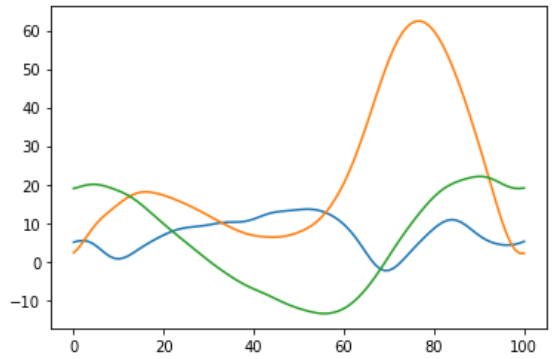
Practical Lab Algorithms in Machine Learning and Their Application

Analysis of Two Gait Datasets



Practical Lab 12.07.2023

Enno Igel, Leo Diedering

universitätbonn

Data and associated Tasks

Data	Tasks

• **physical features** of human gaits
• different **subjects**
• different **conditions**
  • Speed
  • Medical restrictions
	• Classification of **condition**
	  • Learned features might be interesting for therapies
	• Classification of subjects, i.e. **reidentification**
	  • e.g. for surveillance purposes (might be considered less problematic than face recognition)

General Challenges

- Dealing with **time series** data
- High dimensional data
  - → **dimensionality reduction**
- Relatively **small datasets**
  - especially after splitting into train and test data

Outline

- First data set
  - Data summary
  - Dimensionality reduction
  - Classification
    - Condition
    - Subject
  - Using
    - SVM
    - 1D CNN
- Second data set
  - “Larger version” of ReID task
  - Same techniques, some new ideas
- Conclusion

First data set

Multivariate Gait Data
Donated on 12/15/2022

Bilateral (left, right) joint angle (ankle, knee, hip) times series data collected from 10 healthy subjects under 3 walking conditions (unbraced, knee braced, ankle braced). For each condition, each subject's data consists of 10 consecutive gait cycles.

Dataset Characteristics	Subject Area	Associated Tasks
Sequential, Multivariate, Time-Series	Life Science	Classification, Regression, Clustering
Attribute Type	**# Instances**	**# Attributes**
Real, Categorical, Integer	181800	7

Information

For what purpose was the dataset created?
Biomechanical analysis of human locomotion

Who funded the creation of the dataset?
National Science Foundation (#0540834) and Mary .

Additional Information
This dataset is a six dimensional array of joint angle
time points. The data were recored from ten subject
treadmill, walking on a treadmill with a knee-brace
SHOW MORE

Citation Requests/Acknowledgements
Any use of these data should cite the following thre
collected by Shorter et al. (2008), preprocessed by H
...
SHOW MORE

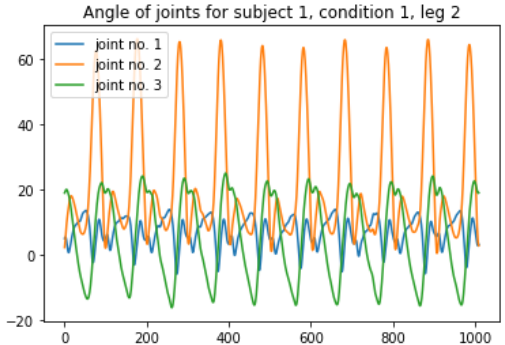
	subject	condition	replication	leg	joint	time	angle
0	1	1	1	1	1	0	4.682881
1	1	1	1	1	1	1	5.073127
2	1	1	1	1	1	2	5.229774
3	1	1	1	1	1	3	5.083273
4	1	1	1	1	1	4	4.652399
...	...	...	...	...	...	...	...
181795	10	3	10	2	3	96	29.778412
181796	10	3	10	2	3	97	29.247559
181797	10	3	10	2	3	98	28.796951
181798	10	3	10	2	3	99	28.432936
181799	10	3	10	2	3	100	28.136438

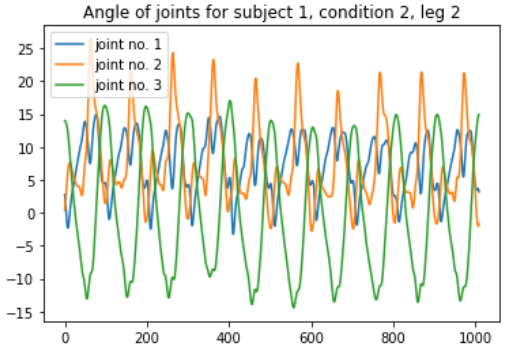
181800 rows × 7 columns

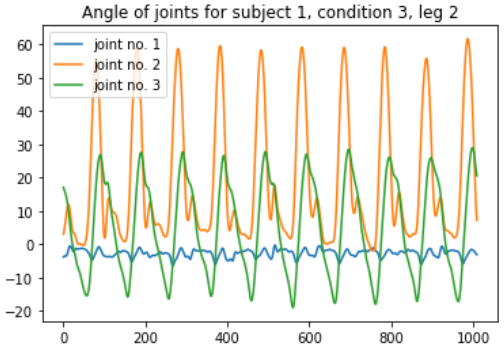
Attribute	Explanation
subject	1,…,10: subject no.
condition	1 = unbraced 2 = knee brace 3 = ankle brace
replication	1,…,10 replication (step) no.
leg	1 = left 2 = right
joint	1 = ankle 2 = knee 3 = hip
time	0,…,100: % gait cycle
angle	Joint angle in degrees (real valued

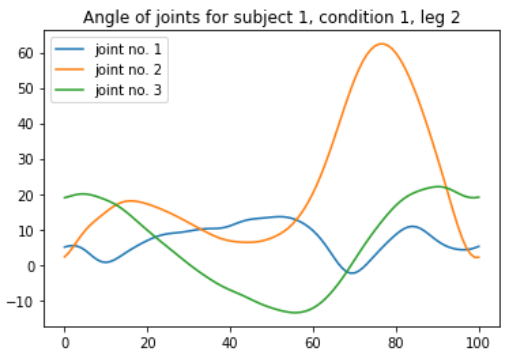
"The data were collected by Shorter et al. (2008), preprocessed by Helwig et al. (2011), and published by Helwig et al. (2016)."

