Towards Efficient Rail Transportation: Bayesian Network Modeling for Predicting Passenger Train Delays Using Secondary Train Information

Maarten Vangeneugden
IDLab, Faculty of Applied Engineering
University of Antwerp - Imec
Antwerp, Belgium
maarten.vangeneugden@uantwerpen.be

Ngoc Quang Luong
IDLab, Faculty of Applied Engineering
University of Antwerp - Imec
Antwerp, Belgium
ngoc.quang.luong@uantwerpen.be

Siegfried Mercelis
IDLab, Faculty of Applied Engineering
University of Antwerp - Imec
Antwerp, Belgium
siegfried.mercelis@uantwerpen.be

***Abstract*—This paper presents the usage of a Bayesian network to predict the delay of a given passenger train on the Belgian rail network, trained with a year of arrival and departure records. In order to improve the prediction of future delays, this paper explores the possibility of incorporating the delays of other trains that visit the same station. While the biggest improvement is to be expected from looking at the train's own delay in previous stations, there's a measurable improvement in the prediction accuracy when secondary trains are taken into account.**

I. Introduction

The Belgian rail network is one of the most saturated transportation networks in the world. Every year, hundreds of thousands of trains travel over more than eight thousand kilometres of tracks. With more than 250 metres of track per square kilometre[1], there aren't many countries in the world with a denser rail network than Belgium[2]. At least half a million Belgians rely on the NMBS/SNCB/NGBE[3] to reach their destinations.

Unfortunately, despite all efforts, the punctuality of the SNCB remains one of its main complaints. The statistics published by themselves suggest that each month, around nine out of ten trains arrive on time.[4] For such an important part of Belgian mobility, this is a high failure rate. These numbers are also disputed by transport advocacy groups and passengers themselves, because suspended trains are not taken into account, and "on time" means that it arrived at the terminus with a maximum delay of six minutes.

When a train is delayed, the SNCB currently uses a best-case scenario for the remaining itinerary; in each subsequent station, the train is assumed to reduce its delay by up to two minutes. The solution's biggest advantage is that, this scenario guarantees passengers that arrive at the station before the "projected" departure time, will not miss their train.[5] However, the main shortcoming is that this scenario practically never occurs, and it is much more likely that the delay actually increases over time. Consequently, this also frustrates the scheduling of other trains; as if the projected delay time seldom corresponds with the actual delay, the rail traffic controllers cannot make informed decisions on how to mitigate a projected delay as well as possible. In rush hour, this can escalate quickly to other trains as well. Moreover, another disadvantage is that this makes passengers hesitant to explore alternative travel options that might get them to their destination faster than the original route.

In fact, the useful delay predictions are difficult to obtain because there are a lot of uncontrollable factors in play. Considering the stringent safety procedures that are in effect at the SNCB, this could explain why they are using the aforementioned scenario for communicating delays with passengers.

This paper explores the possibility of a better delay prediction method, one that not only takes into account the delay of a train itself, but also the delay of other trains on the network. The hypothesis being evaluated is that a train's delay is partially influenced by the delay of other trains. Examples include trains blocking the same tracks and having to wait on a connecting train to facilitate a transfer. Thus, we propose the selection of these so-called informative trains as contextual information in machine learning applications. We support this proposition by investigating and quantifying the usefulness of informative trains, as opposed to only taking the delay of the main train in the previous station into account.

The next section provides a brief literature study of related research. In section III, we will discuss Bayesian networks and explain our choice for this technique in lieu of other

[1] https://infrabel.be/nl/facts-figures

[2] https://en.wikipedia.org/wiki/List_of_countries_by_rail_transport_network_size

[3] Nationale Maatschappij der Belgische Spoorwegen / Société Nationale des Chemins de Fer Belges / Nationale Gesellschaft der Belgischen Eisenbahnen. The railway company prefers to use the French abbreviation when communicating in a non-Belgian language, and this will be used in this paper as well in light of this preference.

[4] https://infrabel.opendatasoft.com/explore/dataset/nationale-stiptheid-per-maand/table/

[5] SNCB train drivers are permitted to depart from a station if the scheduled time of departure has passed, regardless of what is presented in the live data feeds to passengers. So in theory, if a more realistic scenario predicts a 10 minute delay, but the train actually manages to reduce the delay to 8 minutes, a passenger that's 9 minutes late "on purpose" would miss the train, despite what was communicated.

