Department of Computer Science

Submitted in part fulfilment for the degree of BEng

Pollution Forecasting Using Traffic Statistics to Reduce Exposure with a Feed-Forward Neural Network

Jake Schoonbrood

16 June 2021

Supervisor: Iain Bate

ACKNOWLEDGEMENTS

I would like to thank my supervisor, Dr. Iain Bate for his continued assistance, support and understanding throughout the project.

STATEMENT OF ETHICS

This project has no ethical concerns. All data is artificially generated, and no data has been taken from external sources. Real world application will have numerous ethical concerns; however this is outside the scope of the project but is briefly discussed throughout and acknowledged when real world application is considered.

TABLE OF CONTENTS

Executive summary **8**
1 Introduction **1**
1.1 Air Pollution **1**
1.2 Effects of Air Pollution on Human Health **1**
1.3 Vehicle Contribution to Air Pollution **2**
1.4 Dispersion of Air Pollution & Exposure **3**
1.5 Why Vehicle Occupants? **3**
2 Literature Review **5**
2.1 Neural Networks and Pollution Forecasting **5**
2.2 Navigation **7**
2.3 Results Analysis **8**
2.4 Simulation Tools **8**
2.5 Vehicle Emissions Model **9**
2.6 Summary **10**
3 Problem Analysis **11**
3.1 Objective **11**
3.2 Data Usage & Pre-Processing **12**
4 Design And Implementation **13**
4.1 Overview **13**
4.2 Initial Simulations & Data Extraction **13**
4.3 File Processor & Classification Generator Design **14**
4.4 Neural Network Design **15**
4.4.1 Dataset 15
4.4.2 Architecture 16
4.5 Simulation Testing Design **17**
4.5.1 Dynamic Routing Design 18
4.5.2 Cost Function Design 18
4.6 Hurdles **20**
4.6.1 Simulation Optimisation 20
4.6.2 Neural Network 21

5 Results Analysis & Evaluation **22**
5.1 Pollution Exposure **22**
5.2 Route Duration **24**
5.3 Route Difficulty **25**
5.4 Discussion **26**
5.4.1 Ethics 26
5.4.2 Computational Expenses 26
5.4.3 Acknowledgements 27
5.5 Results **27**
6 Conclusion **29**
Appendix A **31**
Appendix B **32**
Appendix C **33**
Appendix D **34**
Appendix E **35**
Appendix F **36**
Appendix G **37**
Appendix H **38**
Appendix I **39**
Appendix J **40**
Appendix K **41**
Appendix L **42**
Bibliography **44**

TABLE OF FIGURES

Figure 1 - A graph showing the increase in traffic across the United Kingdom. [Source: https://roadtraffic.dft.gov.uk/summary] 2
Figure 2 - Pollution Exposure Analysis for Cost A Emission-Optimal Routing Against STF Routing 22
Figure 3 - Pollution Exposure Analysis for Cost B Emission-Optimal Routing Against STF Routing 23
Figure 4 - Total / Average Trip Duration (s) charts for each routing method 24
Figure 5 - Total / Average Trip Distance (m) charts for each routing method 24
Figure 6 - Average Trip Difficulty for each routing method 25

TABLE OF TABLES

Table 1 - List Of Secondary Objectives... 11
Table 2 - Cost A Proposed Weights.. 19
Table 3 - Cost B Proposed Weights.. 19
Table 4 - Result Breakdown for Cost A Emission-Optimal Routes... 27
Table 5 - Result Breakdown for Cost B Emission-Optimal Routes... 28

Executive summary

Modelling air pollution is an incredibly complex task due to the seemingly random behaviour of the dispersion of pollution. Pollution levels depend on a variety of factors, which can often be separated into factors that contribute (often caused by us) and factors that influence the dispersion of air pollution (typically, meteorological conditions). The world health organisation “estimates that around 7 million people die every year from exposure to polluted air”. Governments are beginning to take action to reduce the automotive industry’s involvement in poor air quality, for example the United Kingdom (UK) plans to prohibit the sale of new petrol and diesel cars by 2030 to move towards net-zero emissions.

The shift to electric vehicles will assist in the clean-up of polluted air in densely populated regions. However, a growing demand for Lithium Batteries has shown correlations with changes in environmental conditions at the location of mines which is a vital factor to consider. This transition to electric vehicles is only beginning, despite the ban of new combustion-engine vehicles after 2030, the second-hand car market will still permit the resale of these vehicles. As combustion-engine vehicles will still be road-worthy, air pollution remains a problem. So alternative methods need to be found to reduce air pollution and exposure. Emission-optimal vehicle routing has been explored and has shown impressive reductions in pollution exposure, often requiring live and accurate emissions data taken from sensors to be able to make informed decisions on how to route vehicles. To measure air pollution with sensors accurately on a global scale is not a cost-effective solution to reduce exposure.

Pollution forecasting is a low-cost alternative compared to using sensors, using different forms of machine learning to predict pollution. Neural networks, being one of the most advanced methods have shown an impressive ability for identifying patterns and trends across sizeable datasets that a human mind would not be able to identify. They are exceptional at being able to process large data and making predictions based on this data. More specifically, neural networks have shown better results over standard regression models when forecasting pollution.

The proposed navigation method contained in the paper used multiple basic feed-forward neural networks to classify pollution levels into categories based on traffic data. The cost of each category will act as a weight against the estimated exposure time vehicle occupants will experience travelling along an edge (road). The final solution will be compared against abstract versions of current navigation methods to determine its effect on route quality and pollution exposure.

Due to the difficulty of collecting real world data and extensive cost, data is generated from running simulations on real world roads with random numbers of vehicles to account for all traffic profiles. Unfortunately, this leads to a huge variation of data that is based on random traffic profiles that may not be possible at all in real-life.

The aim of this project is to be able to model air pollution based on traffic data to reduce emission exposure for vehicle occupants by feeding routing algorithms road networks where the weights are emissions-based so that an emission-optimal route may be found. With the hope of directly being able to influence vehicle occupant emission exposure, I anticipate in real world application, rerouting vehicles through routes that are not shortest route optimal will also reduce the strain on city infrastructure and level out the overall variation of pollution making the task of reducing overall pollution more manageable. This may make most journeys more efficient by reducing the time spent in traffic and at idle- however to formalise this as a measurable performance metric is not possible within simulations.

My initial results have shown promising impact, with the emissions-optimal routing outperforming an abstract version of typical vehicle routing. On average my solution reduces maximum pollution exposure. However, it is based on traffic information and to better simulate real world conditions, a pollution model that accounts for all conditions that contribute or effect is needed. A neural network offers the expandability to introduce more factors when simulations (and pollution information) become more advanced.

There are no ethical issues caused by the collection of artificially generated data, although real world application would see numerous ethical issues. For example, apps tracking your current location, speed and destination would have to be approved by the user. The method aims to reduce pollution exposure for vehicle occupants but can result in generating more pollution overall by travelling longer routes. With a secondary goal of evenly spreading pollution to reduce areas of high pollution levels, we can significantly impact someone’s life whether it be indirectly by causing pollution levels to rise aggravating a medical condition, or directly by increasing traffic flow through roads that previous had a lack of traffic.

1 Introduction

1.1 Air Pollution

Since 1990, vehicle emissions have been decreasing thanks to government and manufacture efforts. However, in recent years, the decline of emissions has plateaued [1]. An obvious assumption that manufacturers are struggling to increase the efficiency and emission output of combustion engines past its current point tied with the number of vehicles on the road is increasing yearly.

People are starting to become more aware to their own carbon footprint coupled with their contribution to climate change and the eco-system. Our impact on the environment is becoming more common in main-stream media, increasing awareness and concern regarding the effects of pollution. Countries have begun to try and reduce pollution emitted from vehicles, for example the United Kingdom plan to prohibit the sale of new petrol and diesel engine vehicles by 2030 [2].

Forecasting air pollution can help deter pollution related health problems [3] and raise awareness about effects on the environment. Though, air pollution is rather difficult to model due to the affect meteorological conditions have on its dispersion [4]. Unfortunately, this is not the only factor that makes it difficult, vehicle traffic has shown to impact the dispersion patterns [5].

1.2 Effects of Air Pollution on Human Health

High levels of air pollution coupled with long term exposure can result in both short- and long-term health problems with the main contributor being the combustion of fossil fuels [6]. Chronic Respiratory Disease & Lung Cancer [6][7] are amongst the common health problems associated with exposure to pollutants. Urban areas are prone to high pollution levels due to both the automotive traffic for both industrial and public and industrial development [8][9]. It is important to tackle air pollution in developing cities and densely populated areas as a report estimates that 7 million people die yearly from air pollution, of which 4.2 million of those deaths are a result of ambient air pollution [10].

A report [8] acknowledges the poor air quality in cities, and the contributors can be categorised into a few main categories, specifically motor traffic, industry & powerplants being the most significant. Although the publication of the above mentioned is outdated, it anticipated that there would be continued growth of the automotive industry leading to increased pollution levels and poor air quality.

Figure 1 shows the summary of all registered vehicles in the UK from 1994 to 2018, which shows significant growth every year for all vehicle classes- with cars and taxis being the main vehicles that make up vehicle traffic on the roads.

