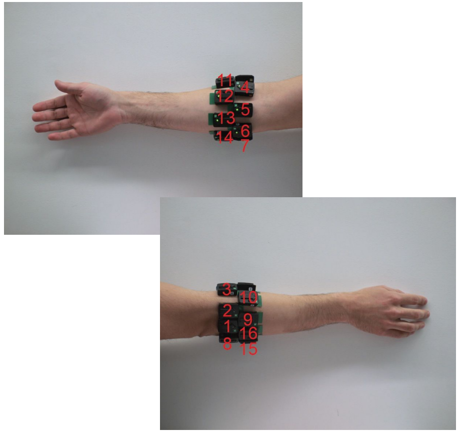
Kinematic Regression from EMG Data

NSSP, Mini Project 3, Alternative 2

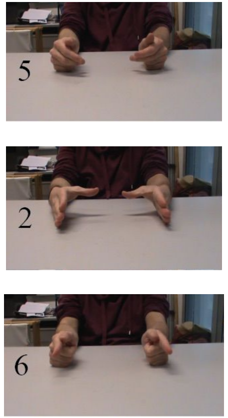
Bär Julian, Calati Veronika, Lee Simon, Merkourios Simos

Mini - Project Outline

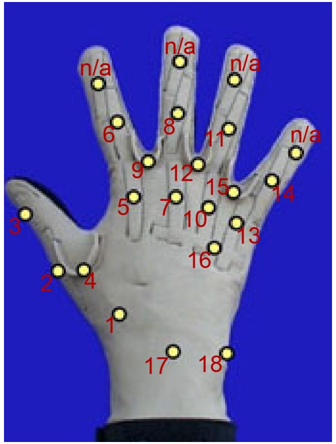
- NinaPro Dataset 8
- EMG, accelerometer regression of digit position



Features (EMG, acc)



Movement data



Labels (glove data)

Data exploration

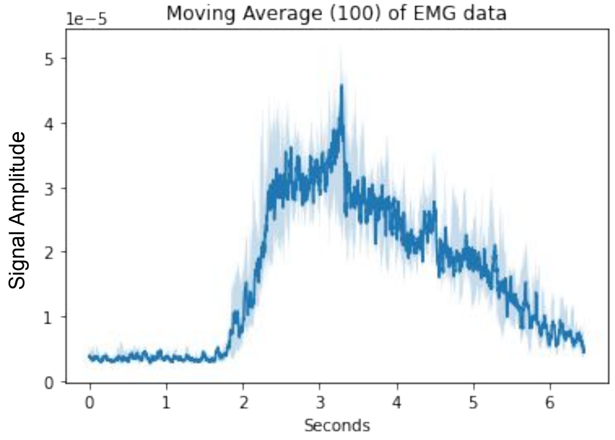
Dataset:

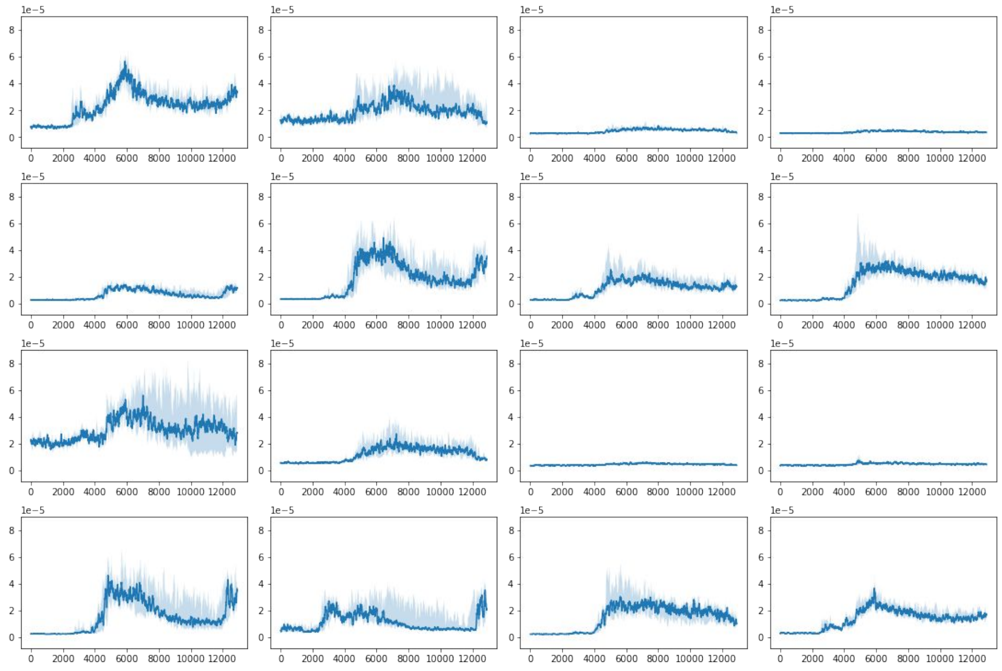
- Subject 1 (able-bodied)
- 9 single-finger and functional movements
- 3 acquisition sessions (10 + 10 + 2 repetitions)

Data exploration

EMG Data:

- 16-channel activity profiles
- Non-negative

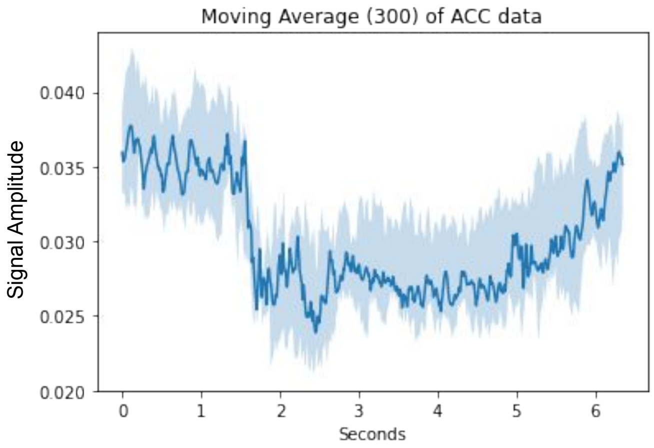


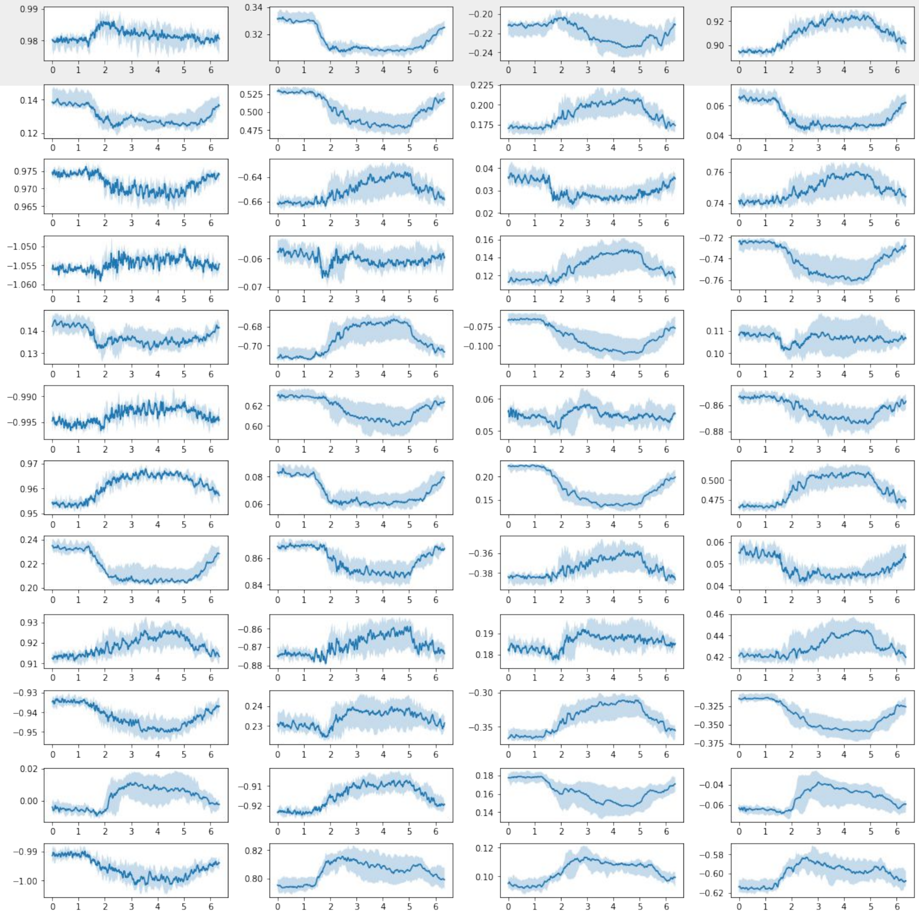


Data exploration

ACC Data:

- 48-channel acceleration profiles

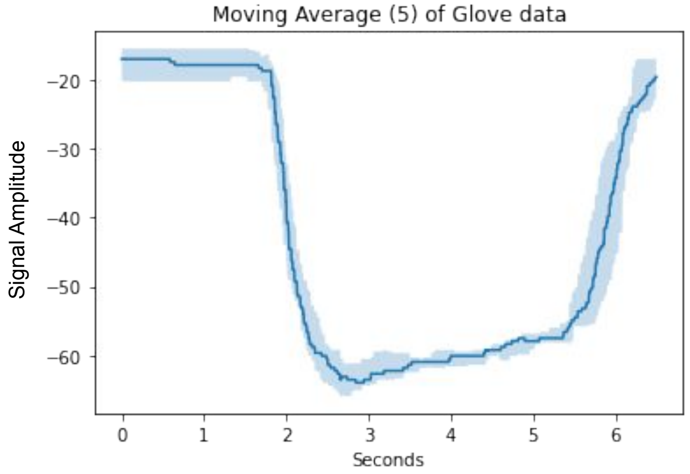


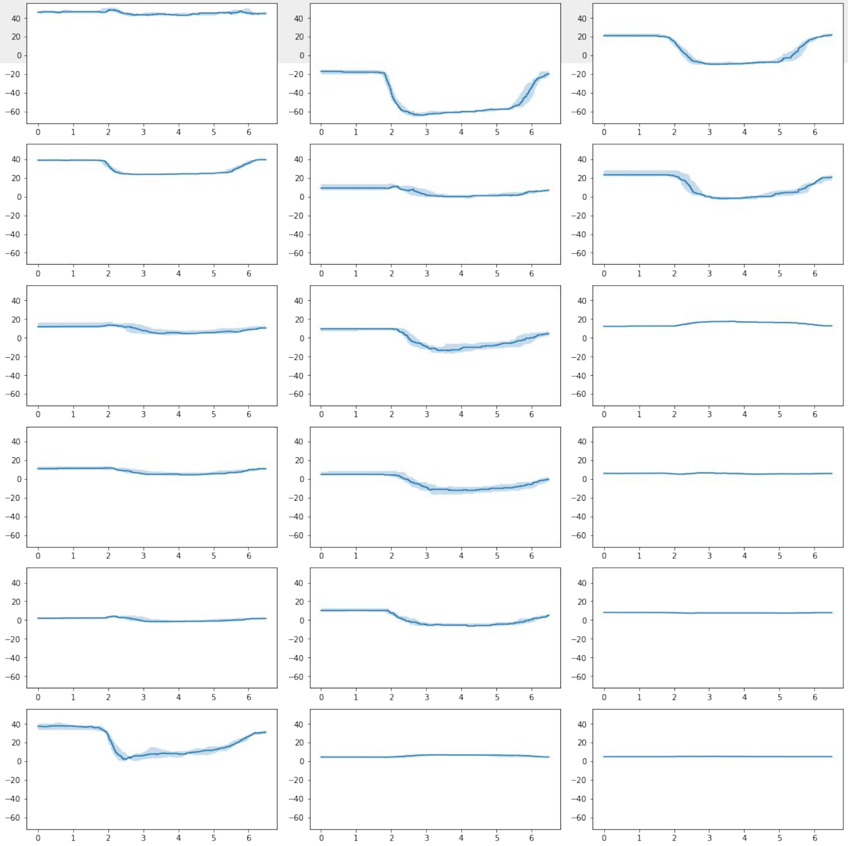


Data exploration

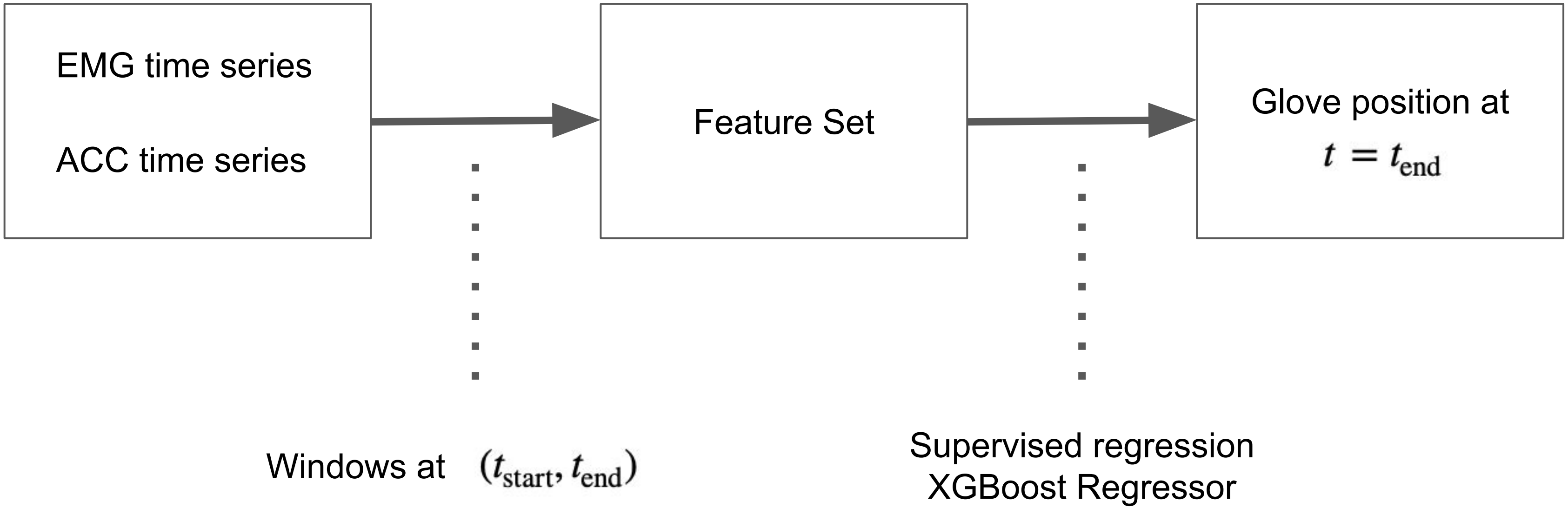
Glove Data:

