# The Influence of Labeling Techniques in Classifying Human Manipulation Movement of Different Speed

Sadique Adnan Siddiqui[1], Lisa Gutzeit[1] and Frank Kirchner[1,2]
[1]*Robotics Research Group, University of Bremen, Bremen, Germany*
[2]*Robotics Inovation Center, DFKI, Bremen, Germany*

Keywords: Movement Recognition, Human Movement Analysis, k-Nearest Neighbor, Convolutional Neural Networks, Extreme Gradient Boosting, Random Forest, Long Short-term Memory Networks, CNN-LSTM Network.

Abstract: Human action recognition aims to understand and identify different human behaviors and designate appropriate labels for each movement's action. In this work, we investigate the influence of labeling methods on the classification of human movements on data recorded using a marker-based motion capture system. The dataset is labeled using two different approaches, one based on video data of the movements, the other based on the movement trajectories recorded using the motion capture system. The data was recorded from one participant performing a stacking scenario comprising simple arm movements at three different speeds (slow, normal, fast). Machine learning algorithms that include k-Nearest Neighbor, Random Forest, Extreme Gradient Boosting classifier, Convolutional Neural networks (CNN), Long Short-Term Memory networks (LSTM), and a combination of CNN-LSTM networks are compared on their performance in recognition of these arm movements. The models were trained on actions performed on slow and normal speed movements segments and generalized on actions consisting of fast-paced human movement. It was observed that all the models trained on normal-paced data labeled using trajectories have almost 20% improvement in accuracy on test data in comparison to the models trained on data labeled using videos of the performed experiments.



## 1 INTRODUCTION

Recognition of human actions is an active research area utilizing both vision and non-vision based modalities. Machine Learning and Deep Learning algorithms have shown promising results in the identification and understanding of human behaviors, which is important to improve the collaboration between humans and robots in several applications. The only major concern with these supervised learning methods is that the effectiveness of these methods desires an ample amount of detailed labeled training data. Despite the need for a large amount of data for training and human supervision for labeling these data, making use of a robust alternative for supervised learning algorithms is a difficult task to accomplish in human action recognition.

The tasks concerning the action classification generally have four major phases: data acquisition, segment labeling, feature engineering, and finally training the classifier. The data can be acquired using different sensing modalities (for example video streams, IMUs, point clouds, etc.). If a sequence of several movements is recorded, the data needs to be preprocessed and segmented into smaller movement entities and action labels need to be assigned to each segment. Then, features have to be extracted from raw motion data and normalized. Lastly, a classifier to recognize and infer the actions needs to be trained. The pipeline for the creation of the dataset and labeling strategies used in this work is depicted in Figure 1. The data labeling phase is a tedious and time-consuming process and poses a major constraint in the creation of a robust action recognition dataset. The data recorded using RGB cameras and RGB-D cameras are easy to obtain, and they provide rich appearance information. Thus, it makes the labeling task less complicated. On the other hand, sensor data recorded either using IMUs or marker-based motion capture system requires careful analysis of time series data to extract the stream of motions and assign a set of actions to it. There are previous works that propose to facilitate the data annotation process for time series data, e.g. Schröder et al. developed a tool support that makes use of a database schema for annotating sensor data (Schröder et al., 2016). Cruciani et al. proposed a heuristic func-



Siddiqui, S., Gutzeit, L. and Kirchner, F.
The Influence of Labeling Techniques in Classifying Human Manipulation Movement of Different Speed.
DOI: 10.5220/0010906900003122
In *Proceedings of the 11th International Conference on Pattern Recognition Applications and Methods (ICPRAM 2022)*, pages 338-345
ISBN: 978-989-758-549-4; ISSN: 2184-4313

