

TUM Data Innovation Lab

Munich Data Science Institute (MDSI)

Technical University of Munich

&

Industrieanlagen-Betriebsgesellschaft mbH

Final report of project:

AI based Analysis of GNSS Data in a Railway Environment

Authors	Tobias Grasberger, Louay Helali, Yash Thirani, Charlotte Winkler
Mentor(s)	Dr. Stefan Baumann, Nikolas Dütsch, Dr. Paulo Alexandre Silveira Mendes
TUM Mentor	Dr. Alessandro Scagliotti
Project lead	Dr. Ricardo Acevedo Cabra (MDSI)
Supervisor	Prof. Dr. Massimo Fornasier (MDSI)

Feb 2024

Abstract

Satellite Navigation is on track to potentially revolutionise railway management systems. Global Navigation Satellite Systems (GNSS) refers to any satellite-based positioning system that provides location and time information on Earth. Examples include GPS and GALILEO. Currently, train positions are monitored by Eurobalises, inertial systems and odometers. These localise the trains into fixed blocks of the railway tracks. Object localisation using GNSS can be used to continuously report accurate train positions, allowing moving blocks to be used for future rail management. Due to the critical nature of train management, GNSS data must meet high standards of accuracy, availability, integrity, and robustness against signal blockages, multipath and interference. Such interference can stem from other radiofrequency transmitters or malicious jamming and spoofing.

The data for this analysis was obtained from the Swiss Federal Railways (SBB) and collected during measurement campaigns over Switzerland's entire standard gauge network. No reliant labels were present within the data, but these are necessary to use Supervised Machine Learning methods. Thus, to conduct a feasible analysis to indicate jamming or spoofing, the data was grouped into categories defined artificially from existing features. Afterwards, the data was examined with Machine Learning methods to identify and flag faults caused by interference. The most promising classical AI methods were Support Vector Machines, XGBoost and Random Forest. These performed similarly across three different subsets of the available features. This allowed the investigation of the influence of different jamming parameters and the time-dependencies within the data. For the future progress of the project, experiments evaluating the artificially created labels and gathering data within a controlled environment or with truly reliable information on jamming events is crucial. This data should then be used to test the AI method's actual performance.

Contents

1 Introduction

Currently, train management is based on fixed track sections occupied by the trains. In the future, the goal is to use the exact and continuous location of each train instead. This would transform rail management from a fixed block to a moving block operation as a part of the European Train Control System (ETCS).[10] The location of each train can be measured using Global Navigation Satellite Systems (GNSS). However, due to the safety criticality of rail management, this positional data needs to comply with high quality criteria regarding reliability, availability, maintainability, and safety. Especially signal interference of different kinds is a challenge when relying on GNSS. If the train is shielded by environmental obstructions such as tunnels, bridges, large buildings, or mountains, the satellite signals might not reach the train antenna. The satellite signals might also be reflected and reach the antenna multiple times with a delay, creating multipath events. The correct computation of one's location can additionally be deliberately prevented by jamming or spoofing. During such an event, the GNSS signal is intentionally overshadowed, making either the GNSS signal non-receivable (jamming) or the receiver is fooled into computing a wrong location (spoofing).[10]
Due to the promising possibilities of using GNSS for rail management, the "European Global Navigation Satellite System based Map Assisted Train Localisation for ERTMS" (EGNSS MATE) was launched. This project is jointly conducted by the Swiss Federal Railways (SBB), the Industrieanlagen-Betriebsgesellschaft mbH (iABG) and the German Aerospace Center (DLR). Within the scope of this research project, SBB operates a measuring train to collect data to evaluate the reliability of GNSS, based on the aforementioned quality criteria. This measuring train commutes the entire SBB standard gauge network for a year and has multiple GNSS receivers. Some analysis of this data was reported by R. Ehrler et al. in [10].

1.1 Problem definition and goals of the project

The dataset collected and maintained by SBB contains a vast amount of real-world data and has yet to be fully analysed. The long-term goal is to estimate the reliability of GNSS data in different environments, extract the most likely areas of intentional GNSS interference and use these to classify the individual railway tracks depending on the estimated possibility of using GNSS for train management. Additional measures such as Eurobalises can be used at tracks classified as less reliable to ensure safe train localisation. As one contribution to this larger goal, our project aimed at data cleansing and first analyses using Machine Learning methods. During data cleansing, the goal was to extract features characterising malfunctions. Due to time constraints, the focus was on jamming events, leaving spoofing events for future analysis. During the analysis, we aimed to classify patterns perceived in the GNSS data based on the different causes. As no label is collected, directly applying Supervised Machine Learning models was impossible. Therefore, the first milestone of this project was to create artificial labels. These classify the data at each timestep, depending on whether the 'train receives a clear signal', the 'train is at a standstill', the 'train drives through a tunnel', 'environmental objects disturb the signal', or 'a jamming attack causes interferences'. Further, we aim to use Machine Learning models to learn and rebuild these labels. While developing these models, we

additionally aim to investigate the importance of the different features further, expanding our knowledge of the possibilities of classifying GNSS data.

1.2 Global Navigation Satellite Services

The Global Navigation Satellite System describes a constellation of satellites used to determine the location of receivers on Earth. GNSS encompasses various satellite navigation systems from different organisations. These include GPS (Global Positioning System), which is operated by the USA; GLONASS (Global Navigation Satellite System), operated by Russia; Galileo, operated by Europe; and BeiDou, which is operated by China. Each consists of at least 24 satellites circling the Earth at a specific distance.

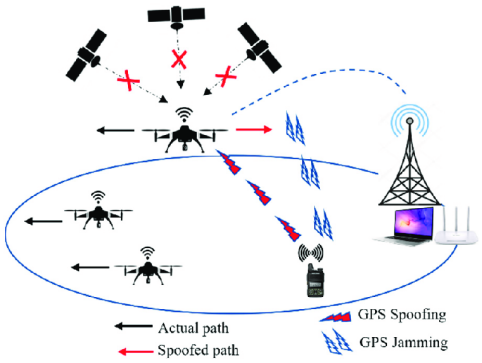


Figure 1: Visualisation of jamming and spoofing

1.2.1 Positioning and interference

The satellites transmit, among other data, their current position and time stamps. Receivers on earth then can use this data to compute their own current position. Theoretically, four signals are enough to compute the location and time at any GNSS receiver on Earth. However, more satellites can be used to increase accuracy and mitigate interference.
Signal disturbances occur frequently in satellite communication and have various forms. Environmental obstructions due to objects such as tunnels, bridges or high buildings are a common cause. Other radiofrequency transmitters such as radars, TV transmitters and mobile communication base stations can account for unintentional interferences. Additionally, malicious jamming and spoofing, as intentional attacks, may affect the channels.
Jamming describes the intentional interference or disruption of signals. A jammer emits interference signals stronger than the satellite signals. This leads to a rise in the noise floor, covering the true satellite signals.
Spoofing is characterised by the emulation of fake signals to confuse or mislead a GNSS receiver. An example of spoofing is the "Record and Replay" method. The spoofer records a GNSS signal and emits it with a time delay. This method is comparatively easy to detect, but more complex spoofing methods, which use rather expensive and sophisticated tools, pose a severe threat.

