EECS 6414 Final Report: Human Motion State Classification from Motion Capture Data

Peter Caruana*
Melissa Kremer*
caruana9@my.yorku.ca
mkremer@my.yorku.ca
York University
Toronto, Ontario, Canada

ABSTRACT

Motion capture data presents a unique challenge to data mining and database search. Several MoCap data formats exist, typically specifying the pose of a human armature over a series of frames. Pose estimation from video and images has been the focus of much research in recent years, as well as methods for querying MoCap databases. We propose a method of classifying MoCap data by activity type using multinomial classifiers. We trained three different models to classify individual frames of MoCap clips, then used winner-take-all voting to classify entire motion clips. Evaluation showed that Random Forests performed best for this task, achieving 100% accuracy on the CMU MoCap dataset. However this work highlights the need for larger and better annotated datasets.

1 INTRODUCTION

Classifying the motion state of humans has been an ongoing area of research spanning at least the last two decades. Most recent research has been focused on estimating motion from a computer vision perspective, using video and/or image data. However, these are not the only types of data which can record human movement. Motion capture (MoCap) data records the 3D position of a human armature as a function time which can be reconstructed into the full motion to be used in applications like computer animation or robotics. The work done in this paper looks at classifying motion capture data into various human activities using machine learning.

One of the challenges with current MoCap or animation databases is accurate annotation for searches. Consider a video game developer searching for a particular animation to use for their human character. They would likely first query some common animation database such as the popular web service *Mixamo*, or game engine asset stores like those of Unity and Unreal. If those yielded no results they may try to look into raw motion capture datasets. However, for anyone who has spent time trying to use these resources it is apparent that it can be very difficult to find specific animations due to the generally coarse nature of how data is annotated. As we begin to see more automated motion capture methods and technologies arise, the need for automated categorization and annotation will be critical for large animation and motion datasets to scale with inflow of data.

*Both authors contributed equally to this research.

EECS 6414 Final Report, April 2021,

2 PROBLEM DEFINITION

Given motion capture data labeled in correspondence to the action being performed, the goal is to train a classification model which can take some or all of a clip of a motion and then assign the correct label.

Bayesian networks (BNs) encapsulate high level representations of probability distributions over a set of variables $\mathbf{X} = \{X_1, X_2, \ldots, X_n\}$. Learning these networks requires both structure learning and parameter learning. To learn the structure, a directed acyclic graph must be constructed from the set X, where each node corresponds to a variable and a directed arc indicates a causal relationship between a parent node and the child node. Once there is a structure, parameter learning can begin which entails finding probability distributions, class probabilities, and conditional probabilities associated with each variable. If $Pa\,(X_i)$ denotes the set of parents of the node X_i, the joint probability distribution is given by

$$P(\mathbf{X}) = \prod_{i=1}^{n} P\,(X_i \mid Pa\,(X_i)) \tag{1}$$

In order to find $P\,(X_i \mid Pa\,(X_i))$, finding the structure is required. Learning an optimal structure of a BN has been proven NP-hard. Instead, the naive Bayes classifier can be used, in which the structure is known. Features in a naive Bayes classifier are conditionally independent given the class, so the only parent of each feature is the class. If C represents the class of example X, then the probability of C given X needs to be found.

$$P(C \mid \mathbf{X}) = \frac{P(C)P(\mathbf{X} \mid C)}{P(\mathbf{X})} \tag{2}$$

Since the features are conditionally independent of each other given the class, this becomes:

$$P(C \mid \mathbf{X}) = \frac{P(C)\prod_{i=1}^{n} P\,(X_i \mid C)}{P(\mathbf{X})} \tag{3}$$

The naive Bayes classifier works by finding these probabilities within the training data, and then using them to predict a class on unseen data [19].

Another classifier used in this project is the Random Forest classifier. If we have a training set T with a features or attributes and n examples, then we can define T_k as a bootstrap training set constructed by sampling T with replacement with m random attributes and n examples. A Random Tree can then be chosen at random from a set of possible trees, with m random attributes at each node.

