Exploring Techniques to Improve Activity Recognition using Human Pose Skeletons

Bharath Raj N.
SSN College of Engineering, Chennai, India
bharathrajn98@gmail.com

Anand Subramanian
SSN College of Engineering, Chennai, India
anand.subu10@gmail.com

Kashyap Ravichandran
SSN College of Engineering, Chennai, India
kashyapravichandran@gmail.com

Dr. N. Venkateswaran
SSN College of Engineering, Chennai, India
venkateswarann@ssn.edu.in

Abstract

Human pose skeletons provide an explainable representation of the orientation of a person. Neural network architectures such as OpenPose can estimate the 2D human pose skeletons of people present in an image with good accuracy. Naturally, the human pose is a very attractive choice as a representation for building systems aimed at human activity recognition. However, raw pose keypoint representations suffer from various problems such as variance to translation and scale of the input images. Keypoints are also often missed by the pose estimation framework. These, and other factors lead to poor generalization and learning of networks that may be trained directly on these raw representations. This paper introduces various methods aimed at building a robust representation for training models related to activity recognition tasks, such as the usage of handcrafted features extracted from poses with the intent of introducing scale and translation invariance. Additionally, the usage of train-time techniques such as keypoint dropout are explored to facilitate better learning of models. Finally, we conduct an ablation study comparing the performance of deep learning models trained on raw keypoint representation and handcrafted features whilst incorporating our train-time techniques to quantify the effectiveness of our introduced methods over raw representations.

1. Introduction

Human pose skeletons play an increasingly important role in the context of activity recognition, owing to the potential for capturing localized context of actions based on body movements. Human Pose Estimation remains an active research problem, with various algorithms providing for fast and robust human pose estimation in recent times [3, 6]. However, there are considerable pitfalls in directly using the coordinates of the human pose skeletons as such for activity recognition. Usage of the raw keypoint coordinates as features suffers from the problems like scale and translation variance, since they vary with changes in the scale and orientation of an image. Keypoints may also be missed by a pose estimation framework during detection, leading to insufficient data in representing the pose of a person. We discuss these potential pitfalls in detail and propose methods to overcome those, in the context of using 2D human pose skeletons for video-based activity recognition with our contribution being as follows:

- An analysis of the pitfalls and difficulties associated with directly using human pose skeletons as features
- A baseline feature extraction method along with techniques such as keypoint dropout and feature propagation to overcome pitfalls
- A baseline heuristic pose tracker that can be used to associate human pose skeletons across frames for video-based activity recognition
- Experiments and ablation studies to verify and validate our suggested solutions

It is important to mention that our proposed experimental setup and contributions are intended to be pose representation and tracker agnostic. We believe the contributions put forth here would serve as useful guidelines for future works that build upon the aforementioned agnostic components, either in isolation or in tandem.

2. Related Work

Activity recognition has long been an active area of research, with different approaches adopted to address the problem. Deep learning approaches involved the usage of CNNs to encode temporal information across multiple frames [9, 17] as well as the combination of LSTMs with CNNs [5] in interpreting sequences of images for the

purpose of activity recognition.

A valuable representation format for extracting information pertaining to variations in a person's activity in a localized manner is the human pose. Human pose extraction from images has witnessed a variety of methods implemented over the past years. Regional Multi-Person Pose Estimation [6], is a top down approach to perform pose estimation. The method first utilized a pre-trained Single Shot Detector (SSD) to detect person candidates from which poses were extracted through the use of a Symmetric Spatial Transformer Network (SSTN) and a Single Person Pose Estimator (SPPE). The OpenPose [3] architecture is a bottom-up approach to human pose estimation, wherein a CNN is used to extract feature maps for the keypoints, which helps in the construction of a bipartite graph used to create the human pose.

As suggested earlier, representation of human activity as a movement of joints and body parts over several images provide an avenue for more accurately localizing and capturing features that describe the variation of activities in videos. Singh et al. [16] employed the use of handcrafted features, namely the orientation or the angles between limbs from the detected pose of individuals to identify individuals engaging in violent behaviour. Arbués-Sangüesa et al. [1] constructed feature vectors for pose keypoints by extracting activation maps from the VGGNet model to incorporate information for tracking of basketball players.

3. Human Pose Skeletons as Features for Activity Recognition

In this section, we provide an overview of the pose skeleton structure, along with a study of the potential pitfalls in directly using the raw keypoints from the human pose for activity recognition.

