Traffic Congestion Prediction in London

Group C

Abstract

Prediction of traffic flow has been an expanding field of study and London, being a densely populated city, has to handle huge amounts of traffic daily. High levels of traffic congestion can seriously impact the lives of citizens. In this study, we collect historic traffic data from the UK Department of Transport and propose multiple approaches to make future predictions of traffic congestion and analyse the performance of each model.

1 Introduction

Road traffic significantly affects the quality of life of citizens and the prediction of traffic flow is an approach used by many researchers to be able to provide traffic condition information to citizens to alleviate traffic congestion. Discussions have been held highlighting the importance of parallel control and management in ITS, underlining the need for accurate traffic flow predictions to enhance traffic management [Wang, 2010]. There also have been demonstrations of how traffic flow prediction is integral to developing responsive and adaptive traffic systems [Zhang et al.(2008)]. However, accurate prediction of traffic congestion is challenging due to the influence of a wide range of factors of traffic such as pollution, accidents, traffic jams, public transport closures, weather, etc. which are not easy to incorporate in prediction models. Currently, human expertise is employed to inform people about current traffic conditions, but with recent advancements in artificial intelligence (AI), machine learning (ML), deep learning (DL) and access to big data, research in this field has been steadily expanding.

London, being one of the most densely populated cities in the country, especially faces high levels of traffic congestion. In this project, we aim to propose common predictive models for road traffic prediction, apply them to traffic data in London, and analyse their performance.

2 Problem Formulation

We begin with logistic regression which serves as a baseline model that provides forecasts and insights into the relationship between features and target variable. To handle the high-dimensional nature of traffic data, the Random Forest model is used, and its hyperparameters are carefully tuned to enhance performance and computing efficiency. Recent advancements in neural networks have highlighted their flexible structures and learning abilities; thus, we deploy two neural networks, an artificial neural network (ANN) model and a Long Short-term Memory (LSTM) recurrent neural network model, which performs well with sequential data that is ordered in time.

Data Description

Traffic congestion forecasting aims to predict congestion based on historical traffic data. Our proposed models were applied to the data collected by the Department for Transport between 2008 - 2010 and 2014 -2017 (‘Road traffic statistics - Download data’(n.d.)). The data was collected hourly from 7:00 to 18:00. It comprised data from different locations, under different local authorities, also mentioning the

road name and classifying them as major or minor. Accurate locations of the vehicles are specified by the latitude and longitude. The count of each type of vehicle added to the diversity of the data.

3 Contributions to Existing Literature

Most of the existing literature is aimed at forecasting traffic flow (i.e., the total number of vehicles passing a point in a given time), which is not informative of the actual traffic conditions. This is because of the lack of information on various factors such as the size of the road, the types of vehicles on the road, speed of the vehicles, etc. which can better provide an insight into the level of congestion than simply the number of vehicles at a given time.

Therefore, we aim to provide more informative results by computing a weighted sum of the vehicles by their size, instead of considering the total number of vehicles. Our reasoning behind this is that since bigger vehicles take up more space, a larger number of bigger vehicles cause more traffic than a similar number of smaller vehicles. Below are the average dimensions of vehicles (with an appropriate safety margin included in the dimensions) by type in London:

- Pedal cycles: 3m, 1.5m (300mm safety on each side) [London(n.d.)]
- 2 wheeled motor vehicles: 3m, 1.5m (300mm safety on each side) [London(n.d.)]
- Cars and taxis: 4.6m, 2m (100mm safety on each side) [Masters(2021)]
- Buses and coaches: 11.4m, 2.7m (100mm safety on each side) [‘New routemaster aesthetic mid-life refurbishment’(n.d.)].
- LGVs: 12.2m, 2.7m (12m length is found; width we took same as HGVs)[‘Weights & dimensions’(2021)].
- HGVs: 13.7m, 2.7m (100mm safety on each side) [Clements(2023)]

Based on this, we calculate the following weights according to the area taken by each vehicle on the road:

- Pedal cycles: 1
- 2 wheeled motor vehicles: 1
- Cars and taxis: 2.04
- Buses and coaches: 6.84
- LGVs: 7.32
- HGVs: 8.22

Based on the sum calculated with these weights, we segregate our data into 4 classes by quantiles (which allows for balanced classes): ‘Less Traffic’, ‘Some Traffic’, ‘Moderate Traffic’ and ‘Heavy Traffic’. We make these classes separately in the case of London’s major roads and minor roads, which allows our models to better understand the occurrence of different traffic conditions based on our features and then make appropriate congestion predictions for future traffic data.