A Random Forest is then defined formally as a classifier consisting of a collection of tree-structured classifiers $\{h_k(\mathbf{x}, T_k)\}, k = 1, 2, \ldots, L$ where T_k are independently identically distributed random samples, and each tree casts a single vote for the most popular class at input x [2, 12]. The random samples of the training set are produced in a similar way to bagging. The node is split by finding the best split on the selected attributes. For each example in the training set, votes are combined only from the tree classifiers that do not contain that example (out-of-bag), such that the out-of-bag estimate for the generalization error is the error rate for the out-of-bag classifier.

The strength of each tree in the forest and the correlation between any two trees influences the error rate of the forest, where the strength can be thought of as a performance measure for each tree. Increasing the correlation between trees increases the error rate, whereas increasing the strength of trees decreases the error rate. If the number of random attributes selected is reduced, however, both the strength and the correlation is reduced [12].

Support Vector Machine (SVM) uses a different approach. Given the training set $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{l}$, where x_i is a real number and $y_i \in \{-1, 1\}$ and $\boldsymbol{\theta} = \left(\boldsymbol{w}^T, b\right)^T$, SVMs attempt to find a decision hyperplane $\boldsymbol{w}^T\Phi(\boldsymbol{x}) + b = 0$. Instead of minimizing empirical training error, SVM tries to minimize an upper bound of the generalization error by maximizing the area between the separating hyperplane and the data, such that the separation between classes of data is also maximized. Formally, the following optimization problem is solved:

$$\min_{\theta} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{l} L\left(1 - y_i\left(\boldsymbol{w}^T\Phi\left(\boldsymbol{x}_i\right) + b\right)\right) \tag{4}$$

where $L(u)$ is some loss function, $C > 0$ is the regularization parameter, and $\Phi(\boldsymbol{x}_i)$ is used to map $\boldsymbol{x}_i$ to a higher dimensional space. We can then show by differentiating that a minimum w satisfies the following:

$$\boldsymbol{w} = C\sum_{i=1}^{l} y_i \partial L\left(1 - y_i f_\theta\left(\boldsymbol{x}_i\right)\right)\Phi\left(\boldsymbol{x}_i\right) \tag{5}$$

where $f_\theta(\boldsymbol{x}) = \boldsymbol{w}^T\Phi(\boldsymbol{x}) + b$ is the decision function. An input (x_i, y_i) that satisfies $\partial L\left(1 - y_i f_\theta\left(\boldsymbol{x}_i\right)\right) \neq 0$ for a given $f_\theta(\boldsymbol{x})$ is a support vector [17]. In this way, SVMs can condense the information in the training data and construct a sparse representation using only a small number of points which are the support vectors.

3 RELATED WORK

3.1 Human Pose Estimation

There has recently been a flourish in techniques and models for estimating the pose of humans from image or video data, which can primarily be attributed to advancements in machine learning [1, 3, 15, 16, 20]. As these technologies progress, pose data will become more and more available. Traditionally, human pose data is acquired through the use of motion capture equipment. Recording human motion in this fashion is quite restrictive in scope and difficult to scale due to the need for expensive specialized equipment as well as requiring fixed studio locations [21]. LCR-NET created by Rogez et al. [16] is able to achieve impressive results in 2D and 3D pose estimation from 2D images by combining a pose classifier with a regressor. LCR-NET can generate pose estimations for multiple humans in a natural scene simultaneously, even given partial occlusion of limbs. Qammaz et al.'s [14] mocapNET was trained using motion capture data from the CMU dataset, taking a two-staged approach which is quite computationally fast. Performance with this architecture is modest compared to LCR-NET and other state of the art models, however, it is the only work we are aware of which outputs directly to BVH file format; making it directly applicable for use in most commercial animation and video game software. Shotton et al. [18] developed a pose recognition framework based on randomized decision forests to be used in conjunction with the *Xbox Kinect*. Their method takes a depth-image as input and estimates joint positions. Similar to the work presented in this paper they utilized a motion capture dataset to train, only considering static frames.

