Project Report: One Step Ahead

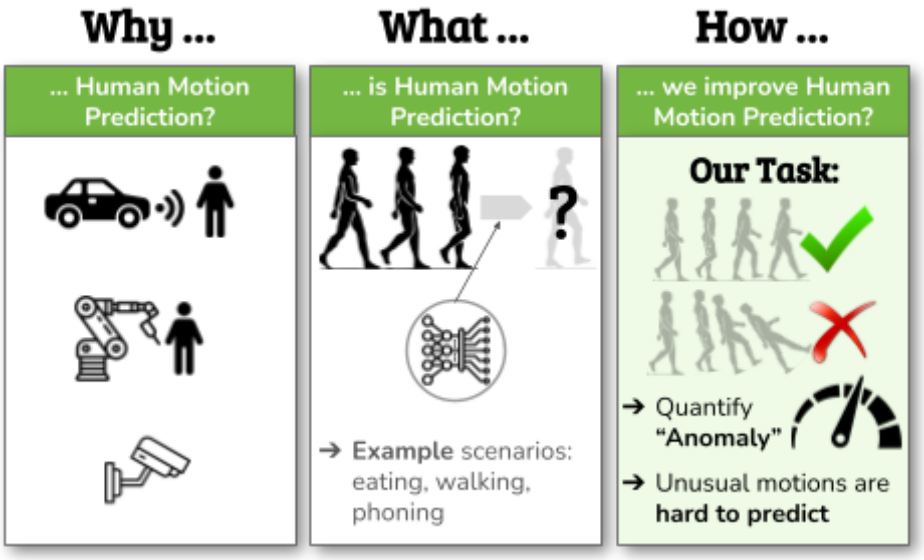
Detecting unusual Human Motions for QualityMinds

This report summarizes the project that we, Jonas, Alaa and Vincent completed as a final project of the data science bootcamp of Constructor Learning. The bootcamp provided us with a broad skillset in the field of data science, such as numpy, pandas, matplotlib, Machine Learning, Deep Learning, NLP, SQL, Git, and Docker. This project helped us apply our skills to a real world problem in the field of human motion prediction for autonomous driving applications.

Table of Contents

Introduction to Human Motion Prediction

This section aims to answer the most important questions regarding human motion prediction, shown in this graphic.



Why HMP? Motion prediction plays a crucial role in robotics, where predicting human movements is vital for ensuring safe and efficient human-robot interactions. Sports analytics leverages human motion prediction to gain insights into athletes' performance, aiding in coaching and training strategies. Autonomous vehicles use human motion prediction to anticipate pedestrian movements and improve pedestrian and cyclist safety. In the healthcare sector, motion prediction helps in understanding and predicting patient movements, assisting in rehabilitation and injury prevention.

What is HMP? Human motion prediction focuses on forecasting the future movements and actions of individuals or groups. Using various algorithms and data sources, human motion prediction aims to anticipate the motion of humans in different scenarios. The idea is, given a data sequence of e.g. 0.4 seconds, it is to be predicted, what the motion in the next second will be.

How we improved HMP? In collaboration with Qualityminds we inspected a dataset of human motions to analyze unusual motions and to quantify this anomaly. The anomalies and the quantification data then helped QualityMinds to improve their motion prediction models, as these unusual motions are hard to detect.

QM Human Motion Prediction Project

In the QM Human Motion Prediction Project (HMPP), the main aim is to utilize different human motion prediction models, such as the STS-GCN (Space Time Separable - Graph Convolutional Network) model to predict human motion sequences. The input data consisted of a sequence of 10 frames (= 0.4 seconds), each containing 32 joints with 3 coordinate values. The goal of the motion prediction models is to forecast the subsequent motion of humans for a duration of 1.0 second, represented by a sequence of 25 frames.

The project encompassed various branches, each serving different use cases and addressing essential aspects of human motion prediction. The branches included:

1. Data Analysis (Learning representation)
2. Adversarial Attacks (Robust Learning)
3. Model Interpretability
4. Efficient Learning (Self-supervised Learning)
5. Simulation

Our subproject was located within the Data Analysis and Learning Representation branch, and specifically our main task was to identify and quantify the degree of anomaly of motion sequences. As a final output we were asked to provide QualityMinds with a multidimensional numerical representation, e.g. a vector with four entries, of the outlyingness of each motion sequence identifiable by an index (e.g., sequence 352 in the action walking). To give an explanation about why a motion sequence might be considered unusual, one example is the motion where different actions are combined, such as "eating" while "walking," or instances of sudden, unexpected movements.

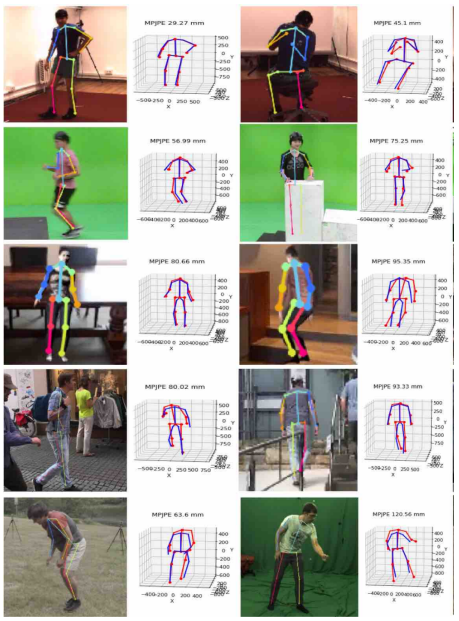
Additionally, as a second goal, we aimed to describe relevant kinematic features of motion sequences, such as the velocity or acceleration of specific joints, using visual and statistical methods.

Detecting outlying motion sequences is crucial within the context of the overall project for several reasons. One of the motivating assumptions behind our task was that outlying motion sequences tend to lead to high errors in motion prediction models. Therefore, by identifying and characterizing these anomalies and validating that they indeed are at the root of high prediction errors of various motion prediction models, we would be able to improve the performance of motion prediction models. By solving this task, we aimed to make motion prediction models more robust and attentive to outlying subsequences that previously caused significant prediction errors. For example, detecting unusual motions and incorporating this information as a regularization or penalty in the loss function, we may be able to support QualityMinds in improving the overall performance of their motion prediction models. Moreover, it was explained to us that the detection of unusual motions will also support the work in several other project branches, e.g. serving as meta information for performing adversarial attacks or helping to improve model interpretability.