4 Proposed Solutions

The motivation behind our proposed solutions is to review some of the most popular approaches used in road traffic prediction and reproduce them for our specific dataset. We decide to use 5 approaches for our task, discussing their specific advantages and disadvantages and then drawing insights from our results.

We first carefully preprocess the data to drop highly correlated features, and missing value rows and encode our data appropriately. Our data is highly skewed; we remove only the most extreme outliers to prevent loss of information (as outliers in traffic data may be genuine, due to various reasons such as traffic accidents or extreme weather).

4.1 Logistic Regression

Logistics regression is a simple yet widely used classification model where an observation is assigned to a class based on the probabilities. It uses a logit function where the output(y) is mapped as the sigmoid function of the inputs(x's). In our case, since we have 4 classes, we implement a multiclass logistics regression model as our benchmark model. To solve for the probability values, the following are the required equations [Caya(2017)].

$$\text{Class 1:} \quad p_1 = \frac{e^{\beta_1^T x_i}}{1 + \sum_{i=1}^{K-1} e^{\beta_i^T x_i}}$$

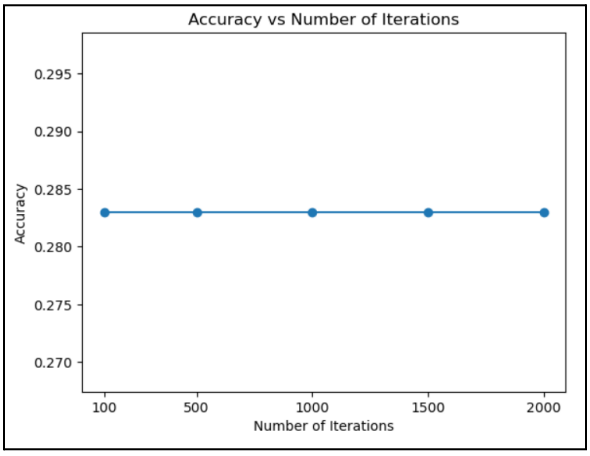
$$\text{Class 2:} \quad p_2 = \frac{e^{\beta_2^T x_i}}{1 + \sum_{i=1}^{K-1} e^{\beta_i^T x_i}}$$

$$\vdots$$

$$\text{Class K-1:} \quad p_{K-1} = \frac{e^{\beta_{K-1}^T x_i}}{1 + \sum_{i=1}^{K-1} e^{\beta_i^T x_i}}$$

$$\text{Class K:} \quad p_K = \frac{1}{1 + \sum_{i=1}^{K-1} e^{\beta_i^T x_i}}$$

To implement this multinomial model, the optimization algorithm that we use is the Limited-memory Broyden–Fletcher–Goldfarb–Shanno (L-BFGS), which is well-suited for optimising smooth, differentiable functions like those encountered in logistic regression. L-BFGS indicates the number of past positions and gradients to store for the computation of the next step ['Two-Class Logistic Regression: Component reference - Azure Machine Learning'(n.d.)].



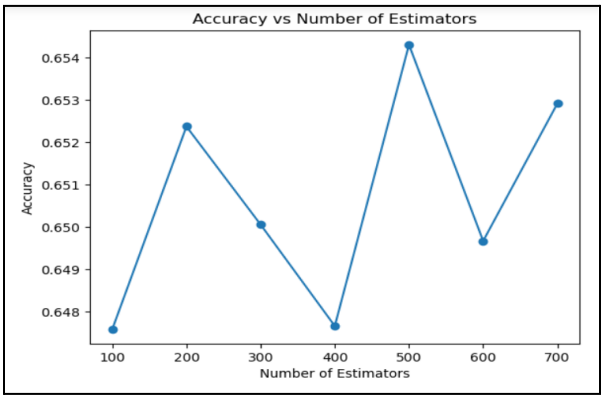
Further, we note that the number of iterations does not affect the accuracy of the model. Thus, we set 'max_iter' to 500 for the optimization algorithm to converge in 500 iterations.