1.3 Theoretical background

The dataset used consists of sequential data, e.g., it contains one value of each feature for every timestep. Therefore, this data can be classified by only considering the respective features at each timestamp. This enables the use of non-time-dependent algorithms. Alternatively, the sequential nature of the data can be exploited, incorporating the change in features between the timesteps. This allows the use of time-dependent algorithms. Non-time-dependent algorithms generally offer better interpretability of features and can be trained and validated faster. However, they do not incorporate sequential information. We experimented with the following algorithms:
Support Vector Machines classify data by finding the largest possible margin between different classes. While a SVM naturally only distincts between two classes, it can be expanded to multi-class usage by introducing one-vs.-one- or one-vs.-many-techniques. By transforming the input data using a kernel, non-linear decision boundaries are possible. Its advantages include dealing with outliers through soft-margin SVM, and that its decision reasons are visually interpretable for low-dimensional data. Its disadvantage is that it generally works best in cases of binary classification. (c.f. [4])
Logistic Regression uses the softmax function to convert a regression score into the probability of belonging to each class. Like SVM, one-vs.-one- or one-vs.-many-techniques are necessary to expand the usage from a general binary classification to a multi-class distinction. Its advantages are that its estimates are comparatively reliable, and the trained model weights imply feature importance. Its disadvantages are that the model expects a linear decision boundary and generally works best in cases of binary classification. (see for example [3] p.119)
XGBoost combines weak decision trees, built sequentially to correct previous errors, into a strong predictive model. One of its advantages is regularisation, an inherent advantage of ensemble models. It also has faster convergence and efficient computation compared to other algorithms. Its disadvantages include a lack of explainability and the challenge of interpreting results and understanding the model's inner workings. It also struggles with handling high cardinality in categorical features. (c.f. [7])
Random Forest ensembles many Decision Trees and uses a majority vote to classify the datapoints. Its advantages are that the decisions of individual trees are easily understandable, and the model can handle multiple classes. Its disadvantage is that it tends to overfit. (c.f. [1])

Time-dependent algorithms can incorporate sequential information and work well with complex data structures. However, they are challenging to train and tune, and their results tend to be less interpretable.
Transformers are one of the most frequently used forms of time-dependent algorithms. Their main idea is that the attention mechanism learns the importance of features and timesteps. Its main advantage is that the training is parallelisable and effectively captures global dependencies. However, its disadvantages are that it requires a large amount of data to train and is challenging to interpret.[8]
The core idea of **Convolutional Neural Networks** is that it uses convolutional layers to find timely distributed patterns in the data and pooling layers for dimensionality reduction. Its main advantage is that it is better at learning patterns in the data,

independent of the patterns' position. Additionally, it has a comparatively fast, parallelisable training process. Its main disadvantages are that it requires a large amount of data, struggles with long sequences, and its decision reasons are often incomprehensible.[9]
Long Short-term memory was first proposed by S. Hochreiter et al. in [2]. It improves recurrent neural networks (RNNs) by incorporating information on the current and past timesteps by saving contextual information in the cell state. The big advantage is its flexibility in defining the size and number of layers, but its decision reasons are often incomprehensible.
Time-series Clustering clusters sequential data into groups using a pre-defined similarity metric, which judges the similarity of characteristics such as frequency, mean, and variance. It could help judge the created labels and lead to a deeper understanding of data. However, the length of individual sequences needs to be predefined, and it is difficult to interpret high-dimensional data.[6]

2 Data Exploration

In this section, we first give an overview of the data and showcase its characteristics and inner relations based on several data analysis case studies. We then based our data processing steps on the previous analysis to prepare our Machine Learning models.

2.1 Data Overview

The Swiss Federal Railways (SBB) introduced a measurement train to assess the reliability, availability, maintainability, and safety of the railway communication system GSM-R using location estimates based on GNSS. This train continuously drives around Switzerland's standard gauge railway network and captures satellite data. The track covered from January until the end of March 2023 is visualised in Figure 2.

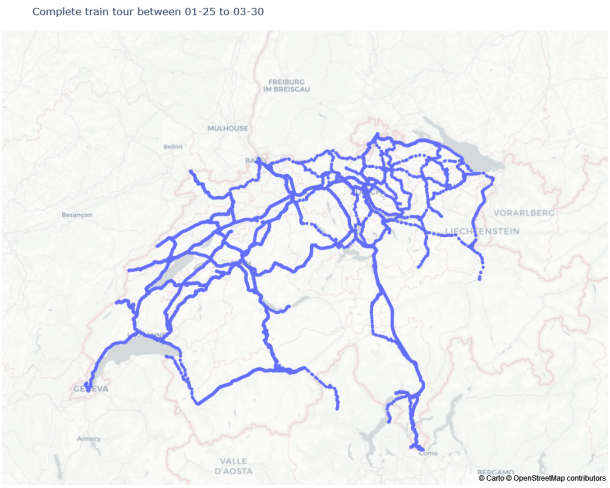


Figure 2: Train route of the measurement train between January and the end of March

The measurement wagon is equipped with receivers, and the data of two was made available to us throughout the TUM Data Innovation Lab to analyse. The first receiver

is called iMar iNAT-RQT-4003 and can integrate satellite data and signals from inertial systems, making the positional data extremely robust to jamming, spoofing and even environmental obstructions like tunnels. Therefore, it can be used as a reference system for localisation and navigation. The receiver provides detailed timestamped **positional data** about the train every 0.1 seconds, including information about the Longitudes, Latitudes and Altitudes, Velocities, Acceleration and Direction (roll, pitch and yaw) of the train and standard deviation and biases for all of the previously mentioned.