979-8-3503-5409-6/24/$31.00 ©2024 IEEE
DOI 10.1109/CAI59869.2024.00254

approaches. After that, the nature of the data itself and the processing methodology is discussed. The last sections of the paper discuss the obtained results and anomalies, and some thoughts about future work that could improve on this paper's findings.

II. Literature review

Trying to predict delays of public transport with artificial intelligence is a topic that has received a decent amount of research already. This section takes a look at related papers and discusses them briefly.

Lessan, Fu, and Wen showed that delays in a train route can be accurately predicted using Bayesian networks (BN). [1] The authors compare a set of different BN schemes. They also correctly remark that a BN structure that takes domain knowledge and expertise into account is able to outperform structures that do not. The structure and goal of their work closely resembles that of this paper, which allows for building the same benchmark. This will be explained when discussing the obtained results.

The authors make an explicit distinction between the arrival and departure delay of the examined train line. While there are valid arguments to do so, we focus on the departure time and forego the arrival time entirely (except for the terminus station).

In the first stage of our research, we compared the departure and arrival times, which revealed that there's a direct linear correlation between both values.[6] We seek to improve the results through non-linear correlations.

The hypothesis that delays on railways are related to problems on the rail network elsewhere is not new, and has already been researched by Ulak, Yazici, and Zhang, who also relied on Bayesian network learning. [2] The authors however did not focus on predicting the delays themselves, but rather on the cause-effect relationship; which stations induce delays, and which ones are susceptible to them. The data used for this was collected using an opt-in feature in a passenger transit information application. Because prediction was not the focus of this research, the authors discarded more than 85% of the data samples, only choosing the logs that were last made before arriving at any given station. While this might be a decent strategy for metro rail networks, it is possible this technique does not hold up for normal rail networks[7].

Tiong, Ma, and Palmqvist created a review of several different papers that aim to predict train delays with a data-driven approach. [3] What is clear is that the operational level is the most popular scope[8]. What makes this paper interesting to mention is that it shows why it is not practical to have a common benchmark amongst related research: All papers use a unique dataset, all with vastly different properties. This provides an opportunity for future research; trying to compare the results of different papers with each other could give closure about the actual effectiveness of different tools and/or algorithms.

While many papers rely on domain knowledge to improve their results, others choose to rely on randomness to discover correlations that might otherwise go unnoticed. Li, Wen, Hu, *et al.* use a random forest regression model to predict train delays up to 20 minutes into the future. [4] While the authors report a high prediction accuracy, it should be noted that weekends were excluded from the training data. Whether or not this has a negligible effect on the results is unknown.

More recent papers have also relied on Bayesian networks. The work of Huang, Spanninger, and Corman shows that, while Bayesian networks are a useful model, the usage of clustering algorithms also has a positive effect on the accuracy of the predictions. [5] The delays of trains would be clustered by their delay evolution, which indeed seems like a good approach, because it's very plausible that delays can evolve in different ways depending on their cause. But as mentioned earlier, it's hard to compare results with other papers, as the authors also use a custom dataset that's not publicly available.

Shi, Xu, Li, *et al.* didn't use Bayesian networks, but Bayesian optimization, which is a very different technique. [6] This, in combination with XGBoost, was then compared with some other models, and was shown to perform favourably. The authors acknowledge the idea that the delay of a given train can influence the delay of another train, and it's one of the only papers to explicitely show this behaviour through an example. However, the variables they use in the model do not seem to take this into account. Instead, they rely on this latent information being available through the other variables they do include.

III. Bayesian networks

A. Properties

In order to show the improvement that multiple trains can provide, the model used in this paper is a Bayesian network. A Bayesian network (BN) consists of nodes and directed edges, respectively representing random variables and causalities. BNs belong to the family of graphical models, which are popular choices when there are good reasons to assume that different random variables have an influence over each other.[7] When these influences are unclear, it is possible to discover them through structure learning.