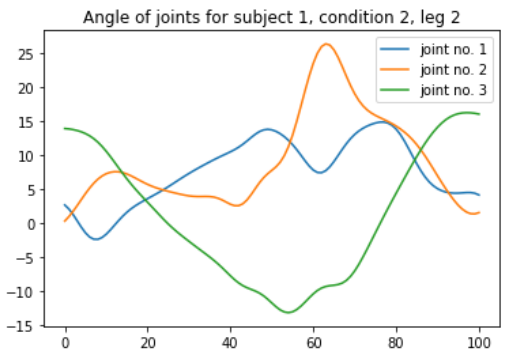
Data introduction

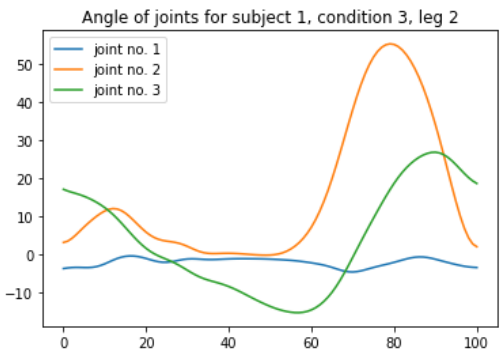












universitätbonn

Processed data format

- X_data has shape:
  - (n_samples, n_features, n_times)

    = (180, 3, 101)

Dimensionality reduction 1

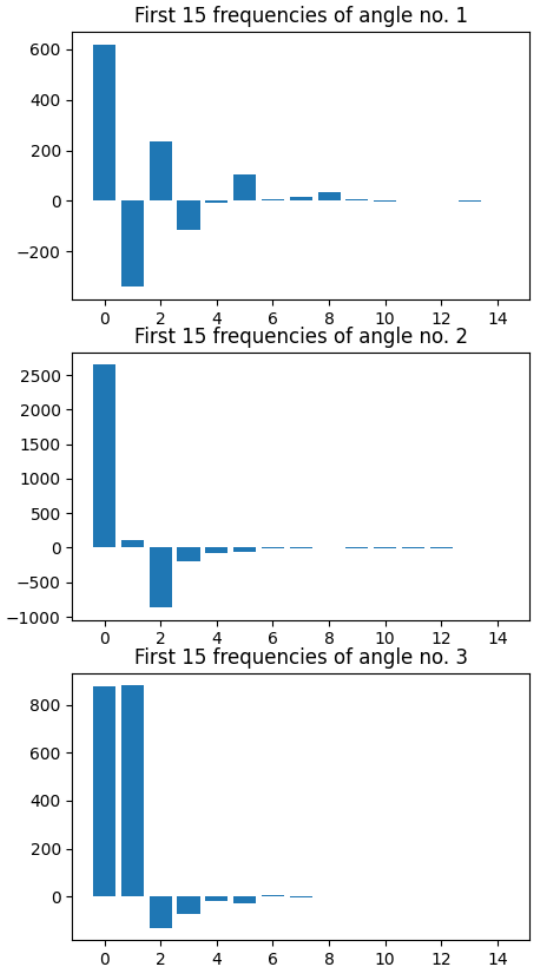
- Periodic time series data

  → FFT (in ptic. RFFT)

- Largest 15 frequencies seem to be sufficient (maybe even less)

- Very different scales

  → normalize



universitätbonn

Dimensionality Reduction 2

- Concatenate normalized frequencies

  → 3*15

- Apply PCA to further reduce dimension
  - For visualization: fit on whole dataset
  - For learning: fit **only** on train data

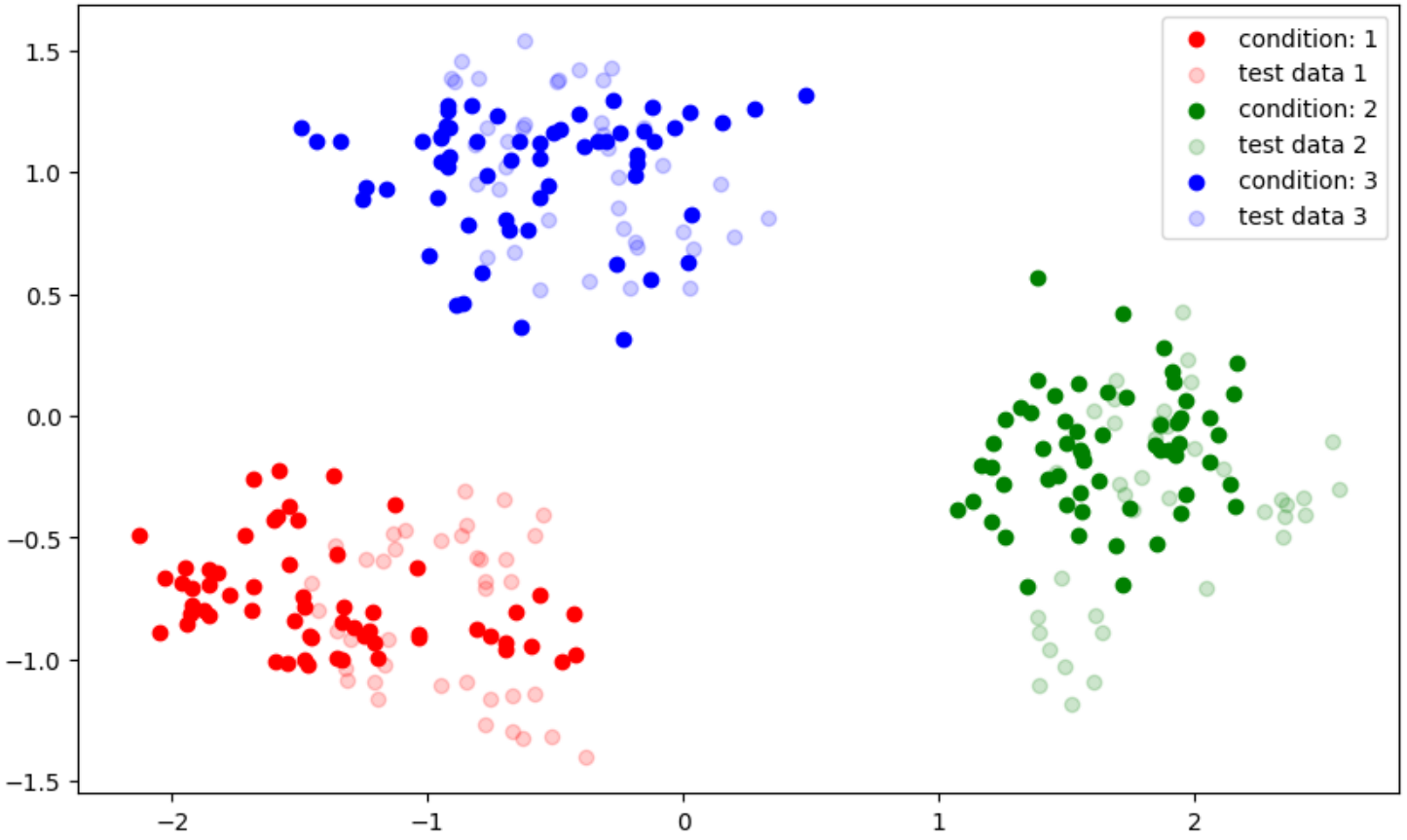
Train-Test-Split for Cond. Class.