- 18-channel position profiles



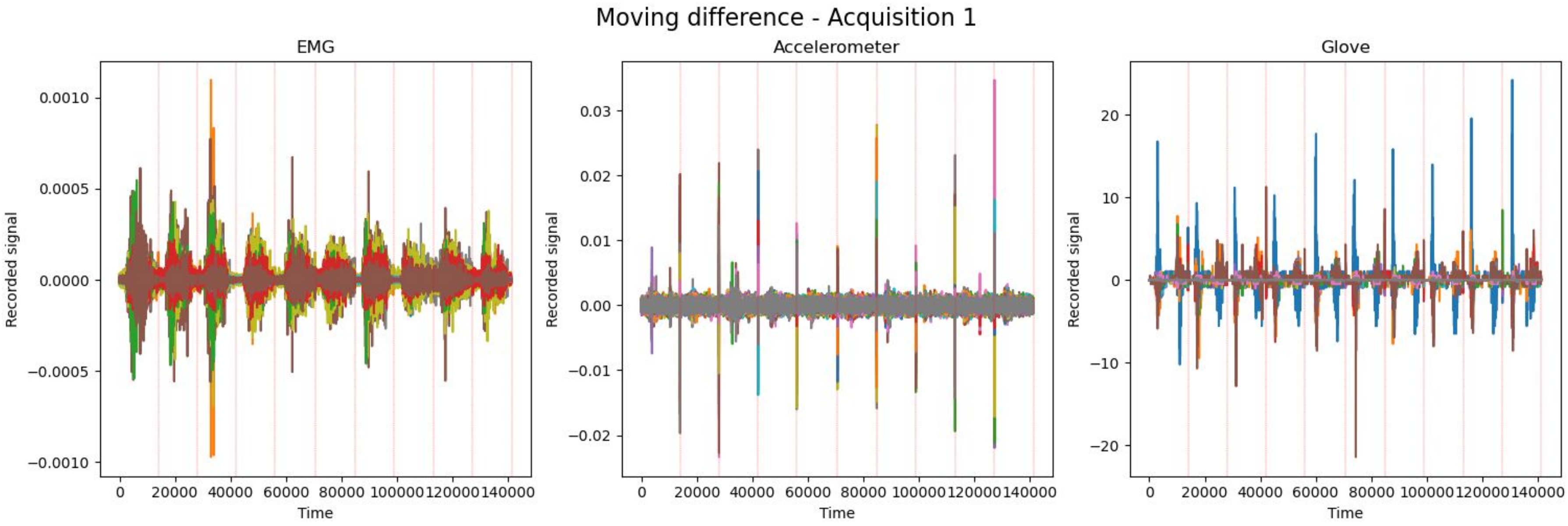


Task Formulation



A note about windowing

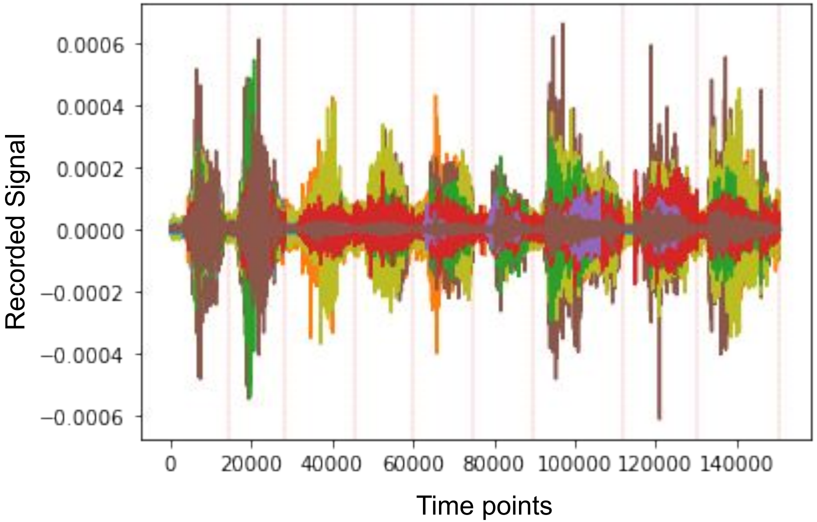
Concatenating time series across repetitions produces big acceleration spikes in $x(t) - x(t-1)$



A note about windowing

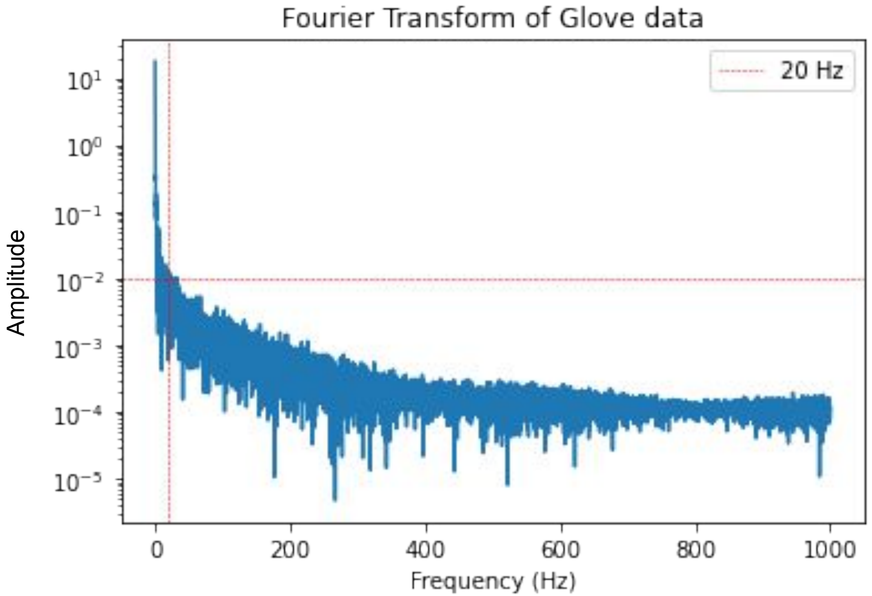
However, concatenating across movements produces no spikes