The way BNs work is by exploiting conditional independences (CI) for representing the joint distribution of all the random variables. Say a hypothetical system that can be described using random variables $A, B, C, ..., Y, Z$, with each letter being conditioned by the previous letter (e.g. B is conditioned by A and only A). Representing the full joint distribution, even assuming all are boolean variables, would give $2^{24} = 16777216$ entries, and learning would require data

[6] While this correlation also exists between different stations, there's a much larger variance in the values, which implies that there are more factors in play that are unaccounted for. The final results suggest that other trains make up a part of those factors.

[7] The main difference between a metro and a "normal" train is that metro railways do not have level crossings and their infrastructure is exclusively used for the metro itself. This is often accomplished by grade seperation, which explains the frequent use of tunnels in urban areas and viaducts in suburban areas.

[8] The scope of this paper can also be classified as operational level.

for the same amount of parameters.
By exploiting CI, it is possible to reduce the size of the representation considerably, since $p(B|A,C,D,...,Y,Z) = p(B|A)$. In this particular example, exploiting CI means the we can use a factorized representation that only requires $24 \cdot 2^1 = 48$ entries, still providing the same expressive power.

This is one of the main reasons for choosing to use BNs in this paper, but there are other advantages compared to other models such as neural networks:

- BNs are often more interpretable than some complex machine learning models. The graphical representation of the network makes it easier to understand the dependencies between variables and the reasoning behind predictions.
  - For our paper, this is especially important, because in order to clearly show the added value of secondary trains as contextual information, being able to manually "disable" particular nodes is a necessity.
- BNs can be updated dynamically as new data becomes available. This is advantageous in transportation systems where conditions may change over time, and the model needs to adapt to new information.
- BNs are relatively fast AI models, both during training and evaluating. Together with their ability to be understandable for laymen in how they operate, this makes them prime candidates for actually deploying them in real world applications.

That is not to say that BNs are without disadvantages. For one, they require a substantial amount of data to be useful. With smaller datasets, a BN cannot properly get rid of uncertainty, resulting in worse error scores than one can obtain with other models.
From a practical standpoint, there are also significantly fewer libraries with support for Bayesian networks. This limits adoptance in other projects, which in turn makes it harder to find solutions for problems that may occur during usage.

Figure 1 shows how the railway network is transformed into a BN. We chose to focus on the IC-05 train line between Antwerp-Central and Charleroi-Central, a relatively busy route that goes through four Belgian provinces and the capital city of Brussels. Brussels itself is especially interesting because of the special regime put in place to handle the large amount of traffic in the North-South connection: Cargo trains are prohibited from entering, whereas passenger trains are allowed to enter the tunnel in rapid succession (up to less than one minute from each other during rush hour), and the usage of the switches on the route is limited as much as possible to avoid any crossings, which would block at least two tracks for a single train. Also important is the fact that IC-05 has a very invariate time schedule; every day, there is one train per hour that arrives at the same minute in each station.[9]

Even though we're focusing on one particular train line, applying the described technique to other train lines should not pose any big challenges, since the process of building the BN is analogous to the one in this paper. That means that it's definitely possible to expand the prediction to more trains across the Belgian railway network.

[9]https://www.belgiantrain.be/-/media/files/pdf/support/riv/ic-leaflets/fr/ic-05-dec2023-fr.ashx

B. Random variables

As mentioned earlier, the nodes in a BN each represent different random variables, and the edges between them represent the causality of those variables with relation to each other.
Typically, the random variables are modelled as discrete events. For example, the throw of a die is a random variable X with six discrete events, with

$$\forall x \in \{1,2,3,4,5,6\} : P(X = x) = \frac{1}{6} \tag{1}$$

For BNs with a small Markov blanket[10] size, this is hardly a problem, as the computational complexity stays low enough. However, this would pose problems with our particular application: Each delay of each train in each station is represented as a separate node, and the delays are in seconds. If we were to represent each second as a discrete event, one would end up with hundreds of possible events per node. Of course, it's possible to use intervals as discrete events instead of each second separately (i.e. "binning"), but even that's not enough.

The joint probability of n binary values needs

$$\mathcal{O}(n \cdot g^k) \tag{2}$$

terms, with g the amount of categories and k the maximum number of parents of a node. This would put a technical limit on the amount of trains that can be taken into account; if the delays are categorized per minute, and limited to just 10 bins, the factorized representation of a BN would reach in the millions. There is also the fact that categorizing continuous values comes with a decrease in accuracy as well.

