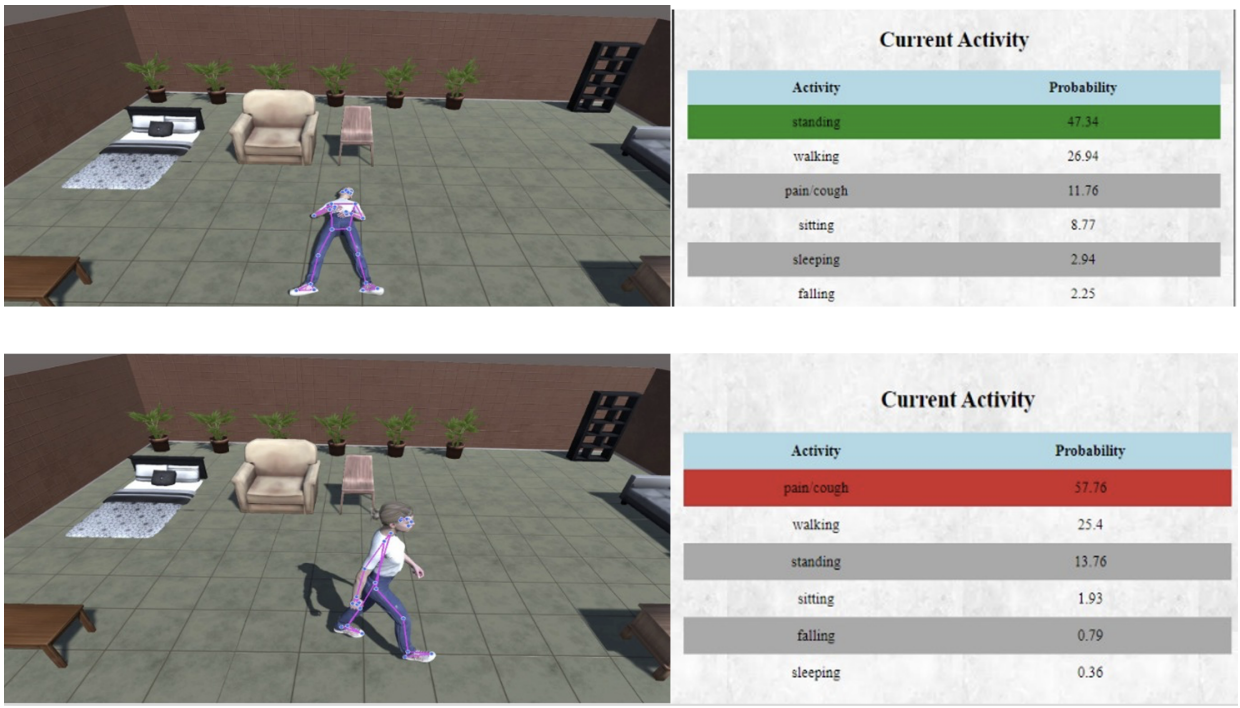

Fig. 9. Result: Simulated Data, Incorrect Classifications

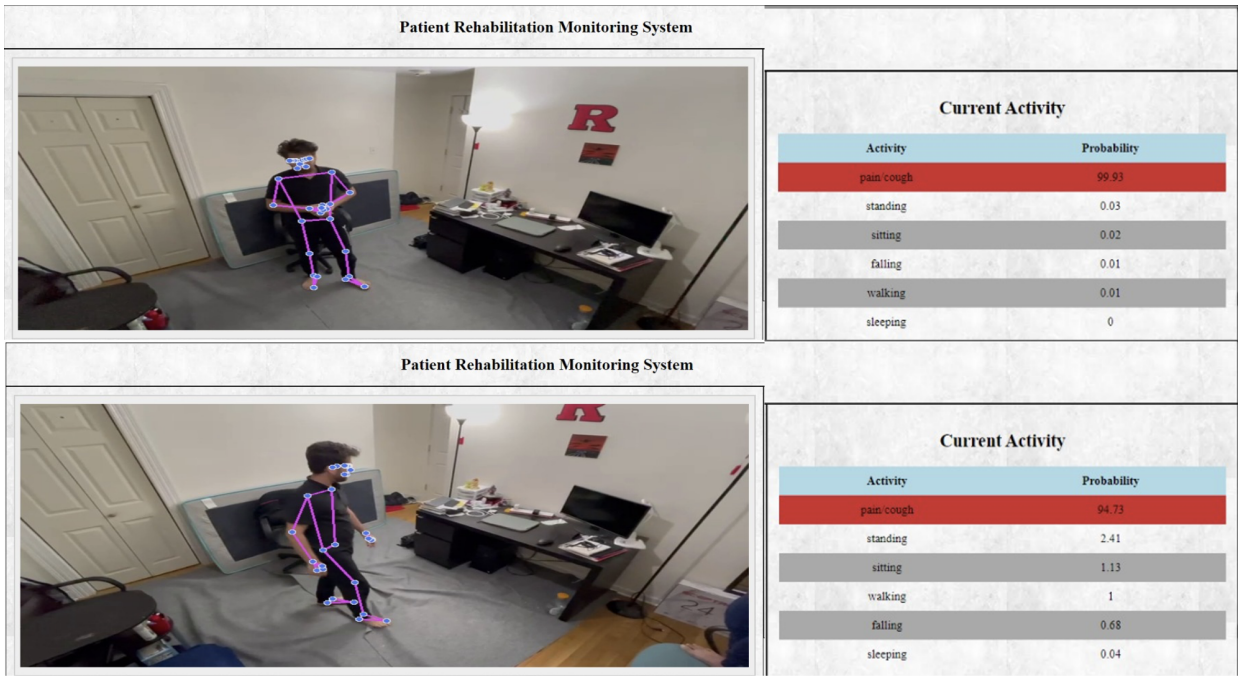