- Problem: 10 is a low number of subjects

  → avoid learning condition features **per subjects**

- Therefore, use data from
  - First 6 subjects as train data
  - Last 4 subjects as test data

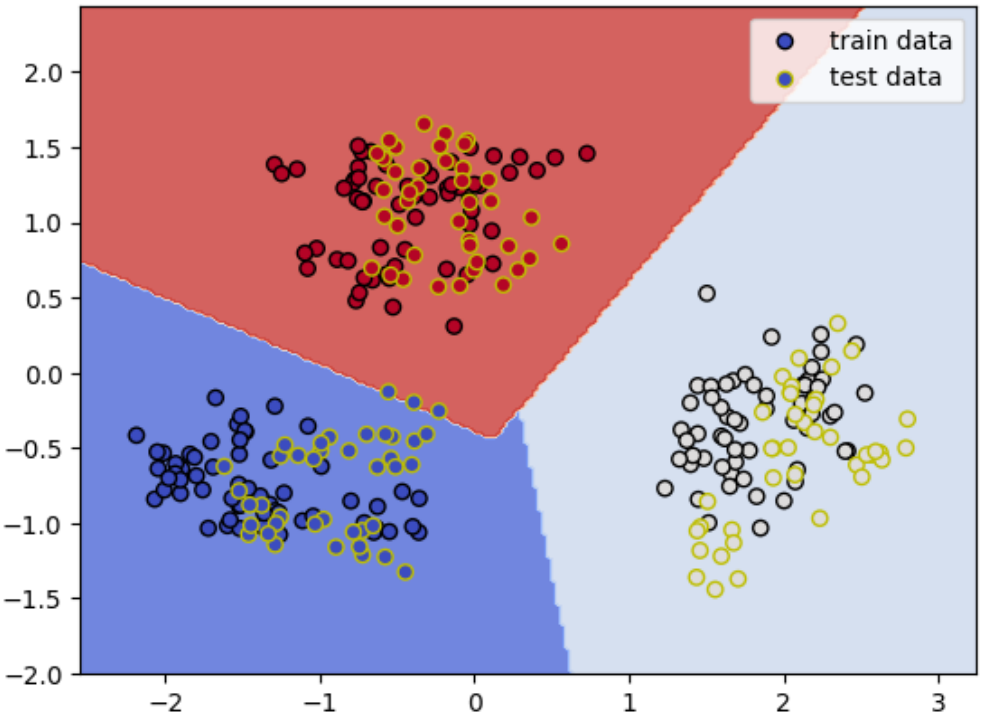
2-dim. PCA result (first 6 freq)



universitätbonn

SVM

- Use first **6 frequencies** as before
- Use PCA result for PCA fitted on **train data**
- Employ **grid search**
  - Use ¼ of train data →
    - C=0.1
    - γ=0.5,
    - kernel = linear
- **Accuracy**:
  - 2-dim: 97,5%
  - 3-dim: 100,0 %



universitätbonn

1D Convolutional NN

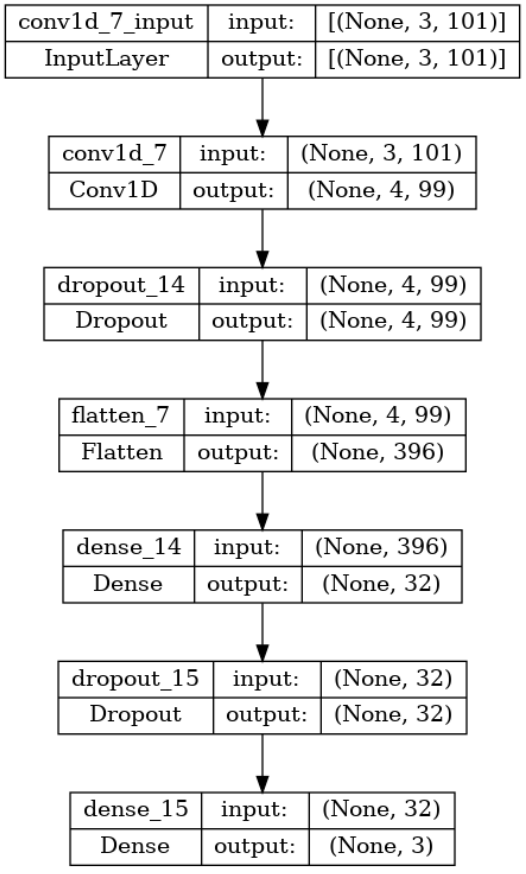
- **Q**: Similar or better results without precomputed features?
- **Idea**: Use 1D CNN to deal with time series data

```
model = keras.models.Sequential()

model.add(layers.Conv1D(4, kernel_size=3, input_shape=X_train[0].shape, data_format = "channels_first"))
model.add(layers.Dropout(0.3))
model.add(layers.Flatten())
model.add(layers.Dense(32, activation="relu"))
model.add(layers.Dropout(0.3))
model.add(layers.Dense(3, activation="softmax"))

model.compile(loss="categorical_crossentropy", optimizer="adam", metrics=["accuracy"])
```

