Classification and Prediction of Bicycle-Road-Quality using IMU Data

Johannes Heidt and Klaus Dorer

Institute of Machine Learning and Analytics, Offenburg University
johannes.heidt@hs-offenburg.de
klaus.dorer@hs-offenburg.de

Abstract. The present work ties in with the problem of bicycle road assessment that is currently done using expensive special measuring vehicles. Our alternative approach for road condition assessment is to mount a sensor device on a bicycle which sends accelerometer and gyroscope data via WiFi to a classification server. There, a prediction model determines road type and condition based on the sensor data. For the classification task, we compare different machine learning methods with each other, whereby validation accuracies of 99% can be achieved with deep residual networks such as InceptionTime. The main contribution of this work with respect to comparable work is that we achieve excellent accuracies on a realistic dataset classifying road conditions into nine distinct classes that are highly relevant for practice.

Keywords: Machine Learning, Deep Learning, InceptionTime, ResNet, Time-series Classification, Road-Quality Prediction

1 Introduction

In November 2019 the Bundestag passed the climate package [1]. There it was decided that at least 900 million Euros should be invested in the expansion and renewal of the German cycling infrastructure. This raises the question of how to obtain an overview of the current state of cycle tracks.

While the Federal Ministry of Transport and Digital Infrastructure (BMVI) is responsible for recording and assessing the condition of the major highways, responsibility for inner-city roads lies with the communes. The BMVI carries out these assessments at fixed intervals of four years, using special measuring vehicles that use laser technology and digital cameras[1]. At the communal level, it depends on individual requirements [2]: Expensive measuring vehicles are used as well, and often random surveys of the residents are conducted on the condition of the roads. With passing the climate package in November 2019, this topic is likely to take on a greater role for local authorities in Germany in the near future [1]. The government decided to invest at least 900 million euros in the expansion and renewal of the cycling infrastructure.

The aim of this work is to develop a new approach for the condition monitoring of cycling tracks. For this purpose, we attach an acceleration, gyroscope and GPS sensor to a bicycle. With the help of machine learning methods, the recorded acceleration values and angular velocities are used to predict the road type and condition. Here, the classifier differentiates between the road types asphalt, cobblestone and gravel as well as the

[1] https://www.bmvi.de/SharedDocs/DE/Artikel/StB/zustand-netzqualitaet-der-fahrbahnen.html

conditions smooth, rough and bumpy. Finally, the system sends the recorded sensor data to a server, which is responsible for the classification and visualization of the results. In this way, our setup determines the condition of the roads concurrently, so that it requires no further monitoring tasks by the cyclist.

For the training and testing of the machine learning algorithms, we collected the required sensor data manually. The setup consists of a smartphone and a Bosch XDK which is equipped with an accelerometer and a gyroscope. Furthermore, we developed an app which communicates with the XDK via Bluetooth – this enables us to annotate the data. Alternatively, the XDK can persist the sensor data on its memory card. In this case it can be labeled manually afterwards. In this way, we created two data sets with different sampling rates and trained, evaluated and compared different classifiers. The models use common algorithms such as tree-based procedures or state-of-the-art residual networks. To test the transferability of the models, we collected a data set with another bicycle and used it for testing the previously trained models. Finally, we created a setup which allows the wireless transmission of the sensor data to a classification server.

2 Related Work

[3] dealt with both the classification of road quality and the detection of bumps. In order to accomplish this, the authors attached a smartphone to the steering wheel of a bicycle and covered a distance of almost 14 km for a total of 16 times [3][p. 40 f.]. During the rides, they used the smartphone to record GPS coordinates at a rate of 1 Hz and accelerometer data at 37 Hz. From this data, the authors extracted five attributes - inclination, speed, as well as mean, variance and standard deviation of acceleration. Finally, they combined the features into segments based on the GPS coordinates and annotated them manually.

In this case, Hoffmann et al. distinguished three classes: smooth, rough and bumpy. For prediction they chose the classifiers k-Nearest Neighbors (kNN) and Naive Bayes. Furthermore, Hoffmann et al. tested different segment lengths (1 m, 2 m, 5 m, 10 m, 15 m, 20 m) and feature combinations. In the end, they achieved the best results by choosing a segment length of 20 m and relying only on the three acceleration features. Moreover, the other two features proved to be unhelpful, because the speed could not contribute to the prediction and the slope even confused the classifiers. The class distribution was not very balanced with 6772 smooth, 2044 rough and 646 bumpy segments. For evaluation, the authors performed a tenfold cross-validation, with the kNN classifier achieving a slightly better result than Naive Bayes with a mean accuracy of 77.457% [3][p. 41 f.]. In our work, we achieve significantly higher accuracies, mainly due to the higher sampling frequency.