Fig. 10. Result: Real Data, Incorrect Classifications

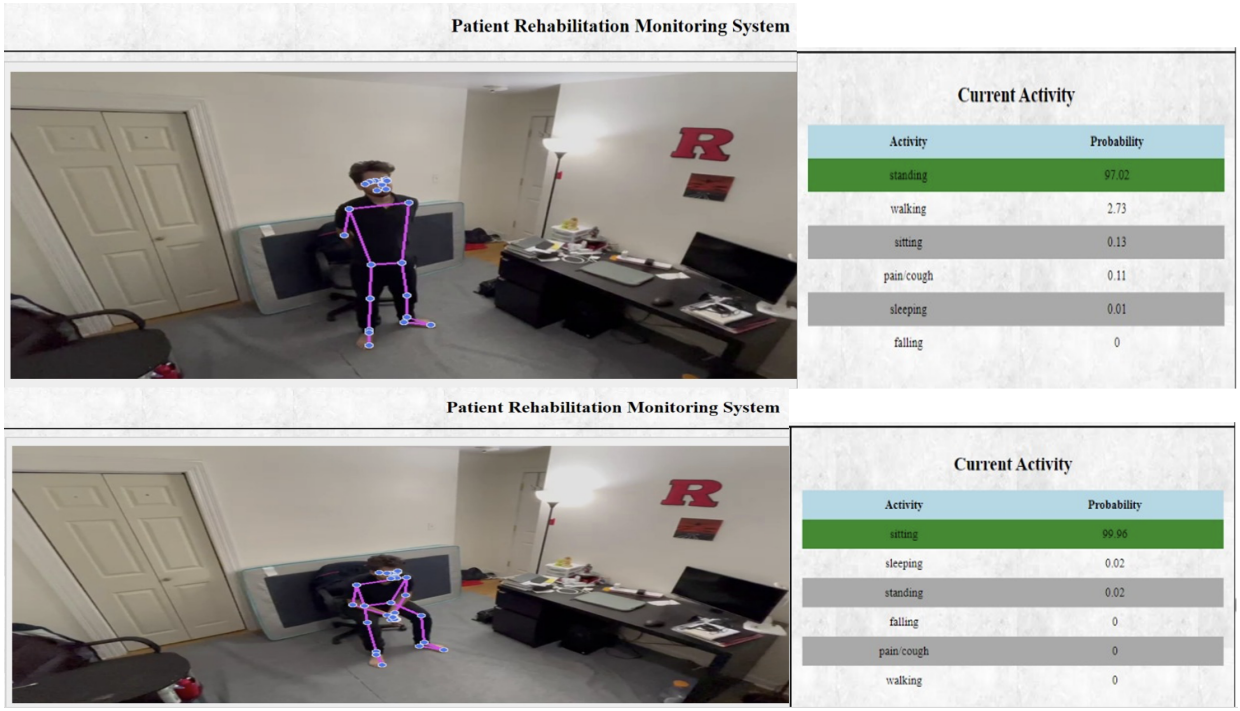

Fig. 11. Result: Real Data, Classification