3.2 Human Pose Classification

The natural extension to the ability to estimate poses is to then recognize and classify them, which is where our work falls. O. Patsadu et al. [13] were among some of the earliest to implement high accuracy pose recognition using an SVM. Using 2D pose estimation from a Kinect camera they were able to classify static poses (sitting, laying down, standing) with near perfect (99%+) accuracy. Vox and Wallhoff [22] attempted to expand on this by training a multi-class SVM to classify motion sequences of 2D poses gathered from a Kinect. In most cases results were acceptable however, motions which primarily involved legs tended to have quite poor results. This is likely because SVMs have difficulty dealing with streaming data. Du et al. [6] argued that the problem can be treated as a time series problem, and demonstrated how a recurrent neural network (RNN) can be used due to its ability to model long term contextual information. They were able to achieve 95% accuracy on average for the HDM05 dataset, which is a motion capture dataset in AMC file format similar to the CMU dataset. Using entire animation clips for categorization however limits the ability to do on-line classification. Classifying motion based on one or a few frames decreases the data and processing overhead significantly. In all cases it is encouraging how well models are able to perform given the relatively small size of most motion capture datasets.

4 MOTIVATION AND DOMAIN DESCRIPTION

Recent years have seen substantial developments in human pose estimation and motion classification. However, most of the work has been in the domain of computer vision, in trying to estimate human pose from video or images [16]. Issac et al. [7] were able to classify gender from videos by analyzing poses across a period of time. Many techniques have been developed for these tasks utilizing machine learning however to the best of our knowledge there appears to be an underrepresentation when it comes to analyzing motion capture data specifically. Developments in 3D pose estimation such as the work done by Qammaz et al. [14] allow for much larger amounts of motion capture data to be generated than is possible using traditional optical tracking systems. A natural addition to

this pipeline is developing robust systems to catalogue and classify these emerging sources of motion capture data.

Motion capture data is a unique data format that presents challenges to database and querying tasks. Kim et al. [10] found that using annotated text tags is by far the most popular method of querying motion capture databases. Kapadia et al. [8] comment that this requires manually annotating the data, which can be costly. Alternatively, some databases which allow public uploads rely on the uploader to submit user-defined tags and metadescriptions. These can often be lost over time or simply inaccurate. Further, some nuances of motion are difficult to capture with text tags. As example imagine a tag "human raises arm". One may then wonder: Which arm? How high did they raise the arm? Was the person sitting or standing? These are the types of specific information often overlooked in the cataloging of animation data.

Another method for querying motion capture databases is through example [4]. An actor can perform the action if they have the necessary equipment and use that to search a database through comparison. This requires a motion capture setup and space as well as requiring that the actor is capable of performing the motion they want to query which is impractical for the vast majority of people. The data may also need to be preprocessed with sophisticated techniques like dynamic time warping to account for temporal differences between the actions in the database and the query action.

There are several other approaches [5, 9, 11], including using a puppet as an actor for example querying, or using sketches which requires the motion to be silhouetted before queries can be made. We believe that motion capture databases could be more efficiently searched if motion capture files could be classified into activities by a machine learning classifier, removing the need for manual annotation or user-defined tags and meta-descriptions. Motion capture data could then be more efficiently organized and categorized as well.

5 METHODOLOGY

Three machine learning models were trained to classify human activities given individual poses. Human poses are highly variable across time, as well as across different people. This makes the problem well suited to machine learning, where an analytic method of classification would be prohibitively difficult.

Similar to video formats, motion capture files are comprised of a series of frames. Each frame can be read as a collection of points representing the joints of the actor in 3D space. Analyzing temporal data can be a difficult task due to the dependence each frame has on each other. To simplify the problem space, individual frames from MoCap clips of activities will be treated as labeled instances. This transforms the problem space to be time-independent, and also expands the number of training samples available. This is important because the number of examples in motion capture data-sets is normally quite small, which can make it difficult to adequately train a classifier.

The MoCap files are in AMC format, used for MoCap data that contains a list of all the joints and their 3D positions and rotations for each frame. An example of the structure of an armature used by an AMC file is shown in Fig. 1. An example of a frame in an AMC file is shown in Fig. 2. It should be noted that some joints have more