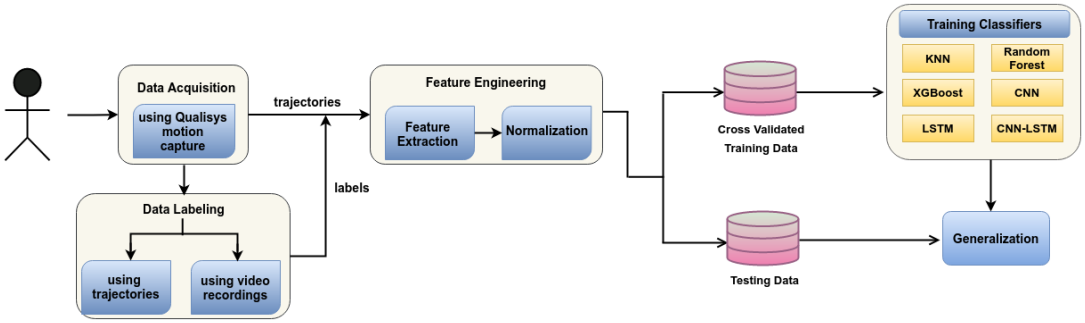


Figure 1: Pipeline for creation of a human action recognition dataset and classification approach. Inspired from (Zhang et al., 2019).

tion based on step count and GPS information for online supervised training (Cruciani et al., 2018). Another technique is to utilize few-shot learning methods as mentioned in (Gutzeit, 2021), where small entities of human manipulation movements can be detected at high accuracy with $\leq 10$ examples per class in the training data. Using the models trained on such a small dataset, it can be generalized to new unlabeled data.

In this paper, the influence of different labeling methods on the classification of human movements is investigated. Human movement is recorded based on a simple stacking scenario using a marker-based motion tracking system that measures the 3D positions of the human arm. Additionally, videos of the movements are recorded. After that, the recordings are labeled using two different methods. In the first labeling approach, the stacking movements are manually segmented using the video data. In the second method, recorded movement trajectories are automatically segmented into manipulation building blocks characterized by a bell-shaped velocity profile of the hand (Gutzeit et al., 2014) followed by manual correction of wrongly segmented data. The stacking movements are labeled carefully by examining the trajectory of the arm while performing the experiments using a labeling tool developed in-house, which visualizes the movement trajectories in 2D and 3D. Six different algorithms that are widely used for human action recognition, the k-Nearest Neighbor (KNN) classifier, Decision-Tree based classifier Random Forest (RF), Extreme Gradient Boosting (XGBoost), Deep Learning algorithms such as Convolutional Neural Networks (CNN), Long Short-Term Memory networks (LSTM), and a combination of CNN-LSTM networks are compared with respect to their suitability to label the movements automatically. The models are trained and evaluated on movements recorded at different speeds in order to study the influence of labeling techniques on the feasibility of transfer of speed in simple action movements. This paper is organized as follows: In section 2, an overview of related work is given. In section 3, the feature extraction and algorithms used for classification are described, along with the evaluation approaches. The data recording and the labeling procedure along with results and discussions are presented in section 4 and section 5 respectively. The paper concludes with future scope of this work in section 6.

## 2 RELATED WORK

There is already a lot of work done on action recognition, involving vision-based and sensor-based modalities. The video streams provide rich spatial information and when it combines with the temporal information, i.e., image frames at time steps, it can be beneficial for the identification of human actions. Starting from the analysis of video streams, earlier works involved the usage of handcrafted feature-based methods such as the position of skeleton joints for action recognition tasks (Wang and Schmid, 2013). In recent times, CNN-based approaches are quite popular because of their benchmarked results in computer vision problems and their ability to extract high-level representation in deep layers. Donahue et al. introduced the Long-term Recurrent Convolutional Network (LRCN) consisting of a 2D CNN and LSTM for extracting RGB features and predicting action labels from each image (Donahue et al., 2015). Ji et al. tried to capture the motion information from several adjacent frames by extracting spatial and temporal dimension features by performing 3D convolutions on the videos (Ji et al., 2013). The vision-based modalities require the use of an appropriately placed camera that poses mobility issues and privacy risks. With the availability of low-cost sensors and activity trackers in smartphones, a sudden shift has been observed in the usage of cameras for action recognition tasks. Halloran et al. presented a comparison of Deep Learning models in human activity recognition