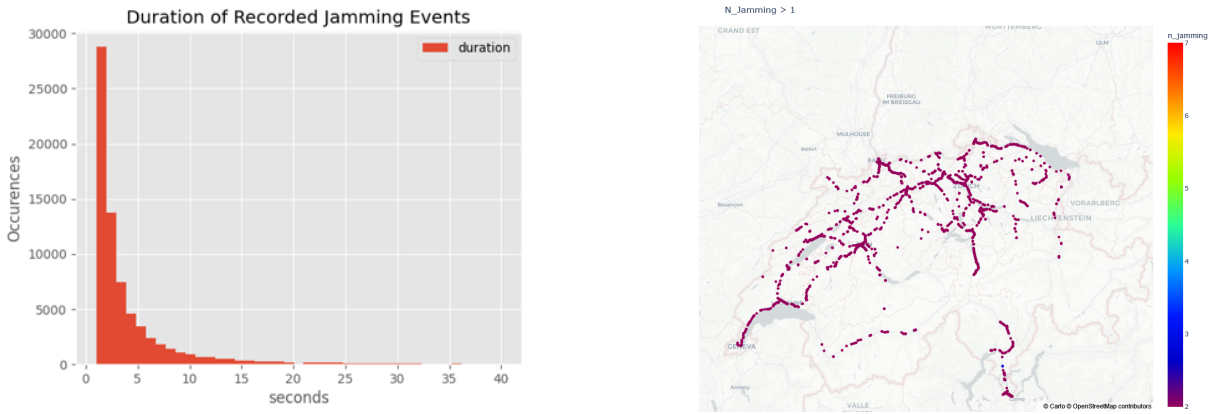
The other receiver is a Septentrio AsteRx-U SSRC7 and is based on GNSS data only. As a result, this receiver is more susceptible to jamming and spoofing attacks and fails to compute a correct location in tunnels. However, it has built-in jamming and spoofing detection mechanisms. The manufacturer has not disclosed how this internal algorithm works. Still, the receiver can report whether it thinks it is being jammed or spoofed and saves the satellite frequency band affected by the biggest disturbance.
The Septentrio receiver provides timestamped data every second. This includes **positional data**, consisting of the current longitude, latitude and altitude, velocity of the train and the standard deviations for all the previously mentioned features.
Furthermore, the receiver provides **interference data**, including the automatic gain control for each satellite band frequency, which describes the receiver's sensitivity. Additionally, *n_jamming* specifies the estimated number of satellite bands jammed by external interference, the frequency and bandwidth of the most prevalent jamming interference and a spoofing flag. This describes whether the receiver thinks it is currently under a spoofing attack.
Lastly, the receiver makes **satellite data** available, with an individual entry for each currently observable satellite, including the name of the satellite, the satellite system it belongs to, the duration of the connection to the satellite, the relative position, and the distance and doppler coefficient of the satellite with respect to the receiver. The received quality of the satellite signal is indicated by the reported signal strength-to-noise ratio C/N_0 and two boolean flags: First, whether the satellite thinks it is faulty and, secondly, whether the satellite is currently actively used to determine the receiver's position.

2.2 Data Analysis

In the following, we review several aspects of the provided data and present specific instances that explain its characteristics and internal relations.

2.2.1 The high false alarm rate of the jamming flag

The main issue with the provided data is that the jamming flag from the Septentrio receiver is overly sensitive and appears to have a high false positive alarm rate. The flag is set extremely frequently for a duration between one and three seconds, which is visualised in Figure 3a . This seems unreasonable, as we expect jamming events to have a longer duration. From the manufacturer's perspective, a large false-positive rate makes sense, as the typical user of such a receiver is rather interested in excluding any timepoint, where interference may be possible. However, this poses the severe problem that we can not fully trust the jamming flag of the receiver. This might cause unnecessary out-times, which affects service continuity. At least, the recorded instances, where the number of



(a) Duration of jamming events

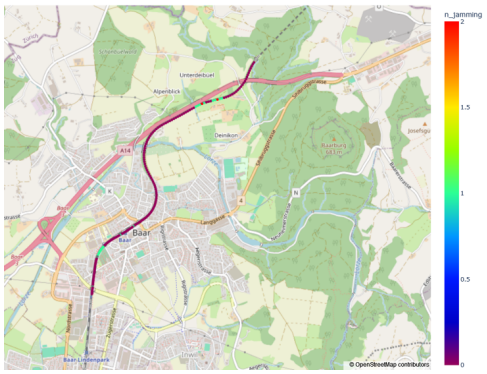
(b) Plot of jamming events where at least two bands were marked as jammed

Figure 3: Characteristics of jamming events

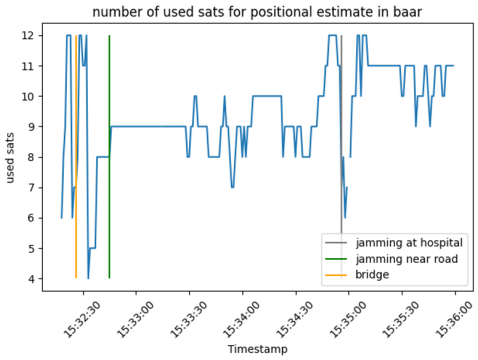
jamming events is reported to contain at least two or more jammed satellite bands, look more realistic than the shorter single-second events, as we can see in Figure 3b.

