Predicting Bus Arrival Times Based on Positional Data

Matic Kladnik†
Jozef Stefan International Postgraduate School
Ljubljana, Slovenia
matic.kladnik@gmail.com

Luka Bradeško
Department of Artificial Intelligence
Jozef Stefan Institute; Solvesall
Ljubljana, Slovenia
luka.bradesko@ijs.si

Dunja Mladenić
Department of Artificial Intelligence
Jozef Stefan Institute
Ljubljana, Slovenia
dunja.mladenic@ijs.si

ABSTRACT

This paper addresses predictions of city bus arrival time to bus stations on an example of a bigger EU city with more than 800 buses. We use recent historic context of preceding buses from various routes to improve predictions as well as semantic context of bus position relative to the station. For evaluation of the results, we developed a live evaluation web application which can compare performance of different prediction systems with various approaches. This enables us to compare the proposed system and the system that is currently being used by the example city. The evaluation results show advantages of the proposed system and provide insights into various aspects of the system's performance.

KEYWORDS

Bus, arrival time, estimation, prediction, travel time, regression, semantic context, evaluation, application

1 INTRODUCTION

Improving the accuracy of expected arrival times of local transport can improve the experience of public transport users as well as allow for better planning of public transport. By using recent historic travel times of other buses and additional semantic context of the bus that is currently in the prediction process, we improve predictions of bus arrival times. These predictions are calculated in a live system and can be used in real-time to inform users of the public transport system as well as to help detect traffic congestions.

The focus of this paper is on the architecture of the live travel time prediction system with which we continuously make predictions of bus arrival times as well as on our approach of evaluating the performance of the proposed system in comparison to the currently used system.

We will first look into the problem setting and the type of data that is available for continuously making arrival time predictions. Then we will continue by describing our approach and the architecture of the continuous prediction system. Lastly, we will look into evaluation approaches that we have taken to compare the proposed system with an existing one.

Information Society 2023, 9–13 October 2023, Ljubljana, Slovenia

2 PROBLEM SETTING AND DATA

The goal of the system is to predict arrival time to specific stations for each bus (more on this in [1][2][6]). To do this, we compute travel time predictions from specific stations to all remaining proceeding stations of the bus, per each bus. The data is suboptimal as we do not know the exact arrival or departure times to or from the stations (similar to [4]), which requires us to do extra processing on data and match bus positions to stations based on coordinates of bus locations and distances to nearby stations.

To address the suboptimal detailedness of data, we deal with detecting vicinities of buses to their applicable stations. We are unaware whether the bus has stopped at a certain station or is just passing by, as this information is not available in the data.

2.1 Bus Routes and Station Details

We use some static data, which gives details about routes. For each bus station, we have a location (latitude and longitude coordinates), along with ID and station name. Bus route is defined with a route number, variation, and list of stations for each variation.

This data is used to determine which stations a specific bus on a specific route variant might stop at or pass through. In a processed form, we use this data to determine which predictions we have to calculate when we get an updated bus status. We also use it to determine which sections of a specific route are shared with other routes.

2.2 Bus Positions

This is the main data that we use for computing predictions. Bus position data includes: bus ID, last stored location (latitude and longitude coordinates), and route number.

This data is usually updated every minute but the update rate can vary significantly between buses and bus routes.

Since we do not have information about exact arrival time to the station or departure time from a station, which would be preferable, we have to process bus positions to be able to use them as input for the prediction models.

To use bus positions as input data, we match a position to the nearest bus station, based on available bus stations on a specific route. Bus position is only matched to a station if it is within a certain distance to the station. For best performance, we use a radius of 50m from the station's position.

3 APPROACH DESCRIPTION

Our system uses recent historic data of travel times to include information about recent traffic flow among features (see [7]). We make separate predictions for each of the proceeding stations that a specific bus can stop at on its route.