on the MHEALTH dataset recorded using a smartphone (O'Halloran and Curry, 2019). They compared machine learning models on sensor-based data utilizing supervised learning methods. Some recent works provide an alternative of using a labeled dataset and propose to train the model using semi-supervised methods, such as label propagation, requiring a less labeled dataset. Cruciani et al. proposed a heuristic function-based method for automatic labeling in an online supervised training approach (Cruciani et al., 2018). The algorithm generated weak labels by combining step count and GPS information. Shamsipour et al. addressed the issue of labeling videos by considering only a few frames depicting the information of humans performing a particular activity (Shamsipour et al., 2017). They randomly selected three video frames instead of employing all the frames, and used CNN for extracting features and SVM for classifying actions from the conceptual features. However, the majority of the approaches in the literature are applied to whole-body human movements, such as walking, running, or sitting. In this work, we compared the classifiers' performance on movement building blocks that can be found in natural and intuitively performed movements and that can potentially be transferred to a robotic system using learning from demonstration (Gutzeit et al., 2019a). To our knowledge, there has been no previous work available that analyzes the influence of labeling techniques in movement classification and examines the possibility of speed transfer in action movements.

# 3 METHODS

In this section, the features that are extracted from the raw movement trajectories captured using a Qualisys motion capture system are described, along with the classification algorithms and the hyperparameter optimization methods used for training those algorithms.

## 3.1 Feature Extraction

Feature extraction is an important procedure in training machine learning algorithms. A meaningful representation of raw data can have a huge influence on the performance of predictive models. In this work, features are extracted from raw motion data in the same manner as mentioned in (Gutzeit, 2021). The data is recorded using a marker-based motion capture system with several markers placed on the arm of the subject as shown in Fig 2. The marker positions are transformed into a global coordinate frame with respect to the markers placed on the back of the subject. The features extracted directly from the raw data are the marker's 3D positions, velocity, orientation, and joint angle between them. The feature trajectories were interpolated to the same length and normalized in the range [0, 1].

## 3.2 Classification Models

### 3.2.1 K-Nearest Neighbor

KNN uses the proximity between the test data and already available training data to classify a sample. The closest proximity of the test data from the training data can be determined using distance metrics such as Euclidean, Manhattan, or Minkowski. We use KNN for comparison in this work because it performed exceptionally well on classifying human movement in a small-sized training dataset (Gutzeit et al., 2019b). The feature trajectories for each movement recording are transformed into a 1-D feature vector. The closest neighbor of each data sample is determined using Euclidean distance, and the number of neighbors K required to classify the test data is tuned using grid search.

### 3.2.2 Random Forest

Random Forest is a bagging algorithm where random bootstrap samples are drawn from the training data and multiple decision trees are constructed. Each individual tree in the random forest outputs a class prediction, and the overall prediction results of the model are obtained by a voting approach on the individual decision tree outputs (Ho, 1995). Due to the random selection of training data and features, the constructed decision trees are independent of each other, which makes the model resistant to overfitting problems. This improves its predictive performance and generalization abilities on unseen data. As the decision trees are generated in parallel at the time of training, it results in a speed-up of the training process. In this work, we have used a Random Forest classifier with grid search to tune the hyperparameters, such as the number of trees and the maximum depth of each decision tree in the algorithm.

### 3.2.3 Extreme Gradient Boosting

XGBoost (Chen and Guestrin, 2016) is one of the most popular machine learning algorithms in recent times, widely used in structured and tabular data. XGBoost is a decision-tree-based ensemble algorithm that utilizes a gradient boosting framework and efficiently makes use of parallel processing and cache optimization for better speed and performance. Boost-

ing algorithms attempt to accurately predict the target variable by aggregating weak classifiers, which in general do not perform so well individually but when combined perform even better than the strongest individual learner. In order to make the final prediction, XGBoost sequentially adds the classifiers and fits the new model on the residual or the errors of the previous predictions that are then combined with previous trees to make the final prediction. It employs gradient descent when introducing new models to minimize the loss.