Architecture

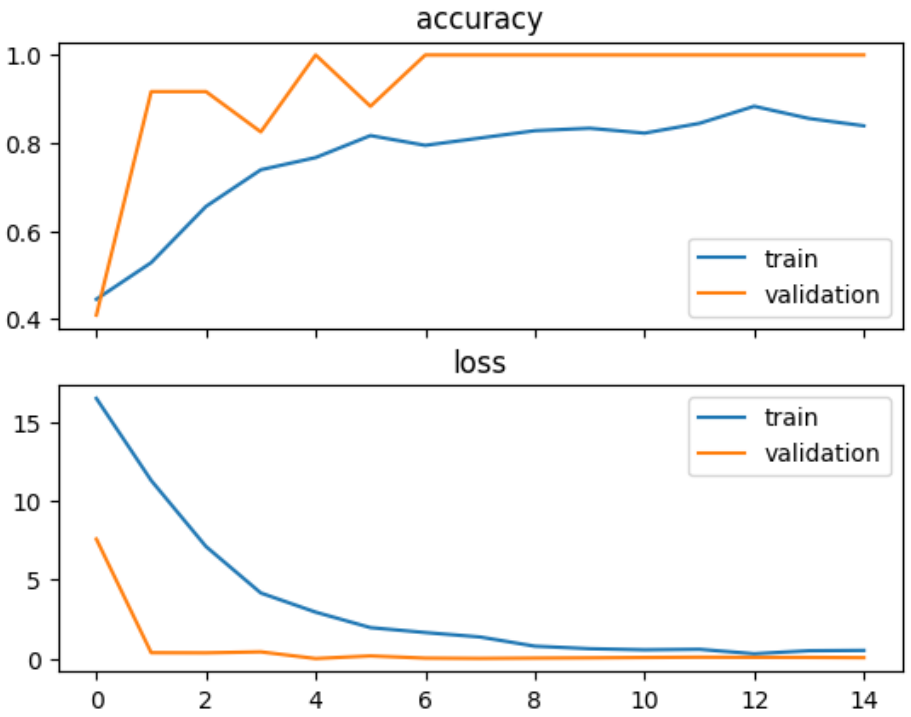
Layer (type)	Output Shape	Param #
conv1d_7 (Conv1D)	(None, 4, 99)	40
dropout_14 (Dropout)	(None, 4, 99)	0
flatten_7 (Flatten)	(None, 396)	0
dense_14 (Dense)	(None, 32)	12704
dropout_15 (Dropout)	(None, 32)	0
dense_15 (Dense)	(None, 3)	99

Total params: 12,843
Trainable params: 12,843
Non-trainable params: 0



universitätbonn

Result

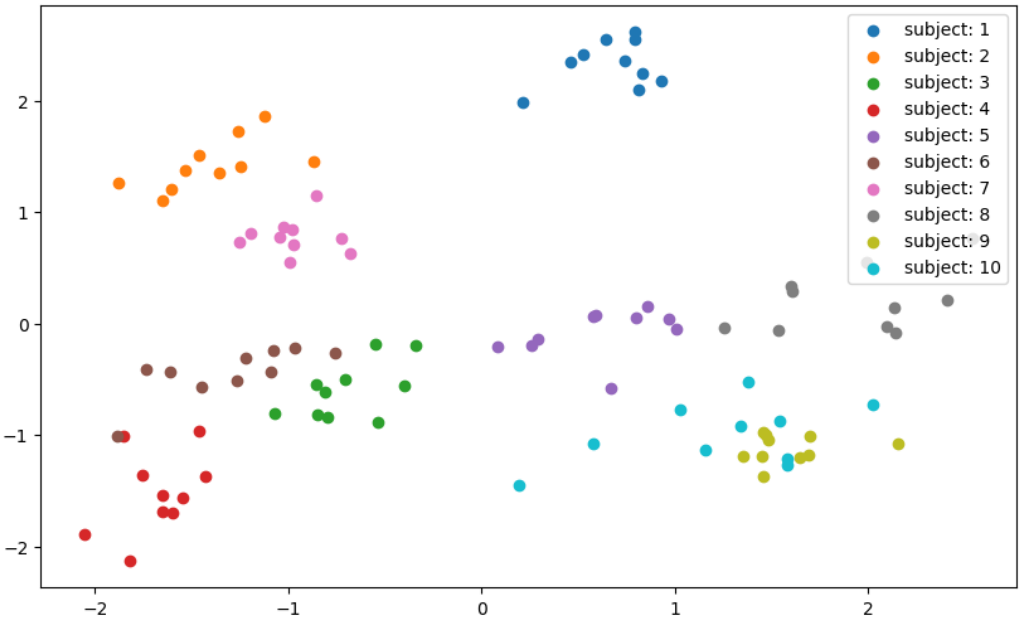


- 100% accuracy after only a few epochs

universitätbonn

ReID Task

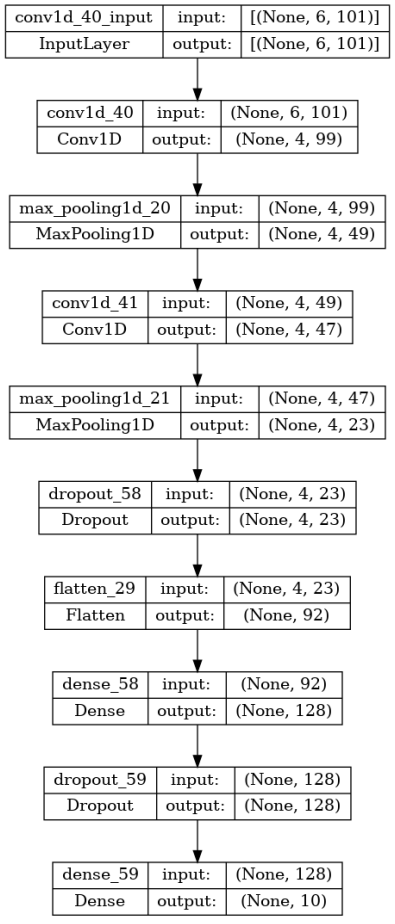
- Train-Test-Split:
  - First 7 steps per subject as train data
  - Last 3 steps per subject as test data
- Precompute features just like before

