Exercise PC Design & Development / Systems and Environments

Assignment 1

11908757 Stefan Haslhofer

1. Remarks

My dataset used for the vehicle type classification may share similarities with the dataset of Jonas Reichhardt as we were meeting up and recording at the same location. Nonetheless, I did the assignment completely on my own.

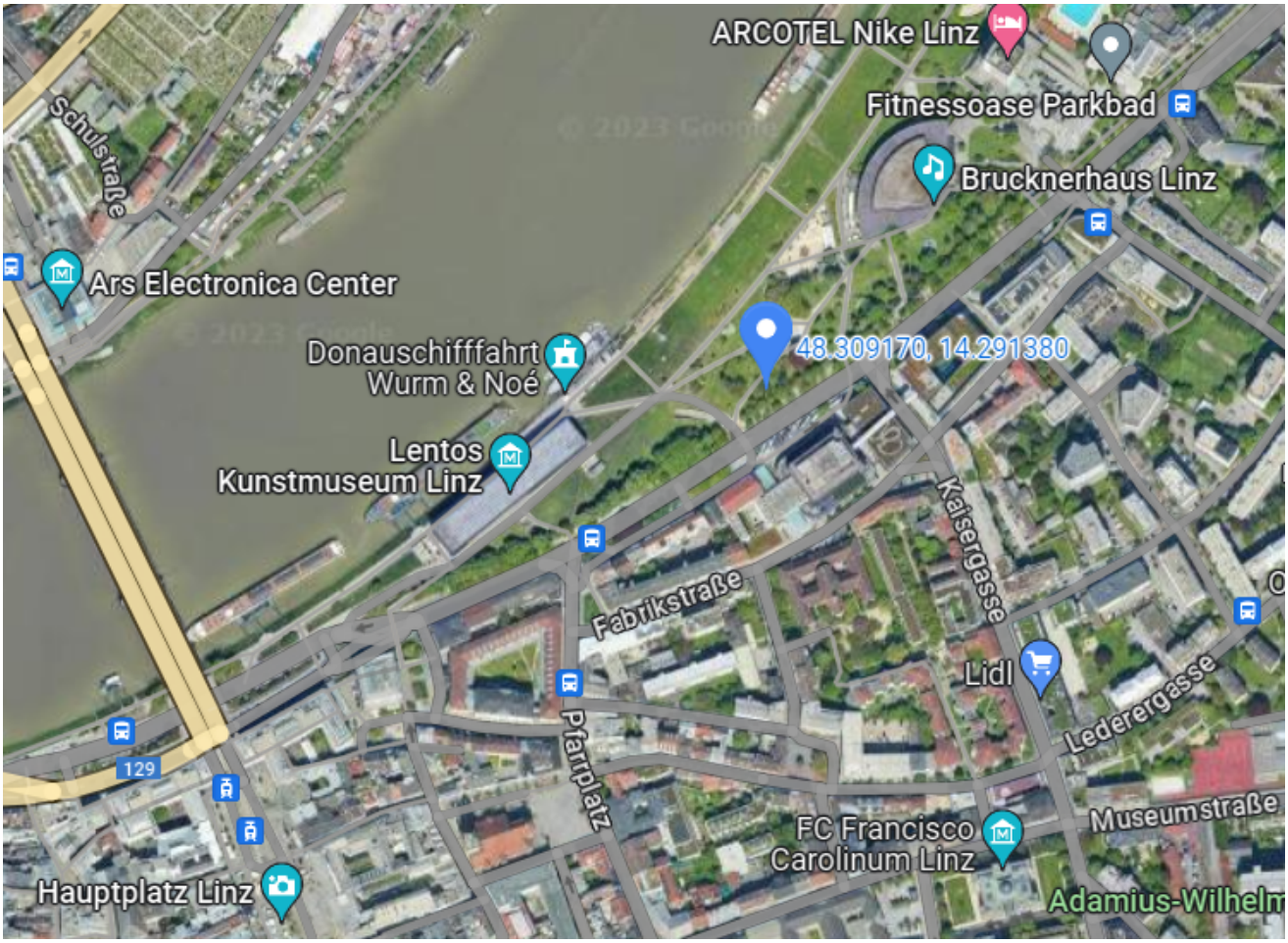
Due to cold weather, I had to reduce the vehicle types to only medium and heavy as there has not been a single 'light vehicle' (motorbike) in sight.