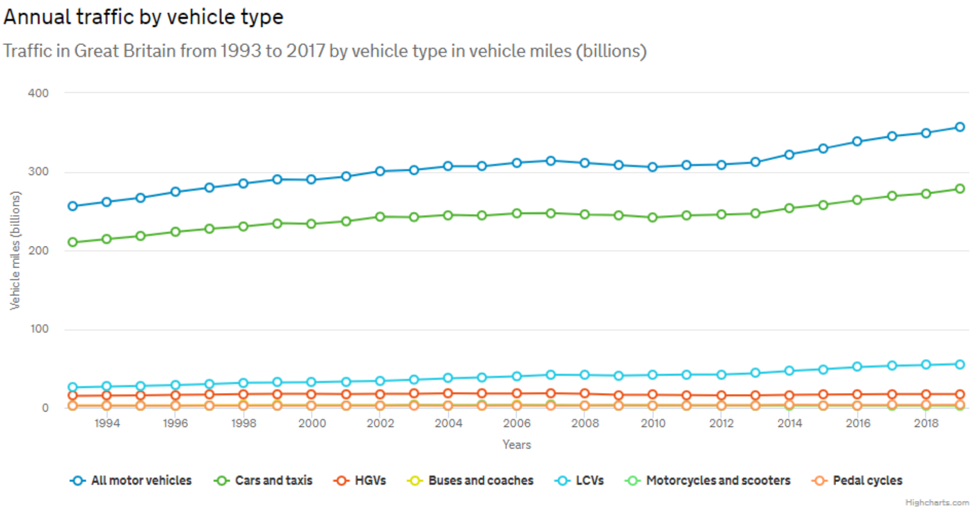


Figure 1 - A graph showing the increase in traffic across the United Kingdom. [Source: https://roadtraffic.dft.gov.uk/summary]

1.3 Vehicle Contribution to Air Pollution

Combustion engine vehicles have significant impact on the mixture of gasses in the air. Some of the harmful by-products from fossil-fuel combustion consist of CO, NO2, HC and PM [11]. The main by-products are briefly discussed.

A combustion engine ignites fuel by either spark or compression. Under most conditions, fuel burn is uneven resulting in incomplete combustion (where the fuel has not completely reacted with the air mixture). A result of this incomplete reaction is Carbon Monoxide (CO), a harmful gas to inhale with short- and long-term effects depending on exposure and is indirectly related to climate change [12][13]. Some long-term effects are severe as heart disease and brain damage given long enough exposure time [13]. Although it is unlikely that long-term problems will occur from traffic as there is not a high enough concentration that vehicle occupants will be exposed to for an extended period. Long term effects are more likely to be caused from leaving a vehicle idling in an enclosed space such as a garage. In extreme cases, vehicles with leaking exhausts can vent exhaust fumes into the cabin, however this is very rare and often noticeable.

Nitrogen Oxides (NOx), which is not considered harmful until Nitric Oxide (NO) reacts to form Nitrogen Dioxide (NO2) which is harmful to health [14]. Inhaling high concentrations of Nitrous Dioxide can directly harm anyone with respiratory diseases [15] which is a concern for people living in populated areas as a study had shown that diesel vehicles produced significantly more Nitrogen Oxides of which most of it was made up of Nitrogen Dioxide [16]. Measures have been taken in some cities to prevent entry to older diesel vehicles that produce these toxic emissions as cities tend to suffer with lack of dispersion.

Several factors are responsible for the dispersion of pollution in the air; while on one hand natural elements such as wind and humidity variations constitute a crucial portion of the problem, on the other hand, high vehicle speeds and static vehicles are major contributors. The build-up of pollution can be mitigated by reducing the number of vehicles that are static at any given time and keeping a consistent flow of traffic. By keeping vehicles moving, pollution will disperse at a higher rate and exposure will be reduced.

1.4 Dispersion of Air Pollution & Exposure

Vehicle induced turbulence, caused by higher average vehicle speeds can affect the transport of air pollution and hence its dispersion [17]. However, given a two-way street in an urban area, it was found that when both sides had the same traffic density and velocity, the induced turbulence from either side essentially offset each other resulting in no significant effect on the dispersion of air pollutants- the main influence was still the wind speed and its direction.

Given a real-life scenario, it is unlikely at any given time on a two-way road, that both traffic profiles on either side will have equal traffic density and vehicle speeds, therefore- the dispersion will always be influenced by the induced turbulence of the vehicles, with a higher influence when vehicle speeds are larger. So, it is possible that pollution exposure can be reduced by travelling on a higher speed road with either less traffic, or a more consistent traffic flow.

Travelling at higher speeds can produce more pollution overall due to the greater resistance against the vehicle and the higher force an engine must produce to overcome this resistance, studies have shown that air pollution near major roads have a higher concentration than minor roads [18]. For an emission-optimal route, the vehicle must travel a route that minimizes emission exposure and emission output by keeping with a consistent speed.

1.5 Why Vehicle Occupants?

A review of numerous pollution studies [19] had found in most studies, vehicle occupants were exposed to higher concentrations of poor air quality compared to non-motorised modes of transport. Most drivers use navigation apps to route themselves through cities, and as vehicles are already bound to roads, it is a lot more effective to route vehicle traffic than pedestrians. However, while vehicle occupants may be exposed to higher concentrations of emission related gases, both pedestrians and cyclists inhale higher doses for longer periods of time as they are exerting energy via physical activity [20].

The routing of vehicles can have a significant effect on exposure to pollution for both vehicle occupants and pedestrians, but it is difficult to measure this effect as it depends on a variety of factors and may not hold under all circumstances. But by rerouting a significant number of vehicles based on pollution exposure, we are reducing the number of vehicles contributing to pollution in congestion zones, where pollution is already likely to be high. This is mainly applicable for urban areas, where there are many pedestrians and vehicles with the same goals (I.e., commuting to the centre of London), yet this may not have a positive effect on all pedestrians, but only the pedestrians travelling through congestion zones. This is an ethical issue that will need to be considered as we may be increasing pollution levels in areas of already low pollution.

2 Literature Review

Directing vehicles along an emissions-optimal route requires air quality information. There are many explored methods of emission-optimal routing, typically requiring either an emissions model (which usually cannot account for all factors that affect air quality) or pollution sensors for live data. In most scenarios, pollution sensors are not available for use due to costs and complexity when implementing vehicle routing. Instead of needing sensors, I propose the use of pollution forecasting as an alternative.

Pollution forecasting is used to predict the concentration of gasses and particles in the air, which eliminates the need for pollution sensors across the globe. Pollution forecasting uses weather models and machine learning techniques such as neural networks to make predictions. To support the planned work, Neural Networks that specialise in Pollution forecasting will be reviewed to establish possible applications that can be used to reduce pollution exposure. We then discuss how we can implement these methods to reduce pollution exposure for vehicle occupants.

2.1 Neural Networks and Pollution Forecasting

Neural Networks are a specialist form of machine learning, they are based on an abstract understanding of how the human brain works. The network consists of a collection of neurons, where each neuron has an assigned weight that was determined during training through several complex calculations. Neural Networks can adapt to perform many different tasks, often performing on non-linear problems which involve complex calculations with no visually recognizable pattern. They also offer great expandability.

A Feed Forward Neural Network is the most common type of neural network, each layer is sequential and the output of a set of neurons from one layer will be fed to the next layer. Therefore, the connection between neurons has no cycles, indicating no advance features like memory. They are often able to perform on most datasets with good results, however, using different forms of neural networks can show significant improvements- depending on the input data.

Each neuron in a neural network is essentially a mathematical function. The inputs to a neuron are multiplied by a weight before being summed to find the average weighted sum. Once the average weighted sum has been found, an activation function can be applied which will determine the neurons output value. The activation functions are often non-linear functions, which give neural networks to advantage of being able to perform on non-linear problems over techniques such as linear regression.

Feed Forward Neural Networks have shown good results for predicting the air pollution index in Malaysia [21]. A common metric to measure a neural networks performance is to evaluate the Root Mean Squared Error (RMSE) of its predictions against the actual values. The best model achieved an RMSE score of 7.151 [21]. The lower the RMSE score, the greater accuracy the neural network has. A key thing to note is that the dataset consisted of weather conditions instead of traffic despite stating urbanization and motor vehicles are the main source for air pollution. The difficulty of collecting live accurate traffic information is a factor for the choice of dataset. This is a common trend amongst most pollution forecasting research [22] [23].

Pollution levels often have a strong correlation strong with pollution levels from previous time intervals (dependent on the time between each interval). So provided a dataset is in a time-series format, specialised neural networks with a form of memory can take advantage of this to influence their predictions.

Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) Neural Networks both implement memory, with both being able to account for previous timestamps, however an LSTM was designed to solve the problem with RNNs struggling to learn when the weights for each neuron are small [24]. This is known as gradient vanishing and results in the network struggling to learn when several time intervals are chosen, which is why they are typically chosen over RNNs.

LSTMs have shown good results when forecasting pollution, with slight performance improvements over the previously mentioned, feed-forward neural network [21]. However, different datasets have been used, but for comparison purposes, the RMSE of an LSTM calculating the next hour’s pollution score in Northern Taiwan achieved an RMSE of 4.03 (the best score) ranging to 9.85 (the worst score) which was predicting the pollution score 4 hours in advance and was in Southern Taiwan, the worst performing dataset out of all 4 possible datasets **(Appendix A)** [23]**.** Although this may be a weak comparison considering both papers used different datasets, there is an obvious, but miniscule performance gain using an LSTM over Feed-Forward given similar datasets.

