THE PRESENT WORK WAS SUBMITTED TO THE INSTITUTE OF GENERAL MECHANICS

RHEINISCH-WESTFÄLISCHE TECHNISCHE HOCHSCHULE AACHEN
FACULTY OF MECHANICAL ENGINEERING

PROJECT REPORT

PROJECT 05.02: BICYCLE-LOAD CLASSIFICATION

Presented by: Mukherjee, Ashutosh (428954), MSc. CAME
Savla, Neel (428689), MSc. MME-CAME
Khamkar, Raj (433267), MSc. MME-CAME
Jain, Sahaj (416009), MSc. CAME
Singhal, Vibhav (427359), MSc. MME-CAME

Group: 02

Abstract

It has become more important than ever to build highly sustainable systems and structures. One technique to warrant the safety and durability of any structure without over-engineering the components is to have a continuous health monitoring system that can sense any alarming peripheral impact which could lead to significant damage at later stages of the structure's life. This study aims to predict the loads on an engineering system, more specifically the weight of the person riding a bicycle. This can then be implemented in operation in order to, monitor the structural health of static as well as dynamic structures or for model-based control systems as well.

Acknowledgements

We would like to thank the entire team behind the organization of the course *Intelligent Monitoring of Engineering Systems* for giving us an opportunity to carry out this study. Specifically, we would like to thank Dr.-Ing., Univ.-Prof. Bernd Markert for introducing us to the vast field of structural health monitoring and always motivating us to gain not only good technical skills but also hone our soft skills to become good researchers. We would like to thank Dr. sc. ETH, Dipl.-Ing. Georg Kocur for giving us a comprehensive overview of signal processing techniques alongside helping us with technical challenges we faced during our work in MATLAB. Additionally, we would like to extend our gratitude to Mr. Rutwik Gulakala and Mr. Jan Stratmann for the assistance they provided with the experiments and the challenges we faced during the execution of the project.

Contents

List of Figures

List of Tables

1 Introduction

1.1 Motivation

In this study, different machine learning methods for classifying weights/loads have been investigated. The ability of these methods to generalize their prediction on an unseen dataset, effectively a modified experimental setup, is a critical factor to judge their performance.

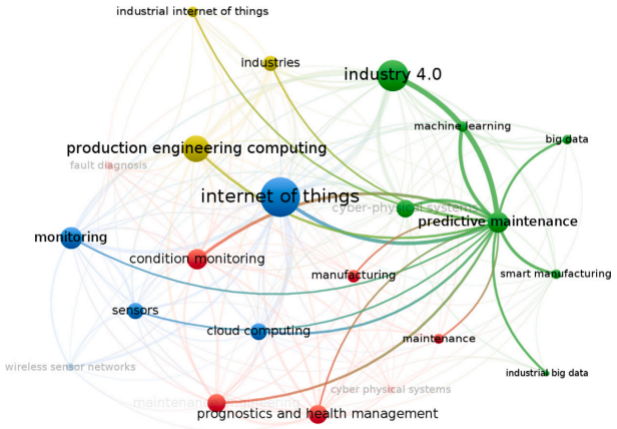

FIGURE 1.1: Predictive Maintenance in context of *Industry 4.0* and *IoT*
Image Courtesy: Zonta et. al [1]

This kind of study is very pertinent in predictive maintenance which is playing a major role in *Industry 4.0* [1], where preventive actions are taken instead of acting based on post-damage evidence [2]. Additionally, this study is invaluable for model-based control systems, which is widely used for controlling highly non-linear systems like exoskeletons [3, 4] and quadcopters [5], and for which model parameters need to be accurately predicted. Instead of depending on classical estimators like the Kalman Filter [3], we can intelligently predict such model characteristics using classifiers.

1.2 Experimental Setup

Seven hurdles were built with long slender wooden planks and were placed equidistantly 5 meters apart. Two bicycles with 28 inch and 24 inch wheel diameters were used to ride over these obstacles at a constant vecocity of $10 km/hr$. A total of six runs were done, three with each bike for each weight category ($65kg$, $75kg$, and $85kg$). The consequent accelerometer (*without acceleration due to gravity*) data were recorded with the smartphone secured on the handlebar of each bike. The data was recorded with a $200Hz$ sampling frequency and the **RWTH inhouse app Phyphox** [6] was used to output the data digitally into *.csv* files.