Our model gives an accuracy of 28.3%, which means that it is not performing well. A major limitation of the logistic regression model is the assumption of linearity between the features and the target; our low accuracy indicates that there are more complex relationships in the data. More powerful and compact algorithms such as neural networks can easily outperform this algorithm [AmiyaRanjanRout Follow(2020)].

4.2 Random Forest

Random Forest is based on its capability to effectively handle high-dimensional spaces and a large number of training examples, without the assumption of a linear relationship between features and the target variable. To make a prediction, each decision tree in the forest considers a random subset of features when forming questions and only has access to a random set of training data points [Koehrsen(2017)]. This increases diversity in the forest leading to more robust overall predictions. The application of Random Forest in our dataset is done considering its ability to manage complex data structures. The Random Forest algorithm's strength lies in its ability to handle complex datasets and mitigate overfitting, making it a valuable tool for various predictive tasks in machine learning [Sruthi(2021)]. To fit the model to our dataset, we first convert our training dataset into a 1-dimensional array that is suitable for the model and use the following hyperparameter in our model:

1. **Number of Trees (n_estimators)**: It refers to the number of trees in the forest. As we note, the number of estimators gives the highest accuracy for the model when set to 500.
2. **Random_state:** It controls the randomness of the sample so that the model always produces the same result as it has a definite value (0) of random state and has been given the same hyperparameters and training data.



In our model, the Random Forest algorithm follows a set of procedures to guarantee precise predictions. Initially, it selects random subsets of data points and features from the dataset, picking 'n' records and 'm' features from the total 'k' records. Utilising these subsets, individual decision trees are constructed for each sample. Each of these trees then generates its output. Finally, the algorithm consolidates these outputs to determine the final prediction. In classification tasks, the most common output among the trees is chosen. This process capitalises on the strengths of multiple trees to produce a more accurate and reliable prediction than a single tree could provide.

The Random Forest model's effective implementation and strong performance demonstrate how well it handles the enormity and scope of our dataset, which may contain high-dimensional feature space and non-linear relations. The performance of the Random Forest model in the dataset is evaluated using accuracy. The model's accuracy is 65%. This implies that the Random Forest correctly classifies 65% of the classes. This accuracy acts as a benchmark level for further models. Some limitations specific to this model are that it takes longer training times, and is less interpretable compared to simpler models [Koehrsen(2017)].

4.3 Artificial Neural Network (ANN)

Artificial Neural Network is a first-order computational model that consists of a set of interconnected processors or neurons [Akhtar and Moridpour(2021)]. They can learn from the data even if the underlying relationships are not apparent and of a non-linear nature. This makes them a useful tool for working with datasets with noise [Lahogianni et al.(2005)]. This model was chosen for this project given this adaptive

ability. The neural network structure is largely chosen on trial and error as there are no fixed rules. This model uses a multilayer perceptron model for predicting the 'traffic class'. To be specific, there are three layers in this neural network. Two hidden layers are adjusted for the complexity of the dataset. The first hidden layer consists of 64 neurons and the other consists of 32 neurons. The output layer has four neurons to account for the four classes that we want to predict using the model.