#### 3.2.4 Convolutional Neural Network

In the last half of the decade, CNN has been one of the most popular variants of Neural Networks because of their substantial contribution and benchmarked results for computer vision, natural language processing (Kim, 2014) and time series based problems (Ismail Fawaz et al., 2019). Contrary to images, CNNs in time series can be seen as a kernel sliding in only one dimension instead of two dimensions. In this work, a very simple Convolutional Neural Network is proposed consisting of two 1D convolution layers followed by a pooling layer, dense layer and an output layer with a softmax activation. The number of filters in the convolution layer, number of neurons in the dense layer, learning rate, and type of optimizer are modulated using Keras Tuner as mentioned in section 3.2.7.

#### 3.2.5 Long Short-term Memory

LSTMs were introduced by (Hochreiter and Schmidhuber, 1997) and are explicitly designed to avoid the long-term dependency problem exhibited in Recurrent Neural Networks. In Recurrent Neural Networks, there are directed cycles between the units, i.e., they propagate data forward and also backward and have the ability to process arbitrary long sequences of inputs using their internal memory. But where the problems of gradient explosion and gradient disappearance arise in RNN, LSTMs are able to avoid these problems with their architecture modifications, such as cell state, and its various gates. These gates are responsible to keep the essential information or forget it during training. In this work, we use a simple structure with two LSTM layers followed by a dropout layer and output layer with a softmax activation.

#### 3.2.6 CNN-LSTM Network

CNNs are known to be a robust feature extractor and capable of creating informative representations of time series data. On the other hand, LSTM networks perform pretty well at extracting patterns for long input sequences. The combination of both the networks has shown good results on challenging sequential datasets (Mutegeki and Han, 2020). A CNN-LSTM model is used comprising two 1D convolution layers with ReLu activation function followed by 1D max-pooling layer and a flattening layer for formatting the feature map so that it can be consumed by the LSTM layer. Afterwards, the flattened feature maps from the previous layers are fed as an input to an LSTM layer followed by a dense layer and output layer with a softmax activation. We have also used a dropout layer, where randomly selected neurons are ignored during training. It helps in making the network capable of better generalization and less likely to overfit the training data. The number of layers, learning rate, and type of optimizer is modulated using Keras Tuner as mentioned in section 3.2.7. All the mentioned Deep Learning models are trained using sparse categorical cross-entropy loss function for 20 epochs using Adam optimizer and early stopping is used to stop the training if the accuracy on a validation dataset did not increase in the last $n$ epochs, where $n$ is called patience value to prevent the model from overfitting the training data. The experiments were automatically tracked using Weights & Biases tool (Biewald, 2020).

#### 3.2.7 Hyperparameter Optimization

Hyperparameter tuning plays a vital role in the performance of machine learning algorithms. Hyperparameters can have a significant impact on model training in terms of model accuracy, training time, and computational requirements. In this work, Keras Tuner (O'Malley et al., 2019) that can be seamlessly integrated with Tensorflow 2 is used for tuning different hyperparameters required in training deep learning models. Methods used for tuning hyperparameters require defining a search space consisting of hyperparameters and their ranges that are needed to be optimized. Some hyperparameters that are optimized in this work are number of filters in convolution layers, number of units in LSTM layer, number of neurons in the dense layer and, choice of optimizers.

