Effects of weather data on traffic flow predictions using an LSTM deep learning model

Nikola Nachev[1]

Supervisor: Elena Congeduti

[1]EEMCS, Delft University of Technology, The Netherlands

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 25, 2023

Name of the student: Nikola Nachev
Final project course: CSE3000 Research Project
Thesis committee: Elena Congeduti, Georgios Iosifidis

Abstract

Accurate traffic forecasts are a key element in improving the traffic flow of urban cities. An efficient approach to this problem is to use a deep learning Long Short Term Memory (LSTM) model. Including weather data in the model can improve prediction accuracy because traffic volumes are sensitive to weather changes. The aim of this study is to show how such a model can be constructed for traffic flow predictions, and how it can be improved with the use of weather data. Results show that an LSTM model gives accurate predictions as a baseline model, and the inclusion of weather data gives a slight improvement in accuracy when predicting single sensors. The improvement was higher on long term predictions of 2.5 hours, and the best prediction results were obtained when adding a lag of 30 minutes to the rain data.

1 Introduction

Accurate traffic forecasts are a key element in improving the traffic flow of urban cities. They can help drivers and passengers make informed decisions about their trips with more accurate and reliable estimates of their trip durations. They can be used to reduce traffic congestion, and carbon emissions, and improve the overall traffic flow of the city.

In recent years, prediction models for traffic forecasting have been a popular research topic. More cities construct sensors for gathering traffic data, and as a result, the need for accurate prediction models grows. Various deep learning algorithms have been developed with high prediction accuracy [1]. However, there is lacking research regarding the usage of weather data in forecasting models, despite the impact that weather conditions have on traffic flow.

Deep learning models are suitable for traffic flow prediction. Traffic forecasting is a time series prediction problem, which means that the traffic data is continuous with time-based recurring patterns, like high traffic flow in the mornings and low traffic flow at night. For this type of sequential data, Recurrent Neural Networks (RNNs) are often used. These are deep learning models that take arbitrary-sized data as input [2] and have the ability to capture non-linear evolution of the data. However, training an RNN can result in exploding or vanishing gradients of the weights [1]. A model that resolves these problems is the Long Short Term Memory model [3], which is an extension of the base RNN. It uses memory cells consisting of an input, an output, and a forget gate as part of the feed-forward algorithm [4]. It is ideal for traffic forecasting as can be seen from [4], where it scored higher than other state-of-the-art forecasting models.

The LSTM model proves to be an effective base model, however, it can be improved by the inclusion of weather data. Pisano and Goodwin [5] found that weather conditions have a high impact on traffic flow, with a correlation between precipitation and traffic flow ranging from 15% to 30% in urban areas. Regardless of this high correlation, effectively using weather data has not always proved to be beneficial, as summarized by Tsirigotis et al. [6]. It needs to be correctly used and applied to the right models to get desirable results.

Related work has been done by Tsirigotis et al. [6] where they used precipitation data as exogenous variables to the linear regression models ARIMAX and VARMAX. These are modifications of the classical ARIMA model [7] to include external variables. However, statistical models like ARIMA generally struggle with unexpected events, and as a result, they underperform compared to LSTM in traffic prediction[4]. Nevertheless, they found a slight improvement in the ARIMAX model, but a significant improvement, between 30-50%, in the VARMAX model when they included the precipitation data. In addition, they discuss that weather changes usually take some time to have an effect on the traffic flow, and they found that adding a 1-hr lagged effect to the data gave the best prediction accuracy. The effects of these methods on the LSTM model are still unknown and will be discussed in this paper.

More closely related to the LSTM model, the authors of [8] have included weather data in a model with a Gated Recurrent Unit (GRU) architecture [9]. They found great performance increases when compared to their base model, up to 25%, however, they also made use of additional external data such as labeling days and time. As a result, it is hard to identify which additional features, or combinations of features, have an impact on the performance results. In addition, they add the external features to only one sampled traffic data point from one sensor. In this paper, data points from multiple sensors will be used in order to potentially explore spatial correlation. Therefore, the input data will need to be formatted differently and can consequently have different performance effects.