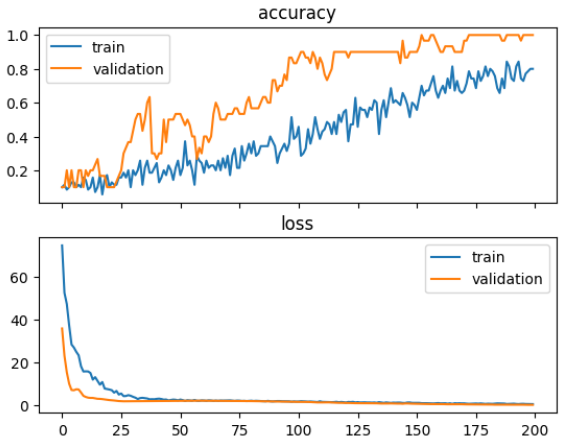


universitätbonn

Short Summary of Results

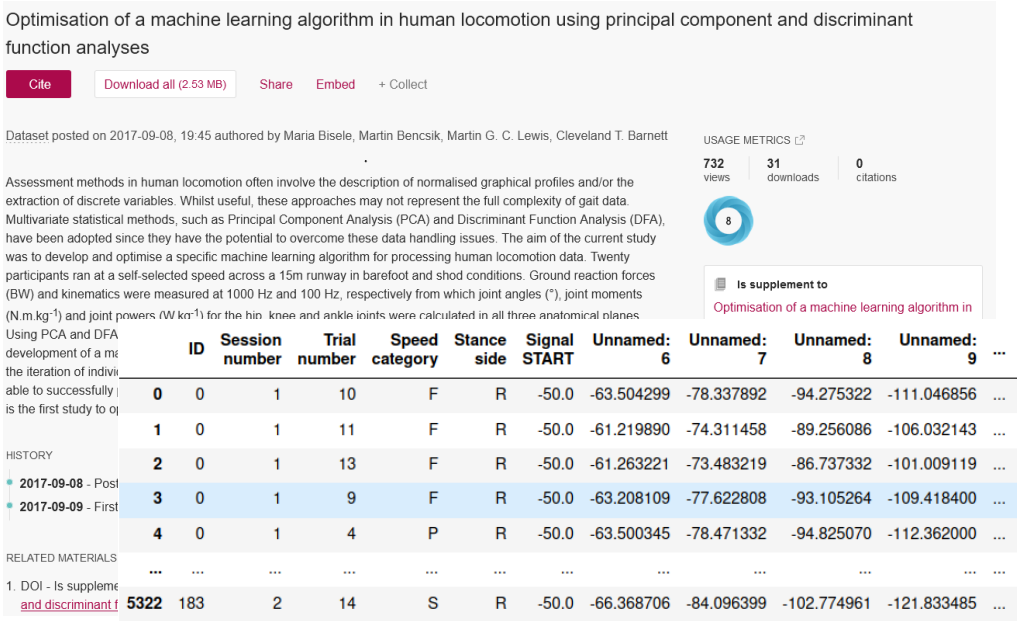
- SVM:
  - 2-dim: 83.3%
  - 3-dim: 96.6%
  - 5-dim: 100.0%
- 1D CNN
  - 2x Conv1D & MaxPooling
  - 100.0% after 200 epochs
- Simple task because of few classes/samples





universitätbonn

Second data set



Optimisation of a machine learning algorithm in human locomotion using principal component and discriminant function analyses

Cite | Download all (2.53 MB) | Share | Embed | + Collect

Dataset posted on 2017-09-08, 19:45 authored by Maria Bisele, Martin Bencsik, Martin G. C. Lewis, Cleveland T. Barnett

Assessment methods in human locomotion often involve the description of normalised graphical profiles and/or the extraction of discrete variables. Whilst useful, these approaches may not represent the full complexity of gait data. Multivariate statistical methods, such as Principal Component Analysis (PCA) and Discriminant Function Analysis (DFA), have been adopted since they have the potential to overcome these data handling issues. The aim of the current study was to develop and optimise a specific machine learning algorithm for processing human locomotion data. Twenty participants ran at a self-selected speed across a 15m runway in barefoot and shod conditions. Ground reaction forces (BW) and kinematics were measured at 1000 Hz and 100 Hz, respectively from which joint angles (°), joint moments (N.m.kg^{-1}) and joint powers (W.kg^{-1}) for the hip, knee and ankle joints were calculated in all three anatomical planes. Using PCA and DFA development of a ma the iteration of indivi able to successfully is the first study to o

USAGE METRICS
732 views | 31 downloads | 0 citations

Is supplement to
Optimisation of a machine learning algorithm in

HISTORY
- 2017-09-08 - Post
- 2017-09-09 - First

RELATED MATERIALS
1. DOI - Is supplem and discriminant f

	ID	Session number	Trial number	Speed category	Stance side	Signal START	Unnamed: 6	Unnamed: 7	Unnamed: 8	Unnamed: 9	...
0	0	1	10	F	R	-50.0	-63.504299	-78.337892	-94.275322	-111.046856	...
1	0	1	11	F	R	-50.0	-61.219890	-74.311458	-89.256086	-106.032143	...
2	0	1	13	F	R	-50.0	-61.263221	-73.483219	-86.737332	-101.009119	...
3	0	1	9	F	R	-50.0	-63.208109	-77.622808	-93.105264	-109.418400	...
4	0	1	4	P	R	-50.0	-63.500345	-78.471332	-94.825070	-112.362000	...
...	...	...	...	...	...	...	...	...	...	...	...
5322	183	2	14	S	R	-50.0	-66.368706	-84.096399	-102.774961	-121.833485	...
5323	183	2	15	S	R	-50.0	-66.631662	-86.734469	-109.992798	-135.498511	...
5324	183	2	16	S	R	-50.0	-62.175174	-76.661237	-93.244695	-111.415776	...
5325	183	2	17	S	R	-50.0	-68.585310	-89.854079	-113.341924	-138.315191	...
5326	183	2	19	S	R	-50.0	-65.396931	-83.588085	-104.188064	-126.383063	...

5327 rows × 305 columns

Attribute	Explanation
ID	1,…,183: subject no.
Session no.	1,2
Trial no.	1,…, 8-10
Speed category	P = Preferred S = Slower F = Faster
Rest	300 time steps of component data