In this paper, a slightly more complex type of Bayesian network is used, the Gaussian Bayesian network (GBN). [8] In these networks, the nodes do not represent a probability table, but a Gaussian distribution. The distribution of a child node is then constructed by a multivariate Gaussian distribution, built using the distributions of the parent nodes.
This offers a lot more flexibility and decreases the network's computing complexity substantially.

C. Loss function

Evaluating artificial intelligence (AI) models is often done by using a so-called loss function. In this paper, the loss function mainly serves as a measure to compare different models with each other.

For GBNs, there are only a couple of loss functions available:

- (Gaussian) Log-likelihood loss (log-loss): Returns the joint probability of the test data based on the parameters of the trained model.[9] This loss function is especially suited when using maximum likelihood estimation (MLE)

[10]A Markov blanket of a node is the only knowledge needed to predict the behavior of that node. This consists of the node's parents, its children and the co-parents, if applicable.

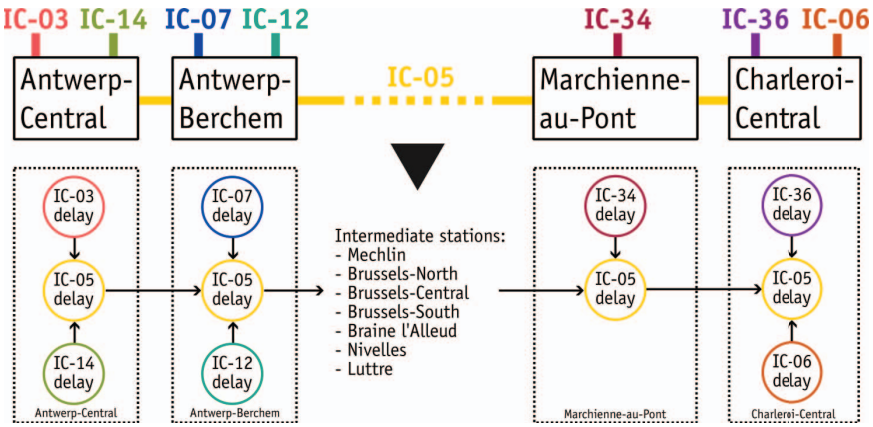

Fig. 1. Visual diagram of how the relevant railway network is transformed to a Bayesian network. We want to predict the delay of train line IC-05, which travels between Antwerp-Central and Charleroi-Central. Each station on the route is also frequented by other train lines. The hypothesis is that the delays on these train lines also have an influence on the delay of IC-05. First, the delay of each line is transformed into a node, one for each station that train line frequents. Then, the delay nodes of IC-05 in each station are connected to the node of the next station, representing the (trivial) influence of the last recorded delay. Finally, the delay nodes of other train lines are connected to that of the IC-05 node in the same station. This results in the graph that's being evaluated in this paper. (Note that there are many more lines per station, but these are not shown here in order to keep it clear and concise.)

during training, because maximizing the MLE is equivalent to maximizing the log-likelihood.

- Predictive correlation: Calculates the correlation between the predicted and actual value in a given node.[10]
- Mean squared error: Sums the squared difference between the predicted value and the actual value. This sum is then divided by the amount of data samples to get the mean value.[11]

It is hard to explain how predictive correlation reaches a given score; the source being referenced[12] in the documentation does not include this concept, and to the best of our knowledge, this technique has not been systemetically described. Therefore, we decide to look at the remaining two loss functions.

A major property of MSE is its sensitivity to outliers. Often, this makes one search for alternative loss functions, but here, it is a desired effect: Small deviations from the actual delay are not a big problem, but the higher the error, the heavier it *should* be penalised.

Another interesting property is that MSE measures the prediction error in a single node, in contrast to log-loss. This makes it possible to discover discrepancies between the different stations, which might remain hidden with log-loss.

These arguments made the choice for MSE the best one in our opinion.

IV. Data processing

Machine learning is only possible with a large dataset of which the content is sampled independently and identically distributed from the real world. These requirements seem to be fulfilled by the punctuality data of Infrabel, the Belgian railway network operator. The total size of the dataset used in this paper is 2GB, and contains all passenger trains of 2018, with the arrival and departure delay in each station measured up to the second, as shown in figure 2.