There is a heavy bias on cars which seems reasonable as most registered motorized vehicles are indeed cars. Furthermore, I recorded the first dataset (used for the vehicle type classification) during the evening rush hour, which meant nearly all vehicles were heading out of town. Unfortunately, this rendered the first dataset useless for the driving direction classification. Hence, I created a second less biased dataset.

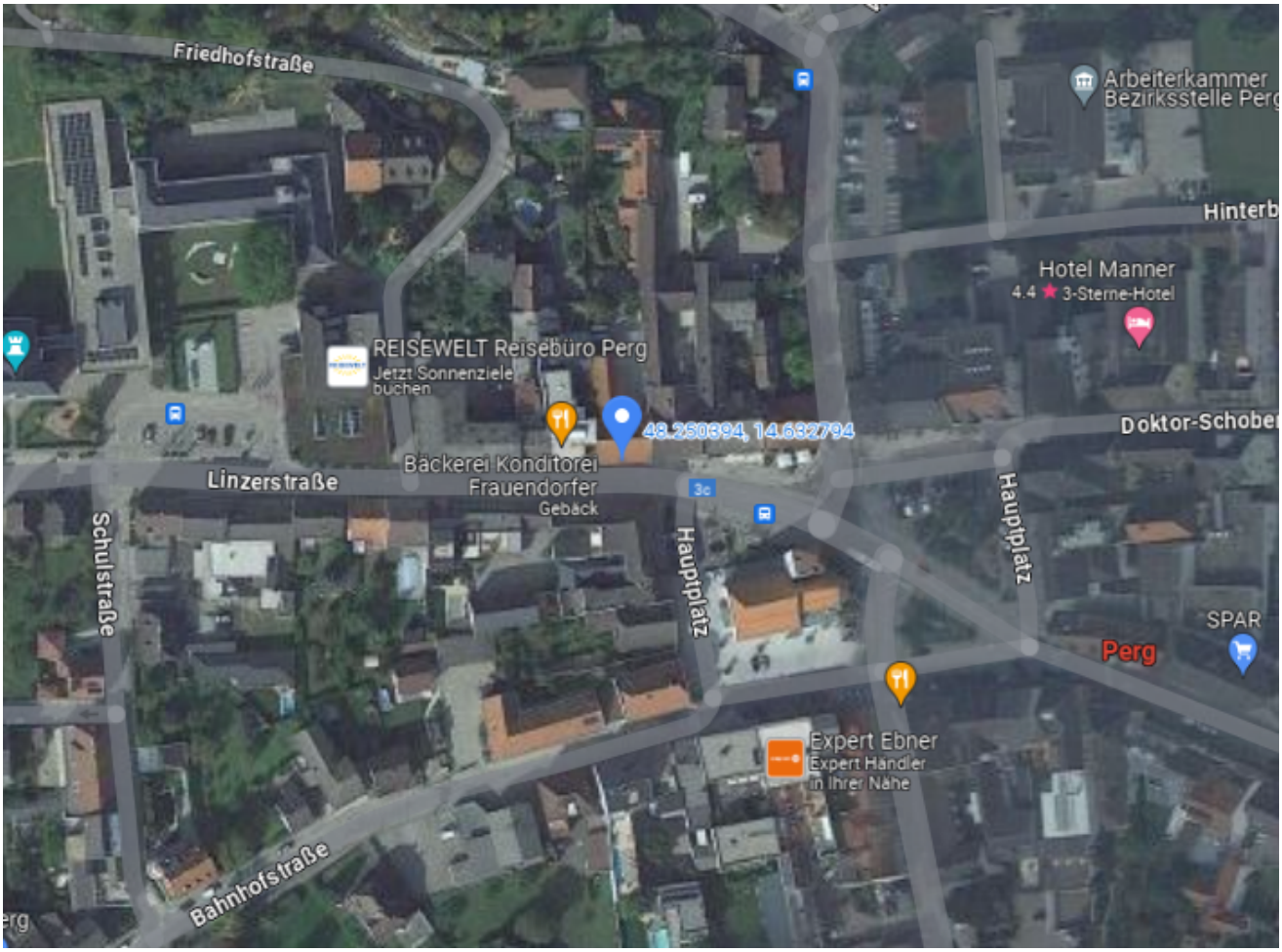
Concerning the raw audio we should include in our submission: I merged all files to one big *wav-file* per classification task. Therefore, I will hand in the **whole raw dataset** alongside all labels with start- and end timestamps as well as all feature sets (according to https://moodle.jku.at/jku/mod/forum/discuss.php?d=118760).