3.1. Overview of Pose Skeleton Structure

The human pose skeleton displayed in Figure 1 has 18 keypoints (or joints). The raw feature vector for a pose skeleton with N keypoints can be represented by a 2N dimensional vector comprising the X and Y coordinates of the keypoints as shown in equation (1)

$$F = [x_1, y_1, x_2, y_2, x_3, y_3, \ldots, x_N, y_N] \quad (1)$$

While this feature representation may seem as a good candidate for training a model, it has a number of pitfalls.

3.2. Pitfalls

3.2.1 Variance to Transformations

Consider the pose representation for the people in Figure 2. Note that, all four people have the same orientation and would be performing the same activity. However, if we extract a feature representation for all four people using equation (1), they would end up having different features due to their location and size. That is, the resultant feature vector is not invariant to linear transformations of the pose since the feature vector formulation of equation (1) is heavily dependent on the coordinates of the keypoints.

Invariance to rigid-body and affine transformations is desired in a feature vector as the person may appear anywhere in the image (with any orientation) having the same pose. Creating a feature vector representation that is invariant to these transformations (some, if not all) can improve the performance and generalization ability of the activity recognition model.

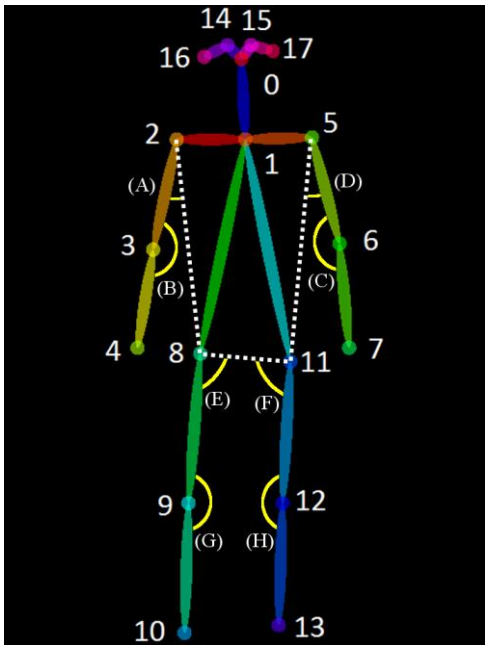

Figure 1: The angles (marked A to H) considered as features are marked in the human pose skeleton.

3.2.2 Need for Temporal Information and Pose Tracking

In the case of activity recognition from videos, temporal variation of the features extracted from the pose skeletons could be considered vital to resolve ambiguity between classes. For instance, in the course of the activities "Jumping Jacks" and "Sit Ups", there is considerable similarity when the person comes to rest towards the end of the activity. An activity recognition algorithm that works on a frame-by-frame basis may fail to resolve these instances. Naturally, by providing the feature information across multiple frames in a video, the algorithm can distinguish the classes with better accuracy.

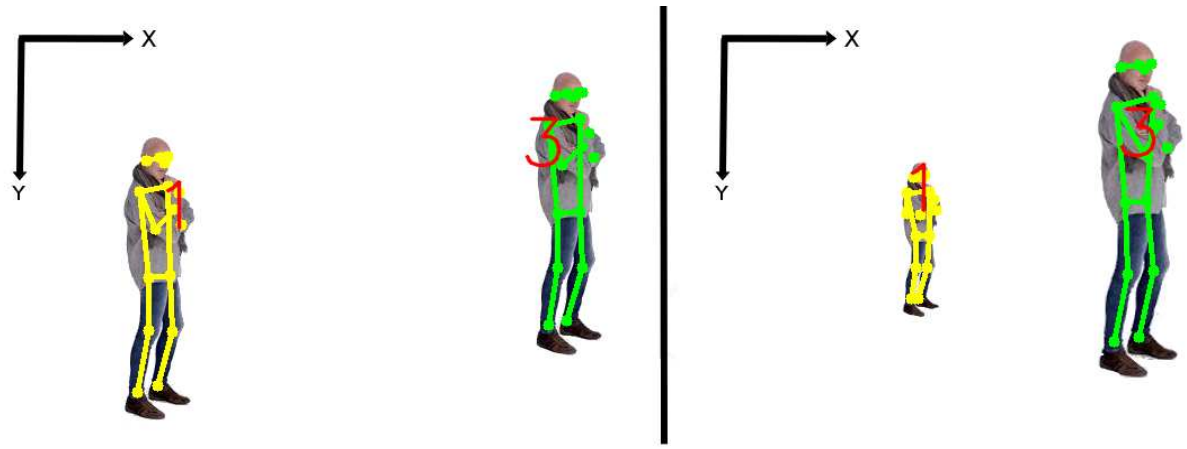

Figure 2: People representing the same orientation but with different translation from the origin and scale.