For each of the above-mentioned parameters, a range of its possible values is provided. Keras Tuner supports different search heuristics (such as random search, Hyperband, and Bayesian optimization) that helps in finding the best value of the defined hyperparameter to enhance the model performance. In this work, a Bayesian optimization search strategy is utilized for the optimization of hyperparameters. It eliminates the problem of choosing a combination of

different hyperparameters randomly that could sometimes result in the abysmal combinations of parameters. Thus, resulting in failure to improve the model accuracy. Instead of choosing random combinations, the Bayesian optimization strategy chooses the best possible hyperparameters based on the model performance of previous combinations. It constructs a probabilistic representation of the performance of a given Machine Learning algorithm, which is modeled using Bayesian inference and Gaussian process.

### 3.3 Evaluation Approach

The six algorithms described in section 3.2 are compared on the movement recordings labeled using two different methods as mentioned in section 4.1. The classifiers are trained on the data comprising arm movements performed by the subject at a slow and normal speed (section 4.1) and evaluated on the movements performed by the subject at a fast speed. The main focus of the experiments was to investigate the impact of labeling techniques on the generalization ability of the model for speed transfer in the fast-paced movement. These movements were appropriate for evaluating the trained model because segmentation of the fast-paced movements into basic action movements is more challenging compared to slower-paced movements and requires precise tracking of the subject's arm position. The training data was divided into 5 folds using stratified cross-validation. In each split, the evaluation is performed on the unseen test data. Finally, generalization accuracy with a standard deviation of mean accuracy and F1 score is reported. The dataset was completely balanced with an equal number of data for each class.

## 4 EXPERIMENTAL DATA

### 4.1 Stacking Scenario Data

The experiment was conducted on a single subject and movements were recorded with a Qualisys motion tracking system that uses infrared light reflecting markers. Additionally, the performed movements were recorded using a video camera. Markers were attached to the right hand, elbow, shoulder and back of the subjects, as shown in Figure 2. The marker positions were tracked with 7 Qualisys cameras and data was recorded at 500Hz. The subject was asked to perform a basic stacking movement as shown in Figure 3 where bricks of different colors were placed on fixed positions on the table and the participant was asked to place the bricks in the middle of the table

Figure 2: Stacking-scenario setup. Positions of markers attached on the arm and the back of the subject are recorded using a camera based motion tracking system.

by stacking it one by one. The experiment was performed by arranging the bricks in different stacking order: the green brick was always kept at the bottom while the other bricks (red, blue, yellow) were arranged in different permutations. Thus, overall, 6 different stacking orders were recorded with three repetitions of each stacking order. The movement for stacking the bricks was recorded at three different speeds (slow, normal, fast) for all 6 stacking orders. The normal and slow speed were intended to provide a comfortable speed for placing bricks from their respective position to the middle of the table one over another, while the fast speed challenged the participant. There were many instances where the bricks were not successfully placed over one another due to the fast arm movement, resulting in bricks tripping over the table.

### 4.2 Labeling Techniques

The movement data was decomposed using two different ways into 8 classes (middle2front, front2middle, middle2left, left2middle, middle2right, right2middle, middle2down, down2middle) based on the position from where the bricks were supposed to be picked and placed. In the first labeling technique, it was segmented using the video recorded for the experiment that was synced precisely with the data recorded from the Qualisys motion tracker. The labeling method was quite tedious and time-consuming, as one has to cautiously track the image frames on the video where the subject picks and places the bricks. Although tracking the movement of the data recorded at slow and normal pace was quite precise, labeling the data recorded at fast pace was very challenging. In the second labeling technique, the arm movement data was automatically segmented using a velocity-based probabilistic segmentation presented in (Gutzeit et al., 2014) into

basic movement units with a bell-shaped velocity. The unnecessary trajectories were removed, and essential segments required for training the model were annotated using a labeling tool developed at our institute, which visualizes the movement trajectories in 2D and 3D. Using this tool, inaccurate segment boundaries of the automatic segmentation approach were corrected. After labeling the data using both methods, the dataset consisted of 144 arm movements each for slow and normal paced recordings and 192 arm movements for fast-paced recordings, that means in total 480 labeled movement sequences were available.