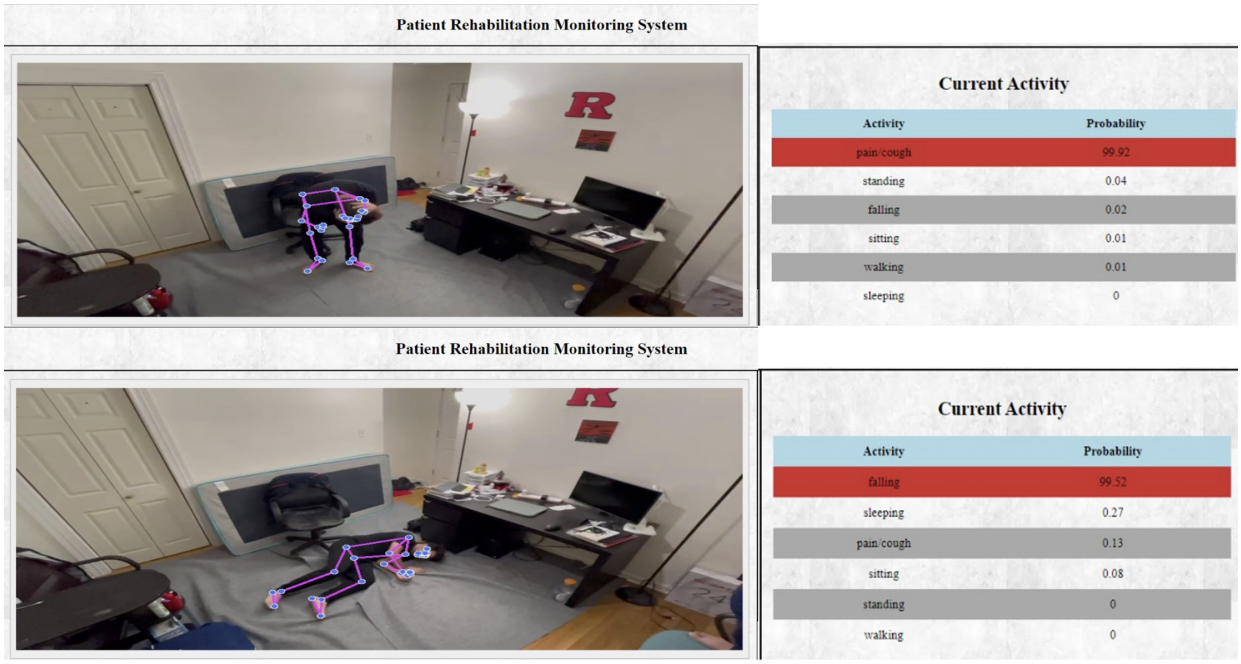

Fig. 12. Result: Real Data, Classification

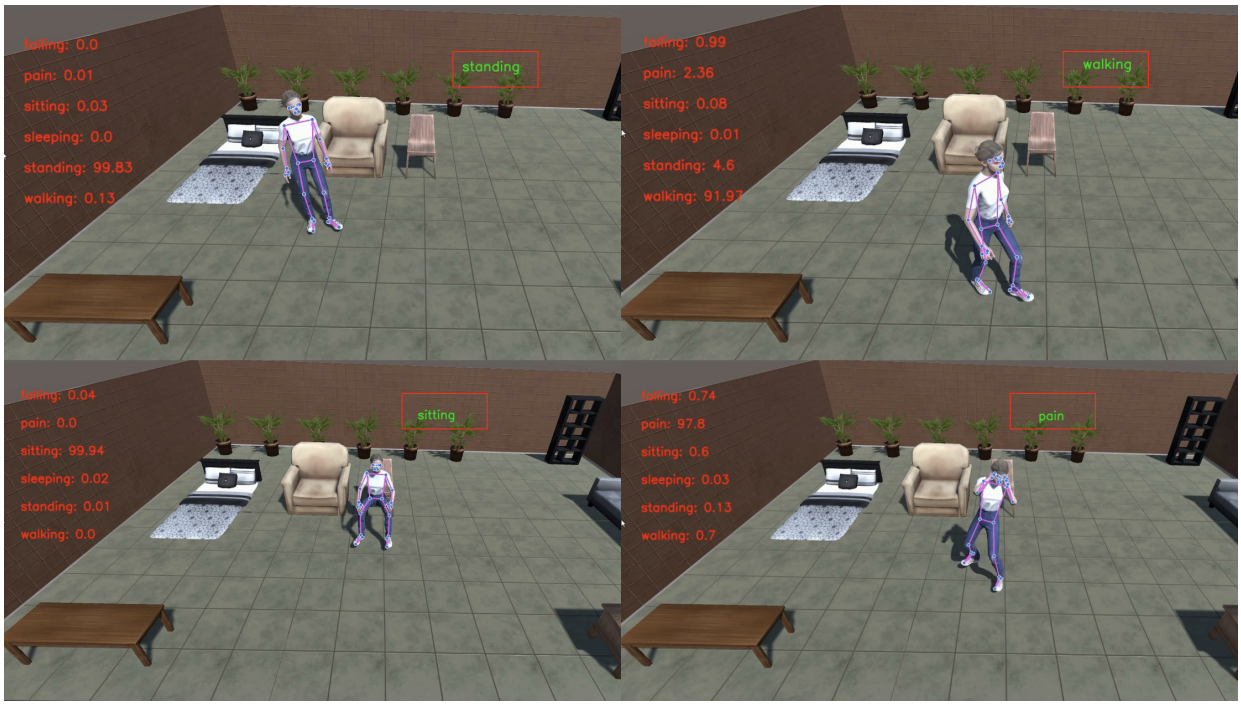