Due to fact I will have to artificially generate data, it is not wise to generate it in time-series format due to obvious overlapping difficulties between road IDs and multiple road networks. An LSTM network may perform minorly better, it is not significant enough that it should be chosen over a feed-forward network unless real world implementation was considered. Therefore, I will be using a feed-forward network to develop my solution.

2.2 Navigation

With the neural network architecture chosen, we need to consider how to implement it with different routing algorithms. Methods of routing can be put into two main categories, static and dynamic routing. Static routing can calculate an initial route between the departure location and destination under current environmental variables (Weather, accidents, traffic). Static routing is not as computationally expensive as dynamic routing and it cannot acknowledge changes to environmental variables either, so the calculated route may differ from the true optimal route.

Dynamic routing [25] addresses these problems by updating the route while the journey is ongoing, these updates can be at set intervals or in response to environmental variable changes. For example, a route may be updated every minute, or a route may be updated if an accident has been reported. Dynamic routing is significantly more computationally expensive as it is expensive is proportional to journey time. However, it can acknowledge and consider changes to the environment to provide a better-quality journey.

Search algorithms can be used in conjunction with emission-based costs, with the weights for travelling each road being emission based. Search Algorithms can be split into two categories, heuristic, and non-heuristic algorithms. For example, Dijkstra is a non-heuristic search algorithm, but its alternative, A* search is a heuristic best-first search algorithm which determines what nodes to explore by a heuristic function (cost function).

A* is similar to Dijkstra, but instead of taking the shortest path, it uses a heuristic function to determine which node to move along next. An example heuristic function is shown below.

$$f(n) = g(n) + h(n)$$

In standard application, f(n) is the overall cost for node *n*, *g(n)* is the cost from the start node to the current node *n* and *h(n)* is the estimated cost from the current node *n* to the end node. In our application, g(n) would be the known emission cost so far to reach the current node and *h(n)* would be an estimate of emission cost to reach the end node. But as we are predicting emissions throughout the journey, we will never know the true emissions on the roads that have been travelled, therefore *g(n)* cannot be defined, so it is not possible to implement this form of heuristic search.

An emissions-optimal routing system was implemented by T. Cannon [26] using the sum of two considered true emission values (Carbon Monoxide and Nitrogen Oxide) to determine vehicle routing. Using continuous emission values and an edge cost function based on emission threshold values determines whether an edge had too much pollution and to exclude it from the route. T. Cannon assumed knowledge of emissions would be known by sharing information between vehicles, however this is difficult considering the number of older vehicles on the road that cannot share such information. Instead of assuming accurate emission values are at hand, we will predict emission value labels that determine how badly a road is polluted and determine routes based on these labels and evaluate the route quality.

2.3 Results Analysis

To determine if the results generated by our solution are effective, we need to consider human behaviour amongst all the statistics. To determine if a route is of good quality can be subjective depending on the individuals' goals. Human behaviour is a difficult to model, the consensus is a driver often wants to take the fastest route to the destination whether that be by time, or distance. However, this is not always the case, choices can depend on the destination, route difficulty, safety, and the reason for travelling [27]. Route choices can also be linked to past experiences, whether good or bad. It is impossible for us to know all these variables to decide for a user what route to take.

The driver is considered an agent with imperfect information and irrational; therefore it will not often choose the method that results in the maximum payoff, often deviating from the optimal choices based on its own views and information fed. It is possible that a driver may ignore suggested routes regardless of the quality of the route, with that being said, it does not mean the performance of the suggested routes does not matter. Despite the driver possibly ignoring the suggest route, the driver is more likely to follow guidance when provided good quality routes continuously.

To account for factors that drivers may find important, time taken to travel to the destination cannot be the only metric we consider when defining the performance of the suggested route. Other performance metrics that will be considered: emission exposure (Carbon Dioxide won't be considered as it is not directly harmful to human health), distance travelled, number of different roads travelled.

2.4 Simulation Tools

To implement the proposed solution, a microscopic traffic simulator is required. A microscopic traffic simulator software can model every vehicle and its behaviours individually [28] which will allow most realistic traffic behaviours.

The software will also need to be capable of providing traffic and emission data, so an initial dataset can be built for a neural network to train with. Secondly, the ability to interface with the simulation and control traffic routes is required to determine the effectiveness of the proposed solution. For obvious reasons, this is not capable with currently available datasets and that's why simulation software is required.

There are many open-source traffic simulation tools available, with community wide support. One traffic simulator stood out above the rest, SUMO [29]. SUMO's an open-source traffic simulator based in Java. SUMO has built-in emission models for all types of vehicles and the ability to extract data by interfacing with its python library, TraCI [30].

2.5 Vehicle Emissions Model

The number of pure electric cars on the road as of 2020 was recorded as approximately 216,000 out of the total 38.6 million vehicles licensed on UK roads [31]. Out of every 5000 vehicles on the UK road, approximately 28 of them are electric. Therefore, it is not essential to have an emissions model that accounts for electric vehicles.

There are many factors that can affect the emissions of vehicles, from engine type, compression ratio & ignition timing all the way to the weather-related factors such as rain and temperature. It is incredibly difficult to model emissions accurately. These emissions can even vary down to the drivers driving style and choices made.

The chosen simulator, Simulation of Urban Mobility (SUMO) uses HBEFA3 [32] as default for emissions modelling. However, not all emissions are implemented from the HBEFA3 Model, the key pollutants have been implemented, which are the ones that will be used. HBEFA3 also summarises emissions produced across vehicle classes into models. The default emissions model, HBEFA3/PC_G_EU4, represents passenger vehicles under the euro norm IV emissions standard [33]. All vehicles within the simulation will be using this emission standard to reduce the complexity of forecasting traffic pollution.

HEBFA3's implemented model within SUMO cannot account for acceleration [34], with constant acceleration and deceleration experienced when driving through populated areas the model may give inaccurate pollution levels. HBEFA3 is better suited to modelling emissions produced by consistent driving, typically on dual carriageways, motorways, and b roads.

2.6 Summary

Emission-optimal routing all suffer from the same problem, either requiring live accurate emissions data, or developing a model that lacks expandability into real world application. T. Cannon [26] suggested the use of VANETs [35] to transfer emissions information from between vehicles, in principle this is an effective solution. But this would require all vehicles on the road to be capable of developing an emissions model for itself based on factors such as acceleration, engine load, temperature and more. Each vehicle would need to recognise the road they are on and communicate the emissions that they are producing on this road. Real world implementation of this would be difficult unless all older vehicles were taken off the road and this system became a manufacture standard.

By using neural networks to model pollution instead, we remove the dependence of needing to know every vehicle's exact emission output and allow variation.

3 Problem Analysis

3.1 Objective

Emission-Optimal Routing has been proven an effective solution for reducing pollution exposure, however, the technology is not so easily available to make this method of navigation viable globally.

The proposed method in this paper simulates information that navigation apps can collect from there users while the application is in use. A study of 500 smartphone users found that 77% of smartphone owners regularly used navigation apps [36]. With most smartphone users using navigation apps, it is possible to update traffic information regularly based on users GPS data with minimal delay depending on the local environment (the degree of urbanization). For example, Waze [37] has already implemented a system to determine the average speed of traffic and notify the user when the average speed is significantly less than the speed limit, indicating congestion. Further implementation details will not be discussed in detail within this report.

Knowing that it is possible to get live-road data globally using crowdsourcing navigation apps / devices, and that neural networks have shown great prediction accuracy when forecasting pollution levels using datasets based on meteorological conditions and time-series data; I have chosen to use a neural network to predict air pollution based on traffic conditions. By using a neural network, we can use a low-cost global wide solution to reduce emission exposure for vehicle occupants in urban areas. The proposed method aims to produce a global solution to emission-optimal routing, to achieve this, I have broken down the solution into objectives.

ID	Goal
Objective 1	Define a neural network that will predict pollution labels given traffic information with respectable accuracy.
Objective 2	Implement custom cost-based vehicle routing method using the neural network to define the cost.
Objective 3	Improve vehicle routing by optimising cost functions and edge choices.

Table 1 - List of Secondary Objectives

3.2 Data Usage & Pre-Processing

All maps are created using real road networks. Selections were made from OpenStreetMaps [38] and exported using either OpenStreetMaps or Planet OSM [39]. From there, all maps were converted from OSM format to a readable format for SUMO (.net.xml in addition to other configuration files). I opted to use my own created simulations rather than open-source simulations for numerous reasons. Firstly, I would like to be able to capture all traffic behaviours, no matter how unlikely, this is best done by having a variation of multiple simulations based in different locations with different traffic profiles. Secondly, the solution needs to work with any road network and not tailor to a single city's traffic profile otherwise the neural network may increase pollution exposure. Lastly, the offered open-source simulations are large and selected parts of the simulation would need to be converted to smaller simulations so that they run in a reasonable amount of time. This would need to be done multiple times and may not even guarantee that all possible traffic profiles have been generated, therefore it is not a good use of time due to uncertainty amongst the datasets.

All vehicles for each map were randomly generated using SUMO's randomTrips.py found within its program files. All trips had random sources, destinations and departure times, the number of vehicles on the road were set in accordance with visual information using SUMO-GUI to determine how many vehicle trips to generate. With the training simulations setup, data is then collected at set intervals during the simulation to collect traffic and pollution information for each edge in the network. This is incredibly time consuming, so simulations were kept small to prevent data becoming corrupt.