For the sake of this paper the data from the 28inch wheel diameter will be referred to as *Bike 1* and the other as *Bike 2*. Data from *Bike 1* was strictly used for training, validation, and testing of the classifier models whereas the data from *Bike 2* was used for making generalized predictions.

2 Methods

2.1 Preprocessing

To pre-process the data, four strategies are used, each based on the learnings of the previous, to arrive at the final strategy that is finally used to pre-process the input raw accelerometer signal. A visual illustration of the data-preprocessing strategies can be referred to from Figure 2.1. The raw signal can be seen in Figure 2.2(a) and the pre-processed data can be seen in Figure 2.2(b).

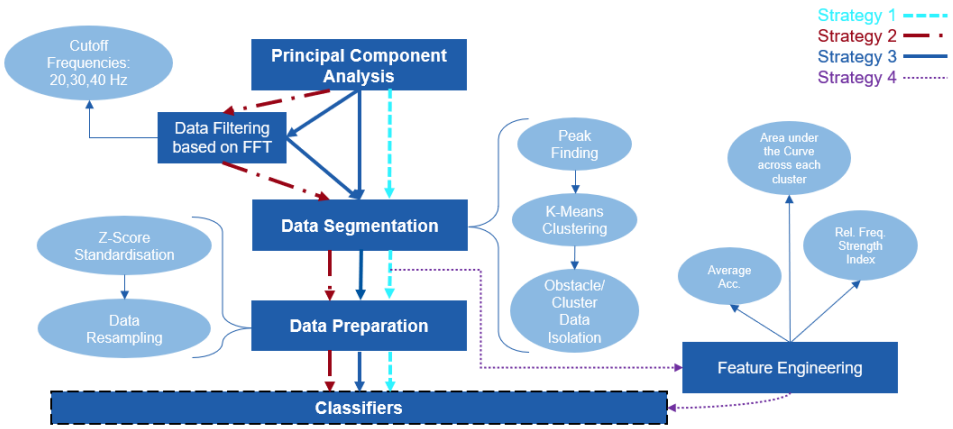

FIGURE 2.1: Data-Preprocessing Flow

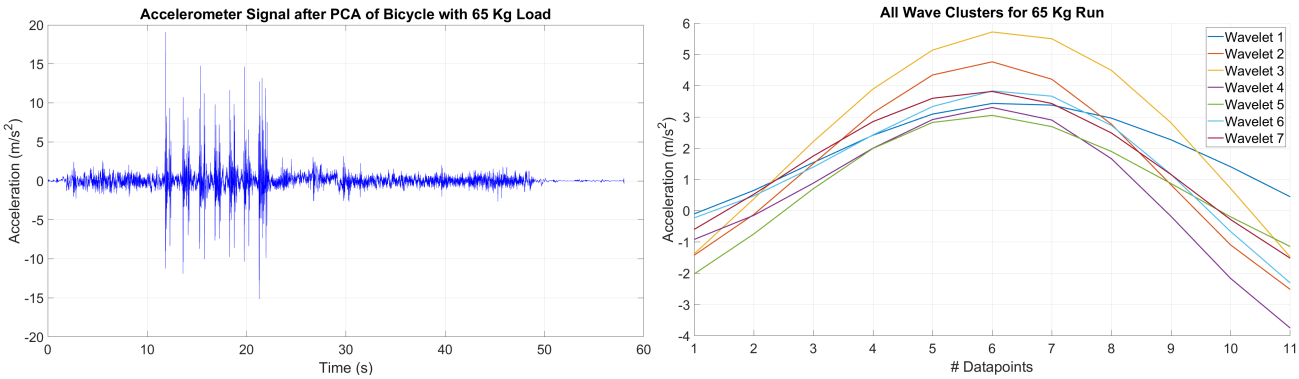

FIGURE 2.2: Acceleration data for 65 Kg weight run passed through all the data pre-preprocessing steps

The need for the second strategy arises from the lack of raw data for the effective training of neural networks. Since there is no point of symmetry in the dataset about which augmentation can be carried out, low-pass filtering of the data was considered. Fast Fourier Transform plots of the raw data revealed that below $40Hz$, there seemed to be a significant difference in the frequency strengths of the datasets of different weight categories. Thus, $40Hz$ was chosen as the upper limit for the augmentation process using filtering. Filtering the dataset at a $10Hz$ cut-off frequency led to a significant data loss, thus $20Hz$ was set as

the lower limit. Finally, the augmented dataset was obtained by filtering the data using a 4^{th} *order Butterworth Low-Pass Filter* with cut-off frequencies: $20Hz, 30Hz, 40Hz$.