This data comes in the form of CSV files that are available under the CC0 license on Infrabel's open data website.[11] The *Pandas* library was used to filter the available data and create a new dataset that could be used for training the Bayesian network.

1) all the data records are bundled per itenerary. This results in many data subsets that each describe a single route, and each record describing a single stop.
2) The IC-05 iteneraries are kept separate; this is the line the model will focus on[12].
3) For each stop in each IC-05 itenerary, the remaining dataset is checked; all trains that have a scheduled arrival within 15 minutes of the IC-05 train's arrival in the same station, is considered as a "possibly influencing train".

By applying these steps to the Infrabel data, we can create a dataset consisting of 3955 samples. Figure 3 shows an excerpt of those processed data samples.

V. Results

Gaussian Bayesian networks do not seem to enjoy the same level of interest as their discrete counterparts, as is reflected in their respective support in software libraries. The only technically viable choice was the bnlearn package for the R programming language[13].

The GBNs are trained and evaluated by using 10-fold cross validation with random sampling. This means that each run,

[11] https://infrabel.opendatasoft.com/explore/dataset/stiptheid-gegevens-maandelijksebestanden/information/?sort=mois

[12] Although the model here is specifically tailored to the IC-05 line, the same methods can be applied to any other train line.

[13] https://www.bnlearn.com

	train_number	relation	pt_car	arrival_delay	departure_delay	planned_arrival	planned_departure	arrival_line	departure_line
284	9,240	IC 35	BE00216	46	46	2018-05-01 16:09:00.000 +0200	2018-05-01 16:09:00.000 +0200	0/6	0/6
285	9,240	IC 35	BE00215	64	101	2018-05-01 16:10:00.000 +0200	2018-05-01 16:11:00.000 +0200	0/6	0/6
286	9,240	IC 35	BE00217	103	103	2018-05-01 16:12:00.000 +0200	2018-05-01 16:12:00.000 +0200	0/6	0/6
287	9,240	IC 35	BE00220	94	[NULL]	2018-05-01 16:15:00.000 +0200	[NULL]	0/6	[NULL]
288	3,791	S2-2	BE00553	9	9	2018-05-01 20:17:00.000 +0200	2018-05-01 20:18:00.000 +0200	36	36
289	3,791	S2-2	BE01174	-30	-30	2018-05-01 20:21:00.000 +0200	2018-05-01 20:21:00.000 +0200	36	36
290	3,791	S2-2	BE00368	-30	53	2018-05-01 20:24:00.000 +0200	2018-05-01 20:24:00.000 +0200	36	36
291	3,791	S2-2	BE00033	53	53	2018-05-01 20:26:00.000 +0200	2018-05-01 20:26:00.000 +0200	36	36
292	3,791	S2-2	BE00648	63	57	2018-05-01 20:27:00.000 +0200	2018-05-01 20:28:00.000 +0200	36	36
293	3,791	S2-2	BE00916	43	43	2018-05-01 20:31:00.000 +0200	2018-05-01 20:31:00.000 +0200	36	36

Fig. 2. Snippet from the raw data in the database. `pt_car` refers to the code of a particular station on the network. Each record pertains to one train in one station/checkpoint. The `arrival_delay` and `departure_delay` are in seconds and are to be summed with the scheduled counterparts. Negative values indicate a train arriving/departing earlier than scheduled.

	arrival_ANTWERPEN-CENTRAAL	departure_ANTWERPEN-CENTRAAL	L 0E_arrival_Antwerpen-Centraal	L 0E_departure_Antwerpen-Centraal	IC 02_arrival_Antwerpen-Centraal	IC 02_departure_Antwerpen-Centraal
0	0.0	11.0	0.0	-13.0	0.0	13.0
1	0.0	37.0	0.0	53.0	0.0	23.0
2	0.0	110.0	0.0	10.0	NaN	NaN
3	0.0	68.0	0.0	-6.0	0.0	20.0
4	0.0	29.0	0.0	13.0	0.0	330.0
...	...	...	...	...	...	...

Fig. 3. Processed data snippet from the entire training set. Each column represents a feature, and is a separate node in the Bayesian network. The row represents the delay of each train in a particular station. The columns shown are the delays of IC-05 (in capital letters) and of secondary train lines. Note that the arrival delay is zero, because Antwerp-Central is a starting station for many train lines.