2.2.2 Behaviour during jamming interference

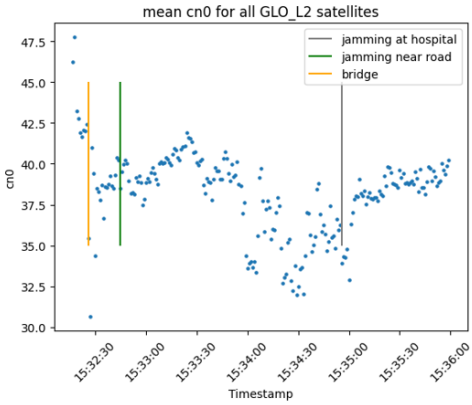
To evaluate the effect of jamming events on the aforementioned features, we analysed several incidents, from which we are presenting one here. These case studies were integral to understanding the relationship between the features more deeply.
On the 10th of February, we observed two jamming events in a small city called Baar, one near a hospital and the other near a big road. Both events are visualised in Figure 4 and last for about 15 seconds each. The vertical lines in grey and green mark the middle point of the interferences, and both the jamming events extend about 7 seconds to the right and left of the line. Additionally, we marked the timepoint, when the train passes under a bridge in orange.
In the jamming event near the hospital, marked in grey, we can observe small drops in the mean C/N_0 values of the received satellite signals of the GLONASS L2 frequency. Signals, even from the same satellite in other frequencies, are unperturbed. On the other hand, in the jamming event near the road, marked in green, we can observe the same effect, but this time, the GLONASS L1 frequency band is affected, while other bands are not. Besides those two drops, both mean C/N_0 values behave quite synchronously and drop simultaneously when the view is obstructed by buildings or similar objects. Furthermore, the C/N_0 values from the same satellites but in different non-jammed frequency bands are unperturbed.
Thus we can generally conclude that a jamming event is happening with a high likelihood if the mean C/N_0 value drops in a subset of the satellite frequencies but not all. If the mean C/N_0 drops for all bands, we expect the perturbance to be due to environmental effects like satellite view obstruction.
We see a drastic drop in the C/N_0 of all satellites when the train passes the bridge. This is visible in the mean C/N_0 plots in Figure 4c and Figure 4d. Simultaneously, the gain parameter, which describes the receiver's sensitivity for each satellite frequency, is



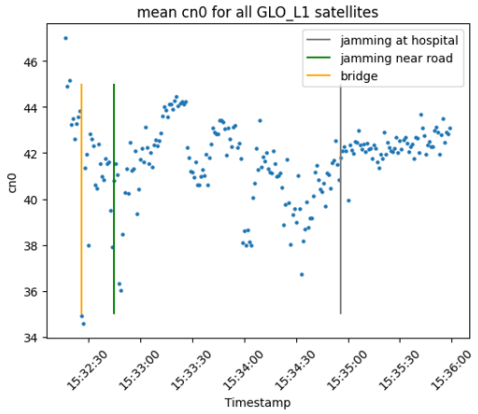
(a) Mapplot of the Baar incidents



(b) Number of used satellites during the jamming events



(c) Mean C/N_0 of all satellite signals, transmitted in the GLONASS L2 frequency



(d) Mean C/N_0 of all satellite signals, transmitted in the GLONASS L1 frequency

Figure 4: Characteristics of jamming events

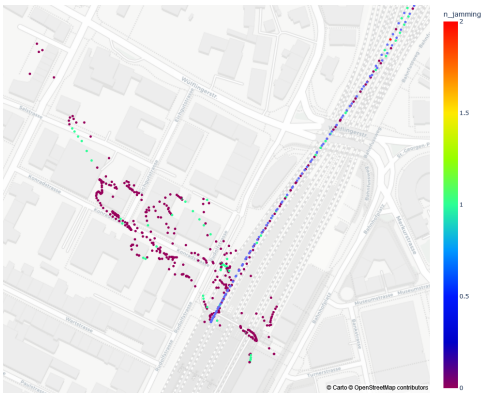


Figure 5: Multipath event including positional drift in Winterthur

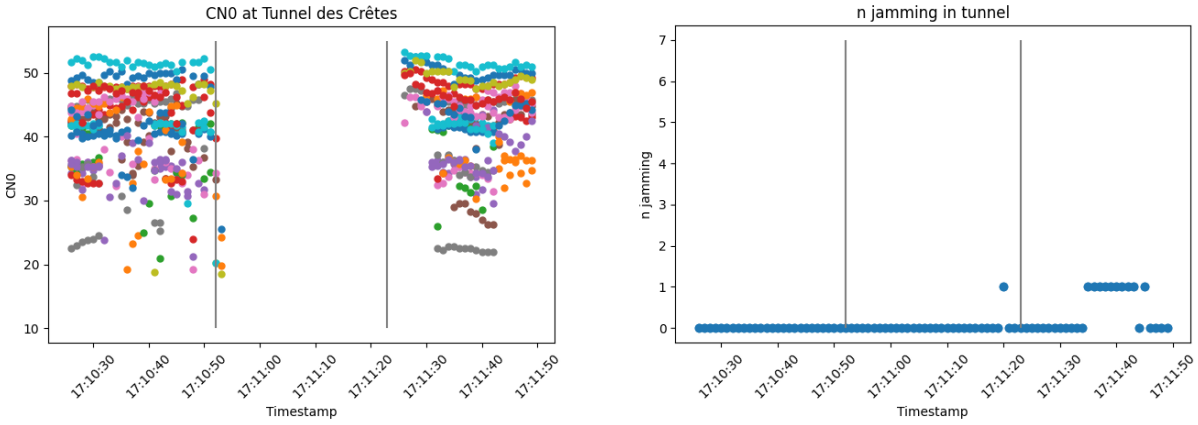
generally unaffected by the bridge. However, the data shows small drops in the gain parameters during the jamming event.

2.2.3 Behaviour during multipath events

Multipath describes the effect of satellite signals reflecting off surrounding walls or mountains to the receiver, leading to a delayed signal of the satellite arriving at the receiver. As a result, the receiving device might trust a false signal and calculate a wrong position. We captured the data of such an event happening in Winterthur displayed in Figure 5. The dark blue points in the plot are the positional estimate of the iMar receiver. The other points are positional estimates of the GNSS-based receiver, colour-coded by the reported number of jammed bands *n_jamming*. As we can see, the train stops at a station under a bridge, and the positional estimate drifts around the station. Interestingly, the spoofing flag of the receiver is not triggered, while the jamming flag is reporting short, irregular, but not concurrent, jamming attacks. The train stays in the station for about 10 minutes. The positional estimate of the inertially supported iMar receiver is unaffected by this multipath event.

2.2.4 Behaviour in Tunnels or during environmental effects

Generally, the GNSS-based receiver loses all satellite signals in a tunnel, as we can see in Figure 6b. Under short tunnels, bridges or other satellite visibility obstructions, the signal-to-noise ratio C/N_0 drops heavily. However, the GNSS receiver is, depending on the severity of the obstruction, usually able to maintain a correct positional estimate. The gain parameter is usually unaffected by a tunnel as is the *n_jamming* feature as seen in Figure 6a.