Additionally, feature engineering was also carried out, but using those features for the Neural networks seemed to decrease its accuracies and SVM was able to perform optimally using only the average acceleration of the dataset.

2.2 Training

2.2.1 Support Vector Machine (SVM)

The SVM employed here uses a linear kernel to classify the input data into multiple classes of $65Kg$, $75Kg$ and $85Kg$ weight categories. This classification decision is made with a one vs one *"ovo"* decision function. With this, the function values are proportional to the distance of the samples to the separating hyperplane. Figure 2.3 depicts the segmentation of the hyperplane based on the training data with the position of the corresponding prediction data being superimposed on it.

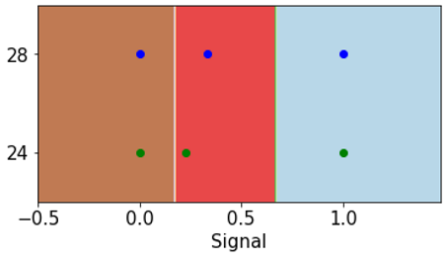

FIGURE 2.3: Superimposition of the generated SVM Hyperlane with prediction and training input data

2.2.2 Fully connected feed-forward Neural Network (FFNN)

The FFNN used here is a multi-layer perceptron (MLP) with one input layer, three hidden layers and one output layer with 3 neurons (1 for each weight class category). The hidden layers use a *Rectified Linear Unit* (ReLU) activation function, whereas the output layer uses a *Softmax* activation function. The Neural Network is compiled with an *Adam* optimizer and a *sparse Categorical Crossentropy* loss function. *Early stopping* is used to monitor the validation loss in order to avoid overfitting of the Neural Network model. Furthermore, *K-Fold validation* is used to verify the sturdiness of the hence trained Neural Network model.

2.2.3 Convolutional Neural Network (CNN)

CNN is a deep learning algorithm that is able to extract features from the given input data signifying a much lower pre-processing required as compared to other classification algorithms [7]. CNNs are generally preferred for image/video recognition problems, but can also be extended to other problems involving continuous data such as time series data (like in our case). The CNN employed here uses an input array of size $(200, 1)$. It further uses two convolution 1D layers with 32 and 64 filters respectively to extract the relevant features from the input data. This convolved data is further processed and passed to the output layer with 3 neurons. Similar to the feed-forward Neural Network the CNN is compiled with an *Adam* optimizer and a *Sparse Categorical Crossentropy* loss function. Early stopping is used here as well in a manner similar to the one used in the FFNN.

3 Results and Discussion

After extracting multiple features from the raw data, the prepared final form of pre-processed data obtained from *Bike 1* was fed into each of the classifier architectures to train and validate them. Accuracy and Loss plots were generated to assess the quality of the models developed and the K-Fold cross-validation technique was implemented to further determine a more accurate estimate of model prediction performance. The generalization ability of each classifier is summarized with the help of the following 3 tools: *Confusion Matrix*, *ROC Curve*, and *Classifier Metrics*.

3.1 Support Vector Machine

3.1.1 SVM: Generalization

In Figure 3.1(a), it can be seen that the SVM was able to classify each data-set correctly, thus achieving an accuracy of 100 %.