Python code: https://github.com/StefanHaslhofer/AudioFeatureExtractor

Recording location of vehicle type classification dataset:



Recording location of the driving direction classification dataset:

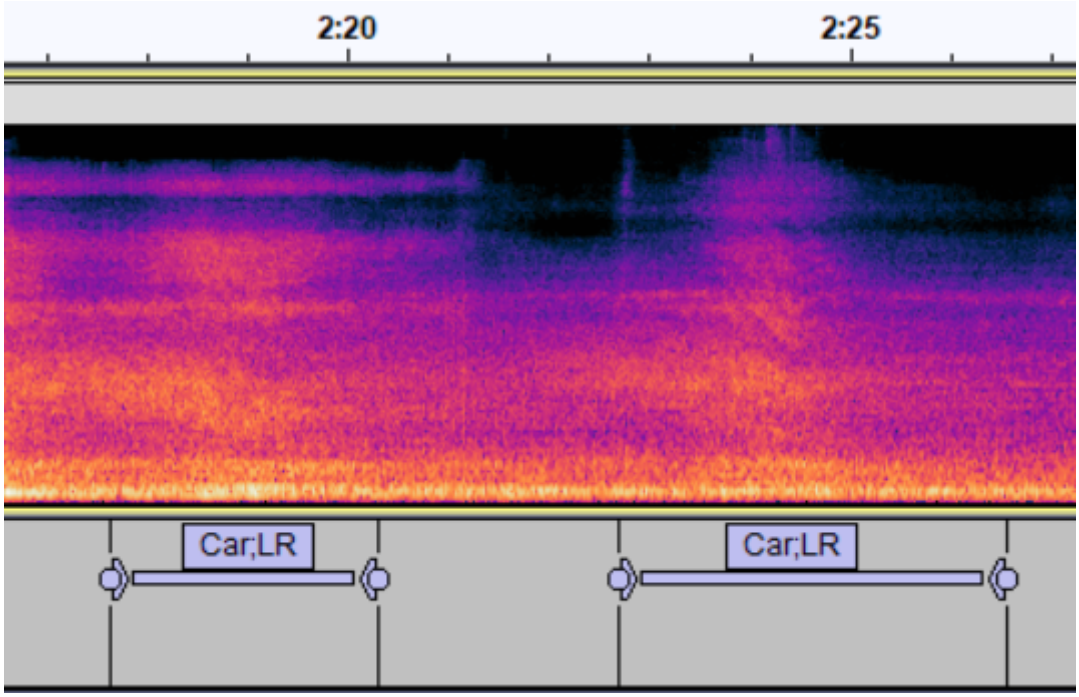


I recorded multiple short videos of the traffic with my mobile phone at approximately the distances shown in the sketch for the sample recording in the assignment sheet.

2. Pre-processing

At first, I merged the raw video data into one large *mp4*-file for each classification task which I converted to *wav* and then labelled within the audio editing software *Audacity*. Due to the recording location having strong background noise further pre-processing was needed. However, *Audacity* provides a noise removal tool which I used to reduce background noise by 6 dB.