~3600 records are used as training set, and 400 serve as unseen testing data, after which the average score of all runs is returned. Because of the random division in training and testing sets, this process is repeated 100 times for each station node. This provides both an average score and a standard deviation metric.

In order to confirm the expected improvements, not all edges as shown in figure 1 were immediately enabled. Instead, a particular order of enabling nodes was used. This order is based on our assumptions of how informative each event would be for predicting the correct delay for IC-05, and the MSE should be strictly descending in each information level. These information levels (called "steps" from now on) are:

1) No information. The delay of a train is said to be completely devoid of any influence, not even the delay in the train's previous station.
2) Delays of all incoming trains, still excluding the delay of the main train (IC-05)'s previous station.
3) Delays of trains that are deemend informative, still excluding the delay of the main train's previous station.
4) Only the delay of the main train's previous station.
5) The delay of informative trains, together with the delay of the main train's previous station.

"All incoming trains" is a set, consisting of the trains that are scheduled to arrive or depart within a time window of [-15,15] minutes, relative to the scheduled arrival time of an IC-05 train.

What constitutes an "informative" train is the result of a bottom-up approach: Starting from step 1 (no information at all), each non-IC-05 node is enabled one at a time. Each time, the MSE is compared to that of the MSE in step 1. If the new MSE is significantly lower (at least a couple of percent points relative to having no info), then that node is marked as being "informative". There is no limit on the amount of informative trains that we select.

The results that were obtained this way are shown in table I. It contains the scores obtained per station, per step. Figure 4 visualises the same information in a diagram, which reveals some interesting things about the data.

There is no relation between the trains that were found informative; sometimes the local trains were more useful, other times the intercity trains, and also a mixture of the two occurs.

One of the most obvious observations is the enormous improvement when we take the delay in the previous station into account. This makes a lot of sense; if a train departs with x minutes of delay, it is very likely it will arrive with a delay close to x in the next station. On average, the loss decreases from 108695 to just 10869.

Another observation is that, even when we do not use this information, and only rely on the delay of other trains, there's still often a noticeable improvement in the delay prediction, a decrease from 174106 to 121817 in the average loss. Only relying on the informative trains (step 3) also reveals the model learns from all the data, but in doing so, tends to overfit on trains that are not informative to the final prediction. This is clearly visible with the improvement going from step 2 to 3, an average loss of 121817 to 112839. This is to be expected from machine learning, and shows that domain knowledge is an important factor for any model to reach the ground truth. However, even in step 2, there's already an improvement, most likely due to large calamities that affect a lot of different trains simultaneously.

TABLE I
RAW MEAN SQUARED ERROR SCORES FOR EACH INFORMATION LEVEL PER STATION, LOWER VALUES ARE BETTER.

Station name	No info	All trains	Informative trains	Previous delay	Informative + delay
Antwerp-Central	88160.97	34007.66	33792.04	N/A	N/A
Antwerp-Berchem	103003.80	61152.12	65980.84	6114.48	**6077.08**
Mechlin	157187.60	77464.27	77011.90	21135.73	**14591.74**
Brussels-North	187577.70	99280.91	89840.65	**17304.48**	18108.03
Brussels-Central	197091.80	102349.50	90129.31	**2563.27**	2640.98
Brussels-South	188398.30	118511.50	102491.10	8053.87	**7323.94**
Braine l'Alleud	206332.30	117234.70	117234.70	**32172.25**	33687.31
Nivelles	198920.70	125921.40	111750.10	6305.28	**5794.42**
Luttre	194622.20	200325.40	206224.90	5713.81	**5528.98**
Marchienne-au-Pont	193002.40	190860.50	191544.40	6166.07	**5873.10**
Charleroi-Central	200874.20	212881.40	155235.40	3165.97	**3002.17**

Step 4 is considered as the baseline model, because this "amount" of information is also being used by Lessan *et al.* [1] By copying the methodology from their peer-reviewed and published paper and applying it to our dataset, we obtain the baseline scores that our proposed methodology ought to improve upon. If taking other trains into account does improve on this score, then our hypothesis would be validated. Since this paper intends to improve upon the current state of the art, the score in step 5 should be better than the score in step 4. This seems to be the case, as the average loss decreases from 10869 to 10262.