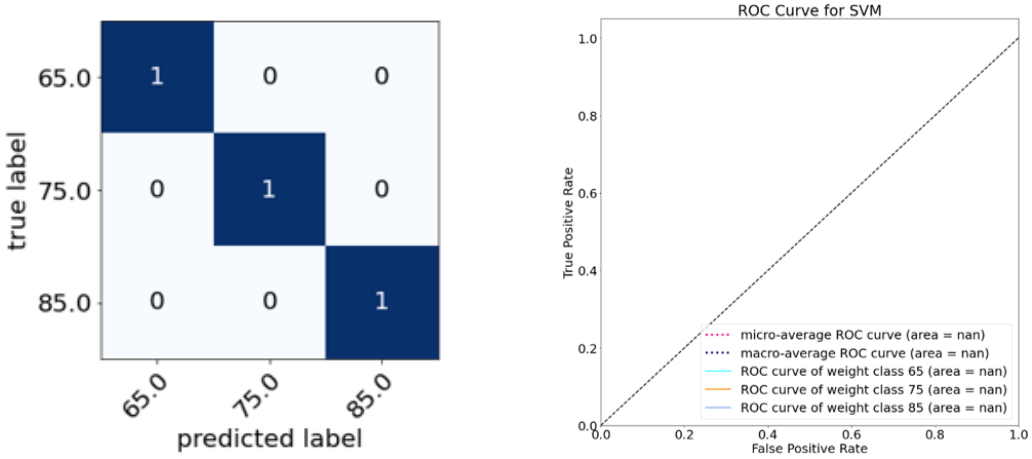

FIGURE 3.1: Confusion Matrix and ROC Curve for SVM for *Bike 2* data

Weight Class	Precision	Recall	F-Score	Accuracy
65.0	1.00	1.00	1.00	100%
75.0	1.00	1.00	1.00	100%
85.0	1.00	1.00	1.00	100%

TABLE 3.1: Classification metrics of SVM for *Bike 2* data

3.2 Fully connected feed-forward Neural Network

3.2.1 FFNN: Cross Validation

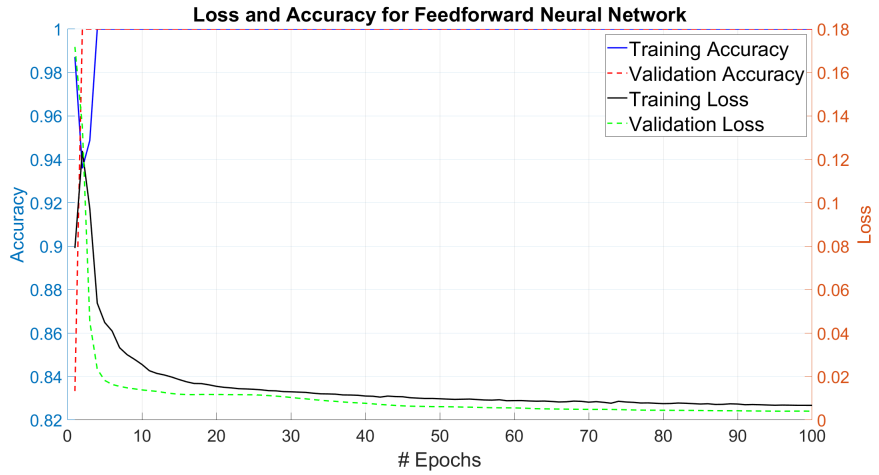

FIGURE 3.2: Accuracy and Loss for FFNN

In Figure 3.2, the validation loss decays faster than the training loss, which is an indicator that the model is getting trained optimally.