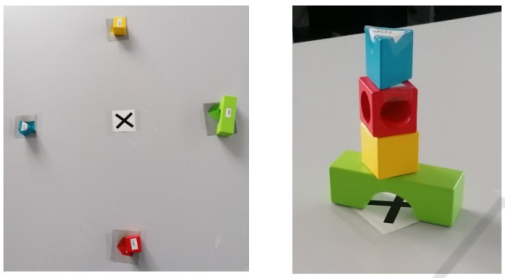

Figure 3: Stacking-scenario setup. The left image shows the different positions of the bricks on the table, and the right image shows one of the stacking examples. The bricks were stacked at the position of the cross.

### 4.3 Complexity of the Dataset

In this section, we compare the structure and diversity of the movement recordings labeled using above-mentioned techniques. For understanding the overlapping between different classes in the dataset, t-SNE (t-distributed Stochastic Neighbor Embedding) introduced by (van der Maaten and Hinton, 2008) is used for exploring a set of points in a high-dimensional space by transforming it into a lower-dimensional space. In Figure 4, we can see that data from the stacking scenarios labeled using the movement trajectories is less complex and are separated more clearly, but one can see overlap in at least 4 classes in the segments labelled using videos.

## 5 RESULTS AND DISCUSSIONS

In this work, the generalization ability of different models, such as KNN, Random Forest, XGBoost, CNN, LSTM, and CNN-LSTM model, to data recorded at different speeds are compared using two different strategies. The main aim was to demonstrate the influence of labeling methods in movement speed transfer within human movements. The classifiers are trained on data consisting of movements recorded at a slow and normal pace and tested on data recorded on fast-paced movements. The results of the generalization capabilities of the classifiers are shown in Figure 5. As we can see from the plots, all the classifiers have a much better generalization on the fast-paced movements when training data is labeled using movement trajectories. The results illustrate the significance of labeling strategies and their impact on classification accuracy irrespective of the choice of classifiers. For the model trained on normal movements and labeled using trajectories, there is almost 20% improvement in accuracy for all the models except LSTM that has an accuracy difference of approximately 35%. CNNs are the best performing model with a mean accuracy of 98% and F1-score of 97.7% for normal movements labeled using trajectories. For the model trained on slow movements and labeled using trajectories, KNN was the best performing model with an accuracy of 99% and an F1-score of 98.5%, and all models except XGBoost have an accuracy difference of approximately equal to or greater than 16%. Precise labeling of the recordings from videos requires meticulous tracking of the arm by the labeling person, and there are chances of human errors in tracking the accurate frames from the videos. That could be the reason for low accuracy on data labeled using videos. As the dataset consists of movements derived from simple stacking scenarios, distance-based algorithms performed considerably better and were almost equivalent to CNNs in performance. The LSTM classifiers fail to generalize well, but providing more data by employing augmentation techniques and training for more epochs could further enhance the results.

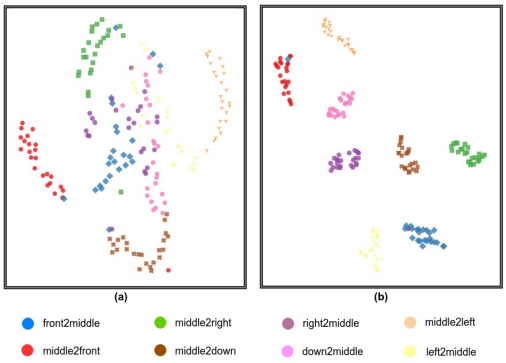


Figure 4: T-SNE plots of the stacking scenarios recorded at a fast pace (a) Data labeled using Videos (b). Data labeled using movement trajectories. Each action movement labels can be identified by a different color.