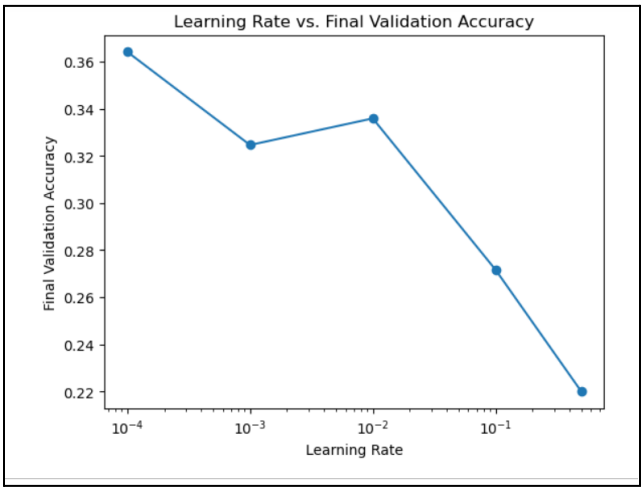
An important step in structuring a neural network is choosing an activation function. These functions have a variety of characteristics, which are deemed essential to successful learning. This model uses the rectified linear unit (ReLU) as the activation function. For classification problems, ReLU performs much better and it is more efficient for deeper neural networks. We used this activation function in the input and hidden layers. For classification problems, ReLU performs much better than other activation functions and it is more efficient for deeper neural networks. We use this activation function in the input and hidden layers. For the output layer, we use the softmax activation function which is suitable for a multiclass classification problem like ours and is widely used for the output layer in classification networks. We implement this structure using the tensorflow and keras library in Python. The feedforward, backpropagation, and gradient descent for updating the weights follow the equations:

FeedForward: $Output_n = Activation(Weights_{n-1} * Output_{n-1})$
Backpropagation: $E_{n-1} = W_n^T * E_n$
Weight Updates: $W_n = W_n * (\text{Learning rate}) * \frac{\delta En+1}{\delta W_n}$
Where E is the error function and W represents the weight.

The following hyperparameters are present in our model:

1. Units in each layer (the number of neurons in each layer)
2. Learning rate (the rate of convergence to the optimal weights while training neural networks. This is tuned and the optimal learning rate used is 0.01)
3. Epoch (tells the model about backpropagation. Here it is 80. This number is a compromise between underfitting and overfitting)
4. Batch size (the number of samples the model takes during one propagation while training, usually a value in multiples of 2)

The Adam optimizer which is a version of stochastic gradient descent is used. The optimiser is called Adam because it uses estimations of the first and second moments of the gradient to adapt the learning rate for each weight of the neural network ['Adam'(n.d.)]. Methods like GridSearchCV and other searching algorithms to tune and find the correct combination of all the hyperparameters were attempted but given this dataset and the complexity in terms of size, it was computationally expensive, and therefore,



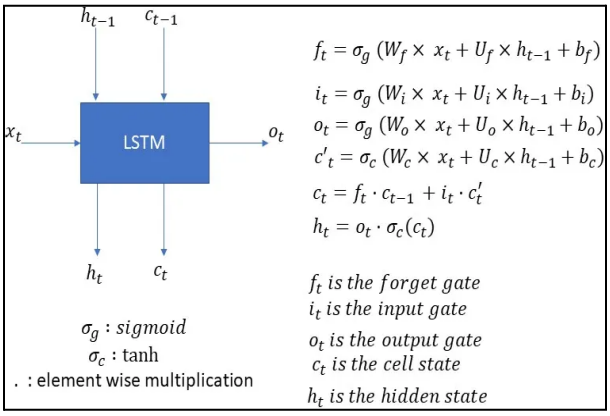
hyperparameters other than learning rate are chosen based on trial and error. For the optimal learning rate, five different learning rates are chosen from 0.0001 to 0.5 and their accuracy on the validation set is checked.

The model is then trained with the chosen parameters. Each epoch is monitored and checked for validation accuracy. If there is no change in accuracy in the validation set over three consecutive epochs, the model training is stopped to prevent overfitting. This model works with an accuracy of 40.23%. This accuracy is pretty low but is higher than naively classifying inputs into four different classes. A potential reason for low accuracy is that this model, unlike LSTM which the project has discussed later, doesn't generalise well to a time-dependent dataset like ours. Therefore, it loses significant information about this time-dependent relationship. With the problem statement in this project, it is imperative to take into account the time factor for traffic class prediction based on traffic flow which this model fails to do.

4.4 Long Short-term Memory (LSTM) Model

Our traffic data consists of sequential observations; order matters since current traffic conditions depend on past traffic conditions. Due to this reason, we propose the Long Short-term Memory model to make predictions for future traffic congestion by understanding the time-dependent relationships in our data.