As every road in the simulation does not have constant traffic, it is likely that a lot of the data extracted will have no use, to prepare the data to be fed into a neural network, multiple steps will need to take place. Firstly, all obsolete data must be removed. Obsolete data is defined as, an edge that has no pollution information or traffic. With all obsolete data removed, each edge with traffic statistics can be mapped to pollution labels.

A roads pollution label is defined as the sum of pollution divided by the length of the road. Boundaries will be used to group the datapoints into individual classes. With all the data setup, the only thing left to do is split the data, randomly into three different datasets, training, validation, and test. With the split being 64%, 20% and 16% respectively, the test dataset does not require a larger split as it is used to see what type of predictions the model is making using techniques such as confusion matrices.

4 Design And Implementation

4.1 Overview

The entire implementation was developed in Python using PEP8 coding style standards [40], except for SUMO that was developed in Java. The code throughout the project is designed to be modular, taking advantage of Python's own class system and object orientated design. This allowed straightforward updates and modifications during the testing phase when deciding the best parameters possible.

Finally, I theorize that emission-optimal routing using neural networks will perform better when travelling long distances and staying away from densely populated areas like city centres, but the latter is not a measurable metric due to decisions on where borders lie.

4.2 Initial Simulations & Data Extraction

Firstly, to be able to predict Pollution levels, a traffic dataset with vehicle emissions output is needed. Multiple simulations will be running to artificially generate data. Locations were randomly picked using OpenStreetMaps [38] and exported into .osm format, using SUMOs own tools, each map was converted to .xml.net files and multiple configuration files were generated, including a route file for vehicles- generated by SUMOs own "*randomTrips.py*" file. With each simulation setup, I was able to interface and extract data using my own "*DataExtraction.py*" module. Data was extracted at 20s intervals (20 simulation steps) to capture all possible scenarios, including serious traffic build-up from accidents and traffic jams which had occurred several times. Data extraction this frequent comes with a significant increase in computational time, however it was deemed essential to gather enough information possible.

The chosen extracted traffic information I consider gatherable through crowdsourcing (i.e., average road speed) from the simulation and known data from maps (i.e., road lengths). Accurate vehicle emissions will need to be known for the training phase as I have chosen to carry out my research with a supervised neural network. In real world application, either an unsupervised neural network will need training, or control environments (urban, rural etc) will need to be setup with pollution sensors and traffic (incl. weather) information will need to be gathered across a large timespan, for example a year would be able to capture all seasons and most weather conditions. Due to the availability of emission models that account for weather accurately, it is not considered in this project.

Carbon Monoxide, Nitrogen Oxide, Particular Matter and Hydrocarbons will be extracted per road at each time interval. Further processing will be required to combine all emission levels together and classify the traffic conditions into classes based on the emissions.

4.3 File Processor & Classification Generator Design

The following section begins the design to achieve Objective 2, before a neural network can predict pollution, the dataset needs converting to a multi-label dataset for a supervised learning neural network that predicts labels. Labelling the dataset has advantages over predicting raw emission values, allowing for a variation of emission levels. Before labelling could take place, the raw data needed processing to remove obsolete information where there was no traffic or emission information.

Obsolete data is removed with "*CSVFileProcessor.py*" by comparing the number of vehicles on the road and the pollution information at that given step and defined in the following pseudocode.

$$IF(CO2 = CO = HC = NOx = PM) \wedge (VehicleNumber \geq 0)$$

"*CSVFileProcessor.py*" creates new processed CSV files keeping the original .csv files intact so that the software is robust and able to return to its previous state without having to reprocess every simulation. Implementation is shown in **(Appendix B)**.

A visual comparison between both versions of the dataset is shown in **(Appendix C)**.

The dataset consists of five different pollutants extracted from the simulations, however, only the four harmful to human health pollutants were considered. The resolution of Carbon Monoxide values was considerably higher than HC, NOx, and PM, so those 3 pollutants were scaled to have the similar resolution so that they offer similar impact.

$$TotalPollutants_i = \frac{(CO_i + w_1 * CO_i + w_2 * HC_i + w_3 * NOx_i)}{w_4}$$

Where i represents the current road and time interval combination.

$$W = < w_1, w_2, w_3, w_4, >$$

W represents the set of all weights that are used to scale emissions to have similar impacts.

With the total pollutant for each datapoint, boundaries are defined by using a linear floor function to define the boundary value. Implementation is shown in **(Appendix D)**.

$$PollutionClass_i = Floor\left(\frac{TotalPollutants_i}{350}\right)$$

The chosen boundary between each class (350) was determined with a series of assumptions and testing; firstly, there will be a high number of roads in the dataset that will have very little pollution. We see this trend in the original dataset before processing **(Appendix C)**, most roads in each simulation at different time intervals had no vehicle traffic or pollution levels. So the processed data is more likely to align itself with the first couple of pollution labels. To combat this, a relatively low boundary value of 350 was chosen, which accepted a higher number of datapoints for the first couple of pollution labels but provided an equal enough spread across the later pollution labels. Secondly, I used 12 pollution labels as I found they were enough to provide pollution information. This determined whether a road had acceptable pollution levels or not. However, using 12 pollution labels has the undesired effect of the last pollution label containing more data than it should. To combat this, increasing the number of pollution labels was also considered, however this would lead to more severe pollution labels being harder to predict and the neural network would lose accuracy due to not having enough data to train with per pollution label. So instead, I opted to address this problem in section 4.5.2 with cost function manipulation.

4.4 **Neural Network Design**

To achieve Objective 1, an effective neural network that can predict with high accuracy is required, the following chapter justifies the design of the neural network and its implementation.

4.4.1 Dataset

An analysis of the input dataset **(Appendix E)** shows strong positive correlation between the number of nearby roads to the average speed of the current road. The more nearby roads, the more options for alternative routes, so traffic can be distributed between roads raising the average speed for all nearby roads including the current one. Mean_Speed has much more informational value than Total_Neighbours, with a larger range of values being of type float(). Whereas Total_Neighbours is integer based and restricted to the number of roads nearby. So if I had to eliminate a predictor variable, it would be Total_Neighbours, however- due to having a low number of predictor variables already, I have opted not to eliminate either variable as neither of them correlate with any other predictor.

Previously researched pollution neural networks have been regression based and shown good results in predicting true emission scores / values, I have opted to create a classification based neural network, due to some important factors regarding my task. Firstly, I am aiming to predict pollution levels on a microscopic scale, meaning I am aiming to classify vehicle emissions per road instead of pollution levels covering a wide area. As I need to predict pollution levels on such a small scale, I need to account for variation and drastic behaviour changes. By allowing a set variation in the pollution label boundaries, the cost of the road will not drastically change unless on the boundary of two pollution labels, therefore a small change or inaccuracy will not be able to significantly affect the predicted emission-optimal route of the vehicle.

The output of the neural network was decided through testing. I found that too small of a resolution (I.e., 4 possible outputs) would not be able to significantly impact emission-optimal routing in a positive way against standard vehicle routing practices, often finding it caused vehicle occupants to be exposed to more pollution, therefore, an analysis of the max-min emission values in the dataset led me to choosing 12 possible emission output labels. **(Appendix F)** shows a comparison between a network that used 4 pollution labels and another network using 12 pollution levels being implemented and tested in a simulation. We can see that the network with 4 pollution labels perform worse against its standard routing counterparts and the simulation with 12 pollution levels showing improvements against its counterparts.

4.4.2 Architecture

A Feed-Forward Neural Network is well suited to this problem and hence the chosen architecture. Using Keras [41] "*Sequential()*" function, I was able to implement the architecture. The following section follows the breakdown of parameters and choices.

There were two main contenders for the position of chosen neuron activation function, Relu and Sigmoid. Both activation functions were bounded between 0 and 1, therefore, always resulting in a positive output. This is important as we do not want to predict negative emissions, by using a bounded activation function, it prevents the model from doing this.

The deciding factor between both possible functions was the design of the pollution label generator. In section 4.3, the processed data was sorted into pollution labels using a Floor Function, having equal boundaries between each class until the final labels, therefore 11/12 emission labels follow a linear pattern, similar to the shape of a plotted Relu function, except for the last class which contains roads with significantly larger pollution values than the rest of the classes. Therefore Relu was the chosen activation function.

The final model consisted of 4 high level layers, with the input layer consisting of 64 neurons and a Relu activation function, this was followed by a "*BatchNormalization()*" layer, known to speed up training times by using a scaling technique for each batch of input data. Then a "*Dropout()*" layer to prevent any form of overfitting, which may be more likely than normal due to keeping both Total_Neighbours & Mean_Speed despite their strong positive correlation. The remainder of the model consisted of two more hidden layers with Relu activations functions, and the output layer used a SoftMax activation function.

Using the "*Dropout()*" technique usually results in the validation data having a smaller loss and higher accuracy than the training dataset, this would typically indicate a poor model, however this is a known occurrence when using "Dropout()" and the Test Dataset was used to verify the model's accuracy out of 5000 predictions.

Keras [41] "$SparseCategoricalCrossentropy()$" was the chosen loss function, capable of multi-label classification and generating probabilities for each pollution label. The label with the highest probability was the predicted emission output label. "$SpraseCategoricalCrossentropy()$" requires the output layer to have a Softmax Activation Function to normalize the probabilities for each emission output class.