However, the problem is that a single frame may comprise multiple people in it. Moreover, the pose estimation algorithm may have spurious detections leading to detection of additional people. In order to use temporal information, we need to know exactly which pose skeleton belongs to the target person. This requires the usage of an object tracker to associate the pose with the person.

3.2.3 Missing Keypoints

At times, our pose estimation algorithm may not detect all keypoints of a pose skeleton. For instance, the OpenPose framework assigns such missing keypoints the coordinate location (0,0) with a confidence score of 0. This is a potential problem for two reasons:

First of all, we need to provide a value for the missing keypoints before using them in a feature vector representation (such as in equation (1)). Care must be taken while giving appropriate values for the missing keypoints; the imputed values should not hinder the ability of the rest of the keypoints to represent the orientation of the person properly.

Second of all, missing keypoints are problematic for trackers that work directly using keypoint information. The tracker must be robust enough to associate pose information across frames even if a few keypoints are missing.

4. Creating Robust Features from Human Pose Skeletons

In this section, we introduce a set of baseline feature extraction methods and train-time techniques. These techniques are introduced with the intention of making the activity recognition models based on human pose skeletons robust to the pitfalls introduced in the previous section.

4.1. Transformation Invariant Features

4.1.1 Centre of Gravity and Angle Features

One such transformation invariant feature representation involves computing the normalized distances of the keypoints from their Center of Gravity (CG) [14]. To obtain these features, we calculate the distance of each keypoint from the CG of the human pose skeleton. We then normalize the above distances by the longest vertical distance (d) in the human pose skeleton. In addition to this, we modify the equation such that it only considers keypoints that are visible (c=1) so that the location of CG is not affected significantly by missing keypoints. The calculation of the CG features are better explained by the following equations:

$$CG_x = \frac{\sum_{i=1}^{N} x_i c_i}{\sum_{i=1}^{N} c_i}; CG_y = \frac{\sum_{i=1}^{N} y_i c_i}{\sum_{i=1}^{N} c_i} \quad (2)$$

$$x'_i = \frac{x_i - CG_x}{d}; y'_i = \frac{y_i - CG_y}{d} \quad (3)$$

$$F_{CG} = [x'_1, y'_1, x'_2, y'_2, x'_3, y'_3, \ldots, x'_N, y'_N] \quad (4)$$

In the above equation, (x_i, y_i) is the XY coordinate of a keypoint, (c_i) is a boolean value that denotes that visibility of a keypoint and N is the number of keypoints in the pose skeleton. We call the resultant feature vector obtained in equation (4) as Centre of Gravity features (F_{CG}). This feature vector is nearly translation and scale invariant. The invariance may be very slightly affected by missing keypoints and variations in the longest vertical distance. However, for our baseline study we find this representation to be adequate.

Another feature representation that is commonly used are the Angle features [16]. In our case, we calculate the angles between certain pairs of the limbs of the human pose skeleton as shown in Figure 1. This is illustrated by

the below equation:

$$\theta_i = \tan^{-1}\left|\frac{m_1 - m_2}{1 + m_1 m_2}\right| \quad (5)$$

$$F_\theta = [\theta_A, \theta_B, \theta_C, \ldots, \theta_H] \quad (6)$$

Each angle (θ_i) is computed by considering the slopes (m) of the lines connecting the keypoints as illustrated in Figure 1. We call the resultant feature vector comprised of different angles obtained in equation (6) as Angle features (F_θ). Besides being invariant to translation and scale, angle features are also invariant to rotation. Figure 1 depicts the angles we have considered for our feature vector.

4.1.2 Affine Transform Augmentation

Typically, it is difficult to construct a feature vector that is truly robust to all rigid-body and affine transformations. Thus, we utilize augmentation mechanisms that randomly flip, shear, scale or rotate human pose skeletons before feature vectors are extracted from them. For all such transformations, the CG is fixated as the center of the transformation. We believe that this would encourage the activity recognition model to be more robust to affine transformations.