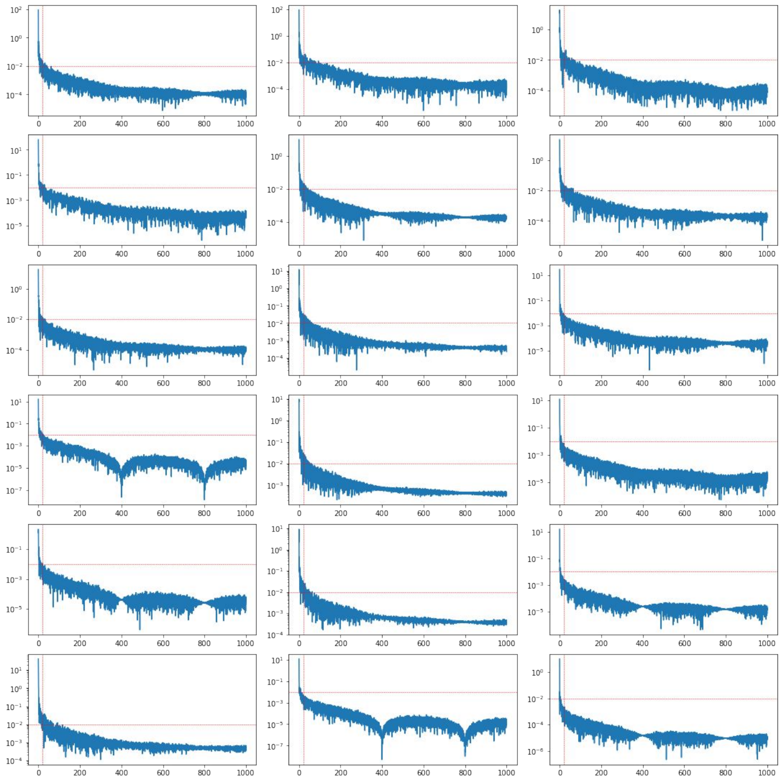
ACC



We chose to avoid computing windows across concatenation points

Window length: lower bound





- Coefficient drops below 0.01 after 20 Hz
- No gain for windows smaller than 50ms

Window Selection

TIME WINDOW

- Lower bound: 50 ms (Fourier Transform)
- Upper Bound: 200 ms (Motor latency)

NUMBER OF WINDOWS

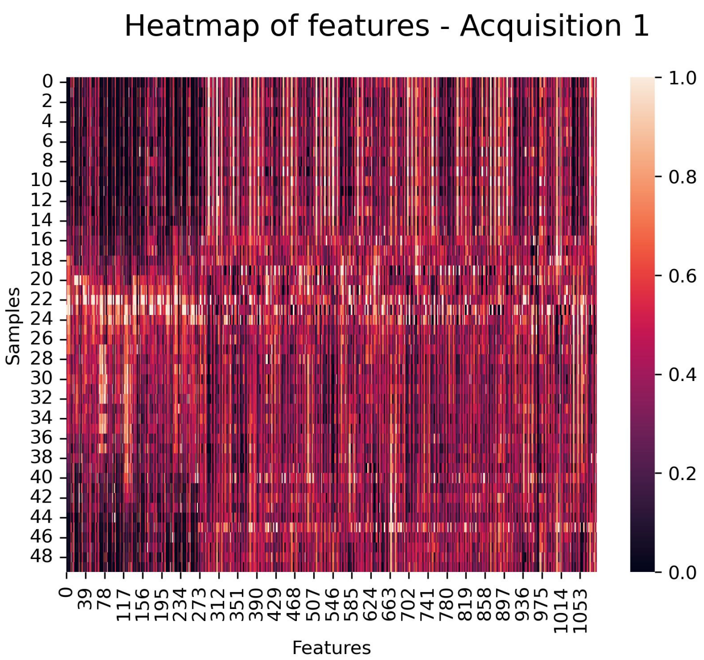
- 50 per repetition (good trade-off accuracy/computation time)

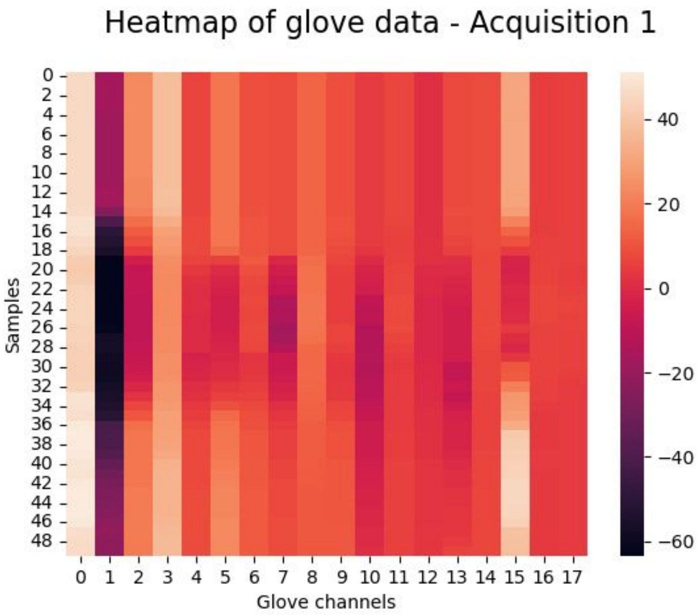
Feature Extraction

- Recommended features: 17 time, frequency, time-frequency features
- Full feature set per EMG and ACC channel
- Features normalized and checked for NaN values
- NaN columns different for each channel, difficult to just remove
- All NaN values were a result of constant-valued features (max = min)
- Therefore, set to 0.5 without loss of information