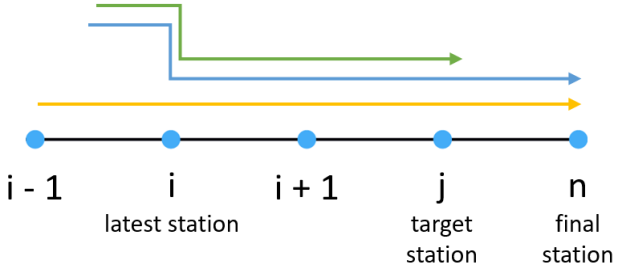

Figure 1: Schematic of bus routes

Let us say that bus *A*, for which we are making predictions, has departed station *'i'* (latest station). To get recent historic data, we check which bus routes share paths between the latest station of the bus *A* and the target station *'j'* for which we are making arrival time predictions. As we can see on Figure 1 above, Yellow route shares the path to target station *'j'* with green and blue routes. Thus, we can use the latest travel times between stations *'i'* and *'j'* on yellow, blue and green routes, to get the most recent data about traffic flow on this path.

coordinates of the bus, active route of the bus and the direction of the route that the bus is taking. After filtering bus stations based on route and direction, we compute distance to each station using the Haversine formula [9]. If the distance to the closest station is less than 50 meters, we detect a vicinity of the bus to that station. Once we have a vicinity match to a bus station, we process and insert the data into a list of detected vicinities to stations.

After each fetch routine, we store detected vicinities to stations to the data manager in the bus travel time predictor's data manager component. For easier comprehension, we can say that detected vicinities to the stations can be viewed as detected arrivals of buses to the station. After the data fetch cycle is complete and updated arrivals of buses to stations are ready in the data manager of the bus travel time prediction component, the regression machine learning model is used to predict travel times for all buses that have a new detected vicinity to a station for all of their proceeding stations.

At any given time, users can send a POST request to our proposed approach's bus prediction server API to get predictions either for all buses, all routes, specific buses, or specific routes. The system returns predictions in a JSON object and provides users with the most updated predictions for each bus.

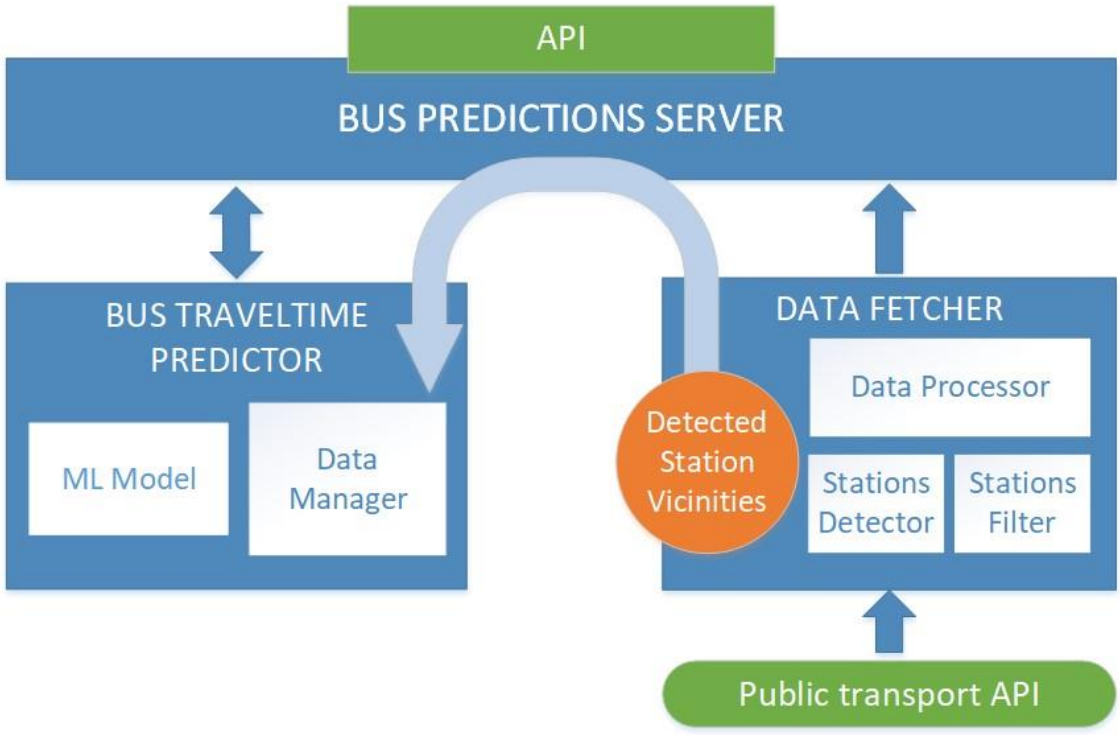

Figure 2: Architecture of the proposed solution

Which is why we also consider data from other routes that share the bus path for which we are making predictions. This way we get a better recent historic context to have a more reliable information about current traffic dynamics. This is especially useful for routes that have less frequent buses (e.g. once every 30 minutes or even less frequent).