Next, I switched to a spectrogram view making it much easier to distinguish passing vehicles from remaining noise as some frequencies suddenly spike in amplitude. Furthermore, I used the video as visual evidence to help me determine the correct driving direction for each sample. Labels can be exported from *Audacity* with start and end timestamp.



start	end	vehicle_type	direction
137.657700	140.323168	medium	LR
142.714615	146.575807	medium	LR

Note: After exporting the labels to a *csv*-File I renamed *Car* to *medium* and *Truck*, *Bus* to *heavy*.

3. Feature extraction

Next, I implemented a python script slicing the large audio into multiple smaller pieces, one for each label (start and duration indicated by the timestamps). For each audio slice I extracted 13 features based on Mel-frequency cepstral coefficients (MFCCs). I used the *librosa*-library's MFCC implementation which breaks the audio into windows and per default calculates 20 coefficients for each window. The chosen window size is equivalent to the number of coefficients in milliseconds.

However, I found that the overall number of correctly classified samples peaks at around 40 MFCCs for the used classifiers.

The following table lists all extracted features and why I initially chose them:

feature	remark

feature	remark
mean	I suspected that there must be a correlation between vehicle size/distance to the vehicle and the mean energy for each MFCC.
mean squared	I added the squared mean to emphasize the larger overall sound energy of more powerful vehicles.
standard deviation	Looking at the spectrogram, one may notice that heavier vehicles have a strong representation in lower frequencies, whereas lighter vehicles tend to be more spread out over the whole spectrum which means there could be a noticeable difference in the deviation from the mean.
variance	I also included variance for the same reason as standard deviation.
median	Similar to mean, the larger and closer a vehicle is the higher the amplitudes will be for each frequency, therefore the median energy should also be larger.
max energy	I thought the dominant amplitude of heavier/closer vehicles stands out compared to the dominant amplitude of lighter/distant vehicles.
max energy bin	I wanted to compare the dominant MFCCs because the maximal energy on its own may be misleading. For example: the amplitude of the loudest frequency (*max energy*) of a truck and a motorbike could be roughly the same even tough the motorbike's dominant frequencies are higher.
min energy	Similar to the *max energy* feature.
min energy bin	Related to *max energy bin* feature: the least dominant MFCC of two vehicle classes may differ even tough the largest amplitude does not.
q1	The frequencies do not seem to be distributed the same for all classes according to the spectrogram. Hence, the quartiles might be different.
q3	see *q1*
skewness	After studying the spectrogram, I noticed the frequency distribution of heavier vehicles is possibly skewed to lower frequencies. Frequencies of lighter vehicles on the other hand seem to be more symmetrically arranged.
interquartile range	If frequency distribution differs widely between classes, so does the interquartile range.

In general I assumed that features that provide information about frequency distribution are more useful to determine the vehicle type whereas features that hold information about the loudness (energy/amplitude) are a good indication for distance measuring and therefore direction (vehicles going from left to right are closer and should therefore be louder).

4. Classification of vehicle type

Weka was used for classifications.

After evaluating the features I found that all vehicle types tend to be silent in a similar frequency range and also have akin maximal amplitudes, rendering *max energy*, *max energy bin*, *min energy* and *min energy bin* useless. Therefore I ignored them in the vehicle type classification, which resulted in an average 2% precision increase.

a) J48

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0,950	0,714	0,884	0,950	0,916	0,301	0,570	0,855	medium

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0,286	0,050	0,500	0,286	0,364	0,301	0,570	0,323	heavy
Weighted Avg.	0,851	0,615	0,827	0,851	0,833	0,301	0,570	0,776	

Using *J48* I achieved an accuracy of 85.1%. I tried to optimize the result by tuning the *confidenceFactor* and the *minNumObj* parameter in *Weka* but this only changed the outcome slightly, therefore I sticked to the default values: 0.25 for *confidenceFactor*, 2 for *minNumObj*.