Feature Extraction and Labels

- Concatenated EMG and ACC features
- labels from glove data: end of the window

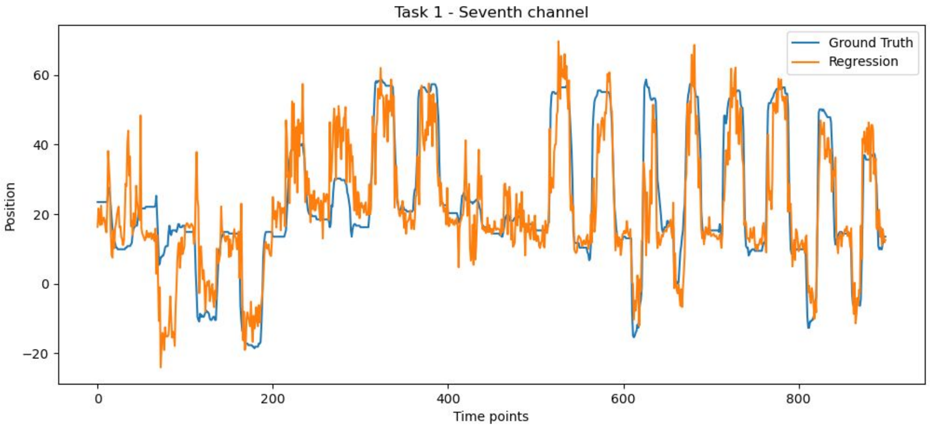


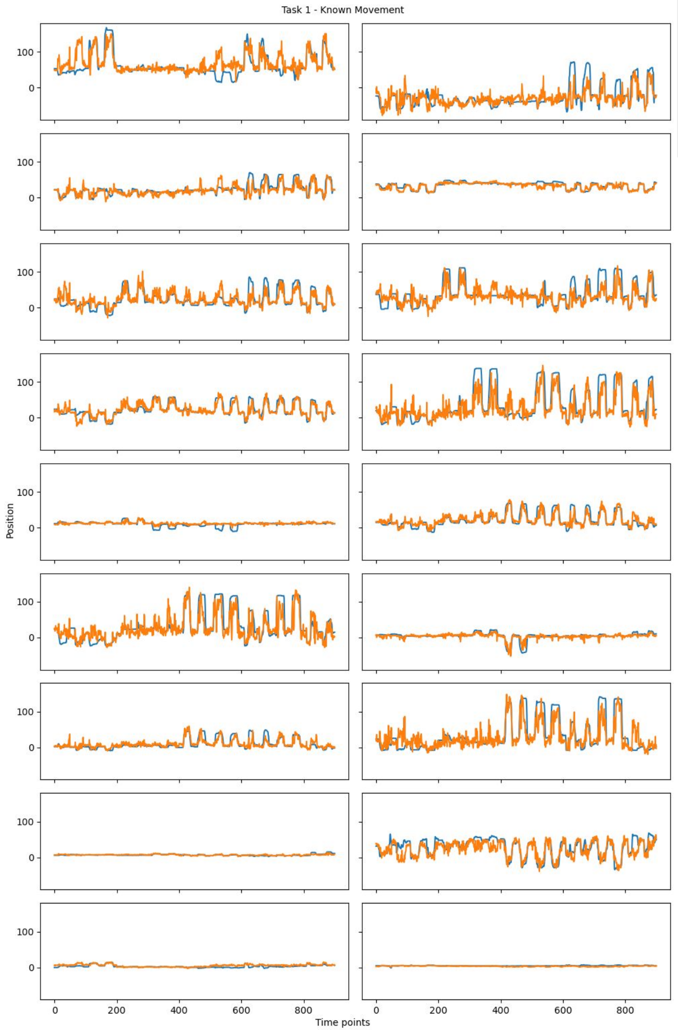


Task 1 - Window Selection

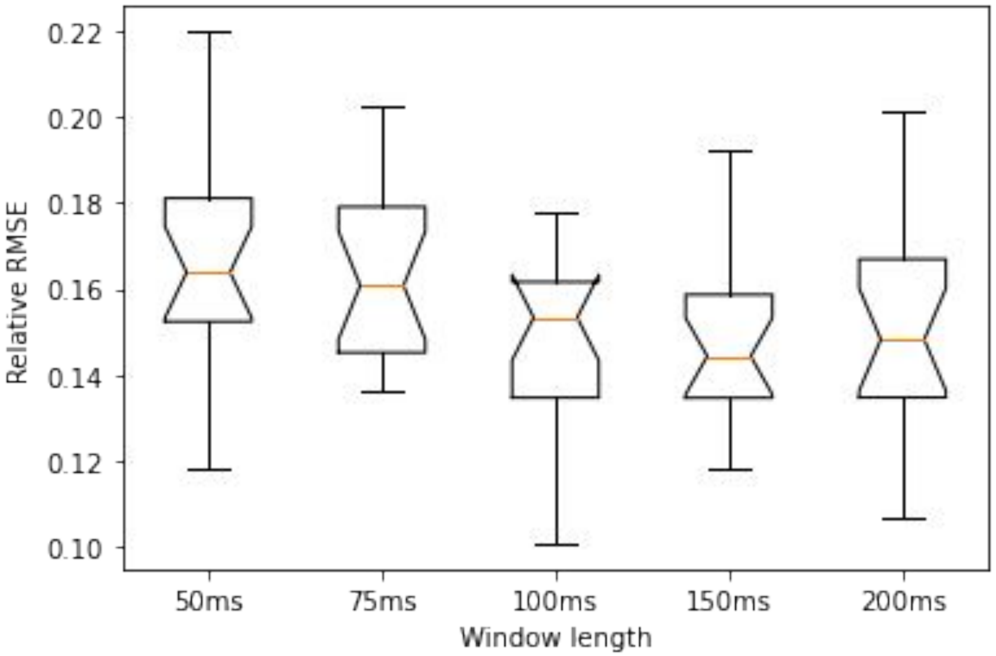
Training: Acquisitions 1 and 2 all movements

Testing: Acquisition 3 all movements





Task 1 - Window Selection



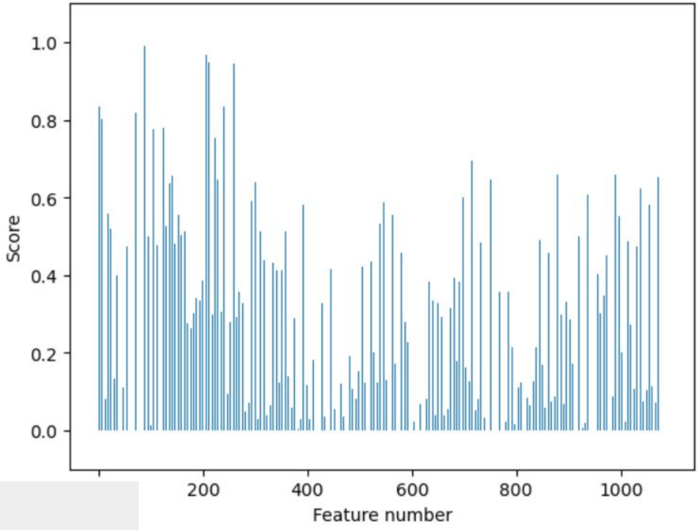
- RMSE computed relative to channel variation, averaged across channels

$$\text{RelRMSE} = \frac{\text{RMSE}(y_{\text{test}}, y_{\text{pred}})}{\max(y_{\text{test}}) - \min(y_{\text{test}})}$$