The goal of this paper is to show how traffic predictions can be accurately made with an LSTM model, to show how that model can be extended with the addition of weather data as exogenous variables, and to analyze the prediction accuracy effects of the usage of weather data in the model.

The second section of this paper will further go into detail about the LSTM model and the experiments conducted to analyze the weather effects on the model accuracy. In the third section, the datasets and experiments will be explained. Then, the results of the experiments will be presented. A discussion about the results and the limitations of the experiments will follow. Finally, in Section 7, conclusions will be drawn and future work and improvements will be discussed.

2 Methodology

The problem of traffic forecasting consists of using observed traffic flow data from previous timestamps to predict the traffic flow of the next timestamp. This is done by training a prediction model on a training dataset and then using the model with new input traffic data to generate predictions. The model takes an input vector $[x_1, x_2, ...x_t]$ of size T, to predict the value of x_{t+1}. x_t represents the number of vehicles observed between timestamps t and $t-1$. T represents the size of the lookback window, or the number of timestamps in the past used for predicting the next timestamp.

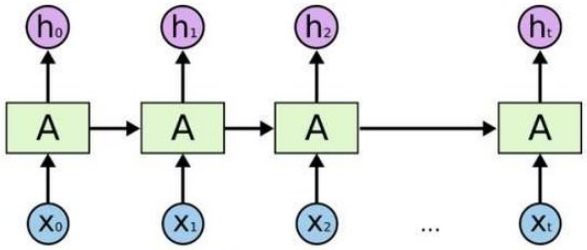

Figure 1: Recurrent Neural Network that takes timestamp inputs from X_0 to X_t to predict h_t

2.1 Long Short Term Memory prediction model

Traffic data is time-series data and in order to make efficient predictions, a model that can take arbitrarily-sized input from multiple timestamps for predictions is needed. Prediction problems of this type can be solved with the use of deep learning models [10]. In this research, the Long Short Term Memory model has been used for traffic flow predictions.

The LSTM is similar to the base Recurrent Neural Network model [2], and its similarities can be seen in Figure 1. Each cell A takes input data x_t corresponding to the data observed at a specific timestamp t. The cell produces output that will be used together with x_{t+1} as input in the next cell. The last cell takes input x_t and produces output h_t, which represents the prediction of x_{t+1}.

The main problem of the RNN is that the output h_t of a cell is multiplied by a weight and directly used as input in the next cell. This can become a problem during the training of the model, in the backpropagation algorithm, when the gradient keeps getting smaller (resulting in a vanishing gradient) or bigger (resulting in an exploding gradient).

The LSTM solves this issue by making the structure of the cells more complex by including additional activation functions and an additional connection between each cell. This can be seen in Figure 2, where each cell has two paths, the top path representing the long-term, and the bottom path representing the short-term memory. The weights together with the activation functions inside the cells dictate how much the long-term and short-term memory needs to be used, how much it should be forgotten, and also how much the input X_t needs to be used and remembered for the next cell.

2.2 Combining multiple sequential inputs

The field of traffic forecasting needs traffic predictions as accurate as possible, and a way to increase accuracy is to make use of additional external factors that have an influence on traffic flow. One example of apparent and easily accessible data is weather data. Similarly to traffic data, weather data is also sequential, which makes it easy to integrate it into the LSTM model.

Weather data is integrated into the LSTM model by combining the traffic input data together with weather data features. For each timestamp, the weather features are appended together with the traffic data, meaning that each traffic data point is transformed into a vector consisting of the data point and the weather features. This can be done when predicting one sensor, or multiple sensors.