The Data we used

Datasets used in Human Motion Prediction

Human motion prediction relies on various datasets that capture human movements in different contexts and environments. These are four main datasets commonly used in human motion prediction research:



AMASS (Archive of Motion Capture as Surface Shapes):

The AMASS dataset is a comprehensive collection of 3D human motions captured from various sources, including motion capture (mocap) data, 3D scans, and parametric body models. The dataset contains over 15,000 motions and covers a wide range of human activities and poses. It includes both mocap data and 3D scans, making it suitable for generating realistic human animations and understanding body shape variations.

Human3.6M (H36M) Dataset:

H36M was captured in a controlled environment using a motion capture system and includes 3D body joint positions and 2D projections from multiple camera views. The dataset contains more than 3.6 million images, covering 15 activities, and multiple subjects performing various movements. H36M is widely used for evaluating human pose estimation and motion prediction algorithms due to its large-scale and diverse nature.

3DPW (3D People in the Wild) Dataset:

3DPW focuses on capturing humans performing daily activities in unconstrained outdoor environments (in the wild) using RGB-D cameras. The dataset contains over 60,000 frames with 3D human pose annotations. 3DPW provides real-world scenarios, challenging researchers to predict human motion in more realistic settings.

ExPI (Extended People in Photo Albums) Dataset:

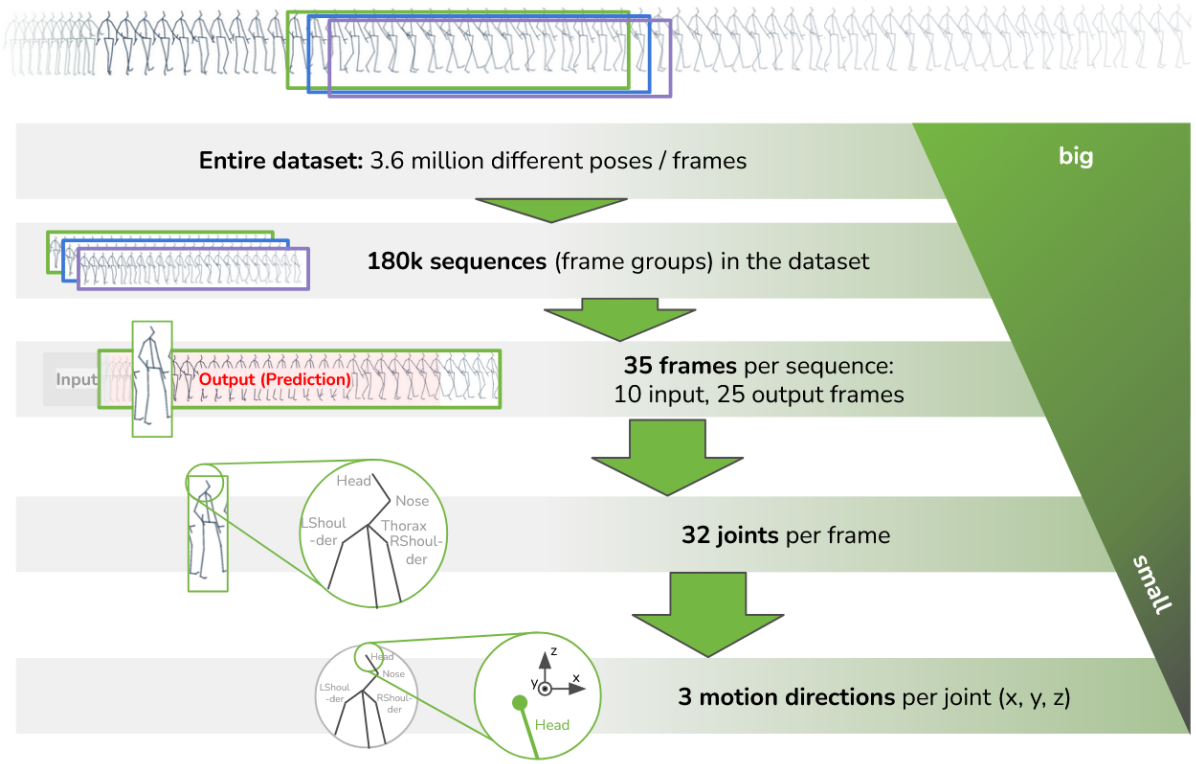
ExPI collects images from personal photo albums and social media platforms, with various humans in unconstrained poses and environments. The dataset contains over 500,000 images and diverse human pose variations. ExPI offers a unique dataset for studying human motion prediction in everyday life contexts, with non-ideal poses and varying clothing.

For this project, the Human3.6M dataset was selected for performing data analysis on OOD detection in the dataset. This is due to the high variation in the dataset and the possibility of generalization using this set.

Structure of the H3.6M Dataset

The H3.6M dataset consists of 3.6 million images/frames/poses. These poses are clustered into 15 actions, which are performed by 11 actors. Each action is performed in two different videos.

The poses were then grouped into 180.000 motion sequences, each consisting of 35 pose frames: 10 frames used as an input to the motion prediction model (0.4 seconds), and 25 frames used as an output (1.0 second). Each frame consists of geometrical information for 32 joints that represent the joints in the body, such as the wrists, hip joints, leg joints, or head. Every joint can move in a three-dimensional space, so in X-, Y-, and Z-direction, hence the amount for these three values is given. The dataset structure is depicted in the following illustration.



When loading the data structure using the loader QualityMinds provided in the given configuration, the result was a dictionary that used the actions as keys and the geometrical data of all sequences with the given action. The result was a four-dimensional numpy array with the following shape:

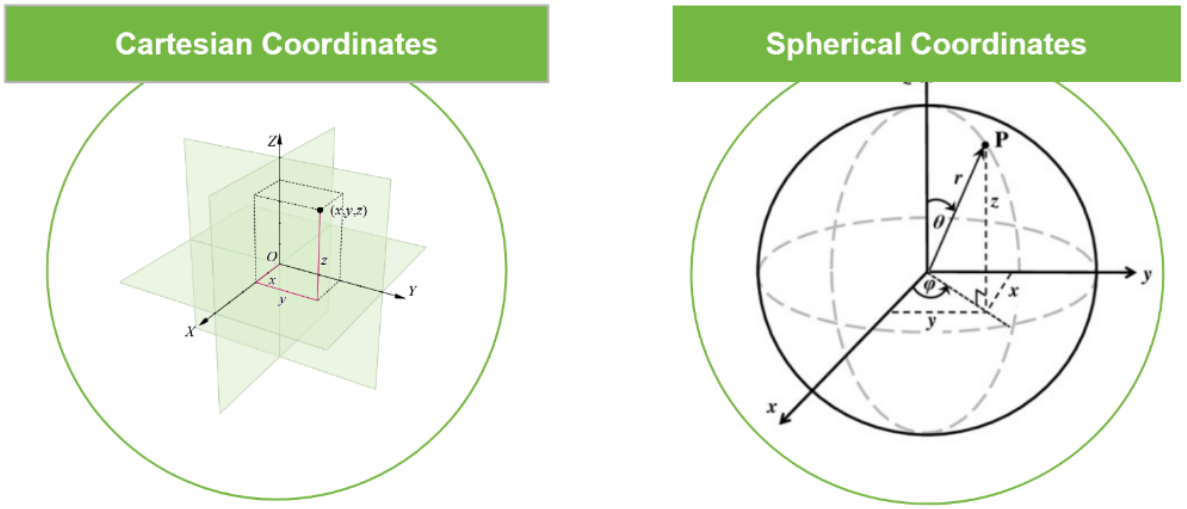
- 180.000 rows in axis 0 (number of sequences per motion)
- 35 rows in axis 1 (number of frames per motion sequence)
- 32 rows in axis 2 (number of joints per frame)
- 3 rows in axis 3 (number of motion directions per joint).

To process this data in the machine learning models, it was then flattened into a two-dimensional Dataframe using pandas. The Dataframe had the following dimensions:

- 180.000 rows (number of all sequences)
- 3360 columns (number of frames x number of joints x number of motion directions)

Cartesian and Spherical Coordinates and why it can be beneficial to use the latter in Human Motion Prediction

Cartesian coordinates and spherical coordinates are two different systems used to represent points in three-dimensional space. In the Cartesian coordinate system, a point is represented by its three perpendicular distances (x, y, and z) from the origin along the x, y, and z axes, respectively. On the other hand, in the spherical coordinate system, a point is represented by its radial distance (r) from the origin, inclination angle (θ) from the positive z-axis, and azimuth angle (ϕ) from the positive x-axis. A depiction of cartesian vs. spherical coordinates is shown below.



For human motion prediction, using spherical coordinates can have several advantages. Firstly, spherical coordinates are more intuitive for representing rotations and orientations, which are crucial in capturing complex human movements. Secondly, they can provide a more concise representation of motion, as certain movements (e.g., rotating in place) can be expressed more simply in spherical coordinates compared to Cartesian coordinates. Lastly, for tasks involving 3D rotations, such as orientation tracking and joint angle estimation, spherical coordinates can simplify the calculations and reduce computational complexity, leading to more efficient and accurate motion prediction models.

The transformation applied to the data

As using spherical coordinates is beneficial for human motion prediction, we applied this transformation to our data. The components are calcuated as follows:

$$r = \sqrt{x^2 + y^2 + z^2}$$
$$\theta = arccos\,(z/r)$$
$$\phi = arctan2\,(y, x)$$
$$c = (c_r, c_\theta, c_\phi)$$

To be able to analyze the data and to see, if this yields to an increase in model accuracy, we also calculated the velocity and the acceleration in spherical coordinates. We did this by deriving the spherical coordinates by the time using the following formula:

$$v_i = (c_{i+1} - c_i)/dt$$

$$v = (v_r, v_\theta, v_\phi)$$

c_2 refers to the spherical coordinates of the joints on timestamp 2
c_1 refers to the spherical coordinates of the joints on timestamp 1
dt refers to the difference of time between the two timestamps. As we were using 25 frames per second, there was a difference of 40ms between the timestamps.

$$a_i = (v_{i+1} - v_i)/dt$$

$$a = (a_r, a_\theta, a_\phi)$$

v_2 refers to the velocity in spherical coordinates of the data on timestamp 2
v_1 refers to the velocity in spherical coordinates of the data on timestamp 1.

The resulting method leads to a reduction of the sequences. The velocity of a sequence only contains 34 velocity datapoints, and 33 acceleration datapoints. This is because the velocity of the last coordinates v_35 cannot be calculated anymore, as there is no v_36.

Methodology

Our methodology for detecting outlying motion sequences involved a multi-step pipeline, starting with data representation and reduction. We aimed to find sequences with higher “degree of anomaly”. We named the “degree of anomaly” as “decision scores”. To achieve a diverse representation, we utilized four different unsupervised out of distribution (OOD) detection methods: CBLOF, HDBSCAN, IFOREST, and LOF.

In the first step, we converted the input data, consisting of 35 frames with 32 joints and 3 coordinates each, into various representations, including Cartesian coordinates, joint positions, spherical coordinates, velocities, and accelerations. Next, we reduced the dimensions using Principal Component Analysis (PCA), retaining 50 principal components that captured approximately 96% of the original variation. Standard scaling (z-score normalization) was then applied to the data to ensure consistency in the OOD detection process.

By using multiple OOD detection methods, each defining anomaly differently, we aimed to achieve a more comprehensive and diverse identification of outlying sequences.

Method	Which sequences are considered anomalous, i.e. have a higher decision score?
CB-LOF	Separating into small/large clusters (k-means) → Sequences with a high distance to the nearest large cluster
HDBSCAN	Sequences not belonging to any density-based cluster or being at the edge of a density-based cluster

IForest	Sequences that are easily isolated in a random tree, i.e. a low number of splits required (averaged over a forest)
LOF	Sequences with significantly lower local density compared to their k nearest neighbors

To detect anomalous sequences using CBLOF, the data are separated into small and large clusters using k-means (or other clustering algorithms), and identified sequences with high distances to the nearest large cluster are considered unusual. HDBSCAN defines anomaly by finding sequences that do not belong to any density-based cluster or are at the edge of a density-based cluster. IFOREST, a tree based method, identifies sequences as unusual, that are easily isolated in a random tree by a low number of splits required to do so. LOF, on the other hand, focuses on sequences with significantly lower local density compared to their k nearest neighbors.

As an unsupervised approach lacks ground truth, we adopted several strategies to check our results. First, we used an explorative strategy to validate the identified OOD sequences. We compared outlying and inlying sequences based on relevant kinematic characteristics such as visualizing velocity or acceleration of joints (e.g., average velocity/acceleration) and looking at them through motion GIFs.

However, our main approach for validation was to create artificial metrics for classification, i.e. to turn our unsupervised learning problem into an artificial supervised learning task. We achieved that by merging the prediction errors from two motion prediction models (STS-GCN and motionmixer) to our data. By defining a ground truth based on high error thresholds for one or all motion prediction models, and OOD sequences with high decision scores for the OOD detection model, we were able to calculate precision and recall in the usual way, considering true positives, true negatives, false positives, and false negatives.