- selected window: 150ms

Feature Selection

- Feature window length at 150ms
- sklearn.SelectKBest using mutual information

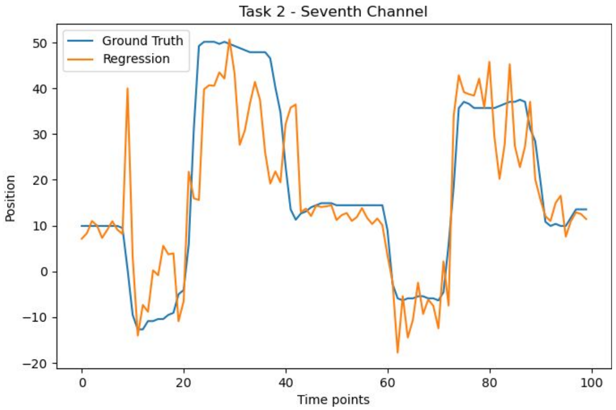


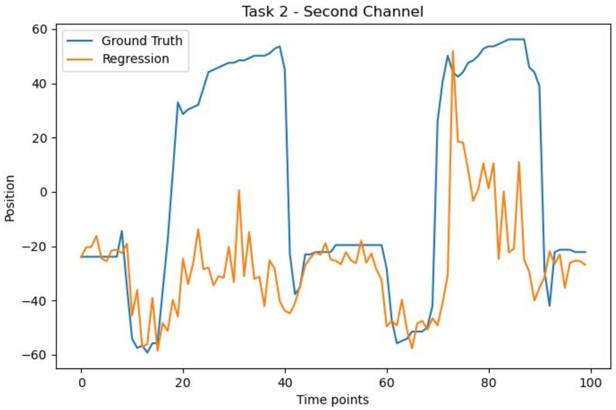
Threshold	0.1	0.2	0.3	0.5	0.7
RMSE	15.90304	15.46936	15.26376	15.43300	19.99233