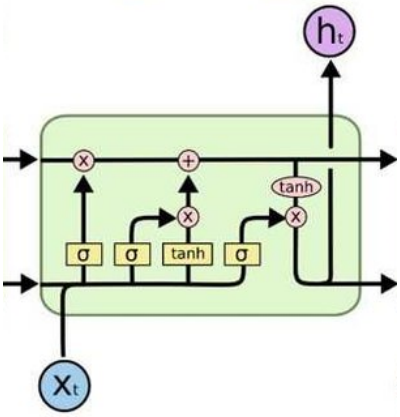

Figure 2: Components of a Long Short Term Memory Cell

The input to the model for predicting 1 sensor is represented by the matrix S.

$$S = \begin{bmatrix} w_1 & w_2 & \cdots & w_t \\ x_1 & x_2 & \cdots & x_t \end{bmatrix} \tag{1}$$

Here s_t represents the sensor traffic flow at timestamp t, and w_t represents the weather data at timestamp t. This input will be used for predicting the value of x_{t+1}.

When making predictions for multiple sensors, the input looks like the matrix M.

$$M = \begin{bmatrix} w_1 & w_2 & \cdots & w_t \\ x_{11} & x_{12} & \cdots & x_{1t} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{mt} \end{bmatrix} \tag{2}$$

The difference here is that for every timestamp t, $x_t = [x_{1t}, \ldots, x_{mt}]$ is a vector of size m, where m represents the number of sensors included in the predictions. Therefore, x_{mt} represents the traffic flow of sensor m at timestamp t. The weather features are again represented by w, which can be either the rain or the temperature data.

2.3 Analyzing weather effects

In order to analyze the effects of weather data on the LSTM model multiple experiments were conducted. Firstly, an analysis was done on the linear correlations between the traffic data and the rain and temperature data. This was done to get a better insight into the relations between the given datasets and to choose the specific traffic sensors that will be further tested.

Then, the effects of shifting the rain dataset were analyzed. Weather changes do not have an immediate impact on traffic flow, and it takes some time to get visible changes in the traffic flow as a result of the weather. In order to check if this can be helpful for the LSTM model, an analysis was done on the rain data with an included lag of different sizes. This included an analysis of the correlation differences between the different lag sizes and also an experiment checking the prediction accuracy differences when using the original rain dataset and the lagged dataset with the highest average correlation. The

dataset with the best prediction results was used for the remaining experiments.

The model was then tested on predicting the sensors with the highest and the lowest correlation with respect to the weather datasets to check the effects of the weather data on the prediction accuracy.

The models in the previous experiments were trained and tested on making predictions for only 1 future timestamp, corresponding to the next 15 minutes. However, LSTM models can also be used for making predictions for multiple timestamps in the future. Additionally, predicting further ahead in the future becomes harder, and therefore the LSTM model might benefit from the weather data more when making long term predictions. Therefore, the model was trained to make predictions for the next 10 timestamps, meaning that it creates predictions for the next 2.5 hours at every 15-minute interval.

Lastly, the model was tested on predicting the intersections with the highest and lowest correlations. This experiment was done to see the impact of the weather data when predicting multiple timestamps at once.

3 Experimental Setup

In this section, the datasets used for the experiments will be explained, and the configuration of the models will be specified. This includes the model architecture, the training strategy, and the evaluation criteria. Lastly, an experimental setting will follow that explains the setup for comparison between the models.

3.1 Description of datasets

To conduct the experiments, the models were trained and tested on data from multiple datasets.

For the traffic data, a dataset provided by the Traffic Department of The Hague Municipality was used. The dataset includes data from 128 sensors that measure the traffic by the number of cars passed. The dataset is on 15-minute intervals, representing the number of cars detected in that interval. Each sensor is placed on an intersection in The Hague. There are a total of 11 intersections with sensors in the dataset. The dataset contains traffic data for the month of November 2019.

For the weather feature data, the precipitation and temperature datasets from the historical weather API[1] from Open-Meteo have been used. This dataset contains observations on 1-hour intervals and the same period, November 2019, has been used for the experiments. The temperature is measured in Celsius degrees and represents the temperature at a given hour. The precipitation data represents the amount of rain in millimeters of the preceding hour. The weather data is at 1-hr intervals, so each data point was split into 4 intervals. The temperature data was duplicated at each interval, while the precipitation data was evenly split among the intervals to account for the average millimeters of rain in 15 minutes. The timestamps were then matched with the timestamps of the traffic data to be used in the model.