Litzenberger et al. also deal with the classification of bicycle tracks using accelerometers [4]. They compared different experimental setups [4][p. 1]. They recorded all data on three flat and straight sections, each 100 m long. Furthermore, the authors chose three different section types: cobblestone, gravel and asphalt. These also represent the classes to be predicted. For the data set generation, they drove each route at three different speeds (10/20/30 km/h) and three different tire pressures (3/4/5 bar). Consequently, each route was covered nine times.

Furthermore, the authors used two different devices for data recording [4][p. 2]: First, Litzenberger et al. attached an accelerometer sensor with a sampling rate of 500 Hz to the fork of a bicycle. On the other hand, the rider had placed a smartphone in his back pocket, which recorded accelerometer data at a frequency of 100 Hz. Subsequently, they formed samples that encompassed time windows of different sizes. Values of one and two seconds were tested here. In the next pre-processing step, they calculated 16 summary

statistics for each of the four channels of the accelerometer (X, Y, and Z direction and total acceleration). These statistics are: average, maximum, and minimum values, average peak distance and amplitude, median FFT (Fast Fourier Transform) signal frequency, maximum and minimum FFT signal frequency, maximum and minimum FFT signal amplitude, average positive and negative slope, maximum and minimum positive slope, and maximum and minimum negative slope.

For the classification task the authors trained and tested different tree-based methods and SVMs in a fivefold cross-validation. In the end, an SVM with a polynomial kernel was able to achieve the highest accuracy in the detection of three classes - both when using accelerometer data (99.2%) and smartphone data (97.7%). In another test, Litzenberger et al. tried to additionally predict speed and tire pressure, resulting in 27 classes. Here an ensemble of boosting trees scored best. With the data from the accelerometer, the classifier scored 97.9%, whereas the smartphone data lead to an accuracy of 48.4%. In all trials, the time window of two seconds produced slightly better results.

Hoffman et al. only relied on one track that is run a total of 16 times [3]. This is not optimal, because it means that similar data is used in the process of training as in the validation. Furthermore, the validation accuracy can possibly be improved. Litzenberger et al. drove three selected routes several times as well [4]. Although they used different tire pressures and speeds, the selected courses are only 100 m long. Consequently, the question arises to what extent the models are suitable for application in practice. In this work, we achieve a similarly high or even higher validation accuracy compared to Litzenberger et al. At the same time, a larger and more varied route selection was performed while attempting not to run the same tracks more than once.

3 Approach

The goal of this work is to find classifiers, that predict road types and their condition with the help of a wireless sensor network. The sensor attached to the bicycle is supposed to transmit the data to a classification server. Ideally, the model should achieve the highest possible prediction accuracy for unknown roads. The pursued approach is described in this chapter.

3.1 Hardware

With regard to the classification task, we chose a Bosch XDK 110 which has a BMA 280 accelerometer and a BMG 160 gyroscope. Both sensors offer a sampling rate of up to 2000 Hz. The measuring range of the accelerometer is $\pm$16 g, that of the gyroscope $\pm$2000 $\circ$ /s. The other sensors of the XDK are not needed in this case. There is also a slot for micro SD cards. Three programmable LEDs and two programmable buttons are available for user interaction. Finally, the XDK offers two options for wireless data transmission: Bluetooth 4.0 Low Energy IEEE 802.15.1 and Wireless LAN IEEE 802.11b/g/n.

3.2 Measurement

For data set generation, we mounted a plastic box modified for this application to the basket of a 28-inch trekking bike (see figure 1 bottom left). Before that, we attached the bicycle basket to a carrier, which is located above the rear wheel (top left). The Bosch XDK can be hooked into the mounted device (middle picture). The lockable plastic box has proven to be a reliable rain protection. Both the bicycle basket and the box device

are securely fixed with several cable ties so that no additional bouncing occurs during the ride. The two components only "vibrate" together with the carrier.

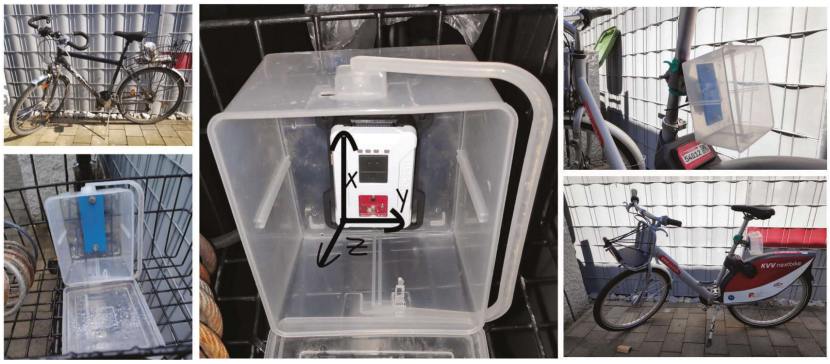


Fig. 1: Setup for data recording.