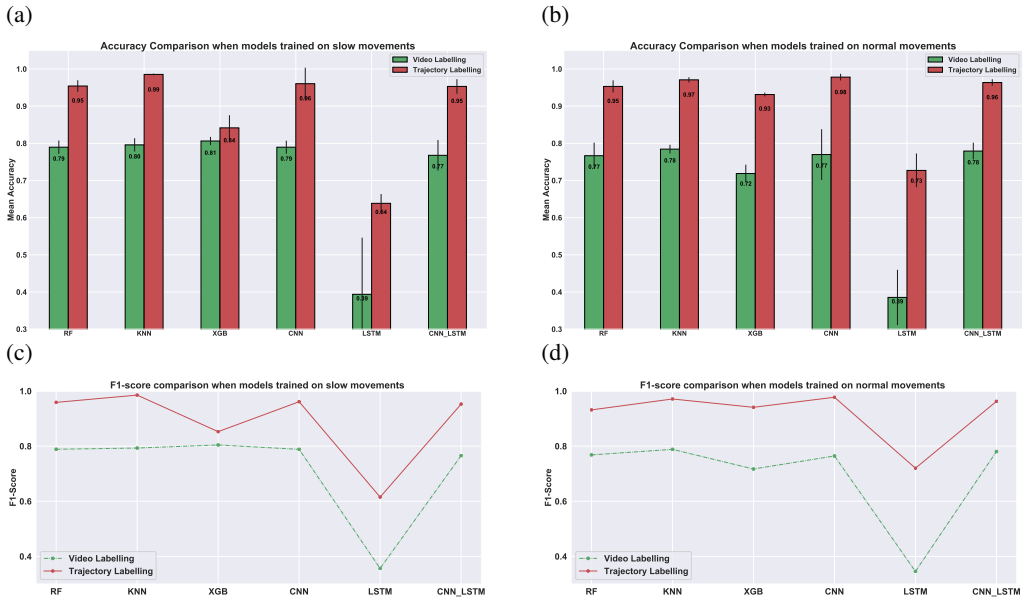


Figure 5: (a) and (c) Accuracy and F1 score comparison when model trained on slow movements. (b) and (d) Accuracy and F1 Score comparison when model trained on normal movements.

## 6 CONCLUSION AND FUTURE WORK

In this paper, we studied the impact of annotation quality on the classification accuracy on data consisting of basic human movements. Two different labeling strategies have been proposed and six different Machine Learning and Deep Learning models were compared. The potential possibility of speed transfer using the model trained on the data labeled using these two strategies was examined. It is found that fast-paced movements are better recognized on data labeled using trajectories of the recorded movements. The best results could be achieved with k-Nearest Neighbor and CNNs, achieving an accuracy of 99% and 98% on the model trained on slow and normal paced movements respectively.

For future work regarding the movement recognition models, some self-supervision methods that showed promising results in the field of Computer vision and NLP domain could be explored and there are possibilities to leverage such networks for sensor data in human action recognition. Although it does not completely discard the usage of labeled data, it learns useful representations of the data from the unlabeled dataset, which can then be fine-tuned on a small number of labeled data. Further, more focus could be given to traditional machine learning methods for model explainability. It would help to prevent model bias and could help to understand the working of a model in a better way. Shapely values (Lundberg and Lee, 2017) and LIME (Ribeiro et al., 2016) can give a rough idea about the features that greatly influenced the performance of the classifiers. Model interpretability has a huge prospect in the AI community, one can compare the models used in this work based on their interpretability to get a better understanding of these black-box models for human action recognition tasks.

Regarding the influence of the labeling techniques, the experiments conducted in this paper were performed with just one subject on simple movement data. For a deeper investigation of this influence, the movements of more subjects and more complex movements should be analyzed. Furthermore, not only the labeling technique but also the experience of the person labeling the data should be taken into account. However, the first small study presented in this paper already shows that accurately segmented data could significantly improve the movement classification accuracy.

## ACKNOWLEDGEMENTS

This work was supported through a grant of the German Federal Ministry for Economic Affairs and Energy (BMWi, FKZ 50 RA 2023).

