
# Task 3

Group-4

IAM Institut für Allgemeine Mechanik | RWTH AACHEN UNIVERSITY

# Team Introduction

**Group 4 Members**

1. Yash Mohta
2. Amal krishna A S
3. Zubin Zaheer
4. Saravana Kumar Krishna Kumar
5. Steffen Tillmann

**Responsibilities**

Data Preprocessing:

Steffen Tillmann, Amal Krishna A S, Saravana Kumar Krishna Kumar

Neural Networks:

Zubin Zaheer, Yash Mohta

**Task**

Use acceleration data from Smartphone 2 with 2-,5- and 10-fold cross validation

# Overview / Timeline

**Goal**: To differentiate between normal and impared walking using Machine Learning

Week 1

Week 2

Week 3

Week 4

### Data Collection

Data collection was done using Phyphox Application

### Data Preprocessing

1. Collection and Filtering the data to remove noise
2. Rotation of data
3. Segment trial into sample
4. Resample to same frame number
5. Check and remove malicious sequence

### Neural Network

Comparing Random Forest and CNN

### Optimization and Result

Tuning the parameters of ML algorithm

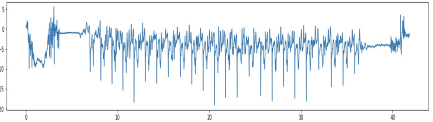
# Packages used

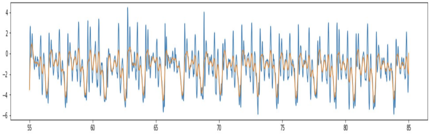
| For the project, following packages were used: | |
|---|---|
| 1. Tensor flow + Keras<br>2. Panda & Numpy<br>3. Scipy<br>4. Sklearn<br>5. Matplotlib (Plotting of results) | Neural Networks<br>Data Preprocessing<br>Signal processing<br>Classification models<br>Plotting of results |

# Data Preprocessing

## Filter Noise and Extract Motion sequence

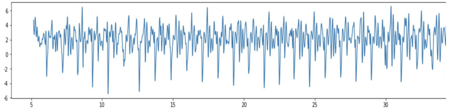
A single Butterworth filter with a cutoff frequency of 10 Hz is used to reduce a large component of noise from the data.
Extraction of the motion sequence is done by the distance between the peaks in the data.





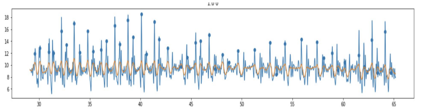
## Rotation of Data

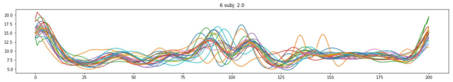
Based on the average mean of oscillations we were able to measure the acceleration due to gravity in the y-axis and rotate the data accordingly

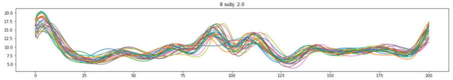


## Segment trail and resampling

Low cut off frequency filter was applied so only a sinusoidal oscillation remains. The peaks were determined and the signal was then split based on these peaks, as two peaks make one step of the walking sequence. This was performed based on the data of the y-axis. Then, the segmented data was resampled to 200 data points.

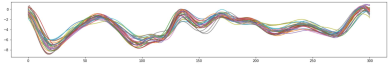


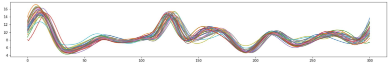


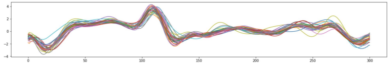


## Checking and removing malicious sequence

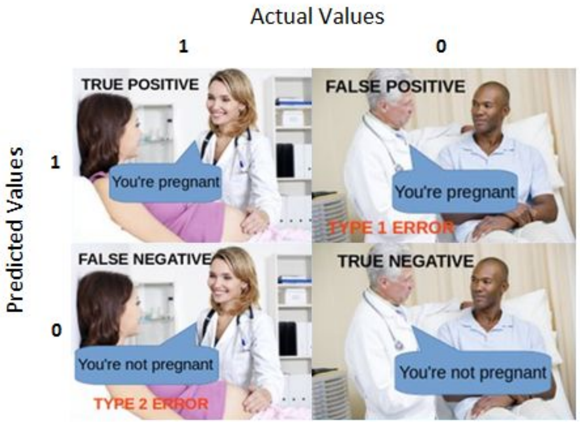
Steps that deviate too much from the mean of all steps from a walking sequence were thrown out.







# CONFUSION MATRIX - KFOLD + Random Forest Classifier




Source: TowardsDatascience.com

Two parameters choose for model study

1. Accuracy : (tp + tn) / N
2. Misclassification rate: (fp+fn) / N

tp: True Positive
tn: True Negative
fn: False Negative
fp: False Positive

| Input Array Shape | 11959 * 600 ( Resample length: 200 * 3 Dimensions | | |
|---|---|---|---|
| | Accuracy (μ, σ) | Misclassification rate (μ, σ) | F - score |
| K – 2 fold | 0.95 ,0.004 | 0.04, 0.004 | 0.96 |
| K – 5 fold | 0.96, 0.003 | 0.03, 0.003 | 0.97 |
| K – 10 fold | 0.96, 0.005 | 0.03, 0.005 | 0.96 |