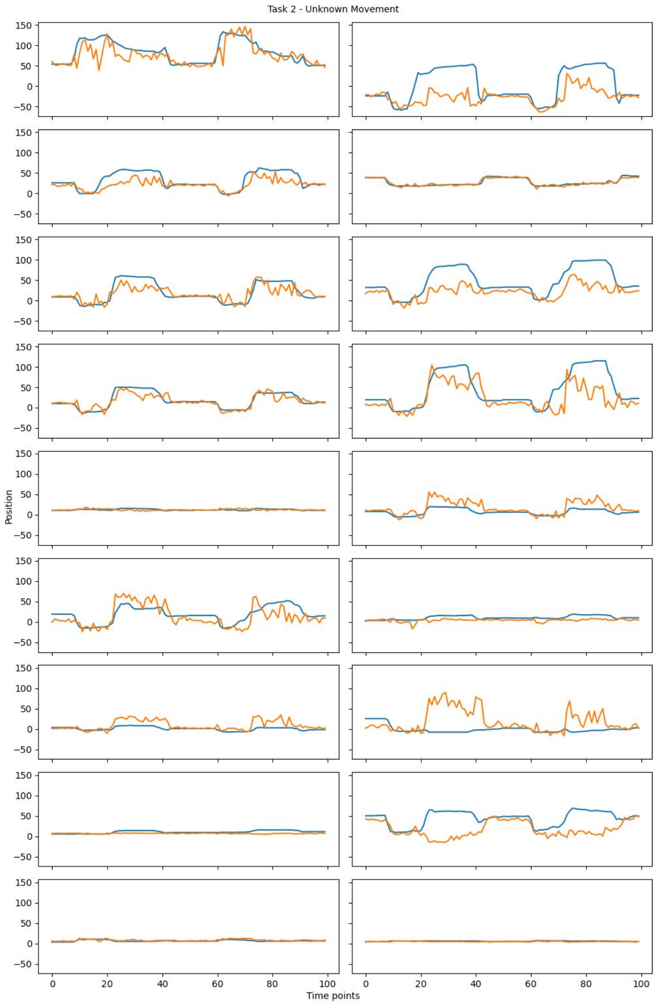
Task 2 - Novel Movement

Training: Acquisitions 1 and 2, movements 1-8

Testing: Acquisition 3, movement 9



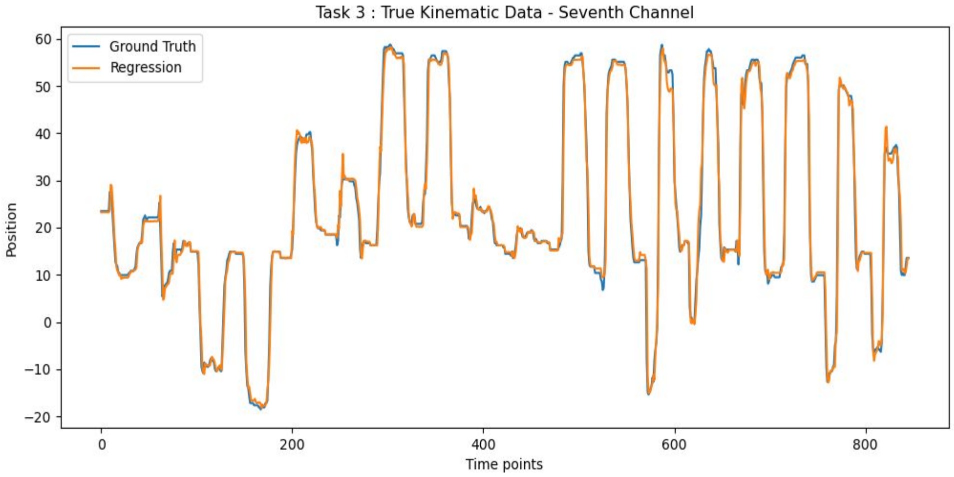


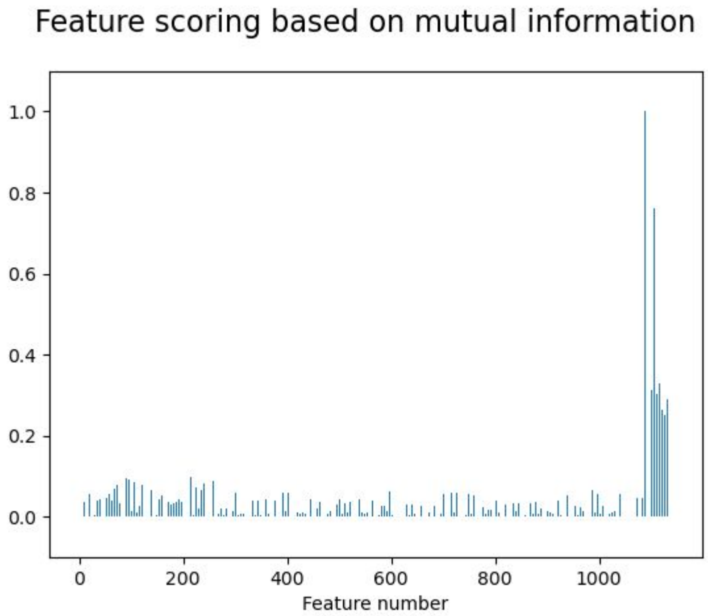


Task 3 - True Kinematic Data

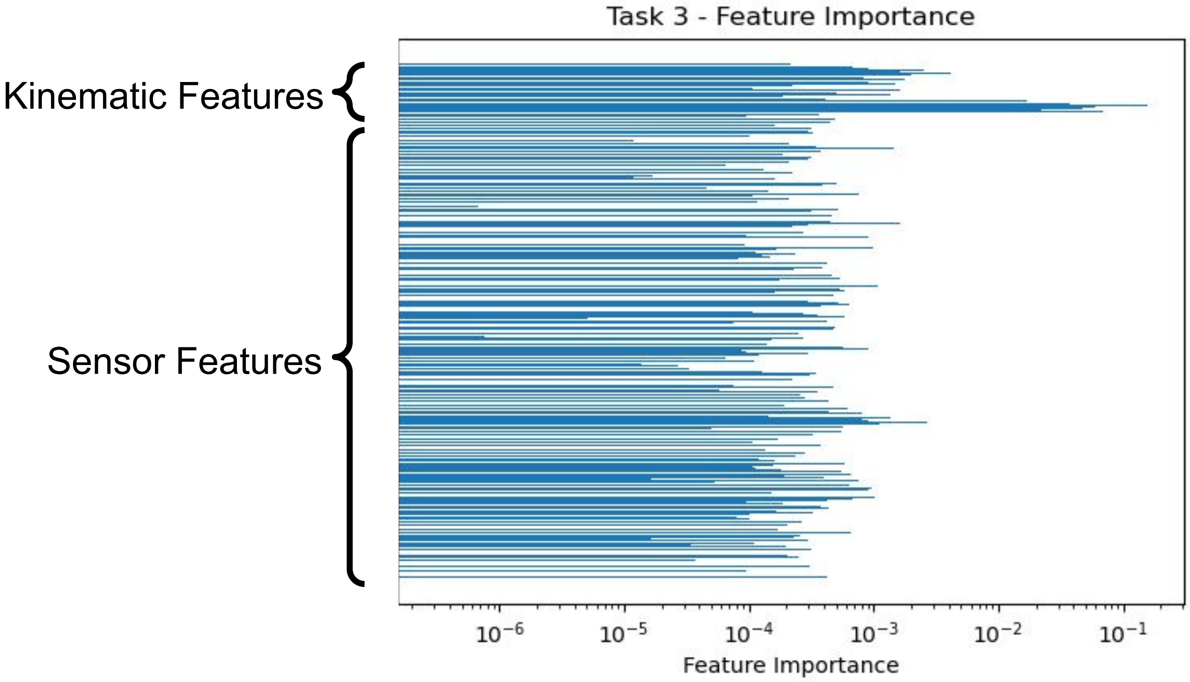
Training: Acq. 1 and 2, all data + last 3 positions

Testing: Acq. 3, all data + last 3 positions





Task 3 - Feature Importance

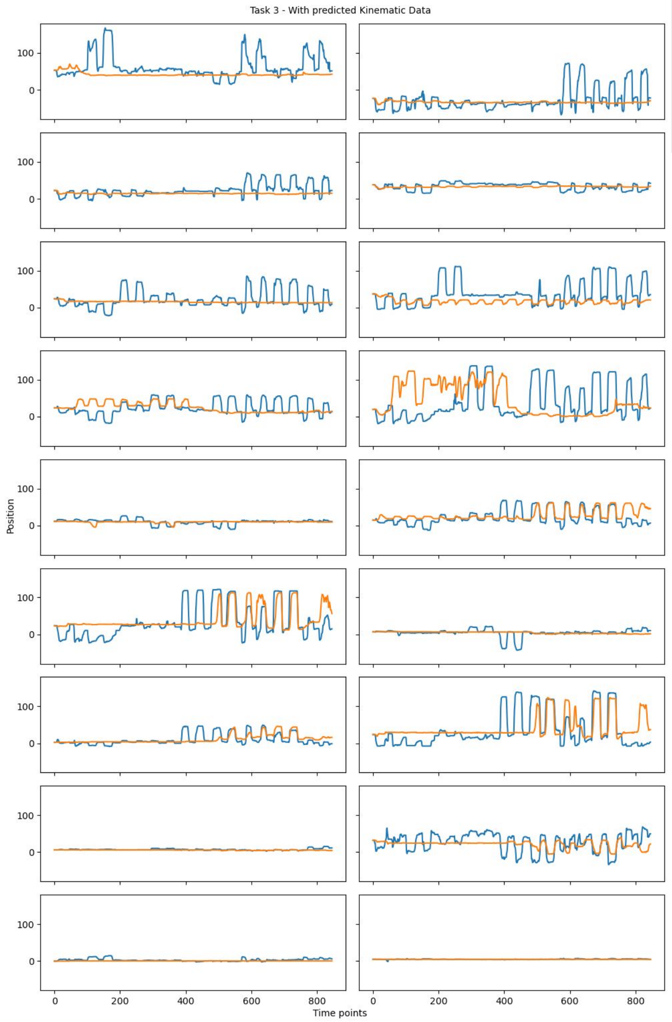


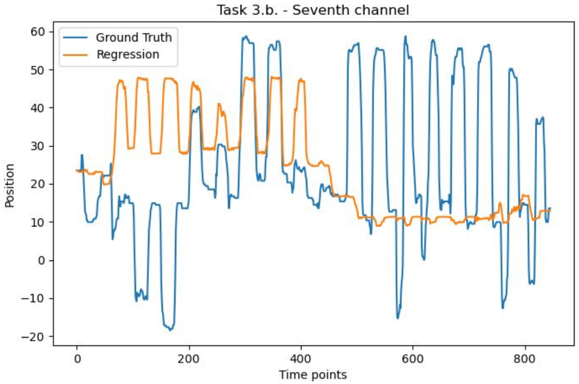
Kinematic features dominate!