b) Naïve Bayes

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0,875	0,429	0,921	0,875	0,897	0,404	0,731	0,919	medium
	0,571	0,125	0,444	0,571	0,500	0,404	0,731	0,393	heavy
Weighted Avg.	0,830	0,383	0,850	0,830	0,838	0,404	0,731	0,841	

Naive bayes has no parameters in *Weka*.

c) kNN

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0,963	0,714	0,885	0,963	0,922	0,337	0,693	0,909	medium
	0,286	0,038	0,571	0,286	0,381	0,337	0,693	0,352	heavy
Weighted Avg.	0,862	0,613	0,838	0,862	0,842	0,337	0,693	0,826	

At first, I increased k to 10, which resulted in the misclassification of all heavy vehicles. The lack of heavy vehicles in the dataset made the kNN-algorithm struggle with false positives for large k in general. In the end I went for k = 3 to get the best possible result.

d) Multilayer perceptron

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0,950	0,571	0,905	0,950	0,927	0,437	0,780	0,944	medium
	0,429	0,050	0,600	0,429	0,500	0,437	0,780	0,443	heavy
Weighted Avg.	0,872	0,494	0,859	0,872	0,863	0,437	0,780	0,870	

The multilayer perceptron performed best using the default parameters:

- learning rate: a small change did not change the performance, however a large increase/decrease worsened accuracy

- momentum: same as with learning rate
- training time: the accuracy was not getting better with increased training time but it got significantly worse when decreased

Summary

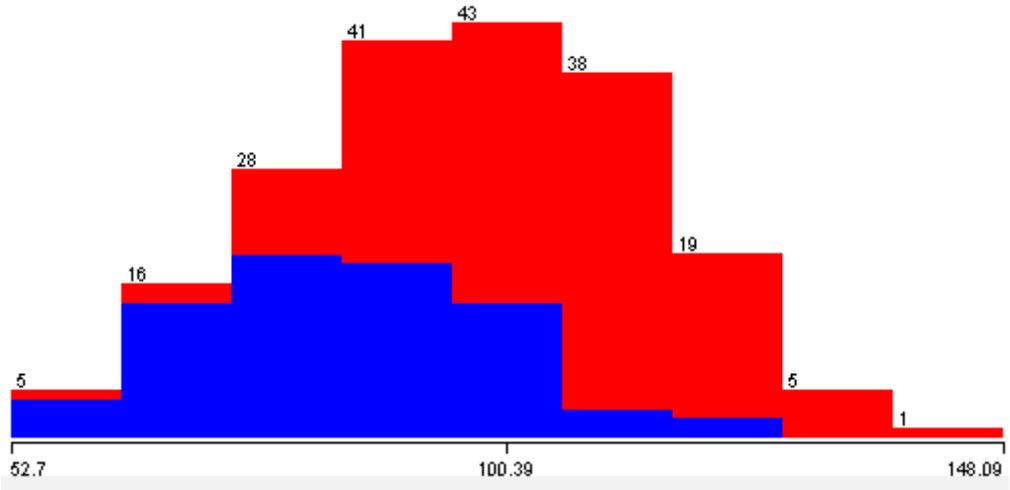
Measured on accuracy the multilayer perceptron performed best. However, the naïve Bayes achieved the lowest false positive rate. As mentioned before, the dataset used for the vehicle type classification is highly imbalanced which makes the PRC area a suitable metric. The highest score is achieved by the multilayer perceptron (87%), closely followed by the naïve Bayes (84.1%).