Optimisers are used to change the weights in a neural network with the goal of minimizing losses. The Adam Optimiser [41] is a standard choice of optimiser and applicable to most situations. The main parameters of the Adam Optimiser consist of Learning Rate, Momentum and Decay rate.

The learning rate determines how the neurons weights are updated across the model, with a larger value indicating a larger change in weight value, and the smaller the value, the less severe the change in weight value. As the model converges to a minimum, the learning rate should decrease to allow the model to converge, otherwise the model will not perform as well as it could. I started with a learning rate of 0.001 and a decay rate of 0.95 per step to allow the model to converge.

Full Implementation is shown in **(Appendix G)**

4.5 Simulation Testing Design

Due to the computational time to process each simulation, it is unreasonable to make the user continuously adjust scripts and run each simulation manually, to optimise efficiency, each simulation needs to be automated.

For each simulation scenario, multiple random routes will be generated for a single vehicle to travel. For each random route generated, the simulation will run 4 times using 4 different routing methods. Once each route has run, the simulation will move onto the next route or scenario and repeat the process. This creates a fully automated testing system, however, may take days to run and complete.

Both static and dynamic routing has been implemented, however only the design for dynamic routing is shown as static routing is implemented by default in SUMO.

The following subsections outlines the design that was implemented to tackle Objective 3. Two edge cost functions were explored with both static and dynamic routing.

4.5.1 Dynamic Routing Design

Dynamic routing has been implemented through time intervals rather than an event-based system, crashes do happen within the simulation. Yet, they do not typically happen frequently enough for Dynamic Routing to update often enough. By using set time intervals, Dynamic Routing can respond to traffic build-up and pollution zones frequently.

When using Dynamic Routing, computation time is significantly increased the further the source and destination roads are apart with static update intervals. Dynamic update intervals were considered and implemented, but no optimum combination of update time intervals was found. During controlled testing, each update interval produced the exact same results finding no optimum, yet when implemented fully, would cause Dynamic Dijkstra to drastically increase pollution exposure. I concluded that a larger test size would be required to find an optimum- however lacked the time to test combinations of parameters and interpolate between the chosen optimum time intervals at each distance marker. The full design is shown in **(Appendix H)**

4.5.2 Cost Function Design

By default, SUMO uses Estimated Travel Time to calculate routes for each vehicle, this is a shortest-predicted time cost function that has no consideration for vehicle emissions. To account for emissions, an emissions exposure cost function needs to be defined. Formally, we can define emission exposure as:

$$EXP(x)_i = EST(x)_i * POL(x)_i$$

Where EXP_i is a unitless representation of Pollution Exposure, EST_i is the estimated time (s) to travel road i, and POL_i is the total weighted emissions on road i that is considered evenly spread across road i with no dispersion. As we have classified pollution with labels, we do not know the real value weighted total pollution per road, however a cost system based on label can be implemented instead. Hence satisfying Objective 2.

$$RoadCost(x) = \left(Cost[PollutionClass] * EstimatedTravelTime(s)\right)$$

Two costs were implemented, with each cost being based on how the original pollution labels were defined in section 4.3. Cost A strictly follows how pollution labels were defined by following a linear increase in cost until reaching the end label where it increases the cost drastically. Whereas Cost B loosely follows the definition for pollution labels, knowing that label 12 will have severe amounts of pollution exposure, it drastically increases the cost for that label. The costs for labels 10 and 11 are also drastically increased as just increasing the cost for label 12 will not have significant impact on routing by itself.

Each cost function was performed on a different set of source and destination roads, so they are only comparable to their counterparts that used the same source and destination roads.

Cost A												
Labels	1	2	3	4	5	6	7	8	9	10	11	12
Cost	1	1.3	1.6	1.9	2.2	2.5	2.8	3.1	3.4	3.7	4.0	10

Table 2 - Cost A Proposed Weights

An analysis of Cost A **(Appendix I)** shows that Emission-Optimal Static Dijkstra (Stat_Dij_Global) reduces maximum pollution exposure, while maintaining an equal minimum pollution exposure to Shortest-Time-First Static Dijkstra (Dij). However, the Median Quartile is slightly larger than before. The average exposure for both Static and Dynamic Emission-Optimal routing is lower than Shortest-Time First routing. With static emission-optimal routing performing the best out of all 4 routing methods.

Cost B												
Labels	1	2	3	4	5	6	7	8	9	10	11	12
Cost	1	1	1	1	1	1	1	1	1	100	100	100

Table 3 - Cost B Proposed Weights

However, for Cost B, Emission-Optimal Routing heavily outperformed its Shortest-Time-First counterparts **(Appendix J)**. Emission-Optimal Static routing (Stat_Dij_Global) has a smaller upper and median quartile than its counterpart, Shortest-Time-First routing (Dij) as well as a much lower maximum exposure. Dij has an outlier (maximum pollution exposure) that is approximately ~3 times larger than the maximum exposure of Stat_Dij_Global. Cost B has shown its effectiveness at reducing maximum pollution exposure when dealing with longer / more congested routes.

For both costs, Emission-Optimal Static routing has outperformed Shortest-Time-First Static routing, the same goes for Emission-Optimal Dynamic routing outperforming Shortest-Time-First Static routing. However, Dynamic routing is theoretically supposed to perform better than Static routing but has not. I theorize that the optimal Dynamic Interval parameters have not been selected, and given the optimal parameters, Dynamic routing should out-perform Static routing.

Ethically, dynamic routing should not be considered until the optimal update interval parameters are found, as we are knowingly increasing pollution exposure despite having better methods that can reduce pollution exposure more effectively. However, we can still compare Dynamic Routing's performance with both Shortest-Time-Cost and Emission-Cost.

4.6 Hurdles

One of the important metrics regarding software efficiency was computational time. With the goal of reducing computational time and automate user interaction, a few optimisations were made.

4.6.1 Simulation Optimisation

Computational time was the main problem that was considered throughout the project. Simulations can often be very demanding, especially with poor CPU utilisation. Knowing that simulation times were poor, I had researched into external simulation modules to help improve performance, coming across sumo-cuda [42] being the main external module that would aid simulation computing time. However, the module was outdated and built for an older version of SUMO. Therefore, non-traditional methods of speeding simulations were considered, due to being bounded to a single processing thread.

Firstly, SUMO had to be run without a GUI, however this showed no clear performance increase as the GUI is ran on a second thread, so having no significant effect on simulation performance. Smaller simulations were used than originally planned and the time between data being extracted from the simulation was marginally increased to allow higher variation between the data & less pausing of the simulation to extract data.

Secondly, the option to use SUMOs inbuilt logging system was considered, however, this was only effective for initial data extraction and with the amount of time available at the time of writing this script coupled with the lack of urgency for the data, I deemed it not important. Whereas, when it came to testing routing methods, all optimisations possible were taken, firstly the intervals between each routing calculation for dynamic vehicle routing was increased to 30steps. The script was designed to be fully automated till completion, many issues were encountered with SUMOs own simulation software stepping out of sync with the python interface library, so error catching was implemented throughout in case SUMO and TraCI did not communicate that the vehicle had reached its destination.

SUMO also had a lack of support for determining if two edges were able to connect to each other through a combination of other edges. SUMO would require the simulation to start and edges to be loaded before it would determine if they were valid, therefore a brute force technique was implemented to pass edges and catch the errors until two edges were deemed connected & valid. This may not decrease computational time, but it does decrease interaction time between the user and software, therefore decreasing overall time taken.

Full implementation can be found in "*TestSimulation.py*"

4.6.2 Neural Network

Continuing with reducing computational time, Tensorflow [42] offers two packages, CPU based processing and GPU based processing. Unfortunately, due to not having access to CUDA 3.5 compatible Graphics Cards, I was not able to take advantage of the increased processing speed that would come with Tensorflow-GPU.

Despite that, I attempted to create the optimum neural network given the dataset provided, a few techniques were implemented to speed up training time. Firstly, a "*BatchNormalization()*" layer was implemented, known to improve training time by normalizing each input batch. Secondly, to reduce time spent training the model when it is no longer improving, early stopping was implemented coupled with call-backs to stop the model when it stops seeing improvements and save the best model.

5 Results Analysis & Evaluation

In Chapter 2.3, I defined a set of performance metrics to evaluate the effectiveness of the proposed approach. Firstly, a broad analysis is conducted to determine whether the proposed solutions have achieved the original goal. Secondly, I will explore the quality of the proposed routing methods.

Pollution Exposure will not consider Carbon Dioxide as it is not directly harmful to human-health. Therefore, the sum of the remaining gasses (Carbon Monoxide, Hydrocarbons, Nitrogen Oxides and Particulate Matter) will determine the value for pollution exposure.

5.1 Pollution Exposure

The main objective of this report was to reduce pollution exposure for vehicle occupants through emission-optimal routing algorithms. Two Emission-Optimal routing variants were implemented, as well as, two Shortest-Time-First (STF) routing variants.