We created a total of three data sets: Two training sets at different sampling rates and a final test set. The data collection took place with as many different routes as possible. At the same time, we ensured that the data set had a relatively balanced class ratio, with each of the nine classes accounting for about one-ninth of the entire dataset. As a result, routes of rarely occurring classes were sometimes used several times. When creating the data sets, the cyclist tried to vary the driving speed. With regard to the amount of data to be collected.

Moreover, we developed a labelling app to create the first training set. It connects via Bluetooth to the Bosch XDK, which in turn sends the recorded sensor data to the app. Via the app the cyclist can assign the data to one of the nine classes. Afterwards the app persists the sensor values on the smartphone. However, since the Bluetooth protocol limits the length of a message to 20 bytes, it is not possible to achieve a very high sampling rate. In this way, the system recorded 734,441 values per sensor axis (x-, y- and z-values of the accelerometer and the gyroscope) in a period of 376 minutes, which results in a sampling rate of 32.5 Hz. We drove 91.34 km with an average speed of about 14.6 km/h. In general, there were no multiple runs on the same route - except for the rarely occurring classes of smooth and bumpy cobblestone. Furthermore, we achieved a relatively balanced class ratio (the distances covered range from 9.2 to 11.3 km per class).

With regard to the second data set, we aimed to maximize the sampling rate. Therefore, we omitted the Bluetooth app and stored the data directly on the SD card of the XDK. For the labelling of the collected sensor data we programmed a simple user interface using the buttons and LEDs of the XDK. With this approach, the XDK was able to record 5,632,000 values per sensor axis over a period of roughly 314 minutes. Compared to the first data set, the total recording time is about 62 minutes shorter, but we collected significantly more data (factor approx. 7.7). This results in an approximate sampling rate of 294.1 Hz - more than nine times higher than in the first approach. We sampled 76.85 km, with the routes largely identical to those of the first data set. Moreover we omitted some shorter tracks for practical reasons, so that we drove in total about 15 km less. The

average speed was about 14,5 km/h and is very close to that of the first data set (14.6 km/h). Lastly, the class ratio is relatively balanced, similar to the first case.

Finally, we created a last data set that is only used for testing purposes and not for training models. By using another bike to collect the sensor data, it should be checked to what extent the trained models are transferable to another bike. Therefore, we chose a bike from the local rental system. As shown in figure 1 (pictures on the right), we placed the plastic box device in a similar way above the rear wheel. For data acquisition again the variant with the memory card was our choice because it offers the higher sampling rate. Apart from the new bike, the procedure is the same as for the creation of the second data set. In about 34 minutes, the XDK recorded 514,500 measurements per sensor axis, resulting in a sampling rate of approx. 252 Hz. Thus the size of the test data set corresponds to roughly 9.1% of the second data set. For each of the nine classes, we recorded a distance of around one kilometer, totalling 9.32 km. Compared to the previous data sets, the average speed of 16.4 km/h is slightly higher. However, the class ratios are not quite as balanced as with the previous data sets: The number of measured values per axis is between 40,000 and 76,000 for the individual classes.

3.3 Data Preprocessing

Before the collected data is used, several preprocessing steps are performed. To reduce noise in the data, idle times are removed from the data set. These can occur when stopping the bike to annotate the data with the smartphone app. Since the timestamps of all sensor data are known, including the GPS coordinates, the idle times can be removed in an automated manner.

Before the data is used as model input, it is normalized. The individual measurements of the six sensor channels are transformed so that they have a mean value of 0 and a standard deviation of 1. In addition, the transformation is performed independently for training and test data. Finally, this is the model input for the deep learning-based methods. For all other methods, the data are statistically summarized before normalization, whereby the following metrics are calculated for the individual features of a sample: arithmetic mean, standard deviation, maximum and minimum value, mean width and amplitude of signal peaks, mean peak distance, mean positive and negative slope as well as maximum positive and minimum negative slope. A total of 14 new features are calculated for each of the six features (accelerometer and gyroscope each with x-, y-, and z-values). Consequently, the original sample matrix with dimensions $s_i \times 6$ becomes a sample vector of length $s_j = 84$. Here, s_i is the sample size respectively the number of time steps and thus a configurable hyperparameter.

4 Results

This chapter presents the results of the classification using several machine learning models that were trained and evaluated using cross-validation.