[1] https://open-meteo.com/en/docs/historical-weather-api#latitude=52.08&longitude=4.30&start_date=2019-11-01&end_date=2019-11-30&hourly=temperature_2m,precipitation&timezone=Europe%2FBerlin

3.2 Baseline LSTM model

The baseline LSTM model that will be used in the experiments has optimized hyperparameters using grid search. The hyperparameters that were optimized were the number of LSTM layers, the number of units per layer, and the look-back window. The baseline LSTM model consists of two LSTM layers with 500 units. It used a look-back window of 80, meaning that for each prediction, data from the last 20 hours was used. The model was trained with the Adaptive Model Estimation (Adam) algorithm [11] and was implemented in Keras. The model used early stopping with a patience value of 20 to stop learning after the validation loss has not improved for 20 epochs. The dataset was divided into a training, validation, and test set with ratios (0.6, 0.15, 0.25). The model used the Root Mean Squared Error (RMSE), calculated using Equation 3, as a loss function and was trained on 200 epochs on the training set.

$$RMSE = \sqrt{\sum_{t=1}^{n} \frac{(\hat{x}_t - x_t)^2}{n}} \tag{3}$$

The validation set was used to determine if the training needs to stop before the 200th epoch, and the test set was used to evaluate the model. The RMSE was used as a prediction accuracy metric to assess the model. Min-max feature scaling was applied to the traffic, precipitation, and temperature data, resulting in values between 0 and 1, calculated using Equation 4.

$$S = \frac{x - x_{\text{min}}}{x_{\text{max}} - x_{\text{min}}} \tag{4}$$

4 Results

In this section, the performance results of the model will be presented. Firstly, the performance results of the baseline model will be demonstrated. Then, a correlation analysis of the weather data will be presented and then the model will be tested in various settings to analyze the effects of the weather feature data.

4.1 Results of baseline model

The baseline model with optimized hyperparameters was run in different settings and gave satisfying results. In Figure 3 the predictions generated by the model are compared with the observation, and it is evident that the model is quite effective, as the predictions are often very close to the correct sensor data. The model becomes inaccurate only at timestamps where the traffic flow changes quickly. Table 1 shows the RMSE of 1 predicted sensor, the average RMSE of all predicted sensors in 1 intersection, and the average RMSE of all predicted sensors in the dataset. It is evident that the model is efficient at making predictions, and the prediction accuracy does not drop significantly when predicting multiple sensors.

4.2 Weather correlation analysis

The linear coefficients were calculated individually for each sensor with respect to the precipitation data, and the results can be seen in Figure 4.

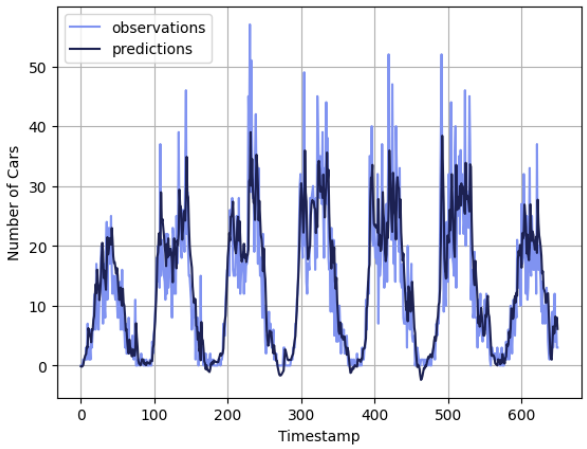

Figure 3: Predictions from the LSTM compared to the actual observations for a period of 7 days for predicting one sensor

Table 1: Root Mean Squared Error values when running the baseline LSTM model for 1 sensor, 1 intersection, and all sensors