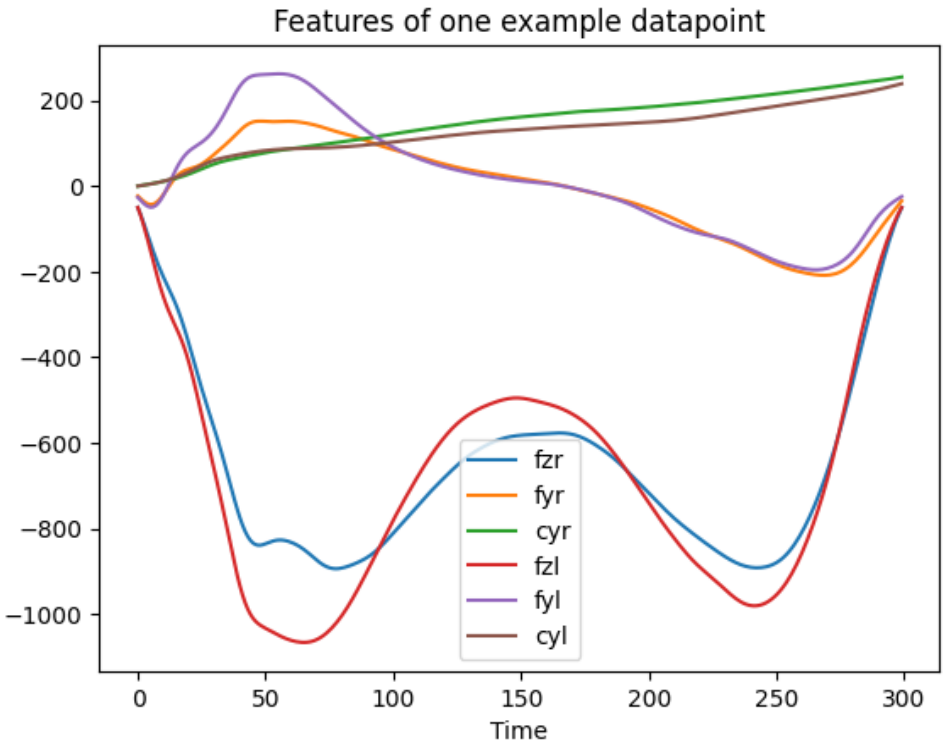
universitätbonn

Components

- Measurements from Force Platforms
- Two Platforms → per leg
- Force, Moment, Center of Pressure
- Direction: x,y,z
- Aim of original paper:
  - determine most discriminative components:

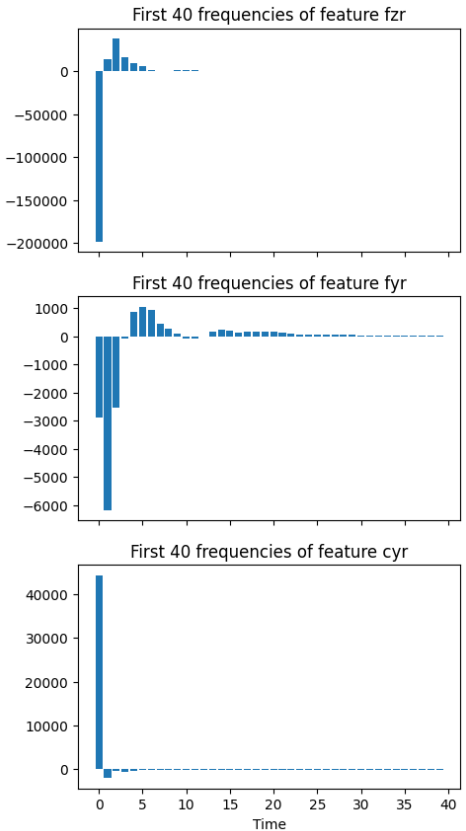
    Fz, Fy, Cy (96.02% in their setup)

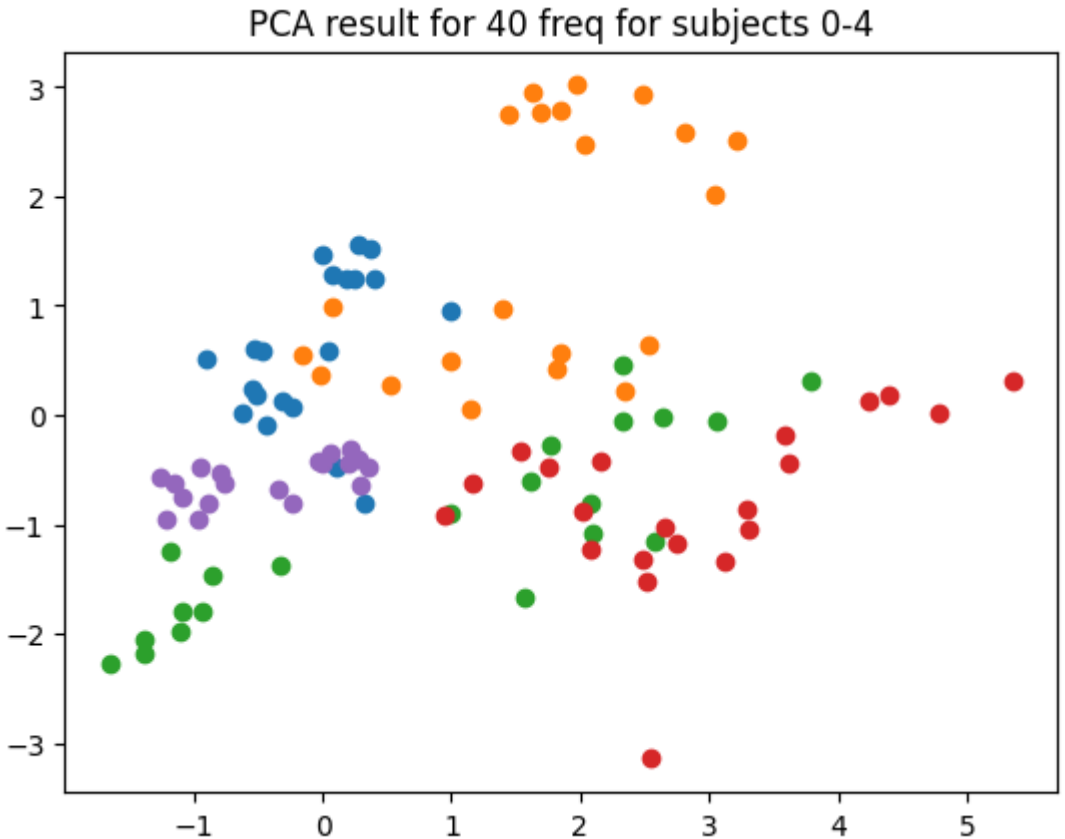
    → we use only these components



universitätbonn

Dimensionality reduction





universitätbonn

SVM

- Use first **40 frequencies** as before
- Use PCA result for PCA fitted on **train data**
- Employ **grid search**
  - Use ¼ of train data →
    - C=100
    - γ=0.1,
    - Kernel = rbf

- **Accuracy**:
  - 5-dim: 49,8%
  - 10-dim: 84,4%
  - 15-dim: 92,2%
  - 25-dim: 95,4%