4.1 Hyperparameter Selection

We used the scikit-learn library to implement the machine learning methods [5]. Only an implementation for XGBoost is missing, so here the XGBoost library is used [6]. Lastly, we implemented the deep-learning based methods using keras [7] and TensorFlow [8]. When implementing our algorithms, we had to choose the parameters carefully. Eventually, we

tested different parameter combinations for each classifier in a grid search, whereby each parameter combination of this grid was tested using a tenfold stratified cross-validation. The final parameter selection is presented in the following chapters. It should be noted beforehand that all the other standard parameters, which are not explicitly mentioned in the following parameter grids, remain unchanged and can be looked up in the according documentation [5–7].

Tree Based Methods In general, tree based methods provide good results for different types of classification problems [6][p. 785 f.]. Furthermore, their decision-making process is easy to interpret. That is why this evaluation process includes random forest and Extreme Gradient Boosting (XGBoost) [9, 10]. In the case of the first data set, we used the following parameter grid for the grid search: $t \in \{5, 10, 25, 50, 100, 200\}$, $max\_depth \in \{10, 25, 50\}$, $min\_samples\_leaf \in \{1, 2, 5\}$, $min\_samples\_split \in \{2, 5, 10\}$ and $n\_estimators \in \{2, 5, 10\}$. Please note that t is not a parameter of the scikit-learn or XGBoost models which we used. It describes the number of time steps for which we calculated summary statistics such as mean values and standard deviations. With the following parameters the random forest has achieved the best result: $t = 200$, $max\_depth = 50$, $min\_samples\_leaf = 1$, $min\_samples\_split = 5$ and $n\_estimators = 100$, resulting in a mean accuracy of 68.546% with a standard deviation of 3.271%. In contrast, the highest mean validation accuracy of the XGBoost classifier is slightly better with 68.963%. At the same time, the standard deviation of 1.646% is slightly lower than with the Random Forest. The parameter selection $t = 200$, $max\_depth = 50$, $min\_samples\_leaf = 1$, $min\_samples\_split = 2$ and $n\_estimators = 300$ leads to this result.

We adjusted the first grid slightly for the second data set, regarding its higher sampling rate: $t \in \{50, 100, 250, 500, 1000, 2000\}$ and $n\_estimators \in \{100, 300, 1000\}$, while the possible values for the other three variables remain the same. The best result of the random forest is achieved with $t = 2000$, $max\_depth = 25$, $min\_samples\_leaf = 2$, $min\_samples\_split = 5$ and $n\_estimators = 300$ - this leads to a validation accuracy of 83.333%. With XGBoost the parameter values $t = 2000$, $max\_depth = 25$, $min\_samples\_leaf = 1$, $min\_samples\_split = 2$ and $n\_estimators = 1000$ result in the highest validation accuracy (86,17%).

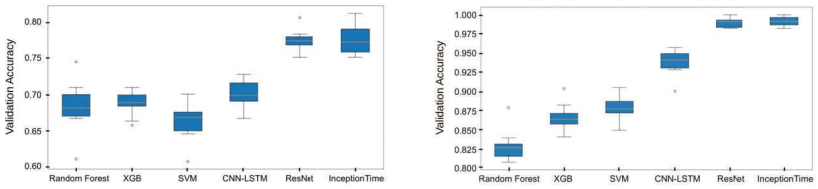
Support Vector Machine The support vector machine (SVM) also proves to be a reliable model for a wide range of classification tasks [11]. Here we use the following parameter grid in connection with the first data set: $t \in \{5, 10, 25, 50, 100, 200\}$, $kernel \in \{linear, poly, rbf, sigmoid\}$, $C \in \{2^{-5}, 2, 2^{10}\}$, $gamma \in \{scale, auto\}$ and $degree \in \{2, 3\}$. Regarding *gamma*, a value of *auto* means $1/n_{features}$ and *scale* means $1/(n_{features} \times \sigma_x^2)$. With the parameter combination $t = 200$, $kernel = linear$ and $C = 2$, we achieved a mean accuracy of 66.539% with a standard deviation of 2.377%, which is the best result.

For training with the second data set, we adjusted only the time steps $t \in \{50, 100, 250, 500, 1000, 2000\}$. With the parameter selection $t = 2000$, $kernel = linear$ and $C = 2$ we achieved the highest validation accuracy of 86.702%.