4.2. Using Temporal Information

4.2.1 Velocity Features

The displacement of the position of a keypoint with time (i.e. velocity) is a useful temporal feature. In our case, we calculate the displacement of the keypoint with respect to the previous frame as shown in the below equation:

$$V_{xi}(t) = \frac{x_i(t) - x_i(t-1)}{d(t)}; V_{yi}(t) = \frac{y_i(t) - y_i(t-1)}{d(t)} \quad (7)$$

$$F_{vel} = [V_{x1}, V_{y1}, V_{x2}, V_{y2}, V_{x3}, V_{y3} \ldots, V_{xN}, V_{yN}] \quad (8)$$

We call the resultant feature vector obtained in equation (8) as Velocity features (F_{vel}). The velocity of each keypoint has the unit (pixels/frame). Much like the CG features, the velocity values are scaled by the longest vertical length (d(t)) of the human pose skeleton. As before, this feature vector is nearly translation and scale invariant. The invariance may be very slightly affected by missing keypoints and variations in the longest vertical distance.

4.2.2 LSTM for Activity Recognition

In order to make optimal use of the temporal information gathered, the use of a time series model such as an LSTM [7] is immensely advantageous.

In our case, we collect the features extracted from a human pose skeleton across multiple frames and then feed it to an LSTM model. In case there are multiple persons in a single frame, a pose tracking mechanism is used to associate the correct poses with each person. The pose tracking mechanism used in this paper is discussed in a later section.

4.3. Handling Missing Keypoints

4.3.1 Feature Propagation

If we opt to use temporal information, we can pass the feature of a keypoint from the previous timestep if that keypoint is missing in the current timestep. If the pose does not change significantly across two timesteps, this technique should be more robust than imputing zero values for missing keypoints.

4.3.2 Keypoint Dropout

To make the model generally robust to missing keypoints, we propose a novel training technique called keypoint dropout. In this mechanism, we manually dropout a percentage of keypoints of the human pose skeleton before feature extraction while training the model. This way, we believe that the model learns to deal with missing keypoints much better. This technique can be used in conjunction with feature propagation.

5. Experiments

To validate the methods proposed in the previous sections, we propose a number of experiments aimed at analyzing the effect of the proposed techniques on the performance of activity recognition models. The problems highlighted are specifically ones that are characteristic of using pose representations for activity recognition, and we look to overcome these and suggest best practices that help in building more robust systems.

We begin with defining the experimental setup with details about the dataset used, feature extraction module, activity recognition methods and our custom pose tracking algorithm. Then, we define the set of experiments and their intended scope. The results of these experiments are analyzed in the next section.

5.1. Experimental Setup

5.1.1 Data and Evaluation Criteria

For our experiments, we require full body poses in order to obtain faithful results. We used a subset of the UCF50 dataset [15] for our experiments. More specifically, we used the "CleanAndJerk", "Jumping Jack", "Lunges" and "ThrowDiscus" classes of the dataset since most instances of these classes have full body poses. Each class had 25 groups of samples, with each group having a number of videos describing the activity

represented by the class.

To evaluate our experiments in a fair manner, we perform 5-fold cross validation. To prevent data leakage, care was taken to ensure that all videos belonging to one group belong in the same train or test split. Accuracy was chosen as the metric to evaluate the performance of our experiments. Here, accuracy is defined as the number of instances for which the model provided a correct prediction divided by the total number of instances.

5.1.2 Feature Extraction Module

We use a pre-trained TensorFlow implementation of the OpenPose[1] pose extraction algorithm to extract human pose skeletons from each frame of a video. The human pose skeleton that is output by this algorithm has 18 keypoints in accordance to the MS-COCO keypoint format[2]. After extracting the human pose skeletons, we convert them to the feature vector representation defined by (1). The X and Y coordinates are scaled by their image width and image height respectively so that both are in the open interval [0, 1). We call this the **Raw** feature vector representation, which is 36 dimensional.

Moreover, using the techniques described in the previous section, we extract CG, Angle and Velocity features in accordance with equations (4), (6) and (8). CG and Velocity features are 36 dimensional each, whereas Angle features are 8 dimensional. We concatenate all three feature vectors to obtain an 80-dimensional **Hybrid** feature vector representation. For certain experiments, we concatenate just the CG and Angle feature to obtain a 44-dimensional **Hybrid-Non-Temporal** feature vector representation.