(a) C/N_0 of selected freq. bands during the tunnel

(b) Plot of jamming events, where at least two bands were jammed

Figure 6: Characteristics of tunnel events

2.3 Data Processing

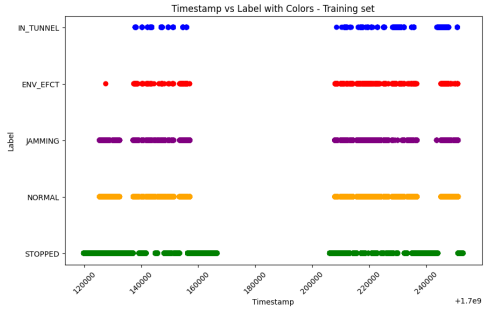
To use Supervised Machine Learning models, creating artificial labels and defining comparable datasets was necessary. This preprocessing of the data will be described in the following.

2.3.1 Creating artificial labels

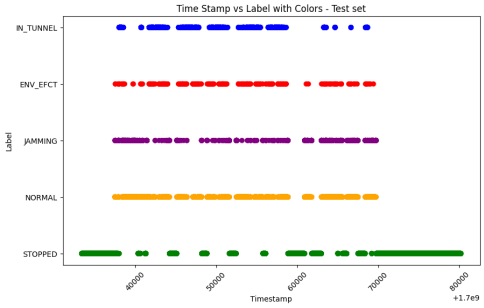
To be able to use methods from the field of Supervised Machine Learning, a label is required. Since no label is available within the existing data set, that reliably determines jamming or spoofing, artificial labels were created from the existing parameters. All data points were classified into *Normal*, *Jamming*, *Tunnel*, *Environment* and *Stop*.
For example, if the train is travelling at a speed below 11km/h, this timepoint is classified as *Stop* in the data set.
All data points likely influenced by surrounding objects such as mountains, bridges or tall buildings are classified as *Environment*. As described in the Section 2.1, the blockage of the satellite signal due to surrounding objects can be detected by a drop in the average carrier-to-noise ratio. If the mean C/N_0 drops below 36dB-Hz in all satellite bands, the datapoint is classified as *Environment*.
First, we evaluated the current position of the train to classify it as being in a tunnel. Therefore, we used a tunnel dataset ([11]), including the length and the approximate midpoint of all railway tunnels in Switzerland. As this did not include any information regarding the orientation of the tunnel, a computationally fast matching to the travelled railway tracks is challenging. We chose to classify any position within the shape spanned by the L1-norm of half the distance of the respective tunnel and its centerpoint as *Tunnel*. This led to far too many points being classified with *Tunnel*. During further development, we chose to instead analyse satellite connections to identify tunnels. Within large tunnels, hardly any satellite signal is observable. Therefore, we classify each datapoint, where we lose all satellite signals, as *Tunnel*. This approach leads to many time points where the train is close to the entrance or exit of a tunnel, not being classified as *Tunnel*. Further analysis showed that the average of the carrier-to-noise-ratios are usually below the threshold close to the tunnel ends, thus classifying the datapoints as environmental interference. Therefore, the *Environment* label covers very diverse causes of natural interference.
The major goal was to create a *Jamming* label that is as reliable as possible. One of the two GNSS receivers in the measuring train determines the number of frequency bands currently affected by jamming (*n_jamming*). As described in the Section 2.1, this value contains many false positives, as visualised in Figure 3a. Therefore, only timepoints with a value of *n_jamming* greater than one or a continuous duration of at least four seconds were classified as *Jamming*.
All timepoints, where none of the aforementioned criteria applies, were classified as *Normal*. This is the class for which reliable rail management based on GNSS localisation would most likely be possible without further measurements.

2.3.2 Training Scenarios

All Machine Learning models presented in the following were trained and tested on the same subset of the data for comparable analysis. This dataset was acquired during an in-person measuring campaign of iABG. This measuring campaign took place between the 15th and 17th of November, 2023. The last two days were used as training data, and the first day was utilised as the test set. This setup and the respective distribution of the labels are visible in Figure 7 and Table 1. In the following, we present three different training scenarios, which differ mainly in the features used.



(a) Label distribution within the training set



(b) Label distribution within the test set

Figure 7: Visualising the distribution of the labels of the training and test set

Label name	Instances in the training set	Instances in the test set
Environment (ENV)	1147 (1.23%)	457 (0.98%)
Jammed (JAMMING)	5956 (6.36%)	1816 (3.88%)
Normal (NORMAL)	30337 (32.41%)	15281 (32.67%)
Stop (STOPPED)	51812 (55.36%)	22500 (48.10%)
Tunnel (TUNNEL)	4346 (4.64%)	6721 (14.36%)
Sum of instances	93598	46775

Table 1: Label distribution in the training and test set

The full dataset As described in the Section 2.1, a multitude of interferences exist. As we aim to classify these into their different causes, different features representing these causes must be extracted. To allow the tracking of the train, using positional data is inevitable. A change in the gain or carrier-to-noise ratios seemed to occur whenever the satellites' signals were obstructed by an environmental object or disturbed by jamming. Averaged values of the carrier-to-noise ratios per frequency band were used to reduce the features. Additionally, the reliability of a GNSS localisation also depends on the number of satellites tracked. The jamming flag of the AsteRx receiver, combined with the number of the frequency bands jammed and the respective duration, was used to select jamming events. As visible in Figure 7 and Table 1, the number of occurrences of the individual labels is very unbalanced. To reduce the vast number of data points labelled as *Stop*, the training and testing dataset neglected the nighttime.

Neglecting the jamming label and jamming duration
In the second scenario, we are interested in predicting the same labels as above, using all features except the number of jammed frequency bands (*n_jamming*) and the duration of the jamming event. Both of these were used to create the jamming label, so by dropping them, we hope to infer more information on the relation between the other features and a possible jamming event.

Time-sensitive dataset using the mean C/N_0s and number of used satellites
Lastly, as we have observed in Section 2, jamming events are usually characterised by a drop in certain mean C/N_0 bands over multiple seconds, the number of used satellites,

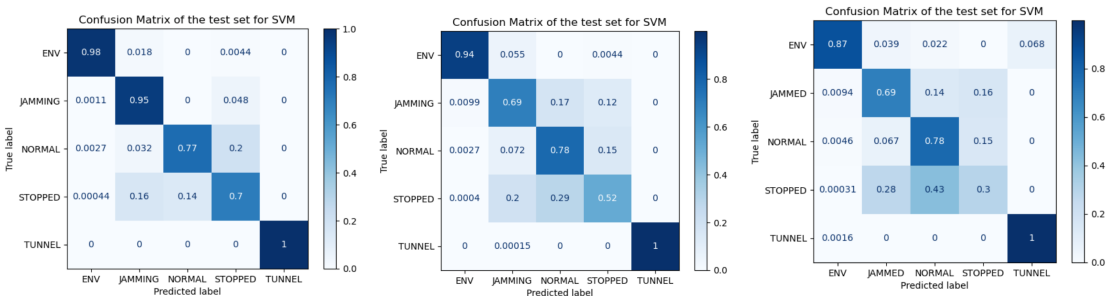
and possibly the gain parameter. Thus, we add the next and previous two seconds of the mean C/N_0 and the number of used satellites as additional available features for the Machine Learning models.