The Long Short-term Memory (LSTM) model is a particular type of Recurrent Neural Network (RNN), which is a neural network that has a feedback loop (in simple words: memory) as compared to feedforward neural networks. The LSTM architecture is similar to the standard RNN, with the hidden layer being the point of difference between the two [Chen et al.(2016)]. In general, RNNs work well when we process data which is in the form of a time series (or is collected over certain time intervals) [Buhl(2023)].



The input data for an LSTM model has to be a 3D vector; thus, we prepare and carefully reshape our data to be inputted in the right manner for the model to then train on. We initialise the model and add 3 LSTM layers to it followed by a dense output layer. The model has several hyperparameters including:

1. The number of time steps (subjective to the data at hand)
2. Units in each LSTM layer (the number of neurons in each layer)
3. Learning rate (the rate of convergence to the optimal weights while training neural networks, between 0.1 and 0.0001)
4. Epoch (tells the model about backpropagation and is usually 50, 100, 200 - it is best to start with lower values first and then increase the number of epochs until its optimum value is found, else there is a risk of overfitting)
5. Batch size (the number of samples the model takes during one propagation while training, usually a value in multiples of 2)

Hyperparameter tuning for an LSTM model has no rigid guidelines and is done specific to the dataset. The tuning of the hyperparameters (2), (3), (4) and (5) is computationally complicated for such a large dataset. So, we adjust the values based on trial and error for each of the parameters, selecting those values (from the above-mentioned 'usual' values) that minimise the loss. Since our data is collected hourly, we decided to use the number of time steps as 3 to predict future hourly traffic conditions based on the 3 previous records of traffic conditions. Specifically, the traffic condition at x_{t+1} is predicted by the sequence: (x_{t-2}, x_{t-1}, x_t).

To evaluate the performance of our model, we use the sigmoid activation function to implement LSTM with a single output node and then compute the accuracy of the model. The model has a loss of 0.1875 and an accuracy of 75%, hence it is performing well.

The advantages of using LSTM include that it is well-suited to process sequential traffic data. It can capture important information from the 3 previous time steps and use that information to make future predictions. LSTM is also less prone to overfitting when provided with large training data, especially with the implementation of 'early stopping' in case of no change of accuracy for three consecutive epochs. This is evident by seeing a negligible difference in accuracy on the test dataset.

It is important to note that the data is not uniformly spaced in time (i.e., the data was only collected between the hours of 7 a.m. and 6 p.m.). This is a common issue that arises when using real data, and it is important to acknowledge the potential limitation of this data while deploying LSTM as it could affect the model during training. Moreover, the tuning of hyperparameters for an efficient LSTM on such a large dataset often requires powerful hardware and thus, is complicated; our selection of parameters based on trial and error could have reduced our efforts to optimise the model during training and deployment [Srivatsavaya(2023)].

5 Conclusion

Model	Result	Time
Logistic Regression	Accuracy: 28.29%	164.065 ms
Random forest	Accuracy: 65.4%	34562.11 ms
Artificial Neural Network (ANN)	Accuracy: 40.54%	25156.13 ms
Long Short-term Memory (LSTM)	Accuracy: 75%	21803.65 ms

For our traffic dataset, the LSTM model has the highest accuracy followed by the Random Forest. SVC has the lowest accuracy as it is a simple Machine Learning model and is unable to capture the complex non-linear relationships between the features and target variables. ANN also performs poorly on our dataset as it fails to capture the time-dependent relationship of the variables.

As compared to most of the earlier studies aimed at predicting traffic flow, our research intends to address the issue of traffic congestion in London by incorporating factors such as the size of the vehicles and road type. Employing an amalgamation of artificial intelligence (AI), machine learning (ML), and deep learning (DL) algorithms on historic data from the UK Department of Transport, we effectively analyse traffic patterns and note the performance as well as time complexity of each model.

5.1 Limitations

Our data was highly skewed and removing all outliers would mean losing a large amount of data. It is also possible for outliers in traffic data to be genuine; so, we only removed the most extreme outliers to prevent the loss of information, which meant that our data had many outliers. Data was only collected from 7:00 to 18:00 which limited the prediction scope as the predictions could only be accurately made for times during this interval. The study might not provide accurate results for timings beyond this interval, i.e., during the off-peak traffic hours in London. Various factors like accidents, road closures, and weather conditions are important as they significantly influence traffic conditions. The current models could not account for these uncertain factors.