Throughout the rest of the paper, we will refer to the various feature vectors described above as **Raw**, **Hybrid** and Hybrid-Non-Temporal (**HNT**) feature vectors.

5.1.3 Activity Recognition Models

We use two different models that take any one of the feature vectors described above as input and output a class prediction for an activity. Both models were coded using the Keras framework.

The first model is a simple Multi-Layer Perceptron (MLP) with three layers of 128, 256 and 128 neurons respectively. BatchNormalization, Dropout (with drop rate 0.2) and LeakyReLU (with alpha 0.3) were applied after each layer, except the last one. Softmax was applied at the last layer. This model only makes predictions on a frame-by-frame basis.

The second model is an LSTM model with two LSTM layers having 64 neurons each, followed by a Dense layer of 16 neurons. BatchNormalization, Dropout and Leaky ReLU were applied with the same parameters as the MLP model. The feature vectors described previously are accumulated over 15 time steps in our experiments before being provided to the LSTM model.

Both models were trained using an AdamOptimizer with a learning rate of 0.001 for 10 epochs. Whilst training, we used kernel and bias constraints of 3 for the LSTM and MLP layers. Throughout the rest of the paper, we will refer to the above two models as MLP and LSTM models respectively.

5.1.4 Tracker

As discussed previously, we need a tracker to associate human pose skeletons with the right person before we can accumulate feature vectors extracted from poses across timesteps. Moreover, many of the video samples in our subset of the UCF50 dataset have more than one person in the video, with only one among them performing the activity defined by the class.

For this purpose, we design our own heuristic-based pose tracking algorithm. Given a frame from a video in the current timestep (t) and the previous timestep (t-1), we first identify the human pose skeletons in both frames. Then, we compare all the human pose skeletons across both frames and try to ascertain the similarity between a pose in the previous frame and a pose in the current frame.

The similarity is ascertained as follows. Consider a small patch of dimensions (32,32) centered around any keypoint of a pose skeleton. A BRIEF [2] descriptor can be extracted from that patch. Now, this descriptor extracted from a keypoint of a pose skeleton from the previous frame ($F_i(t-1)$), is compared with the descriptor extracted from the same keypoint from a random skeleton in the current frame ($F_i(t)$). This comparison is performed by calculating the XOR between both brief descriptors. In this fashion, all keypoints of both skeletons are compared and their XOR scores are stored in a list.

$$Score = \sum_{i=1}^{N} XOR(F_i(t), F_i(t-1)) \tag{9}$$

Now, if the median of this list of XOR scores are less than a heuristically set threshold, then we can ascertain that the two human pose skeletons are the same. In that case, the track ID of the pose skeleton in the previous frame is propagated to the current frame. New track IDs are assigned to new observations of pose skeletons.

Our approach is mostly heuristical and requires tuning of the threshold value. This tracker is to be considered as a baseline approach, with ample scope for optimization of performance, which we consider beyond the scope of this paper. Regardless, we take care to handle missing keypoints and edge cases by imputing zeros in the BRIEF descriptor at such instances. We also use a buffer to hold all unique human pose skeletons seen over the last few frames. By comparing the pose skeletons in the current frame with the ones in the buffer as well, the tracker is

1 https://github.com/ildoonet/tf-pose-estimation

2 https://github.com/CMU-Perceptual-Computing-Lab/openpose

made robust to short-term occlusions.

For our experiments, we use the tracker as follows. Given a video, the OpenPose algorithm is used to extract pose skeletons from every frame and is passed on to the tracker. The tracker assigns a track ID to each pose skeleton for each frame in the video. Since videos in our dataset have only one person doing the target activity, we create annotations for each video noting the track ID of the person performing the action. ID switches are also noted whenever the tracker misidentifies our target person.

Using this method, we extract the human pose skeletons for our target from every frame in the videos. Any combination of a feature extraction method and an activity recognition model can be used on these extracted pose skeletons.

These pose skeletons are extracted for every video and are saved in separate files. Hence, we can directly use these files for our experiments instead of running the pose estimation and tracker online.

5.2. List of Experiments

In each of these experiments, we try various combinations of our feature extraction methods (Raw, Hybrid, HNT) along with activity recognition models (MLP, LSTM). We will refer to a particular combination of feature extraction method X with activity recognition model Y as combination (X+Y). For instance, an experiment may investigate performance on the combination (Raw + MLP) or (Hybrid + LSTM) and so on.