# REFERENCES


Biewald, L. (2020). Experiment tracking with weights and biases. Software available from wandb.com.

Chen, T. and Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA. ACM.

Cruciani, F., Cleland, I., Nugent, C., McCullagh, P., Synnes, K., and Hallberg, J. (2018). Automatic annotation for human activity recognition in free living using a smartphone. *Sensors*, 18(7).

Donahue, J., Hendricks, L., Guadarrama, S., Rohrbach, M., Venugopalan, S., Darrell, T., and Saenko, K. (2015). Long-term recurrent convolutional networks for visual recognition and description. pages 2625–2634.

Gutzeit, L. (2021). A comparison of few-shot classification of human movement trajectories. In *Proceedings of the 10th International Conference on Pattern Recognition Applications and Methods (ICPRAM-2021), February 4-6, Austria*, pages 243–250. SciTePress.

Gutzeit, L., Fabisch, A., Petzold, C., Wiese, H., and Kirchner, F. (2019a). Automated robot skill learning from demonstration for various robot systems. In *KI 2019: Advances in Artificial Intelligence. German Conference on Artificial Intelligence (KI-2019), September 23-26, Kassel, Germany*, LNAI, pages 168–181. Springer.

Gutzeit, L., Otto, M., and Kirchner, E. A. (2019b). Simple and robust automatic detection and recognition of human movement patterns in tasks of different complexity. In *Physiological Computing Systems*. Springer.

Gutzeit, L., Schröer, M., Metzen, J. H., and Kirchner, E. A. (2014). Velocity-based multiple change-point inference for unsupervised segmentation of human movement behavior. In *Proceedings of the 22nd International Conference on Pattern Recognition. International Conference on Pattern Recognition (ICPR-2014), 22nd, August 24-28, Stockholm, Sweden*, pages 4564–4569. IEEE.

Ho, T. K. (1995). Random decision forests. In *Proceedings of 3rd international conference on document analysis and recognition*, volume 1, pages 278–282. IEEE.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

Ismail Fawaz, H., Forestier, G., Weber, J., Idoumghar, L., and Muller, P.-A. (2019). Deep learning for time series classification: a review. *Data Mining and Knowledge Discovery*, 33(4):917–963.

Ji, S., Xu, W., Yang, M., and Yu, K. (2013). 3d convolutional neural networks for human action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):221–231.

Kim, Y. (2014). Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1746–1751.

Lundberg, S. and Lee, S. (2017). A unified approach to interpreting model predictions. *CoRR*, abs/1705.07874.

Mutegeki, R. and Han, D. S. (2020). A cnn-lstm approach to human activity recognition. In *2020 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*, pages 362–366.

O'Halloran, J. and Curry, E. W. J. (2019). A comparison of deep learning models in human activity recognition and behavioural prediction on the mhealth dataset. In *AICS*.

O'Malley, T., Bursztein, E., Long, J., Chollet, F., Jin, H., Invernizzi, L., et al. (2019). Kerastuner. https://github.com/keras-team/keras-tuner.

Ribeiro, M. T., Singh, S., and Guestrin, C. (2016). "why should I trust you?": Explaining the predictions of any classifier. *CoRR*, abs/1602.04938.

Schröder, M., Yordanova, K., Bader, S., and Kirste, T. (2016). Tool support for the online annotation of sensor data.

Shamsipour, G., Shanbehzadeh, J., and Sarrafzadeh, H. (2017). Human action recognition by conceptual features.

van der Maaten, L. and Hinton, G. (2008). Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605.

Wang, H. and Schmid, C. (2013). Action recognition with improved trajectories. In *2013 IEEE International Conference on Computer Vision*, pages 3551–3558.

Zhang, W., Zhao, X., and Li, Z. (2019). A comprehensive study of smartphone-based indoor activity recognition via xgboost. *IEEE Access*, 7:80027–80042.