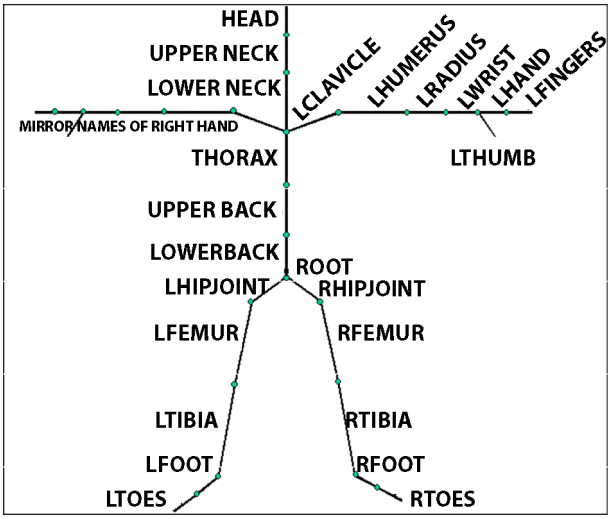

Figure 1: The structure of joints used in the AMC format visualized on a humanoid skeleton.

```
frame 1
root              x y z, rx, ry, rz
lower back        x y z
left arm          x y z
left thigh        x y z
left foot         x y z
head              x y z
          .
          .
          .
```

Figure 2: Example of an AMC file, where each frame specifies (x,y,z) positions of all joints in the armature.

degrees of freedom and can rotate in more directions whereas other joints can only rotate along one or two axes. The root joint contains both a 3D position and a 3D rotation vector. AMC files will be used due to the ease of obtaining them from the CMU MoCap database. Data was arranged such that features are the 3D joint rotations and labels are the activity being performed. Each frame is considered as a tensor of length 29 where each element is a vector with length between 1 and 6 numerical values, specifying the arguments for each joints degrees of freedom.

6 EVALUATION

Data samples consisted of individual frames from motion capture data, with class labels coming from the clip they were drawn from. There were 571,436 examples of MoCap frames across four classes of activities: running, walking, sitting, and jumping; taken from

the Carnegie Melon University dataset (CMU).The dataset is unfortunately severely unbalanced, with walking constituting 87% of samples. The distribution of the classes in CMU is shown in Fig. 3. To mitigate this, class balancing was done by re-weighing the dataset so that each class has an equal weight. Once trained, frames of a single clip can then be considered for pose-based voting (PBV) to classify entire motion capture clips.

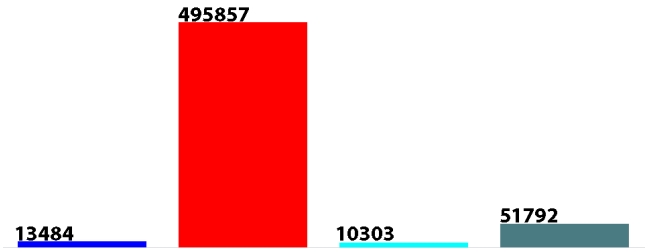

Figure 3: The distribution of the classes for the examples in our dataset. From right to left we have: Running, Walking, Jumping, and Sitting.

Three models of classifier were trained: Naive Bayes (NB), Random Forests (RF), and Sequential Minimal Optimization (SMO), using 10-fold cross validation for training. We then compared performance across each model with regards to the number of correctly and incorrectly identified instances, root mean squared error, shown in detail in Tab. 2. We evaluated the accuracy accross individual frames as well as with voting in order to classify entire animation clips with results shown in Tab. **??**. All models were able to perform competitively, with Random forests being the best, with 100% accuracy, followed by SMO with 92.79% and NB with 83.74%. The results with the inclusion of voting tell us something about how incorrectly classified frames are grouped together. In the case of NB the performance drops rather significantly to 73.51% when trying to classify entire clips through voting. We can gather that the frames this model misclassifies must be mostly localized and concentrated in specific animation clips. On the contrary, SMO gains some performance increases with the inclusion of voting indicating a more even distribution of incorrectly classified frames. This phenomenon is more clearly visualized in Fig. 5, which shows the a vectorized visualization of classifications. The incorrect classifications are spread more evenly across clips for SMO, meaning that more are able to go above the majority threshold. We can also clearly see that in the case of NB, there is much more variation in the labels of misclassifications, versus SMO which has more structured error.