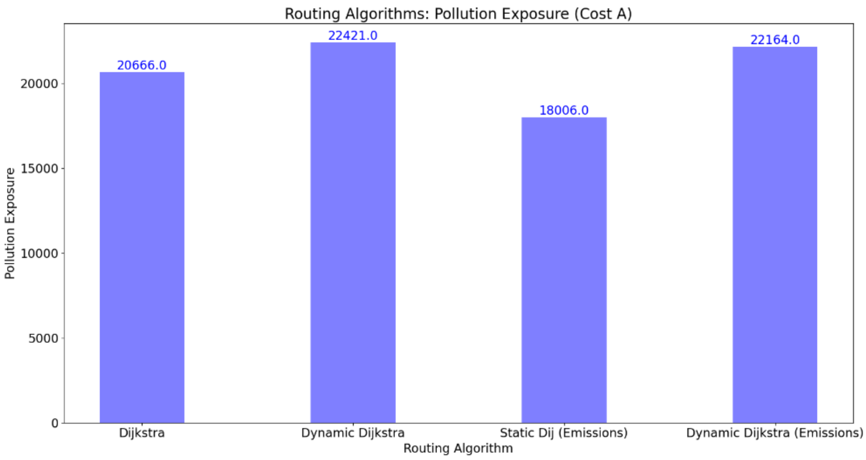


Figure 2 - Pollution Exposure Analysis for Cost A Emission-Optimal Routing Against STF Routing

Figure 2 shows a plot of average pollution exposure per routing algorithm with Cost A (defined in Chapter 4.5.2). Dijkstra and Dynamic Dijkstra are the two baseline pollution exposures calculated from Shortest-Time-First vehicle routing. The Emission-Optimal variants indicated by "(Emissions)" show reduced average pollution exposure, which successfully achieves the original goal of reducing vehicle occupant pollution exposure.

Dynamic Dijkstra consistently underperforms, increasing pollution exposure over its Static Dijkstra counterparts. I theorize that this is due to a particularly small, time interval. When update intervals are too close together, the route will be recalculated too frequently due to the algorithm falsely concluding a better route has been discovered. This would result in the driver increasing the number of edges travelled, and thus also increase the time spent driving, resulting in an increase in pollution exposure.

Two possible improvements could be made based on this theory. First, the time interval could be increased to update less frequently so the route would not excessively recalculate. Alternatively, instead of implementing Dynamic vehicle routing on a time-based system, an event-based system could be used so that routes are only altered in event of traffic build-up or similar obstructions. However, this solution would not be effective on shorter routes as it is unlikely to update frequently enough to alter the suggested driving route.

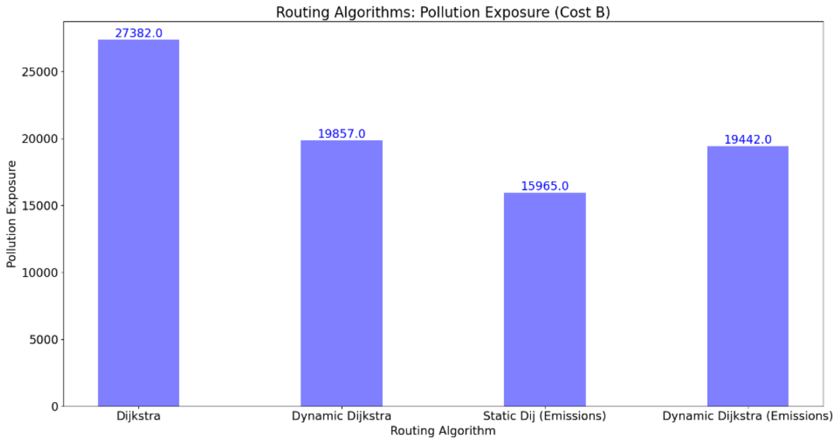


Figure 3 - Pollution Exposure Analysis for Cost B Emission-Optimal Routing Against STF Routing

Figure 3 shows a second average pollution exposure bar chart, except Cost B was the chosen instead of Cost A. The first thing to note is that Dijkstra has a significantly larger pollution exposure average when compared to the alternative routing algorithms. This significant exposure increase indicates that either an accident or a traffic jam occurred during one of the chosen routes for the simulation. Dynamic Dijkstra was able to account for this issue and reroute the test vehicle, however, neither variant of Dynamic Dijkstra achieved an emission-optimal route. Once again, Static Dijkstra with Emission-Optimal routing significantly reduced pollution exposure.

The Dynamic Dijkstra results show that even with a different cost, the route is likely updating too frequently, especially when travelling short distances. The Dynamic Dijkstra update interval is not optimal, and an alternative solution should be found.

5.2 Route Duration

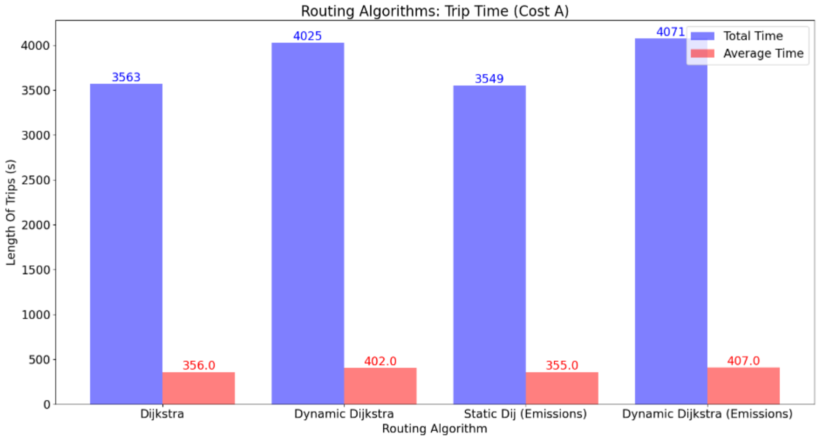


Figure 4 - Total / Average Trip Duration (s) charts for each routing method

Figure 4 shows the total and average route durations for each route per routing algorithm. As predicted, Dynamic Dijkstra variants spend more time travelling than static variants, once again likely due to the short time intervals. However, both Static Dijkstra variants show incredibly similar travel times, with the emission-optimal variant performing slightly better than the shortest-time-first variant. This is an indication that Emission-Optimal routing is producing good quality routes based on the performance metrics defined in chapter 2.3.

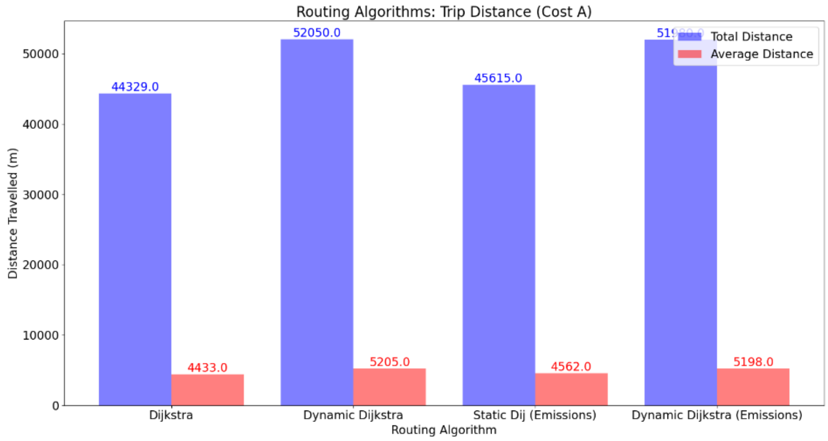


Figure 5 - Total / Average Trip Distance (m) charts for each routing method

Figure 5 shows the total distance travelled by each routing algorithm and the average distance travelled per simulation. Once again, we notice sub-optimal performance from both variants of Dynamic Dijkstra.

Static Dijkstra with Emission-Optimal routing resulted in an increased journey distance of 129 metres, which is negligible when compared to the average distance travelled per simulation. Despite the longer average distance, Figure 4 shows approximately the same time spent driving, indicating that Emissions-Optimal routing has redirected the vehicle to roads that have less traffic, higher vehicle speeds and better flow. Another indication that Emission-Optimal routing has produced good quality routes.

5.3 Route Difficulty

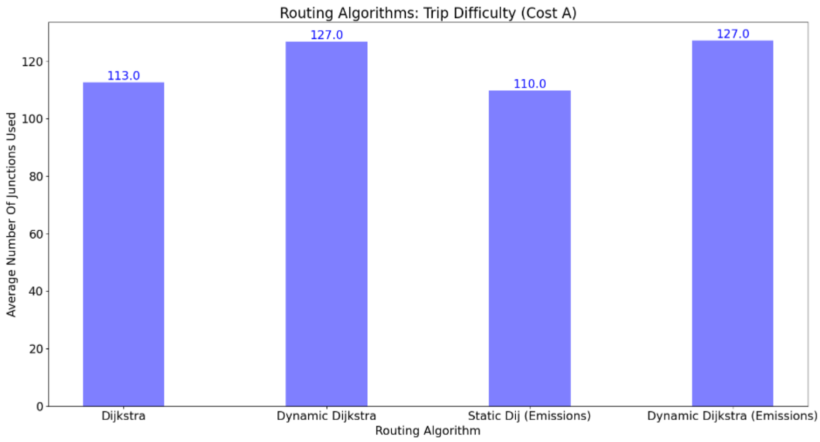


Figure 6 - Average Trip Difficulty for each routing method

Finally, the Trip Difficulty is defined by the number of junctions crossed; albeit this does not represent individual junction difficulty. However, this is a good overall indication of route difficulty. This confirms that Static Dijkstra with Emission Optimal routing is, on average, redirecting road users to longer roads with less pollution, which is an effective method to reduce pollution exposure while maintaining route quality.