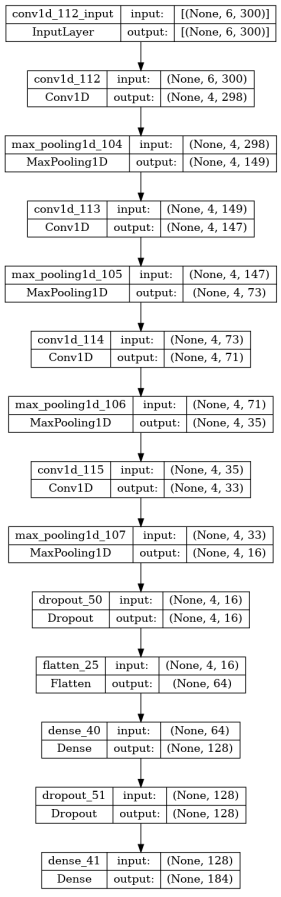
1D Convolutional NN

```
model = keras.models.Sequential()
model.add(layers.Conv1D(4, kernel_size=3, input_shape=X_train[0].shape, data_format="channels_first"))
model.add(layers.MaxPooling1D(pool_size=2, strides=None, padding="valid", data_format="channels_first"))
model.add(layers.Conv1D(4, kernel_size=3, data_format="channels_first"))
model.add(layers.MaxPooling1D(pool_size=2, strides=None, padding="valid", data_format="channels_first"))
model.add(layers.Conv1D(4, kernel_size=3, data_format="channels_first"))
model.add(layers.MaxPooling1D(pool_size=2, strides=None, padding="valid", data_format="channels_first"))
model.add(layers.Conv1D(4, kernel_size=3, data_format="channels_first"))
model.add(layers.MaxPooling1D(pool_size=2, strides=None, padding="valid", data_format="channels_first"))

model.add(layers.Dropout(0.3))
model.add(layers.Flatten())
model.add(layers.Dense(128, activation="relu"))
model.add(layers.Dropout(0.3))
model.add(layers.Dense(184, activation="softmax"))

model.compile(loss="categorical_crossentropy", optimizer="adam", metrics=["accuracy"])
```

Architecture

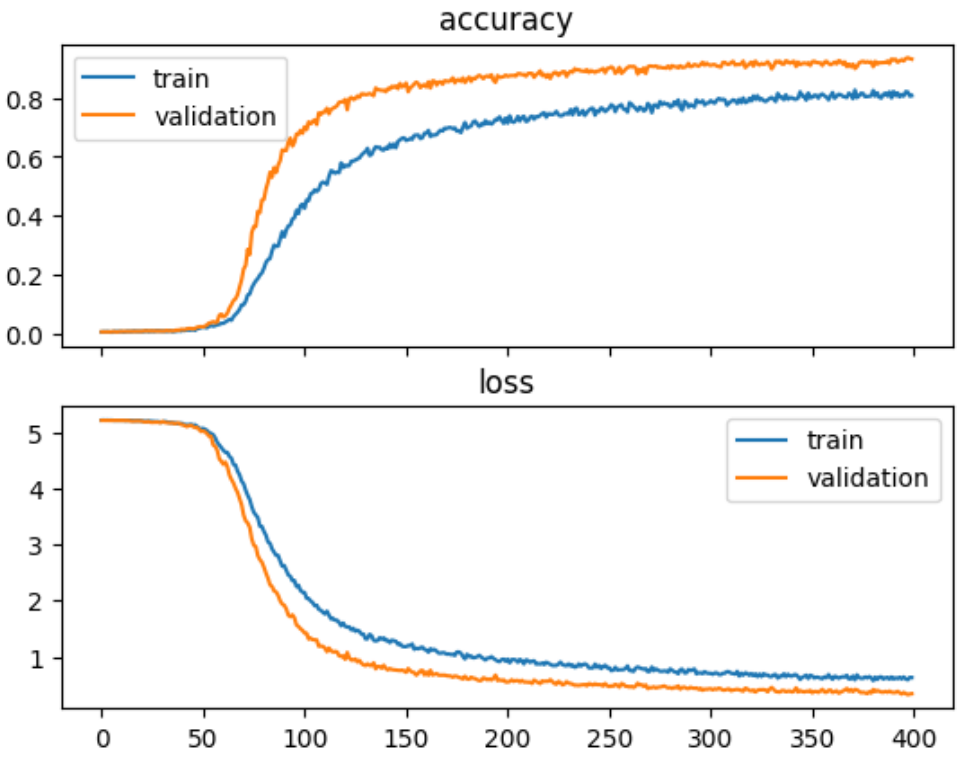
Layer (type)	Output Shape	Param #
conv1d_112 (Conv1D)	(None, 4, 298)	76
max_pooling1d_104 (MaxPooling1D)	(None, 4, 149)	0
conv1d_113 (Conv1D)	(None, 4, 147)	52
max_pooling1d_105 (MaxPooling1D)	(None, 4, 73)	0
conv1d_114 (Conv1D)	(None, 4, 71)	52
max_pooling1d_106 (MaxPooling1D)	(None, 4, 35)	0
conv1d_115 (Conv1D)	(None, 4, 33)	52
max_pooling1d_107 (MaxPooling1D)	(None, 4, 16)	0
dropout_50 (Dropout)	(None, 4, 16)	0
flatten_25 (Flatten)	(None, 64)	0
dense_40 (Dense)	(None, 128)	8320
dropout_51 (Dropout)	(None, 128)	0
dense_41 (Dense)	(None, 184)	23736

Total params: 32,288
Trainable params: 32,288
Non-trainable params: 0



universitätbonn

Result

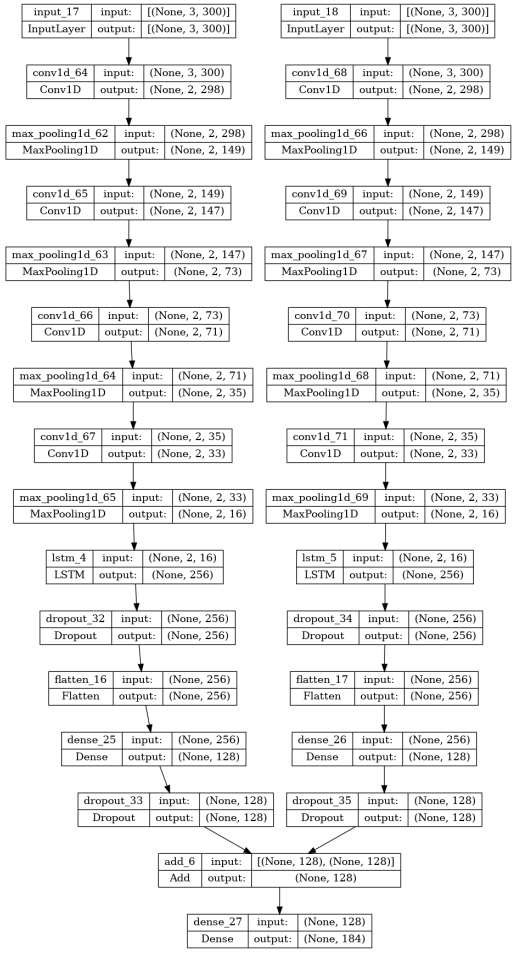


- 93.18% accuracy after 400 epochs
- Note:
  - Slow learning in the beginning

1D Convolutional NN advanced

- Suggestions (from paper):
  - Leg specific features?