Deep Learning Based Methods As deep learning approaches we tested long short-term memory (LSTM) and convolutional neural networks (CNNs) with different architectures [12, 13]. Hence, we tested a single LSTM layer, which is fully connected to the output layer. When working with CNNs, we also adjusted the *sample_size* and *dropout*. The

sample_size specifies the number of instances which we combine into one input. It can be considered as the number of time steps, but in comparison to t we calculated no summary statistics. Additionally, we analyzed different values for the number of $filters$, *kernel_size* and *pooling_size*. Here, we tested different architectures: a simple CNN consisting of two convolutional layers with dropout and max pooling followed by a fully connected hidden layer which is again fully connected to the output layer. Next, we evaluated a CNN-LSTM which has a similar architecture like the simple CNN, we only replaced the hidden layer with an LSTM layer. Lastly, we tested two residual networks from Fawaz et al.: ResNet [14] and InceptionTime [15]. The main part of the architecture are the residual blocks, which contain three convolutional layers and use batch normalization in between. The blocks themselves are also linked via residual connections. ResNet consists of three and InceptionTime of two blocks. Moreover, the residual blocks for InceptionTime include an inception module, which enables the use of multiple, concatenated filter types. In this paper both model architectures remain unchanged.

For the plain LSTM architecture the parameter grid for the first data set looked like this: $sample\_size \in \{20, 50, 100, 150, 200\}$, $recurrent\_dropout \in \{0, 1/4, 1/2\}$ and $units \in \{32, 64, 128\}$. With the parameters $sample\_size = 50$, $units = 128$ and $recurrent\_dropout = 1/4$, we achieved the highest mean validation accuracy - 65.329% with a standard deviation of 1.03%. The tested CNN architecture uses the following grid: $sample\_size \in \{20, 50, 100, 150, 200\}$, $filters \in \{32, 64, 128\}$, $dropout \in \{0, 1/4, 1/2\}$, $kernel\_size \in \{3, 5, 7\}$, $pool\_size \in \{2, 5\}$. With the parameters $sample\_size = 100$, $dropout = 1/2$, $filters = 32$, $kernel\_size = 5$ and $pool\_size = 5$, we achieved a mean validation accuracy of 62.912% with a standard deviation of 1.792%. The CNN-LSTM architecture uses the same grid as the CNN and achieved a slightly better result with a validation accuracy of 70.222% at a standard deviation of 1.699%. The model uses the parameters $sample\_size = 200$, $dropout = 0$, $filters = 64$, $kernel\_size = 3$ and $pool\_size = 5$. In the case of residual nets, we examined only different values for $sample\_size \in \{20, 50, 100, 150, 200\}$. With a value of $sample\_size = 200$, the residual nets achieved the highest mean validation accuracy: InceptionTime scores 77.645% with a standard deviation of 1.925%, ResNet is just below that with 77.484%. But the standard deviation of 1.382% is slightly lower than with InceptionTime.



(a) First dataset (b) Second dataset

Fig. 2: Comparison of classification results.

In the case of the second data set, the parameter grid for all the deep learning based models remained almost the same. We only adjusted the *sample_size* for the higher sampling rate: $sample\_size \in \{50, 100, 250, 500, 1000, 2000\}$. The best result of the LSTM is 71.129% which uses the parameters $sample\_size = 100$, $units = 32$ and $recurrent\_dropout = 0$. When using a CNN, we achieved the highest accuracy of 77.852% with the parameters $sample\_size = 250$, $filters = 128$, $dropout = 1/2$, $kernel\_size = 7$

and $pool\_size = 2$. The CNN-LSTM scored a significantly higher validation accuracy of 91.312%. Already during the tests with the first data set we noticed that the accuracy of LSTMs drops rapidly with $sample\_size >= 150$, which seems to be a common issue [16]. To prevent these problems, we adjusted the pooling size ($pool\_size \in \{5, 15, 50\}$) compared to the first parameter grid. The parameter selection $sample\_size = 2000$, $filters = 32$, $dropout = 0$, $kernel\_size = 7$ and $pool\_size = 50$ finally leads to the best result for this model. All in all, the residual networks provide the best results. With a sequence length of $sample\_size = 2000$, InceptionTime achieves 98.404%, the ResNet scores with 98.227% marginally below.

We trained all models over 200 epochs, only the residual nets over 1000 epochs as it took significantly longer for overfitting to occur. In all cases, we stopped the training process as soon as the training accuracy did not improve over 30 consecutive epochs. Furthermore, we used a local computer with 64 GB of RAM and an nVidia GeForce GTX 1080 graphics card.

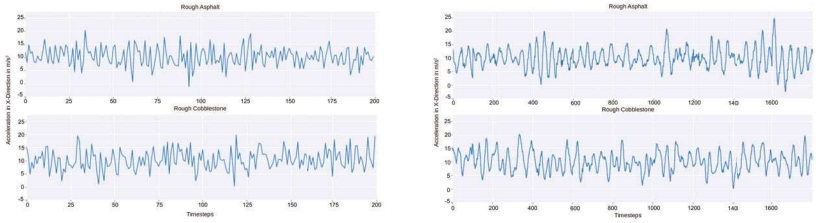
4.2 Classification

Next, we compare the presented models with each other using the hyperparameter selection from the previous chapter.