For the Dynamic Dijkstra variations, the number of edges travelled has increased without needing to. This is a flaw with using time-based update intervals instead of event-based. However, event-based routing would of likely not have performed effectively within these simulations as they were kept to a minimally appropriate size to reduce processing time. The simulations used for these tests had a processing time of approximately 15hours per cost function. However, these trends hold true in larger scale simulations **(Appendix K** visually shows the difference between small-scale and large-scale simulations**)**. The results for a single larger simulation (using Cost A for Emission-Optimal Routing) **(Appendix L)** show that the proposed solution can scale to different problem sizes despite being trained on small simulations. However, it took approximately 20 hours to process this simulation, with only one defined route, making it unfeasible to run all simulations on a larger scale and remain within the defined constraints. These larger scale results follow similar trends to the smaller scale simulations. Static Dijkstra with Emission-Optimal routing once again shows significantly less pollution exposure than all other alternative routing methods.

5.4 Discussion

5.4.1 Ethics

Ethical concerns need to be considered as we are experimenting with the dispersion of emissions. We know that the proposed solution will likely raise pollution levels on roads that typically have low pollution, which can impact human health. The main group it will impact is pedestrians walking alongside that road. Conversely, pedestrians walking alongside a highly polluted road may experience reductions in pollution exposure if vehicles used emission-optimal routing. We can infer from the results, that emission-optimal routing naturally wants vehicles to travel on higher speed roads with less vehicle traffic, which statistically will also have less pedestrian traffic, and would potentially offset the negative impact of the dispersed pollution. I hypothesize that the implementation of this method for all vehicles on the road will distribute pollution almost evenly across roads that are being travelled on, reducing the maximum pollution exposure for everyone, but perhaps increasing the average pollution exposure. However due to additional complexities and processing time to test this hypothesis, I am unable to confirm or deny at this stage.

Maximum exposure has been reduced for the test vehicle; however further testing would be required to confirm that emission exposure is reduced under all circumstances. This would require further cost optimisation & dynamic interval optimisation. There exists a set of costs that will provide the optimal emission-exposure reduction, however finding these optimal emission costs is where the problem lies. An algorithm would need to be developed that would test combinations of costs and interpolate costs. I suggest an analysis of the original dataset is required to look at where data is clustered within each label boundary, where the upper bound of the clustered data can be used as the cost for each label. For example pollution between the range 350 and 700 (label 2) may have clustered data, where most datapoints are grouped together and the largest value of the grouping is 630, making Label 2’s cost 1.630. Unfortunately, this level of testing falls outside of the scope and time constraints of this project.

5.4.2 Computational Expenses

The real world implementation requires the use of a centralised server with traffic data collected live from crowdsourcing and known weather conditions. There are two different ways updating road costs could be performed. First of which would be to take all crowdsourcing information and predict pollution for each road on a centralised server then redistribute local road information to individual users. However, this will be very demanding on the server and may result in delayed updates.

The second option would be to make user devices predict pollution for the road they are currently driving on and report the predicted pollution back to the server to redistribute for route updates. While this method is heavily dependent on having many active users, it could perform effectively with sufficient user density (e.g. built-up urban areas).

5.4.3 Acknowledgements

Finally, due to the variety of factors influencing air pollution, a single traffic model cannot account for all emissions on the road. Rudimentary solutions can be developed to better simulate air pollution behaviour, such as defining that roads in a specified proximity with each other can share 10% of their vehicle emission values. However this is not an accurate representation of real-life, and a better solution would be to have both a traffic and weather emissions model. By doing this, the data would behave in a non-linear fashion, emulating true air pollution. For this very reason, neural networks have been chosen over standard machine learning techniques, such as linear regression.

5.5 Results

	Average Exposure	Maximum Exposure	Average Trip Duration (s)	Average Trip Distance (m)
Dijkstra	29318	115847	429	5224
Dynamic Dijkstra	31100	117894	472	5959
Static Dijkstra (Emissions-Optimal, Cost A)	25649	102081	422	5298
Dynamic Dijkstra (Emissions-Optimal, Cost A)	30866	117894	476	5952

Table 4 - Result Breakdown for Cost A Emission-Optimal Routes

This analysis of Table 4 shows the Static Dijkstra with Emission-Optimal routing has shown improvements in every main performance metric except the Average Trip Distance (m). However, Average Trip Distance is expected to increase due to routing through predicted emissions. Despite the increase in Average Trip Distance, the quality of the route was better than alternative routing methods, with less time spent stopped in traffic or at junctions and more time spent driving.

Dynamic Dijkstra with Emission-Optimal routing performed equally well when compared to the standard Dynamic Dijkstra variant. However, Dynamic Dijkstra based routing did not perform as well as Dijkstra for the many reasons previously discussed (Chapter 5), and therefore the performance can only be compared with itself.

	Average Exposure	Maximum Exposure	Average Trip Duration (s)	Average Trip Distance (m)
Dijkstra	27382	133903	374	3660
Dynamic Dijkstra	19857	49869	337	3867
Static Dijkstra (Emissions-Optimal, Cost B)	15965	36162	311	3696
Dynamic Dijkstra (Emissions-Optimal, Cost B)	19442	43592	349	3990

Table 5 - Result Breakdown for Cost B Emission-Optimal Routes

Table 5 shows similar trends with Table 4, more noticeably, Static Dijkstra with Emission-Optimal routing shows a massive improvement in terms of pollution exposure over all other routing methods.

6 Conclusion

Full implementation of an emission-optimal based routing method has shown impressive initial results in both reducing maximum pollution exposure and average pollution exposure, without sacrificing route quality. In fact, some cases have shown route quality has improved significantly, with less time spent at junctions and more time spent driving at higher average speeds than vehicle routing using shortest-time-first routes. Emission-Optimal with Cost A routing has shown a 12.5% reduction of average emission exposure against its counterpart (Table 4) while maintaining similar route durations. An alternative proposed cost for Emission-Optimal (Cost B) routing has shown an impressive 41.7% reduction of average pollution exposure (Table 5). However this routing method is more volatile, with only the severe pollution exposure labels having weights that exclude them from the network entirely, short of there being no other option to reach the destination. This method of routing vehicles is considered volatile as it can suffer from the same problem of trying to route a vehicle by shortest-time-first and end up being exposed to high levels of pollution in return.

In conclusion, the primary objective has been achieved with extremely promising results that could alter the way pollution is managed in urban environments and beyond. As an initial study, there are many factors that should continue to be explored, not the least of which are the ethical concerns discussed throughout the project, as well as the future development required to improve the algorithms into a production ready project. The idea of evenly dispersing pollution through controlling traffic has both positives and negatives. The original objective was to reduce pollution exposure for vehicle occupants due to the higher concentrations of pollution experienced. I subsequently developed a theory regarding pollution exposure for non-vehicle occupants, stating that if a non-vehicle occupant were to be travelling through a high pollution zone, their own pollution exposure would be significantly reduced if vehicles used emission-optimal routing as vehicles would strictly aim to avoid contribution to the zones of high pollution. You can infer from the results that emission-optimal vehicle routing favours longer roads with higher average speeds, indicating that if enough vehicles were using emission-optimal routing, most vehicles will favour roads with better flowing traffic, theoretically reducing the amount of pollution being contributed by vehicles to high pollution zones in urban areas.

Further Work

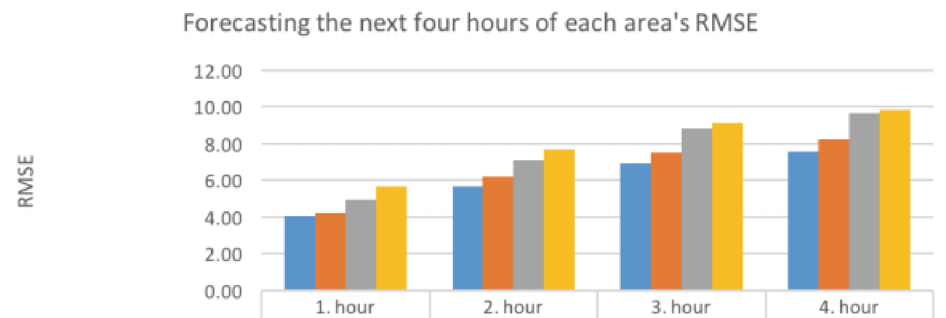
As stated previously, further optimisations need to be made to ensure the lowest pollution-exposure route is always selected without sacrificing route quality. To achieve this, the vehicle emissions model needs to represent real world air pollution and dispersion behaviours, which it currently does not. With a more advanced vehicle emissions model, coupled with the implementation of a weather model, road pollution levels would then be able to exhibit non-linear behaviour.

Routing based on individual road pollution levels may be challenging to implement in the real world, alternative models should be considered to account for this, such as using pollution zones which contain a set of roads sharing similar pollution levels, with slight variation based on traffic and weather information. I believe this method would better represent the behaviour of emission dispersion.

And finally, dynamic vehicle routing would need to be addressed, as clear optimisation problems with the update time interval have been discovered. I suggest using a trigger-based system, however this would only be effective when running large scale simulations. Currently this is not possible due to the simulation software having poor CPU utilisation resulting in long processing times. Unless future updates address this issue, alternative simulation software would need to be considered.

Appendix A

LSTM Pollution Forecasting Results [23].