Model	w/o PBV	PBV
NB	0.8374	0.7351
RF	1.000	1.000
SMO	0.9279	0.9326

Table 1: Average performance for Naive Bayes (NB), Random Forests (RF) and Sequential Minimal Optimization (SMO) without and with pose based voting (PBV)

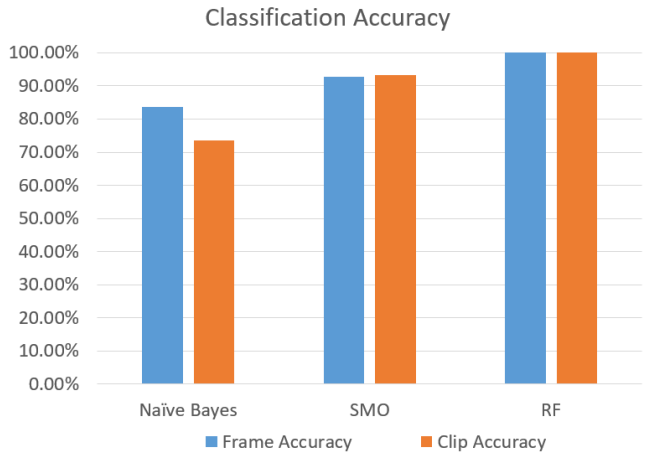

Figure 4: Comparison of classification accuracies for frame and clip classification across algorithms.

This approach scales well to any motion capture database with simple file types such as AMC files. For both SMO and RF, performances with and without voting are only marginally different. In order to classify entire clips it should suffice then to only draw a random subset of frames from each clip. This can even be used to improve existing datasets by classifying across sub-clips to differentiate internal changes in motion state. Any animation format which saves individual poses per frame can easily be used for this type of classification with minor conversions.

All models trained quickly. A model for the Naive Bayes algorithm took just under 13 seconds to build. For Random Forest, a model took 835 seconds or just under 14 minutes to build. SMO took the longest, with a model taking 3782 seconds to build, or just over an hour. Each model was trained using 10-fold cross-validation which increases the training time by 10 times.

7 CONCLUSION

This paper evaluated the use of current motion capture datasets for training classifier models by treating individual animation keyframes as datapoints. Three different types of classifiers were trained on the CMU dataset, all achieving fairly competent results with Random Forest and Sequential Minimal Optimization (SMO) models attaining 100% and 92.8% accuracy respectively for individual frame classification. Using these trained models to classify entire animation clips using winner-take-all voting yielded similarily 100% and 93.3% accuracies, indicating that a sample based approach may be sufficient for classifying larger clips of animation data. However due to severe imbalances in the dataset used, these results need to be replicated using other more balanced datasets. In general, animation datasets are currently sparse and poorly labeled, thus we believe our models would be able to greatly improve the accuracy of existing datasets, as well as be a good tool for efficient querying.

7.1 Data Availability Issues

One pitfall of this evaluation is that the small size and lack of diversity of the CMU dataset makes validation on out-sample results

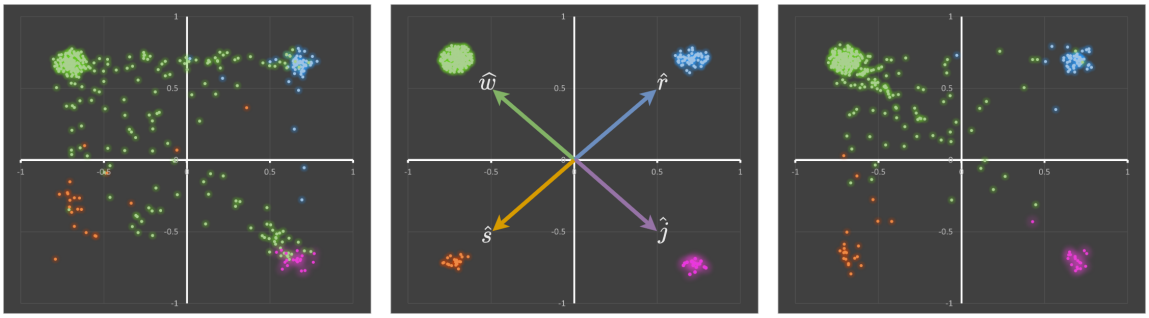

Figure 5: Classification plots for NB (Left), RF (Center), and SMO (Right). Each point represents an animation clip, colours correspond to the clip's true label: Walk (Green), Run (Blue), Jump (Purple), Sit (Orange). Point position is the weighted sum of classifications across every frame of the clip according to: $(x, y) = \frac{1}{N}(W\hat{w} + R\hat{r} + J\hat{j} + S\hat{s})$.

quite difficult. The HDM05 dataset may be a good candidate for testing since it contains human motion capture and is well labeled. While not as substantial as CMU, its size may be at least usable for validation. Another benefit is that examples are in AMC format allowing for direct comparison. The primary roadblock however is that most of the 70 motion classes in HDM05 are very specific. Only walking and locomotion activities could be validated. Being able to support multiple animation formats while using the same model would greatly open up the availability of other animation datasets. Since there are semantic differences in the way animation data is saved, ablative studies would have to be performed to determine the minimal joint information needed without harming performance.