First Data Set As seen in figure 2 (a), the deep-learning-based methods deliver the best results - especially the two deep residiual networks ResNet and InceptionTime. While InceptionTime has the highest mean validation accuracy of the examined models with 77.65%, ResNet with 77.42% has a slightly higher median than InceptionTime (77.26%). Furthermore, the variance in the validation accuracy of the ResNet is lower than that of InceptionTime because the ResNet has a lower standard deviation (1.382% vs. 1.925%) and a smaller interquartile range (1.129% vs. 3.226%). The top two models are followed by the CNN-LSTM, which has an average validation accuracy of 70.22% with a standard deviation of 1.699%. In addition, the median lies at 69.84% with an interquartile range of 2.484%. After that follow the models XGBoost, Random Forest and SVM with median values of 68.90%, 68.23% and 66.88% respectively. The interquartile ranges are 1.553%, 2.958% and 2.605% respectively.

During the tests in chapter 4.1 we observed that almost all classifiers achieved their best results with longer sequences. Consequently, the question arises why no values $t > 200$ or $sample\_size > 200$ were examined. In the end, we set this value as the upper limit for practical reasons: At an average speed of about 14.6 km/h and a sampling rate of 32.5 Hz, 200 time steps result in a time period of about 6.15 seconds and thus an approximate distance of 25 m. In terms of application of the model in practice, a classification of shorter sequences seems preferable.

To gain a better understanding of the strengths and weaknesses of the single classifiers, we examined the precision and recall rates for the individual classes. It became apparent that all models have problems in recognizing the classes rough asphalt, rough cobblestone and bumpy gravel. When also looking at the confusion matrices of these models, it is noticeable that the models often confuse the problem classes with each other. The reason for this could be the sampling rate: At an average speed of about 14.6 km/h, roughly 4.05 m are covered in one second, so that the smartphone app records a sensor measurement approximately every 12.5 cm. However, there are cobblestones that have a shorter length than this. Also in the case of asphalt and gravel, bumps can occur at shorter intervals. Consequently, the tests with the next data set should show what influence the sampling rate can have.



(a) Rough asphalt and rough cobblestone (32.5 Hz)

(b) Rough asphalt and rough cobblestone (294.1 Hz)

Fig. 3: Comparison of selected sequences at different sampling rates.

Second Data Set As in the test with the first data set, the two deep residual networks clearly provide the best results (figure 2 (b)). Between the two, InceptionTime achieves slightly better results: The mean validation accuracy is 99.18% and the median is 99.11%. For the ResNet, it is 98.97% and 99.11%. Again, the variance metrics are nearly equal: While the validation accuracy of InceptionTime has a standard deviation of 0.60% and an interquartile range of 0.98%, the corresponding values of the ResNet are 0.60% and 0.98%. The CNN-LSTM also achieves very good results with a median of 94.14% and an interquartile range of 1.87%. The SVM follows with a median of 87.77% at an interquartile range of 1.54%. Finally, the tree-based methods XGBoost and Random Forest achieve a median of 86.40% and 82.74%, with an interquartile range of 1.42% and 1.60%.

Already during the tests with the first data set we observed that most models benefit from longer sequence lengths. This is also the case with the second data set, but again we limit the maximum sequence length for practical reasons. In case of the first data set we selected 200 time steps as maximum. Since the sampling rate is almost ten times higher for the second data set, we set a corresponding value of $t = 2000$. At a sampling rate of 294.1 Hz, an average sequence of length $t = 2000$ records a period of about 6.8 seconds. Thus, at an average speed of 14.45 km/h, the cyclist covers a distance of roughly 27.3 meters.

Figure 3 (a) shows some selected sequences that indicate why a correct classification can be difficult with a lower sampling rate. Rough asphalt and rough cobblestone are two classes that were very frequently confused with each other in the case of the first data set. The measured value is the acceleration in the x-direction of the XDK (perpendicular to gravity) and it is plotted over 200 time steps. It is noticeable that both time series have very similar characteristics: The measurements vary quite regularly between -5 and 15 m/s^2 with just a few outliers. In addition, the number of peaks and the average period lengths appear to be similar. Consequently, a simple distinction of the classes based on the graphs is not possible. For comparison with sequences at 294.1 Hz, we slightly adjusted the x-axis in 3 (b): 1800 time steps are mapped here, which corresponds to a duration of about 6.12 seconds. This is about the same time period mapped in the left graphs (6.15 seconds). In addition, the measured values of the right-hand figures fluctuate within approximately the same range as in figure 3 (b). Furthermore, the number of peaks is comparable to the left-hand time series. Nevertheless, the higher frequency sequences seem to be more easily distinguishable. In the case of the first dataset, our setup recorded a sensor measurement on average about every 12.5 cm. With the second dataset, the average speed is 14.45 km/h, so that the cyclist covers roughly 4.014 m in one second.