5.2 Future Work

Exploring deep learning models, such as Convolutional Neural Networks (CNNs) or more advanced recurrent neural networks like GRU (Gated Recurrent Units), could provide better feature extraction and temporal pattern recognition. Including more diverse data sources, such as social media feeds, CCTV footage and IoT sensor data could further enhance accuracy and responsiveness to real-time changes. Information about the weather conditions could be incorporated by scraping data using API. Addressing the limitations of the current approaches and exploring the suggested areas could significantly enhance studies in traffic congestion prediction.

6 References

F-Y Wang. Parallel control and management for intelligent transportation systems: Concepts, architectures, and applications. *IEEE Trans. Intell. Transp. Syst.* 11(3):630–638, 2010. doi: 10.1109/tits.2010.2060218.

N Zhang et al. DynaCAS: Computational experiments and decision support for ITS. *IEEE Intell. Syst.* 23(6):19–23, 2008. doi: 10.1109/mis.2008.101.

https://roadtraffic.dft.gov.uk/downloads. Accessed: 2023-12-17.

M O F London. Urban Motorcycle Design Handbook. *https://content.tfl.gov.uk/tfl-urban-motorcycle-design-handbook.pdf*. Accessed: 2023-12-12.

L Masters. What are the average dimensions of A car in the UK? *https://www.nimblefins.co.uk/cheap-car-insurance/average-car-dimensions*, 2021. Accessed: 2023-12-12.

https://foi.tfl.gov.uk/FOI-1423-2223/New%20Routemaster%20Technical%20Specification%20FOI.pdf. Accessed: 2023-12-12.

https://lgvtheory.co.uk/topic/weights-dimensions/, 2021. Accessed: 2023-12-12.

S Clements. New rules on HGV length: How long is a lorry? *https://anthonyjones.com/new-rules-hgv-length/*, 2023. Accessed: 2023-12-12.

P O Caya. Multi-class logistic regression - Pete Caya - medium. *https://medium.com/pete-caya/multi-class-logistic-regression-75e04bea31dc*, 2017. Accessed: 2024-01-01.

https://learn.microsoft.com/en-us/azure/machine-learning/component-reference/two-class-logistic-regression?view=azureml-api-2. Accessed: 2024-01-01.

A AmiyaRanjanRout Follow. Advantages and disadvantages of logistic regression. *https://www.geeksforgeeks.org/advantages-and-disadvantages-of-logistic-regression/*, 2020. Accessed: 2024-01-02.

W Koehrsen. Random forest simple explanation - will koehrsen. *https://williamkoehrsen.medium.com/random-forest-simple-explanation-377895a60d2d*, 2017. Accessed: 2024-01-01.

E R Sruthi. Understand Random Forest algorithms with examples (updated 2024). *https://www.analyticsvidhya.com/blog/2021/06/understanding-random-forest/*, 2021. Accessed: 2023-12-12.

M Akhtar and S Moridpour. A review of traffic congestion prediction using artificial intelligence. *J. Adv. Transp.* 2021:1–18, 2021. doi: 10.1155/2021/8878011.

E I Lahogianni et al. Optimized and meta-optimized neural networks for short-term traffic flow prediction: A genetic approach. *Transportation Research Part C*. 13:211–234, 2005.

https://optimization.cbe.cornell.edu/index.php?title=Adam. Accessed: 2023-12-12.

Y-Y Chen et al. Long short-term memory model for traffic congestion prediction with online open data. *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)* (2016).

N Buhl. Time series predictions with RNNs. *https://encord.com/blog/time-series-predictions-with-recurrent-neural-networks/*, 2023. Accessed: 2023-12-12.

P Srivatsavaya. LSTM — implementation, advantages and diadvantages. *https://medium.com/@prudhviraju.srivatsavaya/lstm-implementation-advantages-and-diadvantages-914a96fa0acb*, 2023. Accessed: 2024-01-08.