3.2.2 FFNN: Generalization

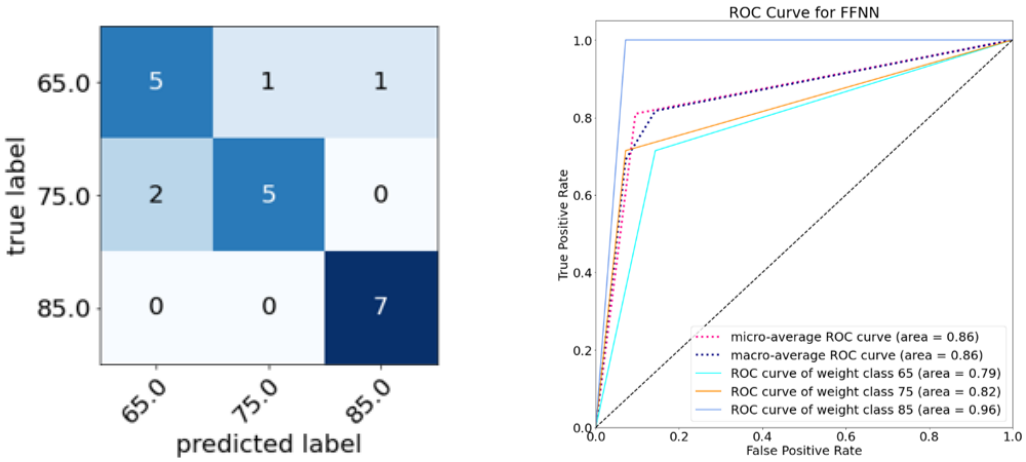

FIGURE 3.3: Confusion Matrix and ROC Curve for FFNN for *Bike 2* data

Weight Class	Precision	Recall	F-Score	Accuracy
65.0	0.71	0.71	0.71	71.4%
75.0	0.83	0.71	0.77	71.4%
85.0	1.00	1.00	0.93	100%

TABLE 3.2: Classification metrics of FFNN for *Bike 2* data

3.3 Convolutional Neural Network

3.3.1 CNN: Cross Validation

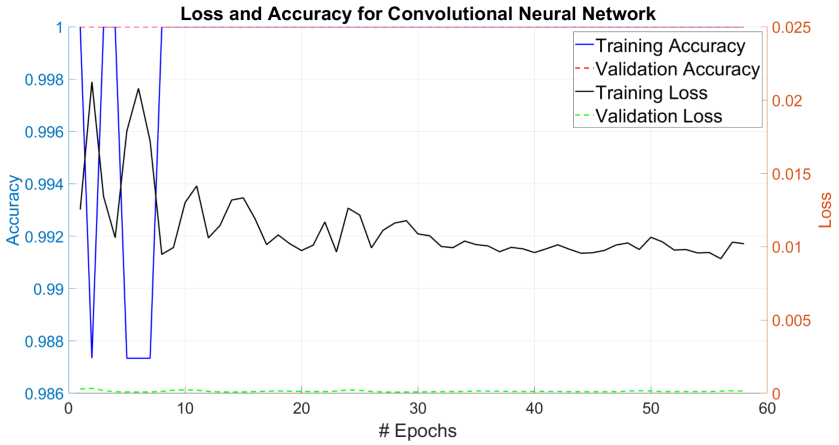

FIGURE 3.4: Accuracy and Loss for CNN

In Figure 3.4, the validation loss is at a very low value from the first epoch itself. This can be attributed to the sufficiently distinguishable input data features coupled with the feature extraction ability of the CNN.

3.3.2 CNN: Generalization

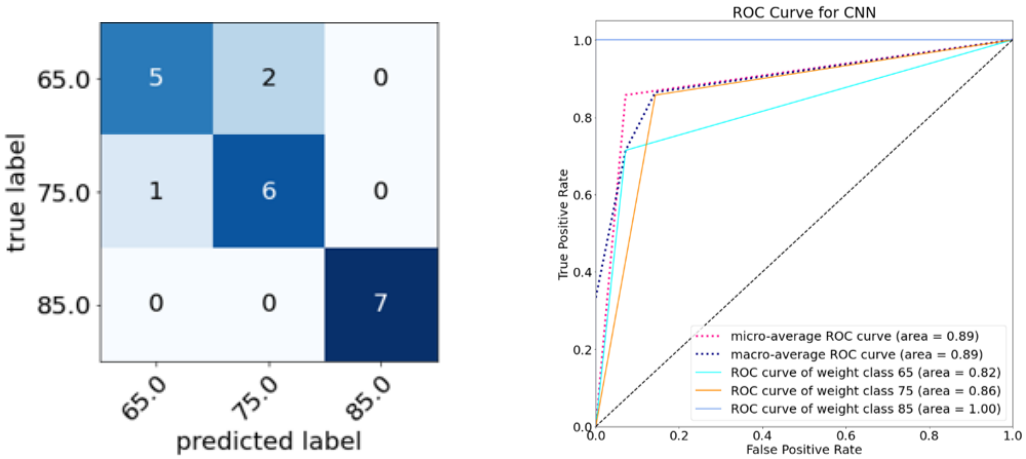

FIGURE 3.5: Confusion Matrix and ROC Curve for CNN for *Bike 2* data

Weight Class	Precision	Recall	F-Score	Accuracy
65.0	0.83	0.71	0.77	71.4%
75.0	0.75	0.86	0.80	85.7%
85.0	1.00	1.00	0.93	100%