5. Classification of driving direction

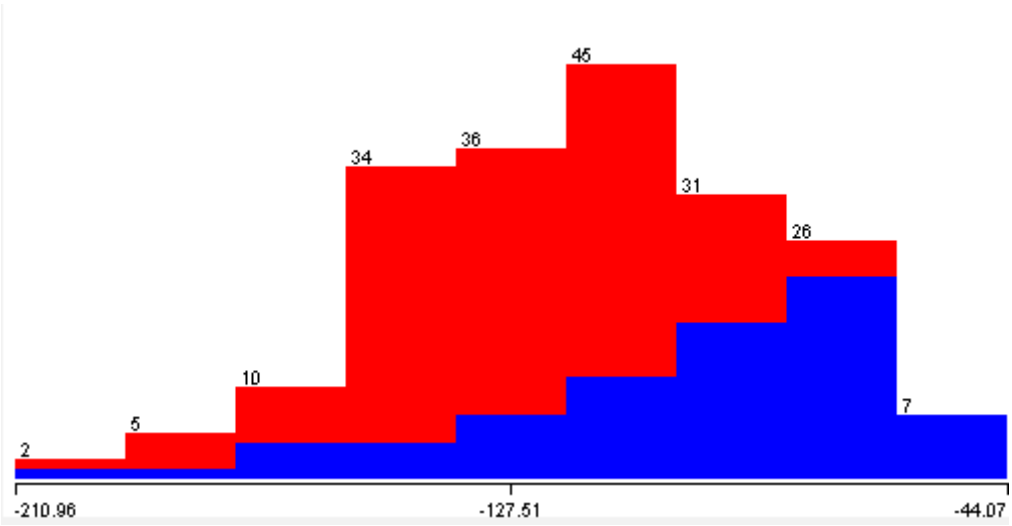
Weka was used for classifications.

Similar to the vehicle type classification I removed *min energy bin* and *max energy bin* because both are the same for all vehicles. Nevertheless, I kept the *min energy* and *max energy* features because these indeed seem to differ in the two classes LR (left-to-right) and RL (right-to-left).

Max energy:



Min energy:



By estimating the expressiveness of each feature with the help of the *Weka* visualization tool I eliminated further features that did not seem to help the cause. At last I was only left with:

- mean
- standard deviation

- variance
- max energy
- min energy

a) J48

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0,635	0,090	0,810	0,635	0,712	0,579	0,758	0,651	LR
	0,910	0,365	0,804	0,910	0,854	0,579	0,758	0,778	RL
Weighted Avg.	0,806	0,261	0,807	0,806	0,800	0,579	0,758	0,730	

The accuracy for the driving direction classification with J48 peaks at 80.6%. Identical to the vehicle type classification task the default parameters produced the best score: 0.25 for *confidenceFactor*, 2 for *minNumObj*.

b) Naïve Bayes

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0,689	0,131	0,761	0,689	0,723	0,570	0,827	0,812	LR
	0,869	0,311	0,822	0,869	0,845	0,570	0,827	0,842	RL
Weighted Avg.	0,801	0,243	0,799	0,801	0,799	0,570	0,827	0,831	

Naive bayes has no parameters in *Weka*.

c) kNN

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0,622	0,049	0,885	0,622	0,730	0,629	0,812	0,787	LR
	0,951	0,378	0,806	0,951	0,872	0,629	0,812	0,831	RL
Weighted Avg.	0,827	0,254	0,835	0,827	0,819	0,629	0,812	0,814	

I increased k until the accuracy declined again, which was at k = 15. In contrast to the vehicle type classification the false positive rate is drastically lower, which is probably attributable to the larger and more balanced dataset.

d) Multilayer perceptron

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0,689	0,098	0,810	0,689	0,745	0,613	0,835	0,804	LR
	0,902	0,311	0,827	0,902	0,863	0,613	0,835	0,849	RL

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
Weighted Avg.	0,821	0,231	0,820	0,821	0,818	0,613	0,835	0,832	

The initial accuracy with default parameters applied was around 80%. At first I doubled the training time as this would lead to an instant 2% growth in accuracy. I also found that even small changes in momentum and learning rate reduces the percentage of correctly classified samples. Additionally, changing the remaining parameters will not further improve the result.

Summary

Overall accuracy seems to be lower, but so is the false positive rate. The dataset is larger and far more balanced which makes the ROC area a suitable metric. The ROC area is noticeably larger in all 4 classifiers. Even tough kNN has a better accuracy, the multilayer perceptron is able to achieve higher scores in FPR and ROC area, which makes it the better classifier.