5.2.1 Experiment A - Effect of Temporal Information

In this experiment we aim to verify the importance of temporal information on the performance of activity recognition algorithms. The combination (Raw + MLP) uses no temporal information whereas the combination (Hybrid + LSTM) makes the best use of temporal information. We will explore the performance on these extremities as well as on other intermediate combinations as well.

5.2.2 Experiment B - Effect of Feature Propagation

In this experiment we aim to test the effectiveness of feature propagation for missing keypoints on the performance of activity recognition algorithms. We use the combinations (Raw + LSTM) and (Hybrid + LSTM) with and without feature propagation for this experiment. Zero values are imputed in the extracted feature vector for missing keypoints to emphasize the performance with this setting.

To accentuate the importance of feature propagation, we used keypoint dropout at test time with a 40% drop probability. The intuition is that if feature propagation is enabled, performance should not be significantly affected even if several keypoints are randomly dropped in every frame.

5.2.3 Experiment C - Effect of Translation Perturbation

In this experiment, we aim to verify if our Hybrid feature representation is robust to the effects of random translation perturbations to the human pose skeletons during test time. We use the combinations (Raw + LSTM) and (Hybrid + LSTM) for this experiment. Care is taken to ensure that the same amount of perturbation is applied across a single video sample to maintain consistency. Results are reported on this perturbed dataset.

We also repeat the above steps by augmenting the training dataset by randomly translating the human pose skeletons. Testing is repeated on the same perturbed test dataset. The intuition is that since Hybrid features are already nearly invariant to translations, the model should not be affected much by the perturbations even if it is not trained using the augmented dataset.

5.2.4 Experiment D - Effect of Keypoint Dropout

In this experiment, we aim to verify if using keypoint dropout during training makes our model robust to missing keypoint information. We use the combinations (Raw + LSTM) and (Hybrid + LSTM) with and without keypoint dropout during training. After the models are trained on these combinations, keypoint dropout is applied to the pose skeletons of the test dataset to simulate conditions where keypoints are missing during testing.

One point to note is that since our dataset is tracked prior to the experiment, this experiment does not consider the effect of missing keypoints on tracking performance. However, since this is not the scope of this experiment, we will assume that we have a tracker that is not affected by missing keypoints.

Combinations	Raw + MLP*	HNT + MLP*	Hybrid + MLP	Raw + LSTM	Hybrid + LSTM
Accuracy	79.21	83.63	90.23	92.45	96.92

Table 1. Effect of Temporal Information

6. Result

6.1. Experiment A - Effect of Temporal Information

In Table 1, we see a clear trend in the increase of accuracy with models that incorporate more temporal information. Feature propagation was disabled for certain combinations (marked with *) as that also conveys temporal information. As expected, the combination that used the most temporal information (Hybrid + LSTM) obtained the highest accuracy. The model that used the least temporal information (Raw + MLP*) obtained the least accuracy.

6.2. Experiment B - Effect of Feature Propagation

In Table 2, we see that accuracy is significantly improved if feature propagation is enabled (with a keypoint dropout of 40% at test-time). This verifies our hypothesis that feature propagation is better than imputing zeros for handling missing keypoints.

Combinations	Without Feature Propagation		With Feature Propagation	
	Raw + LSTM	Hybrid + LSTM	Raw + LSTM	Hybrid + LSTM
Accuracy	71.96	68.19	89.18	88.81

Table 2. Effect of feature propagation

6.3. Experiment C - Effect of Translation Perturbation

The pose skeletons were perturbed by a random amount of up to 40% of the image dimensions for all cases during test-time. In Table 3, we see that the Hybrid + LSTM combination has good accuracy even if it was not trained using the translation augmentation experiments. This can be attributed to its invariance to translation. The Raw + LSTM combination showed improved performance after training with translation augmented examples.

Combinations	Without Train Time Translation		With Train Time Translation	
	Raw + LSTM	Hybrid + LSTM	Raw + LSTM	Hybrid + LSTM
Accuracy	65.97	96.42	84.41	96.84

Table 3. Effect of translation perturbation

6.4. Experiment D - Effect of Keypoint Dropout

We note that while using keypoint dropout during training, our Hybrid + LSTM combination is more robust to missing keypoints during test time. However, there is not much of an effect for the Raw + LSTM combination.