A lot of flexibility was maintained in our definitions, allowing us to use variable thresholds for defining what are OOD sequences and combine single or multiple OOD detection methods and motion prediction models for creating the relevant definitions and calculating the metrics.

Key results / what we achieved

Four anomaly detection models

We provide four unsupervised learning models, namely CBLOF, IFOREST, HDBSCAN, LOF, to produce a learning representation for each sequence.
All the models were trained on a training set of four different subjects from the H3.6M dataset.
We developed three versions for each model:

- 1st version trained on the whole sequence (35 frames)
- 2nd version trained on the time input sequence (10 frames)
- 3rd version trained on the time output sequence (25 frames)

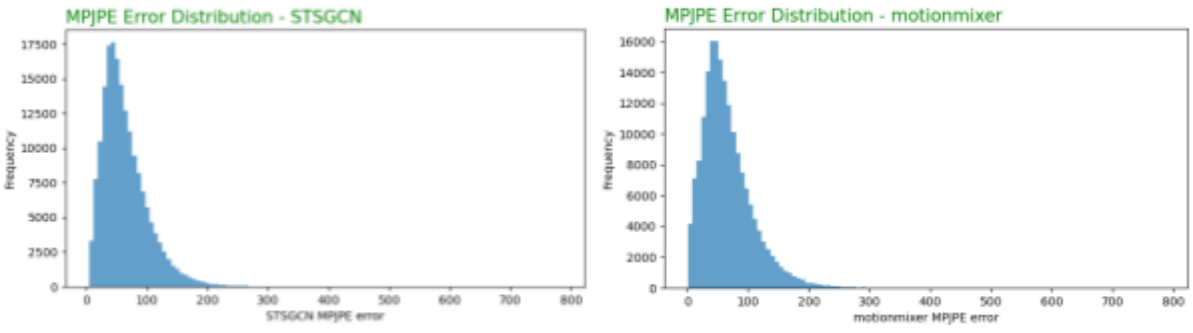
Each model provides us with a decision/OOD score. The higher the score, the more abnormal a sequence is. Out of distribution sequences tend to have higher scores than in distribution sequences.

In the Data_analysis_H36M.ipynb and presentation.ipynb notebook, we provided Analysis ToolKit to help analyze our findings.

Analysis Toolkit: Metric Comparison

To validate our results we used the MPJPE errors from two motion prediction models STSGCN and motionmixer as a ground truth.
We define two different thresholds to turn the unsupervised learning problem into a classification problem. This enables us to compute artificial versions of known metrics to measure ‘precision’ and ‘recall’.

Ground Truth sequences

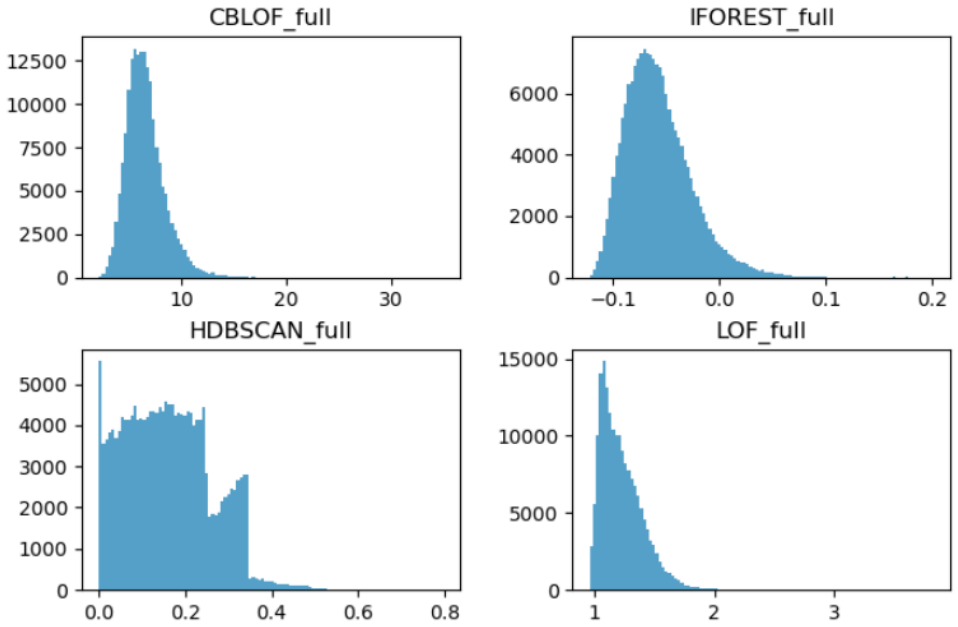


The above figures show us the distribution of the MPJPE error from different models.

1. We define the error threshold, denoted by e_t, as a real number between 0.85 and 1.

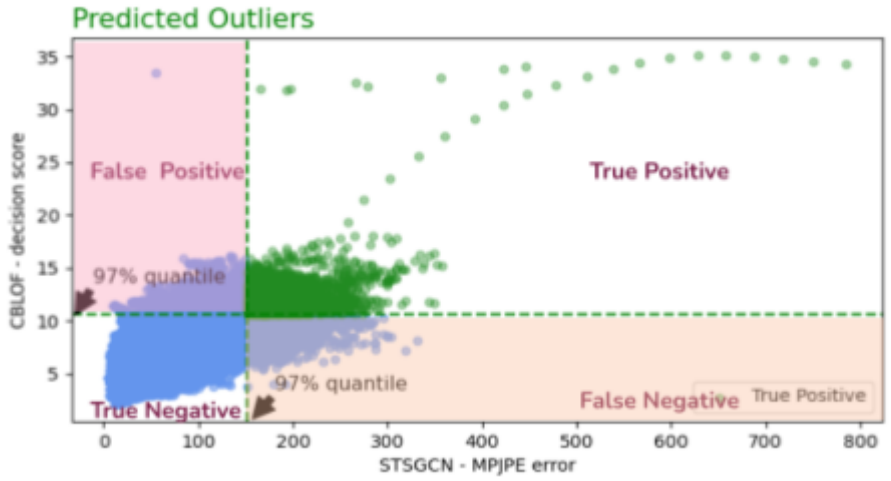
2. Compute the e_t percentile of the MPJPE error distribution for a motion prediction model.
3. We define **ground truth sequences based on the prediction error of the motion prediction models** as the sequences with MPJPE error equal to or higher than the e_t percentile .