The diagram on Figure 2 shows components that are active in the real-time prediction system. We continuously fetch bus positions from Public transport API several times per minute. Bus positions are matched to stations based on geographical

3.1 Positional Semantic Context

Since we have to match bus positions to stations and do not know when exactly a bus stopped, we use a positional semantic context of the bus. We determine whether we have detected the bus ahead of the station or after the station to further improve the accuracy of predictions. When the bus is detected ahead of the latest station we expect it to take longer time to reach the target station in comparison to when the bus is detected beyond the

latest station. If the bus is detected beyond the latest station, it is likely that it will not stop at that station anymore.

To detect the relative position of the bus to the latest station, we use coordinates from the first preceding station (i-1) and the first proceeding station (i+1) in addition to the coordinates of the latest station.

3.2 Machine Learning Models

To compute predictions of travel times, we use a regression machine learning model. We have trained and evaluated models based on several machine learning algorithms. These are: linear regression, SVM (SVR – Support Vector Regressor [3]), and an artificial neural network. We use implementations of these algorithms that are available in Scikit-learn [8], a Python library for machine learning. Models were trained on several weeks of data.

For training the SVM (SVR) model we use the RBF (Radial Basis Function) kernel with the epsilon parameter equal to 10.3. The regularization parameter C is equal to 1.0.

For training the neural network model we use the Multi-layer Perceptron regressor architecture [5] with 2 hidden layers (layer sizes: 15, 8). For solving the weight optimization, we use L-BFGS, which is a Limited-memory approximation of Broyden–Fletcher–Goldfarb–Shanno algorithm. Alpha hyperparameter is equal to 0.5, while learning rate is equal to 0.005.

Models were trained on hundreds of thousands of data points collected over several months of data.

SVM model is the best performing model of the tested ones which is why it is used as the part of our proposed approach in the following evaluation analyses.

4 EVALUATION

We mainly use two metrics to compare accuracies of predictions: MAE (Mean Absolute Error), and RMSE (Root Mean Squared Error).

To get a better overview of the performance of the system as a whole, we developed a web application that serves for analysis of performance of the system.

4.1 Live Evaluation System

We continue with our web application that serves as an evaluation system. With this system we can evaluate performance of our new system in comparison to the currently used system for predicting arrival time of buses. Results of our new solution are in blue color, whereas the results of existing solution are in green color. This web application can also be used for various purposes of evaluation, for example to compare updated models with earlier versions or compare performance of models that are based on different algorithms.

In all of the following figures, our system used the SVM (SVR) model to make predictions of bus travel times. The following figures were generated by evaluating predictions for a single route within a specific week.

To start the evaluation with an initial context of main metrics, the proposed system has MAE equal to 120 seconds and RMSE equal to 11042 seconds. Whereas, the current system has MAE equal to 357 seconds and RMSE equal to 46618 seconds for the selected period on the selected route. Since it is likely that certain extreme values have affected these measurements, we will look into further analyses with which we can also get a more informative understanding of performance of both systems and how they compare to each other.