Predictions for:	1 sensor	1 intersection	all sensors
Training RMSE	5.80	6.61	6.78
Test RMSE	6.59	7.81	7.84

Each column in the figure represents a group of sensors belonging to a specific intersection, and the correlation coefficients for each of the sensors in that intersection are plotted. Additionally, for each intersection, the plotted red lines represent the means of the correlation coefficients of that intersection. The correlation coefficient can have values between -1 and 1, with values above 0 indicating a positive correlation. It is evident that all sensors have a weak positive correlation with respect to the rain data with the mean correlations of the intersections ranging between 0.1 and 0.15. The sensor with the highest linear correlation has a coefficient value of 0.2069 which is a sensor in the K702 intersection, and the sensor with the lowest has a value of 0.00852. These two specific sensors have been tested with and without rain data and the prediction results can be seen in Section 4.4.

The linear correlation of the traffic data with the temperature dataset has been analyzed in the same way, and the results can be seen in Figure 5. From the results, it is evident that the temperature has a slightly higher linear correlation with the traffic data, as all intersections have a higher mean of correlation. However, an average correlation of 0.17 is still considered a low correlation, and there are also two sensors that have a negative correlation with respect to the temperature. The sensor with the highest correlation is part of the K159 intersection, with a value of 0.2805, and the sensor with the lowest correlation of -0.0158 is K701_12_1.

The results show that there is a weak positive linear correlation with the weather feature data. However, the strength of LSTM models is that they are also able to capture non-linear dependencies. Therefore, these linear correlations can be used as additional insight, but can not be used to make conclusions about the performance effect on the LSTM model yet.

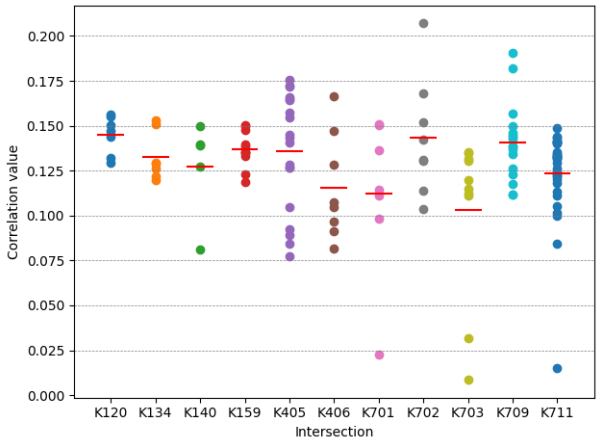

Figure 4: Correlation coefficients values for each sensor, grouped by intersection, with respect to precipitation feature data

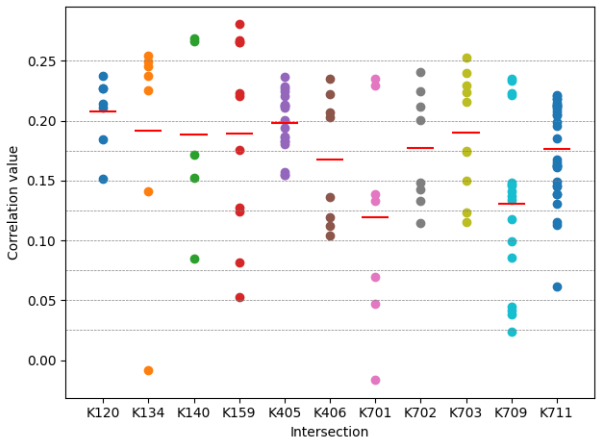

Figure 5: Correlation coefficient values for each sensor, grouped by intersection, with respect to temperature feature data

4.3 Adding a lag to the rain data

Including a lag of different sizes in the rain data was analyzed in order to check if it can improve the accuracy of the LSTM model. This was done by shifting the dataset to the right and appending zeroes at the start of the dataset. Firstly, the correlation effects were analyzed and can be seen in Figure 6.