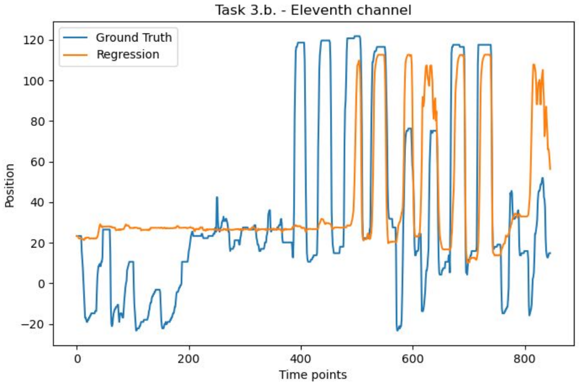
Task 3 - Predicted Kinematic Data

Using the previously trained model

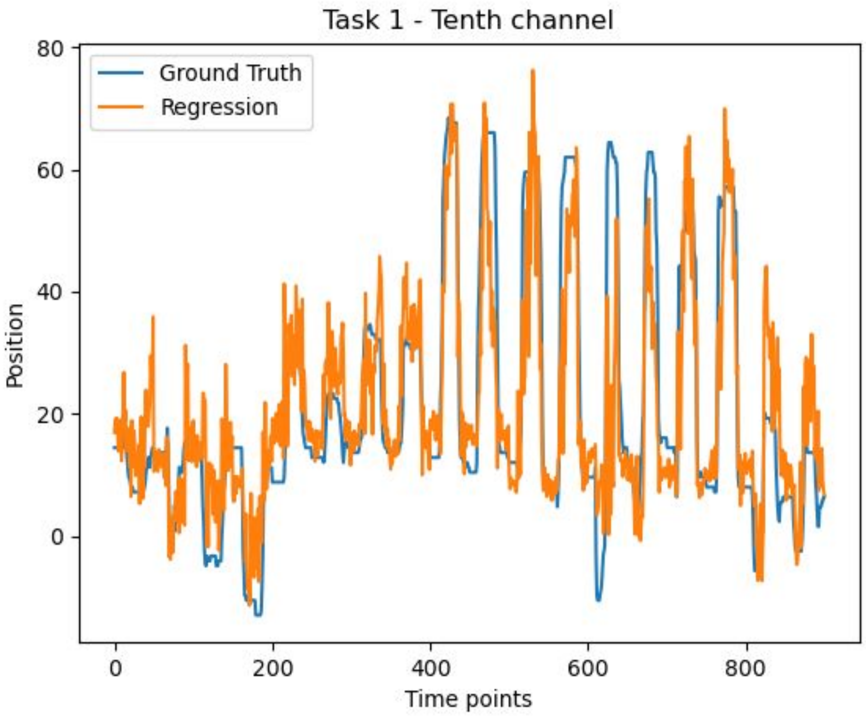
Testing : Acq. 3, all data + last 3 prediction results

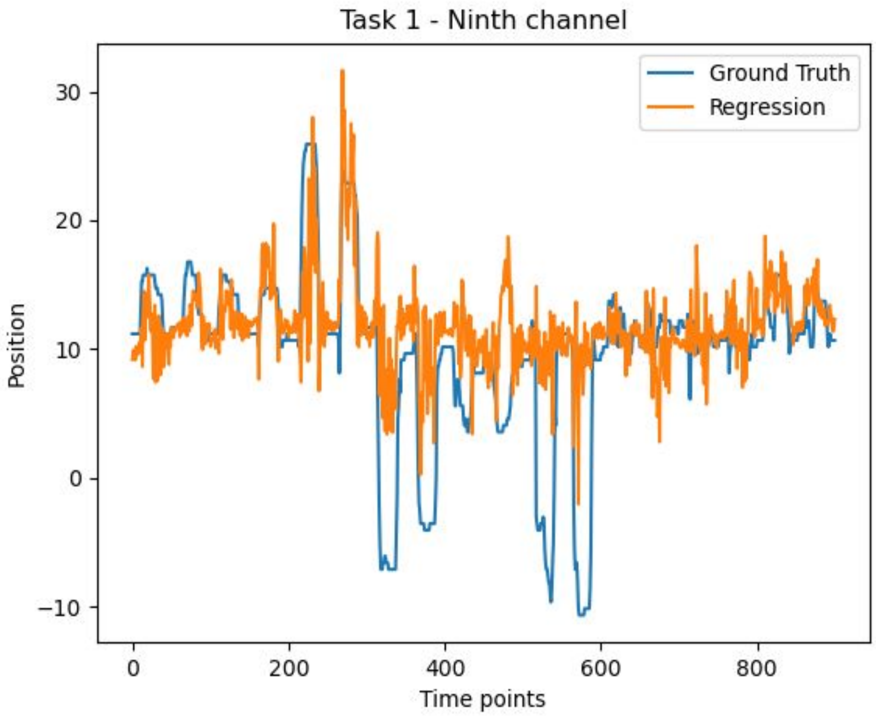






Task 4 - Comparison between glove trackers





Conclusion

- Good performance when movement is known.
- Poor performance when movement is unknown.
- Near-perfect performance with ground truth kinematic data.
- Very poor performance with predicted kinematic data

M

Why did we choose relative RMSE to select the time window length, but RMSE to choose the feature scoring threshold?

‹ 2 / 2 ›

The choice of using relative RMSE to select the time window length and RMSE to choose the feature scoring threshold is based on the goal of each step in the process.

Relative RMSE is used to select the time window length because it allows us to compare the performance of the model on different time window lengths on a relative scale. This is useful when trying to determine the optimal time window length because we want to select the length that results in the lowest error relative to the other lengths tested.

On the other hand, RMSE is used to choose the feature scoring threshold because it provides a measure of the overall error of the model. In this case, we are not concerned with comparing the performance of the model on different thresholds, but rather with selecting the threshold that results in the lowest overall error. Therefore, using RMSE as the metric allows us to directly compare the error of the model at different thresholds and select the one that results in the lowest error.

M

Why should we not normalise time series data when doing time series forecasting?

There are several reasons why normalizing time series data may not be advisable when doing time series forecasting. One reason is that normalizing the data can sometimes remove important information or patterns that may be useful for forecasting. Time series data often contains trends, seasonality, and other patterns that can be useful for making accurate forecasts. Normalizing the data can sometimes remove or distort these patterns, making it more difficult to accurately forecast the future values of the time series.

Another reason to avoid normalizing time series data is that it can sometimes introduce additional noise into the data. Normalization is a mathematical transformation that can sometimes amplify small variations or noise in the data, which can make it more difficult to identify the underlying patterns and trends. This can lead to less accurate forecasts, and in some cases, it can even lead to forecasts that are completely incorrect.

Finally, normalizing time series data can also make it more difficult to compare the forecasts with the actual values. Time series data is often used to monitor and evaluate the performance of forecasting models, and it's important to be able to directly compare the predicted values with the actual values. Normalizing the data can sometimes make it difficult or impossible to directly compare the predicted and actual values, which can make it more difficult to evaluate the accuracy of the forecasts.

In summary, normalizing time series data can sometimes remove important information and patterns, introduce additional noise, and make it more difficult to compare the forecasts with the actual values. These factors can make it more difficult to accurately forecast the future values of a time series, and for these reasons, it's generally not recommended to normalize time series data when doing time series

M

Why did we use equally spaced windows for our features samples instead of a sliding window sampling method?

Using equally spaced windows for our feature samples allows for a consistent and systematic approach to feature extraction. It allows for easy comparison and analysis of the extracted features. Additionally, using a sliding window sampling method can result in overlapping or redundant data, which can increase the complexity and computational cost of the analysis.