The final important observation is thus that combining informative trains with the main train's earlier delay often does give a slight improvement to the overall score. The scores in step 5 seem to support the hypothesis that taking into account the delays of other trains improves the predictive capability of the model.

There are also some deviations on the continuously declining score, the most noteworthy example being Brussels-North and Brussels-Central. This can be explained because of the business of the North-South tunnel; this particular section of the Belgian railway network handles 1200 trains per day, and special regimes and protocols are put in place to handle the special circumstances of this corridor. It Is likely that these protocols decrease the influence of the delay of other trains significantly. This hypothesis seems to be supported by the fact that, when the train has left the North-South connection in Brussels-South, the influence of other train delays becomes an improving factor again.

While this could explain the worse results for Brussels-North and Brussels-Central, it does not explain the worse results in Braine-l'Alleud, which is the only station with a normal regime where taking other informative trains into account worsens the prediction of our main train's delay. It is unclear why that is the case here.

Since Antwerp-Central is the first station of the itenerary, there is no previous delay available. This is visible in the diagram, as it is the only line that's cut off at the third step.

Training and evaluating the model also shows that the reduced computational complexity of a GBN is hugely important for keeping the training times acceptable. On a single CPU core, it takes on average 40 seconds per station to do so, and the model itself is a lot more interpretable than neural networks or other complex machine learning models.

VI. CONCLUSIONS AND FUTURE WORK

By including the delays of secondary trains in the same station using a Bayesian network, we were able to create a delay prediction model that outperforms similar models that do not include this contextual information. We presented a bottom-up approach where we selected only those trains that were actually informative, which also allowed us to discover which trains did not add useful information, effectively offering an easy way to detect and remove noise from our prediction model.

Of course, there are still many open questions and possibilities that ought to be explored in order to find the optimal properties for predicting train delays, some of which we will briefly discuss here.

A. Selection of informative train lines

Currently, the set of informative trains is obtained through a bottom-up approach, as described in section V. While this approach already delivers improved results compared to the baseline, it is theoretically possible that a better set exists. Perhaps there are some train lines that only provide useful information in another influencing scheme. However, time constraints did not allow for this approach, finding the optimal graph is an NP-hard problem. While some influences can definitely be ruled out beforehand, there are still many influence sets that have not been considered.

Trying to find an explanation for the worse results in Braine l'Alleud also seems interesting to do, especially if the model is applied to other train lines and certain stations also seem to suffer from a decrease in prediction power.

Of course, one could also look to include information other than the delays of trains. Special events like strikes, weather forecasts, holidays, ... Incorporating these variables could also show an improvement in delay prediction.

B. Evaluation metrics of the model

While the main properties of the MSE are well known and discussed earlier, there's also one lesser known property: The mean squared error is a *frequentist* loss function. In the frequentist view, probabilities quantify the *frequencies* of