Consequently, our second setup performs a sensor data recording approximately every 1.36 cm. This allows a much more precise "scanning" of the road surface.

Third Data Set As already described in chapter 3.2, we chose the same procedure for data acquisition as for the second data set to record a third data set. The setup only included a different bicycle. The purpose is to check whether the previously trained models are transferable to a new bicycle.

When looking closer at the recorded test data, in particular the standard deviations in relation to the measured values of the individual sensor axes, it is noticeable that these in some cases differ significantly from the training data. Compared to the second data set, almost all standard deviations are about 20-40% lower. Even more significant differences can be found with the gyroscope values, which sometimes deviate by a factor of 3 to 4. The reason for this is most likely the choice of a different bicycle and the slightly different mounting of the XDK support (see figure 1).

Using the third data set, we tested the models from chapter 4.2, which we trained using the second data set. The differences between the training and test data are reflected in the classification results of the models: ResNet achieves the highest accuracy with 26.459%, InceptionTime reaches 26.07%, and all other models fall below the 20% mark. This results conclude that the data from chapter 4.2 is not sufficient to easily transfer the models to any bicycle. Besides the choice of the bicycle, the mounting of the XDK holder and the tire pressure might also influence the recorded sensor data. To train a robust model that delivers solid results regardless of all these factors, the training data must represent these elements.

4.3 Comparison to Related Work

Chapter 2 presented the work of Hoffmann et al. [3], which distinguishes three classes of bicycle tracks: smooth, rough and bumpy. They covered one track with a distance of 14 km 16 times in total and recorded accelerometer data at 37 Hz. The best classification result is a mean validation accuracy of 77.457%, determined by tenfold cross-validation. With respect to the present work this is most comparable to the results of the first data set, since we used a similar sampling rate with 32.5 Hz. However, in addition to the accelerometer data, we also used gyroscope data, which were not available in the case of Hoffmann et al. On the other hand, this work distinguishes nine classes, whereas Hoffmann et al. distinguish three. The data set of this work is smaller by a factor of about 2.5 and we attempted to avoid multiple runs on the same track. Finally, the highest mean validation accuracy with respect to the first data set is 77.645% (InceptionTime) which is slighlty higher than the best result of Hoffmann et al. (77.457%). However, when using our second data set (294 Hz), the validation accuracy of InceptionTime is significantly higher: 99.183%. Itt should also be noted that our training and test data include a greater variety of tracks compared to Hoffmann et al., while avoiding multiple runs of the same routes as far as possible. This makes our models particularly suitable for real-world applications.

The second work discussed is a paper by Litzenberger et al. [4]. They selected three routes of 100 m length each and have run each of the three routes nine times. They combined three different tire pressures (3, 4 and 5 bar) with three different speeds (10, 20 and 30 km/h). They also recorded accelerometer data, firstly with a smartphone (100 Hz) and secondly with a sensor device (500 Hz). Here the authors distinguish the classes of asphalt, cobblestone and gravel, with the highest mean validation accuracy at 99.2% (500 Hz) and 97.9% (100 Hz) respectively, determined with a five-fold cross-validation.

This is comparable to the results of the second data set of this study, since the sampling rate of 294 Hz is almost in the middle of the sampling rates used by Litzenberger et al. There are also differences between the two works: The data set used in this paper is significantly larger and the tracks to be predicted are much more diverse. In contrast to Litzenberger et al., we additionally use gyroscope values. When predicting nine classes, InceptionTime achieves the highest average validation accuracy of 99.18% within the tenfold cross-validation, which is minimally lower than the best result of Litzenbeger et al. (99.2%), but on nine classes making the results more practically relevant.

In order to make our results more comparable to the work of Litzenberger et al., we also tested the classification for three classes - asphalt, cobblestone and gravel. For this purpose, we combined the road types of the second data set so that there is no distinction between the road conditions smooth, rough and bumpy. Consequently, the three combined classes of asphalt, cobblestone and gravel now comprise 1,735,500, 1,887,500 and 2,009,000 measurements per sensor channel. Using this data, we trained the two residual networks InceptionTime and ResNet using the previously determined parameter settings from chapter 4.1. We performed a tenfold stratified cross-validation, whereby the models trained in each run for 500 epochs and the highest validation accuracy is used for evaluation. In this way, the residual nets slightly exceeded the results previously achieved with nine classes: The mean validation accuracy of the ResNet is 99.54% with a standard deviation of 0.163%. InceptionTime performs slightly better with 99.72%, only the standard deviation is a bit higher with 0.213%. To sum up, our models (99.72%) have a slighty higher prediction accuracy compared to Litzenberger et al. (99.2%) At the same time, our route selection for training and testing is significantly larger and more varied, so that our models should be way more suited for use in real-world applications.