For each lag interval larger than 60 minutes, it is evident that the averages of the correlations decrease for all intersections. For the smaller lags, however, different intersections have different correlation effects. In order to determine the lag with the highest linear correlation, the averages of the correlations between the rain data and all sensors in the dataset were taken. These averages can be seen in Table 2, and it seems that the highest linear correlation is achieved when the rain dataset is shifted by 2 timestamps, or by 30 minutes.

These correlation differences, however, are not significant. Additionally, a higher correlation does not necessarily mean

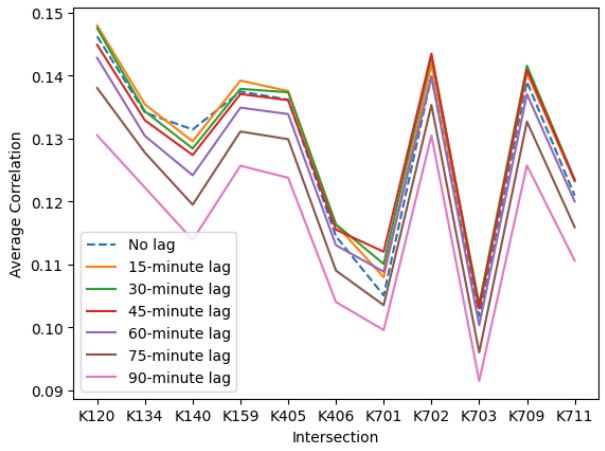

Figure 6: Correlation coefficient values for each sensor, grouped by intersection, with respect to temperature feature data

Table 2: Mean of the correlation coefficients of all 130 sensors with respect to the rain data with different lag sizes

Rain data lag	mean of correlation coefficients
no lag	0.1278
15-min lag	0.1294
30-min lag	0.1296
45-min lag	0.1290

higher prediction accuracy. To check if this is true, the LSTM model was tested on a dataset with no lag and a dataset with a 30-minute lag. The results in Table 3 show that including a 30-minute lag on the dataset improves the prediction accuracy on the sensor with the highest correlation. It resulted in an RMSE improvement of 4.6%. As a result, the lagged dataset was used for the next experiments.

4.4 Predicting chosen sensors

In order to analyze the effects of rain and temperature data on the LSTM model, the model was tested on 4 chosen sensors. The chosen sensors have the highest and lowest linear correlation values with respect to the weather features. This way, it will be shown how the linear correlations relate to the actual prediction accuracy effects on the model.

In table 4 the results are shown when running the model for the sensors with the highest and the lowest correlation with respect to the rain data. These are the sensors K702_01_1 and K 711_08_2 with correlation coefficient of 0.207 and 0.008.

Table 3: RMSE values when running the LSTM model with rain data with and without lag on the sensor with the highest correlation with the rain data

Test RMSE	highest correlated with rain
no lag	3.13
30-min lag	2.99
% Improvement	4.6%

Table 4: RMSE values when running the LSTM model with and without rain data on the sensors with the highest and lowest correlation with the rain data

Test RMSE	highest correlated	lowest correlated
Baseline	3.15	5.49
With Rain	2.99	5.33
% Improvement	5.4%	3%

Table 5: RMSE values when running the LSTM model with and without temperature data on the sensors with the highest and lowest correlation with the temperature data

Test RMSE	highest correlated	lowest correlated
Baseline	5.98	2.23
With Temperature	6.16	2.26
% Improvement	-3%	-1.3%

The results show that the sensor with the highest correlation improves by 5.4% when including rain data, while the sensor with the lowest improves by 3%. In this case, a higher linear correlation corresponds to a higher accuracy improvement. However, it is remarkable that even for a sensor with a very low linear correlation, the results are improved by 3% when adding lagged rain data.

The sensors with the highest and lowest correlation with respect to the temperature data are the sensors K159_712 and K701_12_1 with correlation coefficients of 0.28 and -0.016. Remarkably, the sensor K701_12_1 presents an unusual pattern: all the observations remain under 20 cars per 15-minute intervals, except on three consecutive intervals where there were 105, 201, and 659 cars observed. This made the model less accurate overall and gave highly inconsistent results. To avoid skewed results of the experiment, these three data points were excluded from the dataset used for the experiment.