It is obvious that our definition depends on the choice of motion prediction model used to define the ground truth sequences.



The above figures show us the decision/OOD score distributions from the four OOD models. To define the ground truth based on the motion prediction models for an OOD model, we follow the same technique we used to define ground truth sequences.

1. We start by defining a model threshold, denoted by m_t, as a real number between 0.85 and 1.0.
2. Compute the m_t percentile of the decision/OOD scores distribution for the OOD model.
3. We define **ground truth for an OOD model** as the sequences with decision scores equal to or higher than the m_t percentile.



Definitions:

- We say that a sequence is a **true positive**, for an OOD w.r.t a MP model, if its decision score is **higher than or equal** the m_t percentile of the decision score and its MPJPE error is **higher than** the e_t percentile of the MPJPE prediction error, as shown in above figure.
- Respectively a sequence is **true negative**, if its decision score is **lower** than the m_t percentile of the decision score and its MPJPE error is **respectively lower**, the e_t percentile of the MPJPE prediction error, see the above figure.
- We say that a sequence is a **false positive** for an OOD w.r.t a MP model if its decision score is **higher than or equal to** the m_t percentile of the decision score and its MPJPE error is **lower** than the e_t percentile of the MPJPE prediction error, see the above figure.
- We say that a sequence is a **false negative**, for an OOD w.r.t a MP model if its decision score is lower than the m_t percentile of the decision score and its MPJPE error is **higher** than the e_t percentile of the MPJPE prediction error, see the above figure.

Now we define the precision and recall of an OOD model with respect to a motion prediction model as follows:

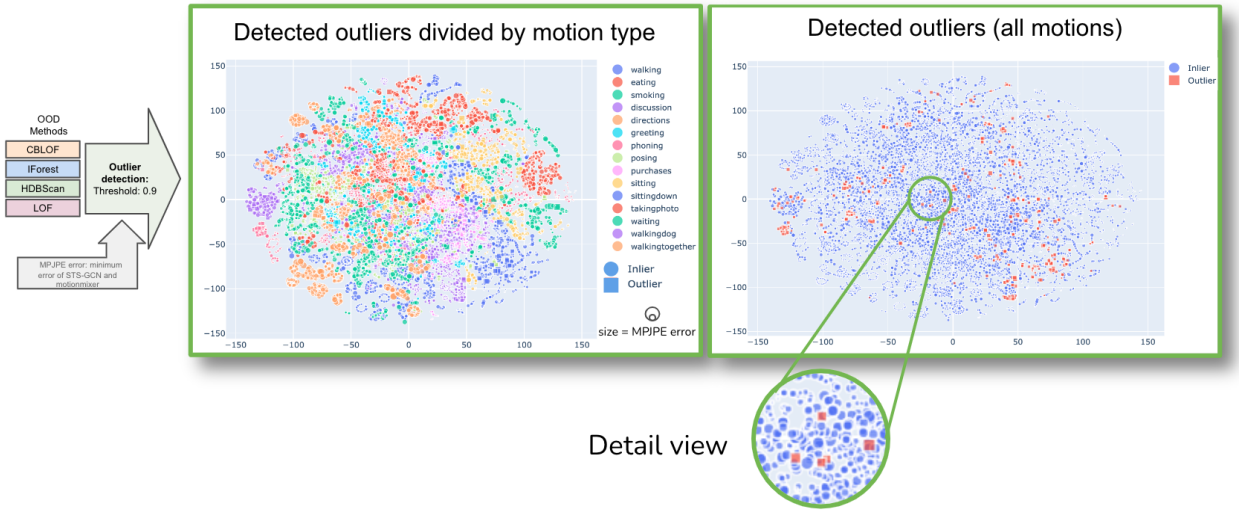
Precision: Rate of predicted OOD sequences with high motion prediction error, i.e.,

precision = True positive / (True positive + False Positive)

Recall: Rate of high error sequences captured by OOD sequences, i.e.,

Recall = True positive / (True positive + False Negative)

We also provide a simple interactive scatter plot in the section.


OOD Methods
CBLOF
IForest
HDBScan
LOF
Outlier detection:
Threshold: 0.9
MPJPE error: minimum error of STS-GCN and motionmixer
Detected outliers divided by motion type
walking
eating
smoking
discussion
directions
greeting
phoning
posing
purchases
sitting
sittingdown
takingphoto
waiting
walkingdog
walkingtogether
Inlier
Outlier
size = MPJPE error
Detected outliers (all motions)
Inlier
Outlier

Detail view

Analysis Toolkit: Generate dataframes with analysis using different thresholds

create_fast_analysis

Based on the above definitions we developed a function to draw analysis to compare the performance of different OOD models and MP models with respect to each other, for different error_thresholds and model_thresholds.

	motion_model_th:0.97	OOD_model_th:0.9	mode	recall	precision	recall_&	precision_&	recall_or	precision_or
0	STSGCN	CBLOF	full	0,71	0,21	0,50	0,30	0,80	0,15
1	STSGCN	IFOREST	full	0,61	0,18				
2	STSGCN	HDBSCAN	full	0,42	0,13				
3	STSGCN	LOF	full	0,65	0,19				
4	STSGCN	CBLOF	Ti_10	0,32	0,10	0,21	0,14	0,44	0,08
5	STSGCN	IFOREST	Ti_10	0,31	0,09				
6	STSGCN	HDBSCAN	Ti_10	0,26	0,08				
7	STSGCN	LOF	Ti_10	0,34	0,10				
8	STSGCN	CBLOF	To_25	0,74	0,22	0,52	0,31	0,84	0,16
9	STSGCN	IFOREST	To_25	0,65	0,19				
10	STSGCN	HDBSCAN	To_25	0,60	0,18				
11	STSGCN	LOF	To_25	0,68	0,21				
12	motionmixer	CBLOF	full	0,71	0,21	0,51	0,31	0,80	0,15
13	motionmixer	IFOREST	full	0,62	0,19				
14	motionmixer	HDBSCAN	full	0,43	0,13				
15	motionmixer	LOF	full	0,66	0,20				
16	motionmixer	CBLOF	Ti_10	0,34	0,10	0,22	0,15	0,47	0,09
17	motionmixer	IFOREST	Ti_10	0,33	0,10				
18	motionmixer	HDBSCAN	Ti_10	0,27	0,08				
19	motionmixer	LOF	Ti_10	0,36	0,11				
20	motionmixer	CBLOF	To_25	0,74	0,22	0,52	0,32	0,84	0,16
21	motionmixer	IFOREST	To_25	0,64	0,19				
22	motionmixer	HDBSCAN	To_25	0,61	0,18				
23	motionmixer	LOF	To_25	0,69	0,21				

In the above table we see the analysis when the error_threshold is set to 0.97 and the OOD model threshold is set to 0.9.
Moreover we provide two ways to combine the results from different OOD models.