Note: The above parameters were determined from the confusion matrix based on record wise sampling to determine effectiveness of our classifier

# Tuning Random Forest

- Used GridSearch CV for getting optimum hyper-parameters.
- Optimum Hyper parameter were overfitting the test data.
- Hence we finally went with a simpler model of the classifier (no of trees =100)

```
{'bootstrap': False,
 'max_depth': 10,
 'max_features': 'auto',
 'min_samples_leaf': 3,
 'min_samples_split': 2,
 'n_estimators': 100}
```

**Optimum parameters from GridSearch**

```
forest = RandomForestClassifier(n_estimators=100, criterion='gini',bootstrap=False, max_features='auto')
```

**Classifier settings used**

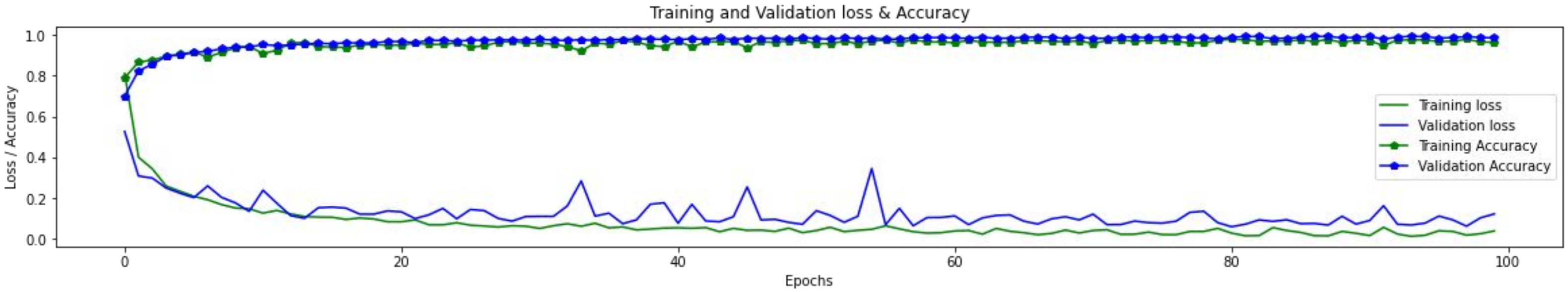
# Dense Neural Network

## Model of the Keras Dense Neural Network

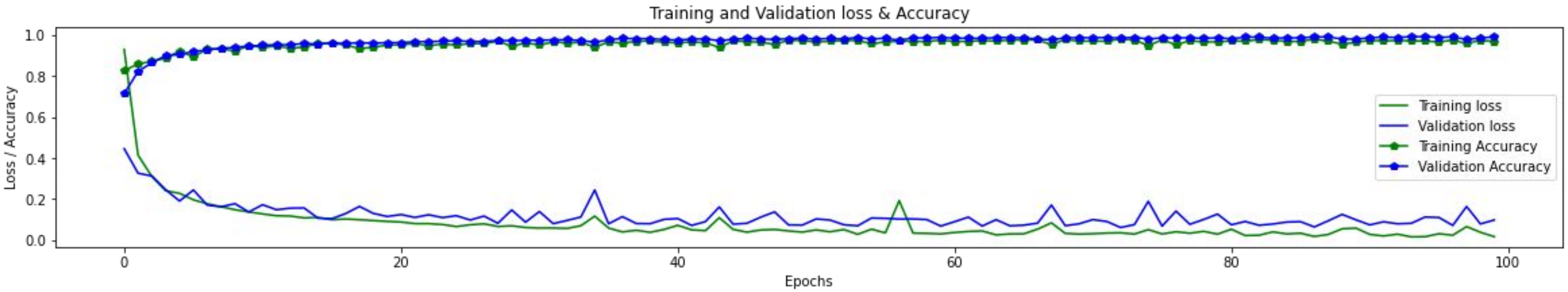
```
model = tf.keras.models.Sequential([
tf.keras.layers.Flatten(input_shape=(600,)),
tf.keras.layers.Dense(1000, activation ='relu'),
tf.keras.layers.Dense(500, activation ='relu'),
tf.keras.layers.Dense(2, activation ='softmax')
])

model.compile(optimizer = 'adam',
          loss = 'sparse_categorical_crossentropy',
          metrics = ['accuracy'])
```

# Neural Network - Validation (keras)

Option A:
Dense 500 +
Dense 100




Option B:
Dense 1000 +
Dense 500




Based on this information, it was decided to go ahead with Option B with 2 Dense layers of 1000 and 500 neurons each as computational time was similar in both cases. Option A had more variation compared to Option B.

# Neural Network - KFold Validation

## Accuracy of the dense neural network based on the number of folds

| | Accuracy in % |
|---|---|
| K – 2 fold | 96.6 |
| K – 5 fold | 97.4 |
| K – 10 fold | 97.2 |

# Conclusion

- Both the Keras Dense Neural Network and Random forest Classifier performed well with the classification task as the train and test splits were done record wise.
- The accuracy that we see here is not actual and is boosted, since during record wise split the ,the model will have seen similar data to the test set during training.

Therefore, both (RF and NN) are good classifiers to differentiate between normal and impared walking as there is not much of a difference in the accuracies obtained.

# Thank you for your attention!