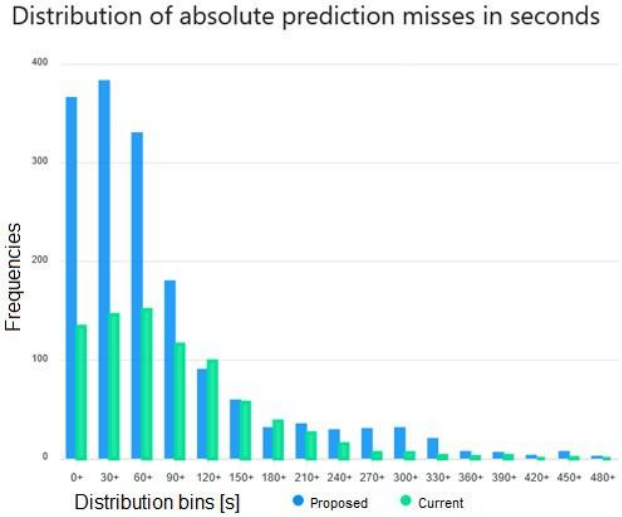

Figure 3: Enriched screenshot of distribution of absolute errors

On Figure 3 we can see how absolute errors are distributed among error bins. Each bin represents a 30 second interval of errors. The most left bin represents errors from 0 to excluding 30 seconds, the second left bin represents errors from 30 to excl. 60 seconds. We have to consider that there are more measurements present of the proposed system (blue bars) than of the current system (green bars). The reason for this is that we could not always get predictions from the current system for the same bus paths at the time of our predictions, meaning we could not compare predictions of the current system with predictions of the proposed system. The same applies to Figure 4 and Figure 5.

Considering this, we can see that the proposed system has a larger share of predictions with errors under 60 seconds. The most common error bin of proposed system is 30+ (30 to excl. 60 seconds), whereas for the current system it is the 60+ bin.

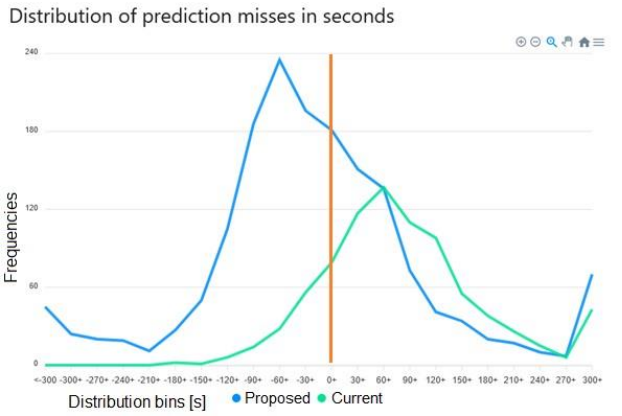

Figure 4: Enriched screenshot of distribution of negative and positive errors

On Figure 4 we can see how positive and negative errors are distributed between the proposed and the current prediction system. Errors are binned into bins of 30 seconds, except for the

most left and most right bins, which consist of all errors that have difference to actual time of more than -300 and 300, respectively.

Notice that the orange vertical line emphasizes the 0+ bin of errors, which consists of predictions with errors between 0 and 30 seconds. Equally well performing bin is the -30+ bin, which consists of errors between -30 seconds up to excluding 0.

In this case a negative error means that we have predicted that the bus will arrive at the station sooner than it actually has. This evaluation approach gives us better information about whether a system is more likely to have negative or positive errors. In case of negative errors, the system undershoots with the predictions. Similarly, in case of positive errors, the system overshoots with the predictions.

We can see that the proposed system is more likely to give predictions with negative errors, which means that the bus is more likely to arrive later than predicted. However, with the current system, predictions are more likely to have positive errors, meaning the bus is more likely to arrive earlier than predicted. Considering this, passengers are less likely to miss a bus if they plan their trip with the proposed system.

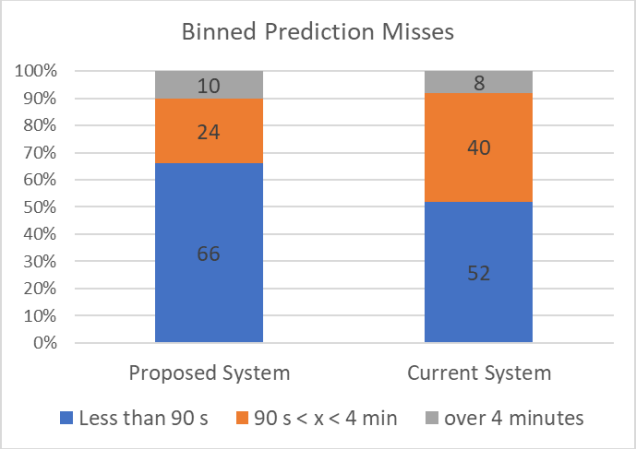

Figure 5: Binned absolute prediction errors

Upon discussion of acceptable prediction errors with the domain experts, they have determined that predictions with less than 90 seconds of absolute errors are the most desirable. Predictions that have absolute errors between 90 seconds and 4 minutes are considered less desirable but still acceptable. Predictions with over 4 minutes of absolute error are considered unacceptable. We have binned predictions into these three bins to further compare performance between the systems.