1. By predicting a sequence as being out of distribution for a combination of OOD models if its decision score is **higher than** the m_t percentile for **all** the OOD models, which we denote by **precision_&, recall_&** in the above table.
2. By predicting a sequence as being out of distribution for a combination of OOD models if its decision score is **higher than** the m_t percentile for **at least one** OOD model, which we denote by **precision_or, recall_or** in the above table.

The function we provided above is pretty flexible; it can take as a list any combination of motion prediction models or OOD models and draw the analysis.

general_analysis

We also provided a function that combines different motion prediction models into the analysis.

```
general_analysis(metrics_df,model_threshold = 0.90, error_threshold= 0.97)
```

	modes_mt=0.9 _et=0.97	recall_&	precision_&	recall_or	precision_or
0	full	0,44	0,34	0,83	0,08
1	Ti_10	0,20	0,13	0,50	0,05
2	To_25	0,54	0,29	0,89	0,10

Here we define a **ground truth sequence based on prediction errors** as the sequence with MPJPE error equal to or higher than the e_t percentile for **ALL** the motion prediction models. This function is very flexible as well; it takes any list of OOD models and motion prediction models.

get_index

The last function in our toolkit accepts model_ threshold, e_threshold, list of OOD models, list of MP models, mode (which can be either the whole sequence of 35 frames or Ti = 10, or To = 25), and return the indices of the samples that are predicted as OOD sequences.

```
SMCf_90_97 = get_index(metrics_df, 0.94, 0.90, motion_models = ['STSGCN', 'motionmixer'], models = ['CBLOF'], mode = 'full')
SMCf_90_97
```

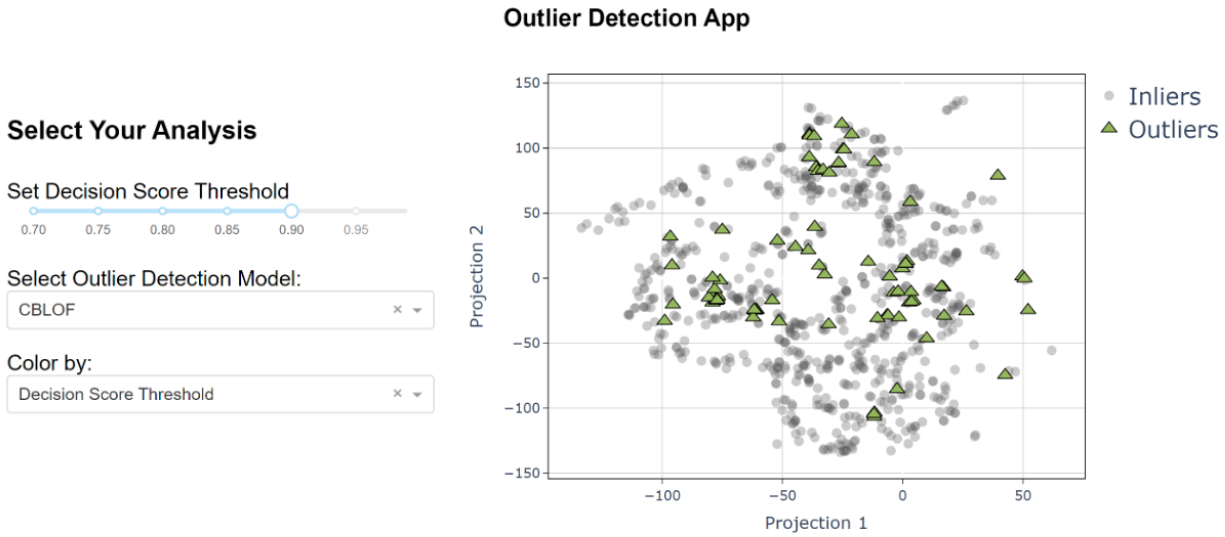
```
total number of ground truth outliers: 14775
total number of predicted outliers: 10805
Total number of True positive predicted outliers: 5950
Here is the dataframe of True positive predicted outliers
```

	idx	action	mean_mpjpe_STSGCN	mean_mpjpe_motionmixer	CBLOF_full	CBLOF_Ti_10	CBLOF_To_25	IFOREST_full	IFOREST_Ti_10	IFOREST_To_25	HDBSCAN_f
3937	3937	walking	124.785430	129.066933	10.371793	7.362556	10.795244	0.001265	-0.071583	0.013221	0.3153
3938	3938	walking	130.291236	138.715216	10.525085	7.177984	11.077783	0.002231	-0.069973	0.015913	0.3262
3939	3939	walking	138.487350	146.490266	10.617262	6.885268	11.276544	0.007945	-0.069029	0.023341	0.3330
3940	3940	walking	148.520770	157.249368	10.641260	6.558383	11.351107	0.016421	-0.075391	0.021537	0.3351
3941	3941	walking	155.824424	171.415822	10.600270	6.318740	11.285074	0.009485	-0.088138	0.027785	0.3327
...	...	...	...	...	...	...	...	...	...	...	
175294	4492	walkingtogether	134.894886	133.234842	9.690693	5.555075	9.557508	0.004726	-0.055282	-0.014773	0.2814
175301	4499	walkingtogether	145.354399	136.335502	9.660747	8.285280	8.693771	-0.001474	-0.032293	0.006574	0.2670
175302	4500	walkingtogether	137.899514	137.468329	9.842678	8.192215	8.574843	-0.006204	-0.042018	-0.008439	0.2621
175303	4501	walkingtogether	133.808679	135.767339	9.843707	8.182288	8.381147	0.003894	-0.031961	-0.008073	0.2543
175304	4502	walkingtogether	126.355372	129.562270	9.721660	8.421561	7.845458	0.002994	-0.017667	-0.014966	0.2455