    → one model per leg
  - Use LSTM layer to efficiently work with time series



universitätbonn

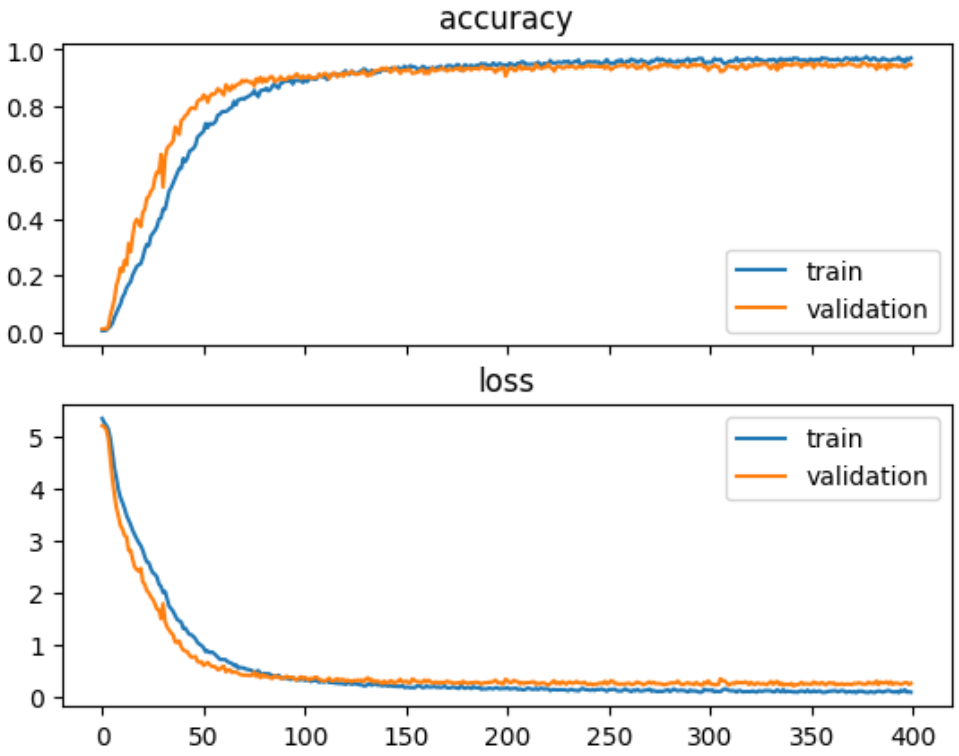
LSTM – Short Explanation

- 16 **time steps** left after 4th convolution & pooling
- **Iterate** over these
  - Keep some information in an **internal state**
  - **Update** internal state with each time step
- Output: **Final state**

Details

```
input_layers = []
output_layers = []
for i in range(2):
    layer_list = [
        layers.Input(shape=X_train_1[0].shape),
        layers.Conv1D(2, kernel_size=3, data_format="channels_first"),
        layers.MaxPooling1D(pool_size=2, strides=None, padding="valid", data_format="channels_first"),
        layers.Conv1D(2, kernel_size=3, data_format="channels_first"),
        layers.MaxPooling1D(pool_size=2, strides=None, padding="valid", data_format="channels_first"),
        layers.Conv1D(2, kernel_size=3, data_format="channels_first"),
        layers.MaxPooling1D(pool_size=2, strides=None, padding="valid", data_format="channels_first"),
        layers.Conv1D(2, kernel_size=3, data_format="channels_first"),
        layers.MaxPooling1D(pool_size=2, strides=None, padding="valid", data_format="channels_first"),

        layers.LSTM(256, dropout=0.3, recurrent_dropout=0.3),
        layers.Dropout(0.3),
        layers.Flatten(),
        layers.Dense(128, activation="relu"),
        layers.Dropout(0.3)
    ]
    # wire layers together
    wired_layer = layer_list[0]
    input_layers.append(wired_layer)
    for layer in layer_list[1:]:
        wired_layer = layer(wired_layer)
    output_layers.append(wired_layer)

# build final model by adding together the outputs of the separate models
added = layers.Add()(output_layers)
out = layers.Dense(184, activation="softmax")(added)
model = keras.models.Model(inputs=input_layers, outputs=out)

model.compile(loss="categorical_crossentropy", optimizer="adam", metrics=["accuracy"])
```

Result



- 94.68% accuracy after 400 epochs
  → slight improvement
- Faster learning within the first epochs
  → more efficient

Conclusion

- High accuracies using 1D (recurrent) CNN
- However:

  Careful feature engineering + “simple” SVM

  → similar performance

Thanks for listening!

Citations

- First dataset
  - Shorter, K. A., Polk, J. D., Rosengren, K. S., Hsiao-Wecksler, E. T. (2008). A new approach to detecting asymmetries in gait. Clinical Biomechanics. 23(4), 459-467. https://doi.org/10.1016/j.clinbiomech.2007.11.009
  - Helwig, N. E., Hong, S., Hsiao-Wecksler E. T., & Polk, J. D. (2011). Methods to temporally align gait cycle data. Journal of Biomechanics, 44(3), 561-566. https://doi.org/10.1016/j.jbiomech.2010.09.015
  - Helwig, N. E., Shorter, K. A., Ma, P. & Hsiao-Wecksler, E. T. (2016). Smoothing spline analysis of variance models: A new tool for the analysis of cyclic biomechanical data. Journal of Biomechanics, 49(14), 3216-3222. https://doi.org/10.1016/j.jbiomech.2016.07.035
- Second dataset
  - Bisele, Maria; Bencsik, Martin; Lewis, Martin G. C.; Barnett, Cleveland T. (2017). Optimisation of a machine learning algorithm in human locomotion using principal component and discriminant function analyses. PLOS ONE. Dataset. https://doi.org/10.1371/journal.pone.0183990