7.2 Feature Importance Studies

Future work includes feature selection analysis to reduce the feature space. It is expected that certain joint positions are not important for determining the pose, such as the fingers and toes. However, it would depend on what types of poses are being classified. For example to classify poses involving hands the fingers would be more important. If we consider the poses classified in this paper, we expect that joints such as the pelvis, knees, and elbows would probably be most important. Further testing would be needed to validate this hypothesis.

Other algorithms could also be tested, as well as other poses. Data availability is a problem, but if there was enough data available, many kinds of poses could be classified. In this case, Random Forest may or may not perform as well. Other algorithms that have not been tested yet could be compared and may outperform Random Forest in a situation where there is more data and more poses. Further investigation could reveal which joints are important for which kinds of poses.

To address the data availability problem, data collection studies could be run. In this day and age, motion capture can be performed with affordable cameras and a very simple setup. By running data collection studies, many different poses could be captured.

Class	# Predicted	# Actual	Accuracy	Precision	Recall	F1 Score	ROC Area
NB							
Run	48812	13484	93.42%	0.92	0.25	0.40	0.960
Walk	431584	495857	84.45%	0.85	0.97	0.90	0.938
Jump	29478	10303	96.53%	0.97	0.34	0.50	0.987
Sit	61562	51792	93.07%	0.71	0.6	0.65	0.951
Weighted Avg			83.74%	0.85	0.84	0.81	0.940
NA							
SMO							
Run	29754	13484	97.00%	0.97	0.44	0.60	0.988
Walk	462493	495857	93.04%	0.93	0.99	0.96	0.961
Jump	20303	10303	98.19%	0.98	0.5	0.66	0.999
Sit	58886	51792	97.35%	0.92	0.81	0.86	0.979
Weighted Avg			92.79%	0.93	0.93	0.92	0.970
RF							
Run	13484	13484	100.0%	1.00	1.00	1.00	1.000
Walk	495857	495857	100.0%	1.00	1.00	1.00	1.000
Jump	10303	10303	100.0%	1.00	1.00	1.00	1.000
Sit	51792	51792	100.0%	1.00	1.00	1.00	1.000
Weighted Avg			100.0%	1.00	1.00	1.00	1.000
NB w/ PBV							
Run	60	93	89.95%	0.61	0.95	0.75	NA
Walk	288	198	75.77%	0.99	0.68	0.81	NA
Jump	19	61	89.18%	0.31	1.00	0.47	NA
Sit	21	36	94.59%	0.50	0.86	0.63	NA
Weighted Avg			73.51%	0.87	0.75	0.77	NA
SMO w/ PBV							
Run	73	60	96.11%	0.98	0.81	0.89	NA
Walk	288	266	93.26%	0.92	0.99	0.95	NA
Jump	24	19	98.70%	1.00	0.79	0.88	NA
Sit	23	19	98.45%	0.95	0.78	0.87	NA
Weighted Avg			93.30%	0.94	0.93	0.93	NA
RF w/ PBV							
Run	60	60	100.0%	1.00	1.00	1.00	NA
Walk	288	288	100.0%	1.00	1.00	1.00	NA
Jump	19	19	100.0%	1.00	1.00	1.00	NA
Sit	21	21	100.0%	1.00	1.00	1.00	NA
Weighted Avg.			100.0%	1.00	1.00	1.00	NA

Table 2: Comparison of calculated metrics for individual frames and with voting

REFERENCES

[1] Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Bernt Schiele. 2014. 2d human pose estimation: New benchmark and state of the art analysis. In *Proceedings of the IEEE Conference on computer Vision and Pattern Recognition*. 3686–3693.

[2] Leo Breiman. 2001. Random forests. *Machine learning* 45, 1 (2001), 5–32.

[3] Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh. 2019. OpenPose: realtime multi-person 2D pose estimation using Part Affinity Fields. *IEEE transactions on pattern analysis and machine intelligence* 43, 1 (2019), 172–186.