5950 rows × 16 columns

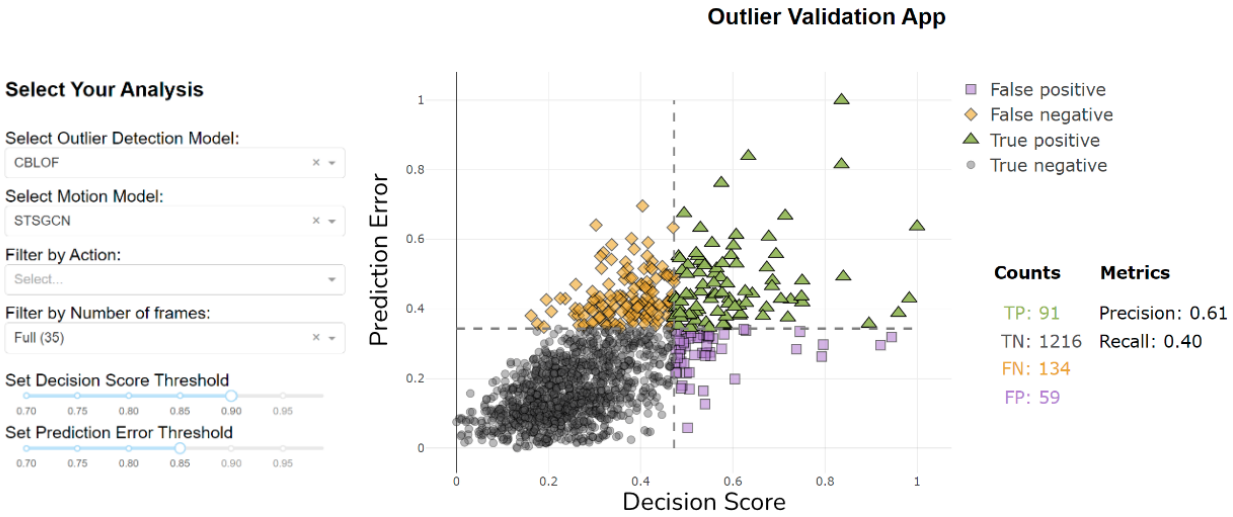
Analysis Toolkit: Generate two interactive apps

OOD Detection App



This interactive scatterplot displays motion sequences as data points, highlighting the detected outliers as green triangles and the inliers as gray circles. By hovering over an outlying data point, one can access detailed information such as the motion's index and action, enabling further analysis. Via the menu on the left, one can flexibly create different analysis scenarios. By changing the decision score threshold, one can define the fraction of motion sequences considered as outliers. Using the dropdown on the left, one can select among four different outlier detection models, each providing a unique perspective on what motion sequences are regarded as unusual. Moreover, one can color the data points by different actions such as walking, eating or sitting, allowing to compare outliers, marked as triangles, and inliers, marked as circles, both within and between actions.
Finally, to focus on specific actions only, one can select these from the legend on the right, allowing for an in-depth analysis.

OOD Validation App

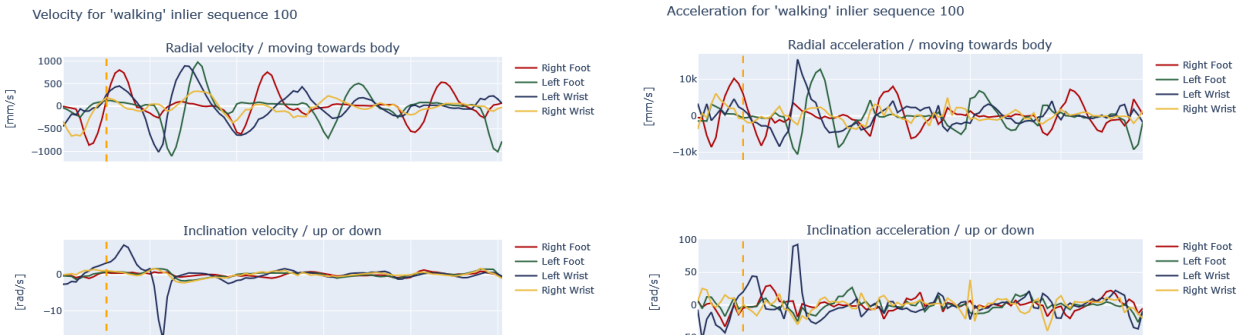


To validate our results, our second interactive plot shows the relationship between a motion sequence's degree of anomaly on the x-axis and its prediction error on the y-axis. It reveals a strong positive correlation as motion sequences with higher decision scores tend to have a higher prediction error. This suggests that the motions we identified as outliers are indeed one cause of high prediction errors.
As our first app, this visualization tool is fully interactive, allowing QualityMinds to assess detailed information on single outliers or choose between different analysis scenarios via the menu on the left. To further quantify the performance of our outlier detection analysis, we apply two different thresholds on decision scores and prediction errors. These thresholds divide the scatter plot into four regions.
The region colored in gray is where we have usual motions, that the motion model was able to predict well. The region colored in green is where unusual motions live, that the motion model failed to predict. The region colored in orange is where we have usual motions, but for some reason the motion model failed to predict. The region colored in violet is where unusual motions live, but the motion model was able to predict well. Based on the number of samples in each of these regions, our app interactively calculates two key metrics of precision and recall in the lower right, enabling the user to efficiently compare different analysis scenarios.

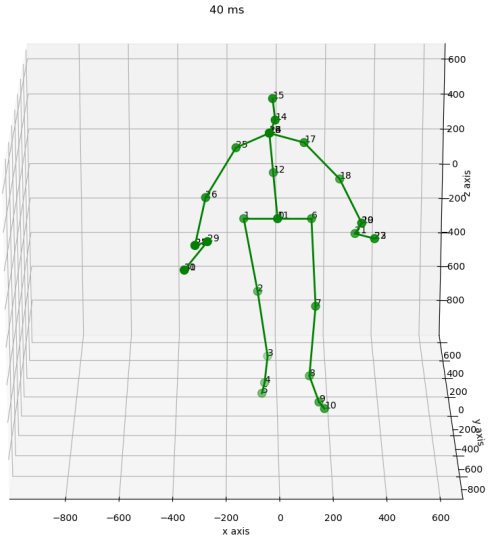
Analysis Toolkit: Kinematic Comparison Toolkit

Velocity and acceleration together are referred to as "kinematics", this terminology is used in this report and in the code. The kinematics data was included in the analysis toolkit and can be plotted for spherical coordinates. Given, the action "walking" is to be analyzed, the following kinematics are obtained for an in distribution sequence (ID) sequence (sequence 100):

Velocity and Acceleration for "walking" Frame 100 (ID sequence)