3 Results

In the following, we will describe the performance of different AI algorithms. Each method's performance is visualised in a confusion matrix, and the accuracy is communicated in each section as well as comparison tables.

3.1 Support Vector Machines

As mentioned in Section 1.3 Support Vector Machine is one of the typical classification models. After cleaning, preprocessing, and splitting the dataset into training and testing data, the model's design is one of the important steps to have an efficient and performant model. One important step is to choose the kernel. This is a function which transforms the input data into a higher-dimensional space. One of the most common kernels is the Radial Basis Function (RBF): $K(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|^2}{2l^2}\right)$.
This equation is presented by [4] as a stationary kernel. It is also known as the squared exponential or Gaussian kernel, where l is a free parameter, and x_i and x_j are represented as feature vectors in some input space.
The features are weighted to account for an imbalance in the labels, i.e. less occurring labels are weighted with additional importance to improve the overall performance of the model.
Using the full dataset leads to the best overall accuracy of 0.85, with higher accuracy for the jamming and environmental effect labels. If the duration of jamming and *n_jamming* are neglected, we notice a drop in the accuracy of the jamming label to 0.69, which leads to a drop in the overall accuracy to 0.79. Using the time-dependent dataset, the accuracy of the jamming label drops to 0.69, leading to a drop in the overall accuracy to 0.57.



(a) Using the full dataset (b) Neglecting *n_jamming* and duration (c) Using the time-sensitive dataset

Figure 8: Confusion matrices for SVM

3.2 Logistic Regression

Like Support Vector Machines, Logistic Regression aims to split the data using linear decision boundaries. Just as the data can be transformed into a higher dimensional data space when pre-processing it for the SVM using kernels, a similar method can be applied to prepare the data into higher dimensions when working with Logistic Regression. But, for SVMs, finding the best-suited kernel, tuning it and transforming the data is part of the pipeline implemented in sklearn [5]. For Logistic Regression, this is not the case. This makes finding and tuning a well-suited Logistic Regression model far more challenging. As an accuracy of 0.69 obtained by the Logistic Regression model for the full dataset is far below the accuracies reached by the other three approaches, we chose to neglect this method.

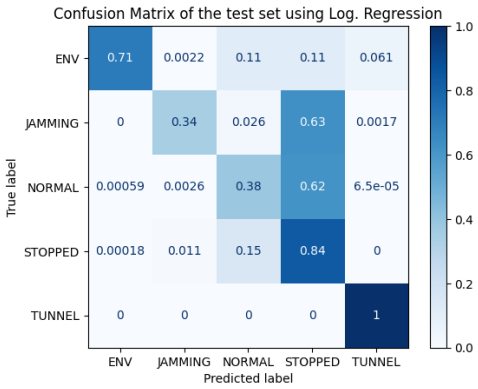


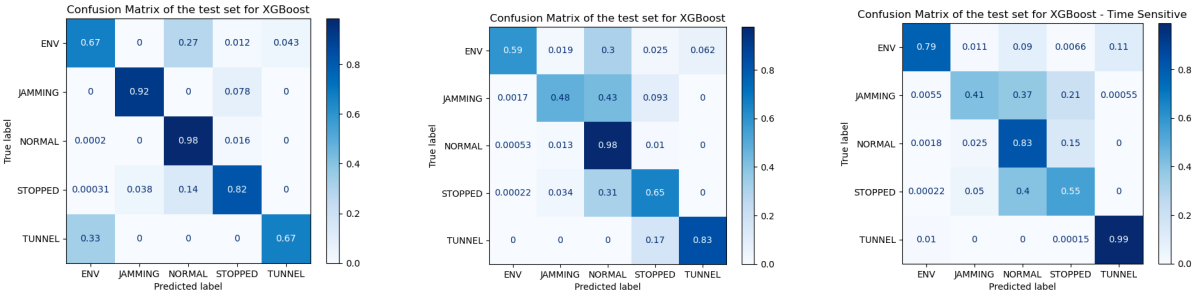
Figure 9: Confusion matrix for Logistic Regression, using the full dataset

3.3 XGBoost

XGBoost aims to incrementally build an ensemble of decision trees by correcting each tree's errors in subsequently trained trees. While experimenting with several trees, 100 trees were a good compromise for performance. Tuning the relationship between the number of features used and accuracy, the best results came from including all features, i.e. not excluding any columns.
Overall, the model's performance, using the full dataset, has an accuracy of 0.89, with 0.92 accuracy in the Jammed class. Neglecting the number of jammed frequency bands and the jamming duration leads to a drop in the accuracy of the data points labelled as Stopped, from 0.82 to 0.65. The overall accuracy then is 0.77. For the time-sensitive case, the accuracy is 0.70, with the poorest subclass accuracy in the Jamming class.

3.4 Random Forest

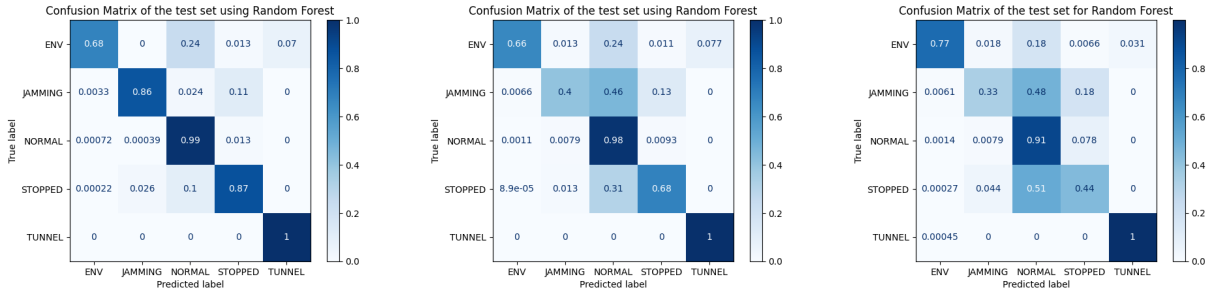
Similar to the pipeline described to design the previous models, several steps are necessary to train a well-performing Random Forest. Splitting the data into a training and test set was done as explained in Section 2.3. In order to find the best possible hyper-parameters, hyper-parameter tuning was performed. As described in Section 1.3, a Random Forest consists of many Decision Trees. Each decision tree splits the data using sequential, linear