[4] Yueguo Chen, Gang Chen, Ke Chen, and Beng Chin Ooi. 2009. Efficient processing of warping time series join of motion capture data. In *2009 IEEE 25th International Conference on Data Engineering*. IEEE, 1048–1059.

[5] Zhigang Deng, Qin Gu, and Qing Li. 2009. Perceptually consistent example-based human motion retrieval. In *Proceedings of the 2009 symposium on Interactive 3D graphics and games*. 191–198.

[6] Yong Du, Wei Wang, and Liang Wang. 2015. Hierarchical recurrent neural network for skeleton based action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1110–1118.

[7] Ebenezer RHP Isaac, Susan Elias, Srinivasan Rajagopalan, and KS Easwarakumar. 2019. Multiview gait-based gender classification through pose-based voting. *Pattern Recognition Letters* 126 (2019), 41–50.

[8] Mubbasir Kapadia, I-kao Chiang, Tiju Thomas, Norman I Badler, and Joseph T Kider Jr. 2013. Efficient motion retrieval in large motion databases. In *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*. 19–28.

[9] Daisuke Kasai, Toshihiko Yamasaki, and Kiyoharu Aizawa. 2009. Retrieval of time-varying mesh and motion capture data using 2D video queries based on silhouette shape descriptors. In *2009 IEEE International Conference on Multimedia and Expo*. IEEE, 854–857.

[10] Dohyung Kim, Minsu Jang, and Jaehong Kim. 2016. Example-based retrieval system for human motion data. In *2016 6th International Conference on IT Convergence and Security (ICITCS)*. IEEE, 1–2.

[11] Naoki Numaguchi, Atsushi Nakazawa, Takaaki Shiratori, and Jessica K Hodgins. 2011. A puppet interface for retrieval of motion capture data. In *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 157–166.

[12] Thais Mayumi Oshiro, Pedro Santoro Perez, and José Augusto Baranauskas. 2012. How many trees in a random forest?. In *International workshop on machine learning and data mining in pattern recognition*. Springer, 154–168.

[13] Orasa Patsadu, Chakarida Nukoolkit, and Bunthit Watanapa. 2012. Human gesture recognition using Kinect camera. In *2012 ninth international conference on computer science and software engineering (JCSSE)*. IEEE, 28–32.

[14] Ammar Qammaz and Antonis A Argyros. 2019. MocapNET: Ensemble of SNN Encoders for 3D Human Pose Estimation in RGB Images.. In *BMVC*. 46.

[15] Grégory Rogez and Cordelia Schmid. 2016. Mocap-guided data augmentation for 3d pose estimation in the wild. *arXiv preprint arXiv:1607.02046* (2016).

[16] Gregory Rogez, Philippe Weinzaepfel, and Cordelia Schmid. 2017. Lcr-net: Localization-classification-regression for human pose. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 3433–3441.

[17] Xin Shen, Lingfeng Niu, Zhiquan Qi, and Yingjie Tian. 2017. Support vector machine classifier with truncated pinball loss. *Pattern Recognition* 68 (2017), 199–210.

[18] Jamie Shotton, Andrew Fitzgibbon, Mat Cook, Toby Sharp, Mark Finocchio, Richard Moore, Alex Kipman, and Andrew Blake. 2011. Real-time human pose recognition in parts from single depth images. In *CVPR 2011*. Ieee, 1297–1304.

[19] Sona Taheri and Musa Mammadov. 2013. Learning the naive Bayes classifier with optimization models. *International Journal of Applied Mathematics and Computer Science* 23, 4 (2013), 787–795.

[20] Alexander Toshev and Christian Szegedy. 2014. DeepPose: Human Pose Estimation via Deep Neural Networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[21] Daniel Vlasic, Rolf Adelsberger, Giovanni Vannucci, John Barnwell, Markus Gross, Wojciech Matusik, and Jovan Popović. 2007. Practical motion capture in everyday surroundings. *ACM transactions on graphics (TOG)* 26, 3 (2007), 35–es.

[22] Jan P Vox and Frank Wallhoff. 2017. Recognition of human motion exercises using skeleton data and SVM for rehabilitative purposes. In *2017 IEEE Life Sciences Conference (LSC)*. IEEE, 266–269.