TABLE 3.3: Classification metrics of CNN for *Bike 2* data

4 Conclusion and Outlook

4.1 Classifiers Comparison

Based on the detailed studies presented in Chapter 3, it can be concluded that the SVM works best for classifying the weight of a bicycle rider since the algorithm reached a prediction accuracy of 100%. The Neural network models, on the other hand, were unable to attain such a level of accuracy. Additionally, we can use preparation time as a metric too for comparison of the classifiers. SVMs require minimal data pre-processing, and for this particular case, it was able to give a perfect prediction using only one data feature, so the effort required in feature engineering was also minimum. On the other hand, the FFNN required the most amount of data pre-processing and feature engineering and still couldn't match the accuracy level of the SVM. Ideally, CNNs also require less data pre-processing effort than FFNNs, since they have intrinsic feature extraction capabilities, but for this particular case, the CNN was giving decent predictions only after a significant amount of data-preprocessing of the time series. So SVMs are a better classifier even when it comes to preparation time.

4.2 Outlook and Future Scope

The limitation of data is a major factor in assessing the performance of the above-presented Neural networks. SVMs can easily set thresholds for classifying a limited amount of data. On the other hand, for the Neural networks, the same data needs to be finely pre-processed to make it distinguishable for proper classification. On the contrary, with abundant data, the latter two can display improvement in their performance compared to the present case for the classification problem. Whereas, the SVMs in this case may get confused in setting the thresholds due to the closeness or in some cases even overlap of feature values in practical scenarios. Thus, the above conclusion about the comparison of the classifiers can be objectively drawn only for this particular use case with the given amount of data and a fair comparison between the classifiers can be made in a future similar experiment where the available amount of data is much more than the current scenario.

Bibliography

[1] Tiago Zonta et al. "Predictive maintenance in the Industry 4.0: A systematic literature review". In: *Computers and Industrial Engineering* 150.August (2020), p. 106889. ISSN: 03608352. DOI: 10.1016/j.cie.2020.106889. URL: https://doi.org/10.1016/j.cie.2020.106889.

[2] Daniel Frank Hesser, Kubilay Altun, and Bernd Markert. "Monitoring and tracking of a suspension railway based on data-driven methods applied to inertial measurements". In: *Mechanical Systems and Signal Processing* 164.May 2021 (2022), p. 108298. ISSN: 10961216. DOI: 10.1016/j.ymssp.2021.108298. URL: https://doi.org/10.1016/j.ymssp.2021.108298.

[3] Fatai Sado et al. *Exoskeleton robot control for synchronous walking assistance in repetitive manual handling works based on dual unscented kalman filter*. Vol. 13. 7. 2018, pp. 1–36. ISBN: 1111111111. DOI: 10.1371/journal.pone.0200193.

[4] Fatai Sado et al. "Design and control of a wearable lower-body exoskeleton for squatting and walking assistance in manual handling works". In: *Mechatronics* 63.August 2018 (2019), p. 102272. ISSN: 09574158. DOI: 10.1016/j.mechatronics.2019.102272. URL: https://doi.org/10.1016/j.mechatronics.2019.102272.

[5] Ghulam Farid et al. "On control law partitioning for nonlinear control of a quadrotor UAV". In: *Proceedings of 2018 15th International Bhurban Conference on Applied Sciences and Technology, IBCAST 2018* 2018-Janua (2018), pp. 257–262. DOI: 10.1109/IBCAST.2018.8312233.

[6] Staacks S et al. "Advanced tools for smartphone- based experiments : phyphox". In: *Physics Education* (2018).

[7] Towards Data Science. *A Comprehensive Guide to Convolutional Neural Networks*. 2018. URL: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53 (visited on 07/28/2022).

Supplementary Material

The codebase for this study can be accesed at: https://git.rwth-aachen.de/raj.khamkar/imes_group2