Fig. 13. Result: Simulated Data, Correct Classifications

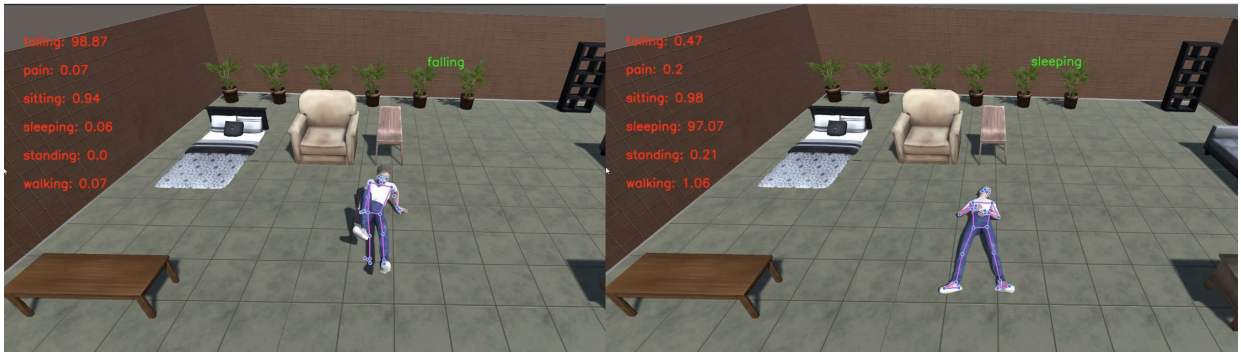

Fig. 14. Result: Simulated Data, Correct Classifications