5 Conclusion and Future Work

This paper describes the development of an approach for condition assessment of cycling tracks. In total, we collected three different data sets: Two training sets with different sampling rates (32.5 Hz and 294 Hz) as well as a test set using a different bicycle. For the first data set, the two residual networks achieved the highest validation accuracy of about 77%. However, all models show difficulties in recognizing certain classes, and mix-ups often occur. A closer look at these classes led to the assumption that a higher sampling rate could solve these problems. With the second data set, all models were able to achieve a significantly higher validation accuracy with the residual nets achieving 99% accuracy.

The evaluation of the transferability of the models to a different bicycle did not lead to good results. The recorded data of the third set differs too much from the training values of the second data set, so that none of the models can make reliable predictions. Once again, the two residual networks record the highest test accuracy, but they reach only about 26%. Despite the poor performance of the third data set, the initial results suggest that this is a very promising approach. The models developed provide higher validation accuracy than existing approaches. At the same time, we use a much larger and more varied selection of cycling tracks, which results in robust models that should be more suitable for use in real-world applications.

In future work, a variety of routes, bicycles, riders, speeds and tire pressures should be taken into account when creating the training data. Furthermore, the final setup for the transmission of sensor data could be adapted so that it can be used for the annotation of training data. For this the already developed bluetooth labelling app would have to be adjusted only slightly. In this way, the user experience could be made

much more convenient compared to the previous recording with the XDK's memory card. Lastly, the final test setup could be made more user-friendly by removing the smartphone dependencies from the current architecture. At the moment, the GPS sensor of a smartphone is still required. To resolve this dependency, the Bosch XDK could be extended with a corresponding sensor via its GPIO ports. Finally, the XDK needs a Wifi-hotspot - for this purpose a mobile router could be mounted on the bicycle. With these changes, the recording and classification of the road condition could be completely automated, so that the cyclist can fully concentrate on the riding itself.

References

1. EMU: Klimaschutzprogramm 2030 der Bundesregierung zur Umsetzung des Klimaschutzplans 2050. https://www.bundesregierung.de/breg-de/themen/klimaschutz/klimaschutzprogramm-2030-1673578 (2019)
2. Maerschalk, G., Krause, G., Socina, M., Köhler, M., Stöckner, M.: Daten und Methoden für ein systematisches Erhaltungsmanagement innerörtlicher Straßen. Forschung Straßenbau und Straßenverkehrstechnik (1079) (2013)
3. Hoffmann, M., Mock, M., May, M.: Road-quality classification and bump detection with bicycle-mounted smartphones. In: Proceedings of the 3rd International Conference on Ubiquitous Data Mining-Volume 1088, CEUR-WS. org (2013) 39–43
4. Litzenberger, S., Christensen, T., Hofstätter, O., Sabo, A.: Prediction of road surface quality during cycling using smartphone accelerometer data. In: Multidisciplinary Digital Publishing Institute Proceedings. Volume 2. (2018) 217
5. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research **12** (2011) 2825–2830
6. Chen, T., Guestrin, C.: Xgboost: A scalable tree boosting system. In: Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. (2016) 785–794
7. Chollet, F., et al.: Keras. https://keras.io (2015)
8. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X.: TensorFlow: Large-scale machine learning on heterogeneous systems (2015) Software available from tensorflow.org.
9. Breiman, L.: Random forests. Machine learning **45**(1) (2001) 5–32
10. Friedman, J.H.: Greedy function approximation: a gradient boosting machine. Annals of statistics (2001) 1189–1232
11. Cortes, C., Vapnik, V.: Support-vector networks. Machine learning **20**(3) (1995) 273–297
12. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**(8) (1997) 1735–1780
13. LeCun, Y., Boser, B., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W., Jackel, L.D.: Backpropagation applied to handwritten zip code recognition. Neural computation **1**(4) (1989) 541–551
14. Fawaz, H.I., Forestier, G., Weber, J., Idoumghar, L., Muller, P.A.: Deep learning for time series classification: a review. Data Mining and Knowledge Discovery **33**(4) (2019) 917–963
15. Fawaz, H.I., Lucas, B., Forestier, G., Pelletier, C., Schmidt, D.F., Weber, J., Webb, G.I., Idoumghar, L., Muller, P.A., Petitjean, F.: Inceptiontime: Finding alexnet for time series classification. arXiv preprint arXiv:1909.04939 (2019)
16. Bai, S., Kolter, J.Z., Koltun, V.: An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. arXiv preprint arXiv:1803.01271 (2018)