(a) Using the full dataset (b) Neglecting *n_jamming* and duration (c) Using the time-sensitive dataset

Figure 10: Confusion matrices for XGBoost

decision boundaries. Therefore, the number of these splits, equalling the tree depth, the minimum number of samples that need to be grouped together per split, and the overall number of trees used, can be chosen. The selected hyper-parameters were adjusted for each of the three datasets.
Overall, the performance of the model using the full dataset has an accuracy of 0.93. Neglecting the number of jammed frequency bands and the jamming duration especially leads to a drop in the accuracy of the data points labelled as *Jammed* from 0.86 to 0.40. The overall accuracy then is 0.81. Including information about the surrounding data leads to an overall decreased accuracy of 0.68 on the test set and significantly worse results, with about 0.33 recall for the jamming detection.



(a) Using the full dataset (b) Neglecting *n_jamming* and duration (c) Using the time-sensitive dataset

Figure 11: Confusion matrices for Random Forest

3.5 Comparison of the classical AI methods

Logistic regression proves to be significantly weaker than the other methods in identifying jamming events, which makes it less suitable. SVM and XGBoost show good performance at picking up jamming events, and the SVM has generalisable results across classes with a minimum accuracy of 0.82 on all classes. The best overall performance is found in Random Forest, but it has a lower accuracy at identifying jamming (0.86). SVM again has high accuracy in the Jamming class when the model does not receive duration, and

the number of jammed bands as inputs, but in general, all AI methods have noticeably worse accuracy across classes. All methods perform poorly on the time-sensitive dataset.

Class accuracy	Log. Regression	Random Forest	XGBoost	SVM
Environment	0.71	0.68	0.67	0.98
Jammed	0.34	0.86	0.92	0.95
Normal	0.38	0.99	0.98	0.77
Stop	0.84	0.87	0.82	0.7
Tunnel	1.00	1.00	0.67	1
Overall accuracy	0.69	0.93	0.89	0.85

Table 2: Performance comparison of the selected classical AI methods on the full dataset

Class accuracy	Random Forest	XGBoost	SVM
Environment	0.66	0.59	0.94
Jammed	0.40	0.48	0.69
Normal	0.98	0.98	0.78
Stop	0.68	0.65	0.52
Tunnel	1.00	0.83	1
Overall accuracy	0.81	0.77	0.79

Table 3: Performance comparison neglecting *n_jamming* and duration

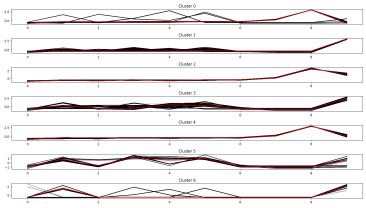
Class accuracy	Random Forest	XGBoost	SVM
Environment	0.80	0.79	0.87
Jammed	0.32	0.41	0.69
Normal	0.68	0.83	0.78
Stop	0.59	0.55	0.3
Tunnel	1.00	0.99	1
Overall accuracy	0.68	0.70	0.57

Table 4: Model performance for the time-sensitive dataset

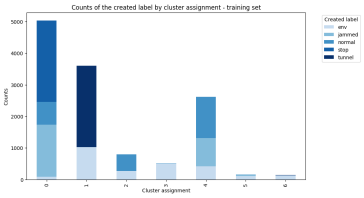
3.6 Clustering time sequences

As described before, the dataset provided by SBB did not include a reliable label, which could have been used to train Supervised Machine Learning models. We created artificial

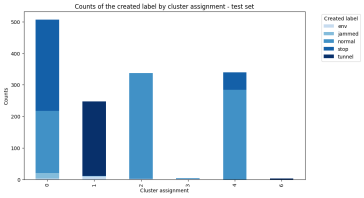
labels by defining the corresponding metrics each data point needs to fulfil to be assigned the respective label. Time-series clustering is an alternative approach to gathering additional information about patterns visible in the dataset. It is defined as the unsupervised classification of time-series data so that sequences with similar patterns, based on a defined similarity metric, are grouped together.[6] Within the project scope, time series clustering was used to analyse whether the data instances of each of the classes show different inherent patterns. For this, a subset of two hours (15.11.23 12 a.m. until 2 p.m.) was selected, cut into sequences of ten seconds and split into a training and test set. The training set was upsampled to allow the clustering algorithm to train on a balanced data set, with each label appearing with the same frequency. During the EGNSS MATE's future progress, it is planned to simulate jamming and spoofing events within a test environment. This will generate labelled data. Then, time-series clustering might be a promising option for extracting patterns inherent in the data assigned to the different classes.



(a) Data structures of the individual clusters



(b) Label distribution within the clusters during training



(c) Label distribution within the clusters during testing

Figure 12: Exemplary usage of time-series clustering to extract patterns in the sequences assigned to the different classes

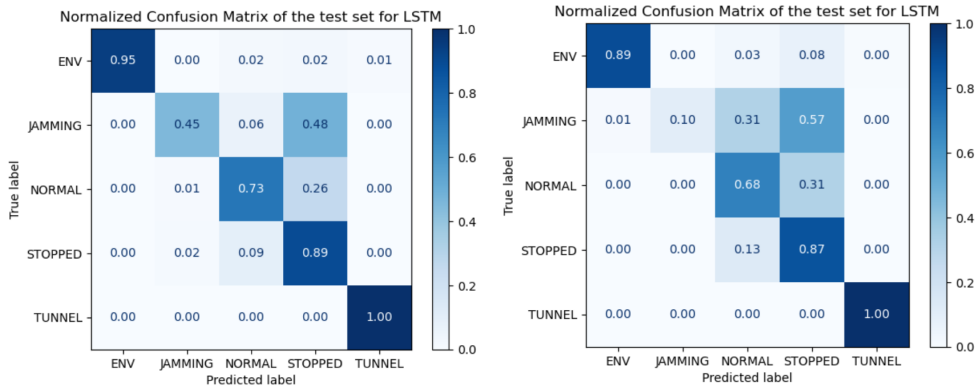
3.7 Long Short-Term Memory

LSTM is a promising tool to use for time-series datasets. The choice of hyper-parameters is the most important step within the training process. In this case, we implemented two hidden layers with 50 units each and a softmax activation function.
The softmax function