The results in Table 5 show that the LSTM model did not benefit from the temperature data. Training on the highest correlated sensor with temperature data had a 3% decrease in performance, while the sensor with the lowest correlation had a decrease of 1%. In this case, there is no correspondence between the linear correlation of the datasets and the model performance, like with the rain.

4.5 Long term predictions

The results of the experiment for predicting the next 10 timestamps, or 2.5 hours, can be seen in Table 6. For each prediction, the RMSE is calculated by averaging the RMSE of all 10 predicted timestamps. From these results, we can see that the model improved by 7.3% with rain data. Additionally, this improvement is higher than the 5.3 % improvement when predicting for 1 timestamp only, or the next 15 minutes. The RMSE values here are higher than the RMSE values for predicting one timestamp only, meaning that it is harder for the model to predict multiple timestamps in the future. Therefore, the additional weather data is more useful and the model gets a higher improvement in accuracy.

Table 6: RMSE values of the LSTM model with and without rain data for predicting the next 2.5 hours on the sensors with the highest correlation with the rain data

Test RMSE	highest correlated sensor 2.5 hours predictions
Baseline	4.13
With Rain	3.85
% Improvement	7.3%

Table 7: RMSE values when running the LSTM model with and without rain data on the intersection with the highest and lowest correlation with the rain data

Test RMSE	highest correlated	lowest correlated
Baseline	7.54	8.34
With Temperature	7.8	8.25
% Improvement	-3.3%	1%

4.6 Predicting intersections

The model was tested on the intersections K120 and K703 for predicting multiple sensors at once. K120 had the highest average correlation with rain with a value of 0.147, and K703 has the lowest with 0.104. The results in Table 7 show that in both cases the model does not benefit from weather data, with a performance improvement of 1% on K703 and a 3.3% decrease on K120. This means that the model is not able to properly use the rain data when making predictions for multiple sensors. A discussion of these results is given in Section 6.

5 Responsible Research

When doing research in the machine learning field, the main privacy or safety concerns may arise from the data that is being used. However, the datasets used in this research are completely anonymized and have no safety or privacy issues. The reproducibility of the results, on the other hand, is a common issue with LSTM models. At the start of each training session, the LSTM randomly initializes the weights and the training process will be different every run, resulting in trained models with different final weights. This means that each trained model will give different evaluation scores, and it makes it hard to exactly reproduce the results gathered from the experiments. To address this, every experiment was run 5 times and the evaluation scores in the results represent the mean of the results of the individual runs. Additionally, it was checked that the variance between the results was not too high to ensure that the inherent randomness of the model does not affect the validity of the results.

6 Discussion

The results of the experiments show that including precipitation data in an LSTM model increases the accuracy of predicting single sensors, however, the increase of up to 7.2% is not very significant compared to related previous work. Zhang and Kabuka [8] found a 25% increase when adding weather data to a similar GRU model. However, a big part of that increase was due to the time label features that were added to the model. Additionally, they used much larger datasets (39,000 sensors over 2 years). The models in this paper used data of 1 month and this might not be enough for the model to properly learn the rain correlations. In addition, the model is also not able to capture seasonal weather correlations for the same reason.

The model was also not able to properly use weather data when predicting multiple sensors at once. The reason for the bad results here might be due to the input format of the data. The rain data is appended to the vector of sensors' data forming a vector $X_t = [x_1, x_2, .., x_m, w]$. With this type of input, the LSTM might be trying to train one weight for the rain for all the sensors, but the rain affects each sensor differently. Modifying the feature vector to a two-dimensional vector like Matrix S_t in Equation 5 might fix this issue, as each sensor will have a separate weight for the rain. The combined input will then be of the form $S = [S_1, S_2, .., S_t]$. However, Keras does not allow for two-dimensional feature vectors, so this idea will be explored as a possible future direction of this work.

$$S_t = \begin{bmatrix} w & w & \cdots & w \\ x_1 & x_2 & \cdots & x_m \end{bmatrix} \tag{5}$$

7 Conclusions and Future Work

The aim of this paper was to show how an LSTM model can be used for traffic flow predictions, to explain how that model can be extended with the use of weather data, and to analyze the effects that the weather data had on the predictions of the model.