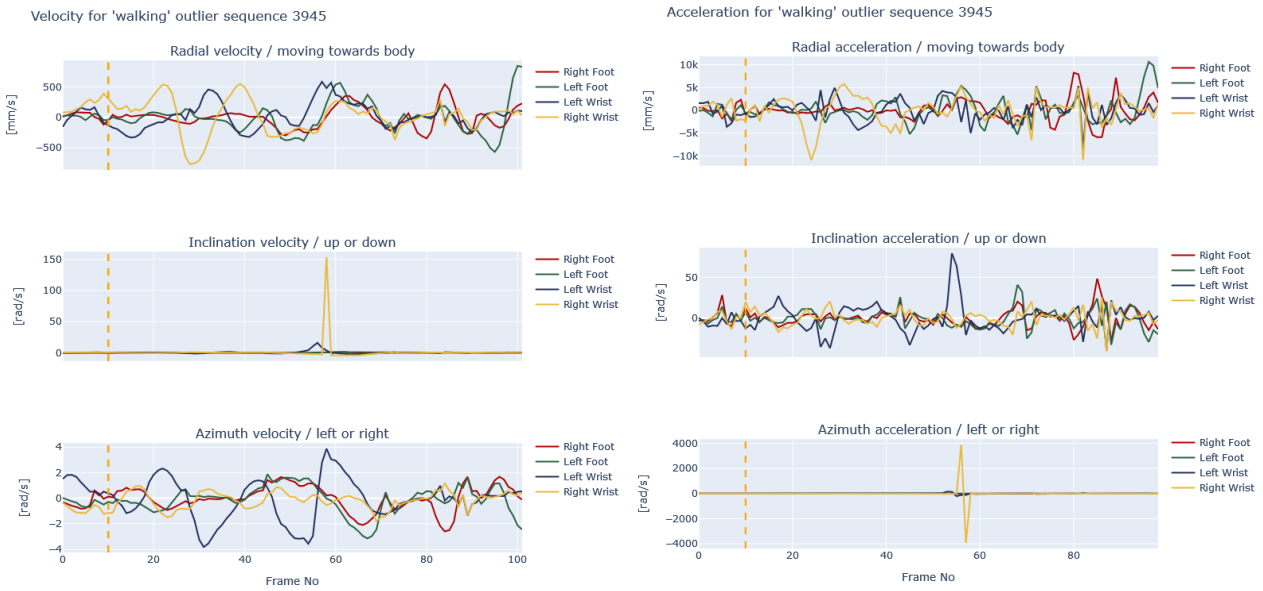


It can be seen very well from the previous graph, that there is a periodic motion of the right foot (red) and the left foot (green). This cyclic motion can be predicted by the model very well. The sequence 100 is shown in the following animation:

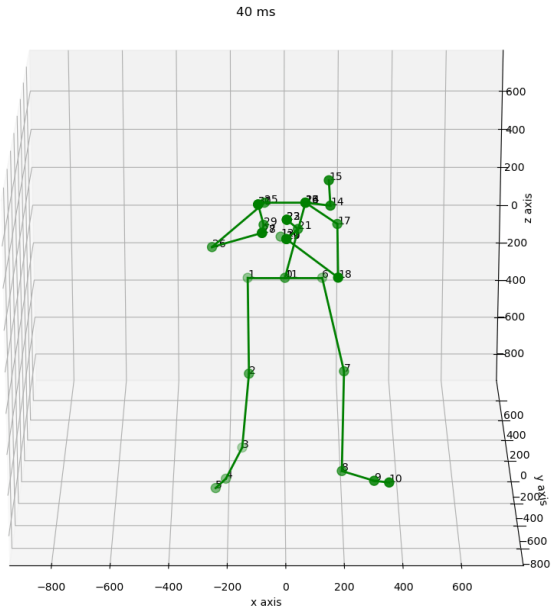


When analysing an OOD sequence from the “walking” sequences, the following data is obtained:

Velocity and Acceleration for “walking” Frame 3945 (OOD sequence)



From this graph, it can be seen that the regular motion is not there anymore. The left and the right foot are moving in an irregular pattern, and the hands are moving irregularly as well. The walking sequence 3945 is shown in the following animation:



With the functions implemented in the notebook, there are also further analyses possible:

- Plot the average kinematics for one action (e.g. walking)
- Plot the average kinematics for multiple or all actions
- Plot the average kinematics of all ID/OOD sequences for a given action
- Plot the average kinematics for all ID/OOD sequences

These analyses did not lead to many further insights, as the blending of sequences into one did not lead to a clear image. Maybe performing a Fourier Transformation on single motion

sequences will lead to further insights into analyzing the classification split of ID and OOD sequences. Using velocity and acceleration as features for training the OOD detection model did not lead to improved results.

Thus, the velocity and acceleration analysis provide a good visual feedback about why an OOD sequence is considered as such by the model.

Outlook / Future steps

In the following, it is highlighted, how our results can be used by QualityMinds to improve their motion prediction models.

Expanding Analysis to Different Datasets

The current project was using H3.6M dataset for the analysis of human motions. Now, this analysis can be broadened to other datasets, to include AMASS, 3DPW, and ExPI datasets. This broader examination will provide insights into dataset variations and challenges, enhancing the robustness of HMP models.

A fourier transformation for the motion sequences could also be considered to analyze the reasons for the classification of motion sequences as ID and OOD sequences.

Enhancing Motion Prediction Model

To improve motion prediction model quality, two approaches are to be explored:

- **Regularizing the Loss Function:** Various regularization techniques will be investigated to penalize incorrect predictions of ID and OOD sequences differently. This fine-tuning will increase sensitivity to OOD sequences while maintaining ID prediction accuracy.
- **Introducing a New Layer:** A novel layer will be integrated into the model architecture, prioritizing the correct prediction of ID over OOD sequences. This will enhance the model's handling of challenging cases.

Utilizing OOD Representation in Project Branches

The next step is to apply the analysis done in this project to the other branches of the Human Motion Prediction Project branches:

- Adapt the adversarial attacks using the knowledge, which of the samples are ID and which ones are OOD sequences
- Enhance the model interpretability
- Support the self learning of the model training with only ID
- Simulate crashes with a focus on motions with usual characteristics.

Acknowledgement

We want to express our gratitude towards QualityMinds in general, and in particular to Edgar, Namrata, Leyong, and Michael for a friendly and respectful communication and a supportive collaboration. We have very much enjoyed working together on this project, having insight into the tools and knowledge you provided for us. The project in itself was very interesting and insightful, also for our further careers, to apply our newly acquired data science skills. And we also appreciated the warm welcome you gave us on our onsite visit. We also want to thank the Constructor Learning team for providing us with this skillset and the support on our way to finish this project. Especially Ekaterina, Adriano, Katherina, and all our instructors that made this possible.
All the best from us,

Alaa, Jonas, and Vincent