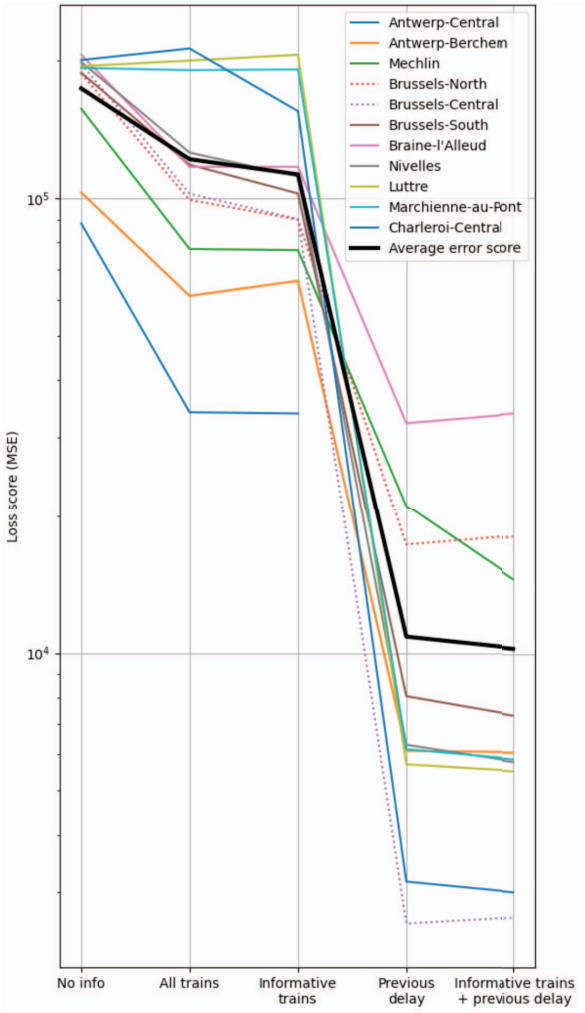

Fig. 4. Mean squared error per station and per step obtained from the trained model. The first three steps (resp. no info, all trains, informative trains) show that there's already a decent amount of predictive information to be obtained from simply looking at other trains alone. Step 4 (Previous delay) is the baseline model score, and the model proposed in this paper is given by step 5 (Informative trains + previous delay). Brussels-North and Brussels-Central are dashed lines because there are grounds to assume that the results in step 5 are heavily influenced by the special regime used in the tunnel connecting these stations. Note that Antwerp-Central does not include step 4 and 5, because for these steps, the previous delay is required, but that's not available in the starting station.

certain events. This contrasts with the Bayesian view, in which probabilities quantify the *uncertainty* of certain events. The MSE will penalise any deviation from the correct value. But this seems a bit exaggerated with regards to the goal of the model: Providing a reliable prediction of a train's delay. For example, if the prediction is off by 30 seconds, that does not pose a major problem by any means. Nor does a deviations of 2 seconds, but the former will result in a much larger loss score nonetheless.

More interesting would be to work with timespans to which the model assigns a very high, predetermined certainty. This would also provide a given user of the model with a tool to determine how certain a prediction would need to be in order to act upon it. It also makes sense to score deviations asymmetrically; if a passenger arrives 1 minute too early in the train station, it is not that big of a deal, but arriving 1 minute too late means that passenger could be spending an hour longer in transit. The currently used metric does not take this into account.

References

[1] J. Lessan, L. Fu, and C. Wen, "A hybrid bayesian network model for predicting delays in train operations," *Computers & Industrial Engineering*, vol. 127, pp. 1214–1222, 2019, ISSN: 0360-8352.

[2] M. B. Ulak, A. Yazici, and Y. Zhang, "Analyzing network-wide patterns of rail transit delays using bayesian network learning," *Transportation Research Part C: Emerging Technologies*, vol. 119, p. 102 749, 2020, ISSN: 0968-090X.

[3] K. Y. Tiong, Z. Ma, and C.-W. Palmqvist, "A review of data-driven approaches to predict train delays," *Transportation Research Part C: Emerging Technologies*, vol. 148, p. 104 027, 2023, ISSN: 0968-090X.

[4] Z. Li, C. Wen, R. Hu, C. Xu, P. Huang, and X. Jiang, "Near-term train delay prediction in the dutch railways network," *International Journal of Rail Transportation*, vol. 9, pp. 1–20, Nov. 2020.

[5] P. Huang, T. Spanninger, and F. Corman, "Enhancing the understanding of train delays with delay evolution pattern discovery: A clustering and bayesian network approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 9, pp. 15 367–15 381, 2022.

[6] R. Shi, X. Xu, J. Li, and Y. Li, "Prediction and analysis of train arrival delay based on xgboost and bayesian optimization," *Applied Soft Computing*, vol. 109, p. 107 538, 2021, ISSN: 1568-4946.

[7] H. David, *A tutorial on learning with bayesian networks*, 2022. arXiv: 2002.00269 [cs.LG].

[8] M. Grzegorczyk, "An introduction to gaussian bayesian networks," in *Systems Biology in Drug Discovery and Development: Methods and Protocols*, Q. Yan, Ed. Totowa, NJ: Humana Press, 2010, pp. 121–147, ISBN: 978-1-60761-800-3.

[9] W. Jon, *Bayesian and Frequentist Regression Methods*. 2013.

[10] S. Marco. "Cross-validation for bayesian networks." ().

[11] A. Ethem, *Introduction to Machine Learning*. 2014.

[12] K. Daphne and F. Nir, *Probabilistic Graphical Models: Principles and Techniques*. 2009, ISBN: 9780262013192.