Combinations	Without Keypoint Dropout		With Keypoint Dropout	
	Raw + LSTM	Hybrid + LSTM	Raw + LSTM	Hybrid + LSTM
Accuracy	89.18	88.81	89.05	94.31

Table 4. Effect of keypoint dropout

7. Conclusion

Thus, the pitfalls of using human pose skeletons directly for activity recognition were studied, and methods to overcome the same were introduced. On analyzing the results of our experiment outcomes, we conclude that our proposed methods show considerable promise in overcoming the pitfalls identified. Further avenues could be explored in future research works, such as the usage of learning-based feature extractors instead of using hand-crafted features. We hope that these experiments serve as a generic guideline for designing more complex activity recognition systems based on human pose information.

References

[1] Arbués-Sangüesa, Adrià, Coloma Ballester, and Gloria Haro. "Single-Camera Basketball Tracker through Pose and Semantic Feature Fusion." arXiv preprint arXiv:1906.02042 (2019).

[2] Calonder, Michael, Vincent Lepetit, Christoph Strecha, and Pascal Fua. "Brief: Binary robust independent elementary features." In European conference on computer vision, pp. 778-792. Springer, Berlin, Heidelberg, 2010.

[3] Cao, Zhe, Tomas Simon, Shih-En Wei, and Yaser 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.

[4] Chen, Yilun, Zhicheng Wang, Yuxiang Peng, Zhiqiang Zhang, Gang Yu, and Jian Sun. "Cascaded pyramid network for multi-person pose estimation." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 7103-7112. 2018.

[5] Donahue, Jeffrey, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. "Long-term recurrent convolutional networks for visual recognition and description." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2625-2634. 2015.

[6] Fang, Hao-Shu, Shuqin Xie, Yu-Wing Tai, and Cewu Lu. "Rmpe: Regional multi-person pose estimation." In Proceedings of the IEEE International Conference on Computer Vision, pp. 2334-2343. 2017.
[7] Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." Neural computation 9, no. 8 (1997): 1735-1780.
[8] Hwang, Jihye, Jieun Lee, Sungheon Park, and Nojun Kwak. "Pose estimator and tracker using temporal flow maps for limbs." In 2019 International Joint Conference on Neural Networks (IJCNN), pp. 1-8. IEEE, 2019.
[9] Karpathy, Andrej, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. "Large-scale video classification with convolutional neural networks." In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, pp. 1725-1732. 2014.
[10] Luvizon, Diogo C., David Picard, and Hedi Tabia. "2d/3d pose estimation and action recognition using multitask deep learning." In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5137-5146. 2018.
[11] Ning, Guanghan, and Heng Huang. "Lighttrack: A generic framework for online top-down human pose tracking." arXiv preprint arXiv:1905.02822 (2019).
[12] Papandreou, George, Tyler Zhu, Nori Kanazawa, Alexander Toshev, Jonathan Tompson, Chris Bregler, and Kevin Murphy. "Towards accurate multi-person pose estimation in the wild." In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4903-4911. 2017.
[13] Pishchulin, Leonid, Eldar Insafutdinov, Siyu Tang, Bjoern Andres, Mykhaylo Andriluka, Peter V. Gehler, and Bernt Schiele. "Deepcut: Joint subset partition and labeling for multi person pose estimation." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4929-4937. 2016.
[14] Pismenskova, Marina, Oxana Balabaeva, Viacheslav Voronin, and Valentin Fedosov. "Classification of a two-dimensional pose using a human skeleton." In MATEC Web of Conferences, vol. 132, p. 05016. EDP Sciences, 2017.
[15] Reddy, Kishore K., and Mubarak Shah. "Recognizing 50 human action categories of web videos." Machine vision and applications 24, no. 5 (2013): 971-981.
[16] Singh, Amarjot, Devendra Patil, and S. N. Omkar. "Eye in the sky: Real-time Drone Surveillance System (DSS) for violent individuals identification using ScatterNet Hybrid Deep Learning network." In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 1629-1637. 2018.
[17] Wang, Limin, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. "Temporal segment networks for action recognition in videos." IEEE transactions on pattern analysis and machine intelligence 41, no. 11 (2018): 2740-2755.
[18] Xiao, Bin, Haiping Wu, and Yichen Wei. "Simple baselines for human pose estimation and tracking." In Proceedings of the European conference on computer vision (ECCV), pp. 466-481. 2018.