$$\sigma(z) = \frac{\exp z_i}{\sum_{j=1}^{K} \exp z_i} \qquad \text{for } i = 1, ..., K \quad \text{and} \quad z = (z_1, ..., z_k) \in \mathbb{R}^K$$

takes an input of K real numbers and normalises them into a probability distribution consisting of K probabilities proportional to the exponentials of the input numbers. Therefore, the Machine Learning model outputs the probability for each datapoint to be classified as each of the labels.
When using all features, we have a good overall accuracy of 0.94. Unfortunately, the jamming prediction was only 0.69. For the dataset neglecting the *n_jamming* and jamming duration features, we notice a dramatic drop in the accuracy of the jamming label to 0.13, which leads to a drop in the overall accuracy to 0.77.
However, we can see from Figure 13 that the model majorly mixes the jamming and

stopping cases. Therefore, we might have a better jamming prediction accuracy with the right weighting and more epochs.



(a) Using the full dataset

(b) Neglecting *n_jamming* and duration

Figure 13: Confusion matrices for LSTM

4 Outlook

In the future, different approaches to analyse the data for perturbances could be evaluated. Some ideas are presented in the following.

Instead of working with mean C/N_0, one could work with the raw C/N_0 values per satellite and extract patterns. However, this amount of data may be hard to analyse, as the positioning of each satellite and its respective degradation in signal quality needs to be accounted. Another idea for future research could be to analyse the intensity and position of the jammer. For this, the reported jamming bandwidth or the duration of the jamming event, dependent on the train's velocity, could be used. To detect spoofing, the core strategy would be to compare the known position of the satellite with the calculated relative position estimate of the receiver. Furthermore, if there is a visual obstruction between the satellite and the receiver, but the signal comes through, a spoofing or multipath incident must be present. More ideas to detect spoofing include looking for sudden changes in the pseudoranges, the Doppler effect or the satellite signal carrier phase.

Additionally, assessing the risk of being jammed depending on external features would be a valuable goal. The most promising features we identified were distance to the nearest busy road, parking lots, radio antennas, and embassies. Furthermore, an index of nearby car traffic, the nearby population, or the presence of high buildings or mountains could be important to assess the reliability of GNSS-based train management. The distance between the reliable inertially assisted location estimate vs. the GNSS-only based location can give information on the risk for multipath events. This risk rises near tunnels and bridges. For this, a dataset provided by the Swiss Federal Council [11], which includes the length and position of the midpoint of all Swiss rail tunnels, can be used.

Once we have a reliable way to determine whether there has been a jamming event, one

could train a linear logistic model to estimate the risk increase of being jammed or spoofed based on these external features.
With the problems we have due to the high false alarm rate of the jamming flag, it would be helpful in the future to create and maintain a benchmark data set where it is certain when we are being jammed or not. Such a dataset could be generated by simulating the effect of a jamming event or signal disturbances of other sorts within a controlled environment. Otherwise, an evaluation of the raw satellite data might allow a more reliable classification of the nature of a disturbance. This dataset can then be used to train better and trusted Machine Learning models capable of labelling new unseen data. Keeping one part of such a benchmark dataset as a test set is always important. This allows us to assess an error rate, and a Machine Learning model performance benchmarked on the test set can be trusted for new unseen data.

4.1 Use Cases

Following the above-mentioned analysis, detecting interferences could be valuable for alternative use cases. These might include a model to flag interference events, help with security enhancement, efficient scheduling, and emergency services.
Detecting jamming and spoofing events can enhance the security of the railways by identifying potential threats to the navigation used in train control systems. This can improve communications too. Additionally, detecting interference can help optimise schedules and prevent disruptions caused by unreliable information. In case of an emergency, accurate location information is critical for effective response. Detecting and addressing GNSS interference could improve first responders' aid.

5 Conclusion

In summary, this project represents an important first step into the future of railway management based on GNSS navigation. First, a deep analysis of core relations in the data collected by the SBB measuring train was conducted, extracting data patterns pointing to jamming events. Based on this, three different sets of features to examine the prospect of predicting jamming events were created. After thorough pre-processing of the provided data, several AI-based solutions for jamming detection were implemented and compared. Against the intuition that including time-dependent data in the models would improve the overall performance, the learned AI methods suffered from the curse of dimensionality and dropped in performance. In the future, careful feature engineering and clean data will be mandatory to achieve a more reliable and trustworthy result. Lastly, multiple promising approaches were listed to further improve jamming and spoofing detection and pave the way to a trustworthy risk assessment of GNSS-based railway management.

Bibliography

[1] Tin Kam Ho. "Random decision forests". In: *Proceedings of 3rd international conference on document analysis and recognition*. Vol. 1. IEEE. 1995, pp. 278–282.

[2] Sepp Hochreiter and Jürgen Schmidhuber. "Long short-term memory". In: *Neural computation* 9.8 (1997), pp. 1735–1780.

[3] Trevor Hastie et al. *The elements of statistical learning: data mining, inference, and prediction*. Vol. 2. Springer, 2009.

[4] Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *JMLR 12*. 2011, pp. 2825–2830.

[5] Fabian Pedregosa et al. "Scikit-learn: Machine learning in Python". In: *the Journal of machine Learning research* 12 (2011), pp. 2825–2830.

[6] Saeed Aghabozorgi, Ali Seyed Shirkhorshidi, and Teh Ying Wah. "Time-series clustering–a decade review". In: *Information systems* 53 (2015), pp. 16–38.

[7] Tianqi Chen and Carlos Guestrin. "Xgboost: A scalable tree boosting system". In: *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*. 2016, pp. 785–794.

[8] Ashish Vaswani et al. "Attention is all you need". In: *Advances in neural information processing systems* 30 (2017).

[9] Zewen Li et al. "A survey of convolutional neural networks: analysis, applications, and prospects". In: *IEEE transactions on neural networks and learning systems* (2021).

[10] Roman Ehrler et al. "Jamming and Spoofing Impact on GNSS Signals for Railway Applications". In: *Proceedings of the 36th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2023)*. 2023, pp. 4153–4167.

[11] Switzerland : Bundesamt für Landestopografie swisstopo, France : INSEE IGN NaturalEarth. *SBB Railway Tunnels Location Dataset*. `https://data.sbb.ch/page/licence`. Last updated: June 2023. Switzerland, France, 2023.