The results of the experiments showed that including precipitation data in the LSTM model increased the prediction accuracy of individual sensors by up to 5.4%. Some sensors benefitted more from this additional data, and performing a correlation analysis proved to be useful for highlighting such potential sensors. The LSTM model was better able to utilize the rain data when it was lagged by 30 minutes, with an increase of 4.6% in accuracy. The model had a higher benefit from the rain data when performing longer-term predictions, with a 7.2% increase in accuracy when predicting for the following 2.5 hours.

The model was also unable to appropriately utilize the weather data when making predictions for multiple sensors at once. This remains an open question, as it turned out not to be trivially solved by our model. Future work can be done on this and there are multiple possible directions. If the original structure of the model needs to be kept, then an implementation of the model with a more flexible framework, such as PyTorch, will be necessary. This will allow for multidimensional feature data and will give more flexibility in experimenting with the input format. Another option in Keras would be to implement a Convolutional 2d LSTM layer that will also be able to accept multidimensional feature data. Another possible approach is to individually predict each sensor with an independent LSTM model. However, with this approach, the models will not be able to learn the spatial relations between the sensors, which could give worse performance results overall.

The prediction accuracy improvements found in this paper were not very significant. However, if the model is used on more diverse datasets that span across a larger period of time, more significant performance improvements could be observed when applying the findings of this paper.

References

[1] A. Miglani and N. Kumar, "Deep learning models for traffic flow prediction in autonomous vehicles: A review, solutions, and challenges," in *Vehicular Communications*, The Institution of Engineering and Technology, 2019.

[2] R. Pascanu, C. Gulcehre, K. Cho, and Y. Bengio, "How to construct deep recurrent neural networks," arXiv preprint arXiv:1312.6026, 2013.

[3] S. Hochreiter and J. Schmidhuber, "Long short-term memory," in *Neural Computation*, vol. 9, p. 1735–1780, 1997.

[4] Z. Zhao1, W. Chen1, X. Wu1, P. C. Y. Chen2, and J. Liu1, "Lstm network: a deep learning approach for short-term traffic forecast," in *IET Intelligent Transport Systems*, vol. 11, pp. 68–75, The Institution of Engineering and Technology, 2017.

[5] P. A. Pisano and L. C. Goodwin, "Arterial operations in adverse weather," in *Institute of Transportation Engineers 2004 Annual Meeting*, FHWA Road Weather Management Program and Mitretek Systems, 2004.

[6] L. Tsirigotis, E. I. Vlahogianni, and M. G. Karlaftis, "Does information on weather affect the performance of short-term traffic forecasting models?," in *International Journal of Intelligent Transportation Systems Research*, vol. 8, pp. 1–10, 2012.

[7] B. Williams and L. Hoel, "Modeling and forecasting vehicular traffic flow as a seasonal arima process: Theoretical basis and empirical results," vol. 129, pp. 664–672, 11 2003.

[8] D. Zhang and M. R. Kabuka, "Combining weather condition data to predict traffic flow: a gru based deep learning approach," in *IET Intelligent Transport Systems*, vol. 12, pp. 575–585, The Institution of Engineering and Technology, 2018.

[9] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," in *IET Intelligent Transport Systems*, arXiv preprint arXiv:1412.3555, 12 2014.

[10] B. Lim1 and S. Zohren, "Time series forecasting with deep learning: A survey," arXiv preprint arXiv:2004.13408, 2020.

[11] D. P. Kingma and J. Ba, "Adam: a method for stochastic optimization," in *3rd International Conference for Learning Representations*, arXiv preprint arXiv:1412.6980, 2015.