On Figure 5 we can see the comparison of distributions of predictions when taking opinions of domain experts into account. Blue parts of the bars represent the most desirable bins, orange parts present less desirable but still acceptable bins and grey parts represent unacceptable bins.

We can see that in 66% of the cases, predictions of the proposed system are sorted into the most desirable bin, compared to 52% of the cases of the current system. The proposed system has significantly less acceptable but undesirable predictions: 24% of selected predictions, in comparison to 40% of selected predictions of the current system. However, the current system does perform slightly better when focusing on the share of unacceptable predictions. 10% of predictions from the proposed system have unacceptably high errors, while 8% of predictions from the current system belong to the unacceptable bin.

When considering all angles of analysis, we can determine that the proposed system generally performs better than the currently used system.

5 CONCLUSION

We have overviewed the approach that we take as the basis for our system for predicting travel and consequently arrival times of buses. We looked into the architecture we implemented to support our approach and continuous computation of predictions for arrival times of buses. We then followed with a more detailed description of our evaluation system with which we can more easily compare two prediction systems – either the proposed system with the current system or different versions of the proposed system.

With the help of the evaluation application, we have also determined that the proposed system generally performs better than the currently used system.

For further improvements of the system, we could include the Relative Mean Absolute Error (often known as MAPE – Mean Absolute Percentage Error) as a metric in the evaluation system. This metric would give us a better understanding of the size of an error, relative to the time taken for the bus to finish the path for which the prediction was computed. We could further improve the evaluation application by adding a feature for comparing the distributions of errors with normalized values in bins, instead of only absolute values. This would streamline the analysis when example numbers differ between both systems.

We could also train additional machine learning models based on other algorithms, such as random forest and XGBoost, as well as include additional architectures of neural networks for a greater selection of models. We could then compare performances of all trained models with the use of our evaluation system.

ACKNOWLEDGMENTS

This work was supported by Solvesall, Carris, the Slovenian Research Agency and the European Union's Horizon 2020 program project Conductor under Grant Agreement No 101077049.

REFERENCES

[1] K. Birr, K. Jamroz and W. Kustra, "Travel Time of Public Transport Vehicles Estimation," in *17th Meeting of the EURO Working Group on Transportation, EWGT2014*, Sevilla, Spain, 2014.

[2] M. Čelan and M. Lep, "Bus arrival time prediction based on network model," in *The 8th International Conference on Emerging Ubiquitous Systems and Pervasive Networks (EUSPN 2017)*, 2017

[3] H. Drucker, C.J.C. Burges, L. Kaufman, A. Smola, V. Vapnik, "Support Vector Regression Machines," in Advances of Neural Information Processing Systems (NIPS), 1996

[4] A. Kviesis, A. Zacepins, V. Komasilovs and e. al., "Bus Arrival Time Prediction with Limited Data Set using Regression Models," in e 4th International Conference on Vehicle Technology and Intelligent Transport Systems (VEHITS 2018), 2018.

[5] F. Murtagh, "Multilayer perceptrons for classification and regression," in Neurocomputing, Volume 2, Issues 5-6, 1991

[6] D. Panovski and T. Zaharia, "Long and Short-Term Bus Arrival Time Prediction with Traffic Density Matrix," *IEEE Access (Volume: 8)*, vol. 8, pp. 226267 - 226284, 2020

[7] T. Yin, G. Zhong, J. Zhang, S. He and B. Ran, "A prediction model of bus arrival time at stops with multi-routes," in *World Conference on Transport Research - WCTR 2016*, Shanghai, 2016.

[8] Scikit-learn: https://scikit-learn.org/

[9] Haversine formula: https://en.wikipedia.org/wiki/Haversine_formula