	1. hour	2. hour	3. hour	4. hour
Northern Taiwan	4.03	5.66	6.89	7.56
Middle Taiwan	4.23	6.20	7.51	8.23
Southern central Taiwan	4.96	7.08	8.82	9.67
Southern Taiwan	5.67	7.67	9.11	9.85

Appendix B

CSVFileProcessor.py, removes obsolete data.

```
# Sets Up Directories
current_dir = Path(os.path.dirname(__file__))
if sys.platform == "linux" or sys.platform == "linux2":
    data_dir = os.path.join(current_dir.parent, 'CSV/')
    new_dir = os.path.join(current_dir.parent, 'PROCESSED_CSV/')
elif sys.platform == "win32":
    data_dir =  os.path.join(current_dir.parent, 'CSV\\')
    new_dir = os.path.join(current_dir.parent, 'PROCESSED_CSV\\')

# Pre-Processed Datafiles
data_files = os.listdir(data_dir)

for fname in data_files:
    processed_fname = (new_dir + str(fname.replace(".csv", "_processed.csv")))
    with open(processed_fname, 'w', newline='') as processed_file:
        writer = csv.writer(processed_file)
        with open(os.path.join(data_dir, file_name), 'r', newline='') as file:
            reader = csv.reader(file)
            data = list(reader)
            for row in data:
                # Row (1-5): Emissions, Row (8): No. Of Vehicles
                if (row[1] == row[2] == row[3] == row[4] == row[5] and
                    int(row[8]) >= 0):
                    pass
                else:
                    writer.writerow(row)
        file.close()
    processed_file.close()
```

Appendix C

A visual comparison between a dataset before and after processing (removing the obsolete data).

Edge	CO2_Emis	CO_Emiss	HC_Emissi	NOx_Emis	PMx_Emis	Mean_Spe	Estimated	Traffic_Le	Total_Nei	Length	Step_Count
-10473123	0	0	0	0	0	22.22	7.074707	0	10	157.2	20
-10473123	0	0	0	0	0	22.22	0.461296	0	12	10.25	20
-10473123	0	0	0	0	0	22.22	0.646265	0	12	14.36	20
-10473123	0	0	0	0	0	22.22	0.009001	0	12	0.2	20
-10473123	0	0	0	0	0	21.6959	3.055877	1	12	66.3	20
-10473123	0	0	0	0	0	22.22	4.206121	0	12	93.46	20
-10473123	0	0	0	0	0	22.22	0.009001	0	12	0.2	20
-10473123	0	0	0	0	0	22.22	1.570207	0	12	34.89	20
-10473212	0	0	0	0	0	13.89	53.97984	0	11	749.78	20
-10473212	0	0	0	0	0	13.89	7.38085	0	11	102.52	20
-10473212	0	0	0	0	0	13.89	8.233261	0	12	114.36	20
-10473212	0	0	0	0	0	13.89	0.014399	0	12	0.2	20
-10473212	0	0	0	0	0	13.89	16.62923	0	12	230.98	20
-10473212	0	0	0	0	0	13.89	29.2527	0	13	406.32	20
-10473212	0	0	0	0	0	13.89	35.57379	0	12	494.12	20
-10473212	0	0	0	0	0	13.89	2.974082	0	12	41.31	20
-10473212	0	0	0	0	0	13.89	11.10223	0	12	154.21	20
-10473212	0	0	0	0	0	13.89	18.25918	0	12	253.62	20
-10473212	0	0	0	0	0	13.89	0.064075	0	12	0.89	20
-10473212	0	0	0	0	0	13.89	0.014399	0	12	0.2	20
-1E+08	0	0	0	0	0	13.89	26.22534	0	10	364.27	20
-10473398	0	0	0	0	0	5.56	74.94964	0	11	416.72	20
-10473398	0	0	0	0	0	5.56	108.1978	0	13	601.58	20
-10473398	0	0	0	0	0	13.89	10.14687	0	10	140.94	20
-10473398	0	0	0	0	0	13.89	4.349172	0	12	60.41	20
-10473398	0	0	0	0	0	13.89	6.815695	0	11	94.67	20
-10473398	0	0	0	0	0	13.89	3.809215	0	11	52.91	20
-10473398	0	0	0	0	0	13.89	0.406048	0	10	5.64	20
-1E+08	0	0	0	0	0	13.89	5.826494	0	9	80.93	20
-10473398	0	0	0	0	0	13.89	7.62995	0	10	105.98	20
-10473398	0	0	0	0	0	13.89	0.019438	0	12	0.27	20
-10473398	0	0	0	0	0	13.89	3.102952	0	12	43.1	20
-1E+08	0	0	0	0	0	13.41	13.12901	0	12	176.06	20
-1E+08	0	0	0	0	0	13.89	3.276458	0	8	45.51	20
-1E+08	0	0	0	0	0	13.89	2.424046	0	8	33.67	20
-10473442	0	0	0	0	0	13.89	0.7509	0	8	10.43	20
-10473442	0	0	0	0	0	13.89	6.692585	0	12	92.96	20

Edge	CO2_Emis	CO_Emiss	HC_Emissi	NOx_Emis	PMx_Emis	Mean_Spe	Estimated	Traffic_Le	Total_Nei	Length	Step_Count
-27947212	2248.41	11.00924	0.098945	0.72744	0.013295	11.25584	33.41556	1	12	376.12	20
-36943638	3545.312	0	0.070666	1.113309	0.042867	20.49655	21.38946	1	14	438.41	20
-37948403	3501.955	41.40749	0.268499	1.332716	0.047136	10.71556	4.906882	1	10	52.58	20
-39107459	10148.21	139.8408	0.883423	4.311801	0.213601	17.02719	2.618752	1	10	44.59	20
-49000300	4013.732	135.8117	0.712578	1.762437	0.084235	3.555406	34.12269	1	10	121.32	20
104734422	6282.304	144.4592	0.807357	2.757678	0.132143	8.040162	10.59556	2	12	85.19	20
37948285#	4378.158	33.99846	0.257049	1.657864	0.064486	14.22281	9.886936	1	12	140.62	20
37948285#	5161.878	113.2957	0.637444	2.211298	0.100574	7.038935	1.703383	1	10	11.99	20
391074592	3937.346	150.7585	0.779262	1.755975	0.08732	2.465745	18.13245	1	10	44.71	20
4021002#5	7828.269	138.59	0.818562	3.397026	0.162133	9.823633	5.741257	1	12	56.4	20
49000030#	2624.722	164.7778	0.811944	1.204444	0.065972	0	232740	1	12	232.74	20
63664885#	6207.316	73.05387	0.480297	2.526872	0.11249	13.98573	8.728895	1	10	122.08	20
-1.2E+08	2220.014	8.732304	0.087886	0.708787	0.012153	11.46263	52.81771	1	6	605.43	40
-15801485	6519.472	58.17706	0.42334	2.515632	0.120566	20.54275	19.80358	1	12	406.82	40
-15801604	5307.911	34.13115	0.28538	1.892818	0.092047	22.73983	5.148235	1	12	117.07	40
-18651172	5765.802	43.40315	0.338882	2.111685	0.103788	22.61822	15.38273	1	14	347.93	40
-19374049	5166.53	135.065	0.737156	2.259358	0.106979	5.307448	81.12562	1	12	430.57	40
-19374049	12882.89	189.4825	1.179135	5.528378	0.286281	20.11534	56.43056	1	7	1135.12	40
-26704742	7197.073	139.5798	0.807485	3.135196	0.149364	8.431848	16.74366	1	9	141.18	40
-27947212	3977.581	44.73166	0.2955	1.537541	0.057996	11.48561	32.74706	1	12	376.12	40
-31671458	9044.569	156.5744	0.930684	3.951164	0.192104	10.59538	3.83658	1	12	40.65	40
-31671458	15162.23	242.7165	1.482326	6.487303	0.354611	25.17317	7.095649	1	10	178.62	40
-36937984	5272.013	103.4407	0.595116	2.23528	0.10006	8.107239	10.11565	1	12	82.01	40
-36943638	2624.722	164.7778	0.811944	1.204444	0.065972	0	4820	1	14	4.82	40
-36943638	6701.966	64.95129	0.45742	2.652643	0.123811	18.3968	0.738172	1	12	13.58	40
-49000030	3776.498	38.32867	0.261371	1.435448	0.05222	11.76994	19.77324	1	12	232.73	40
-57171022	10118.02	160.4293	0.975148	4.398832	0.215929	12.9194	9.477218	1	11	122.44	40
111771216	10781.84	158.3452	0.982888	4.643922	0.230566	15.66756	7.368091	1	11	115.44	40
193740491	5286.524	145.107	0.78606	2.330997	0.112361	4.745574	94.66717	1	11	449.25	40
36943849#	12864.41	197.9625	1.215344	5.607944	0.284627	16.43195	4.813185	1	12	79.09	40
37948285#	2644.622	0	0.048072	0.836635	0.019099	13.05149	14.11486	2	10	184.22	40
37948285#	2707.293	112.2029	0.5727	1.151984	0.051131	3.33318	4.839223	1	11	16.13	40
4021002#0	6375.021	90.43356	0.563242	2.657062	0.119745	11.94948	12.57461	1	11	150.26	40
49000030#	2615.383	0	0.014388	0.79055	0.017357	15.66305	10.99786	1	12	172.26	40
49000300#	4947.713	123.0633	0.676889	2.140761	0.099065	5.884119	35.19473	1	14	207.09	40
571710224	3734.878	149.3594	0.767951	1.665252	0.082893	2.289232	5.241933	1	12	12	40

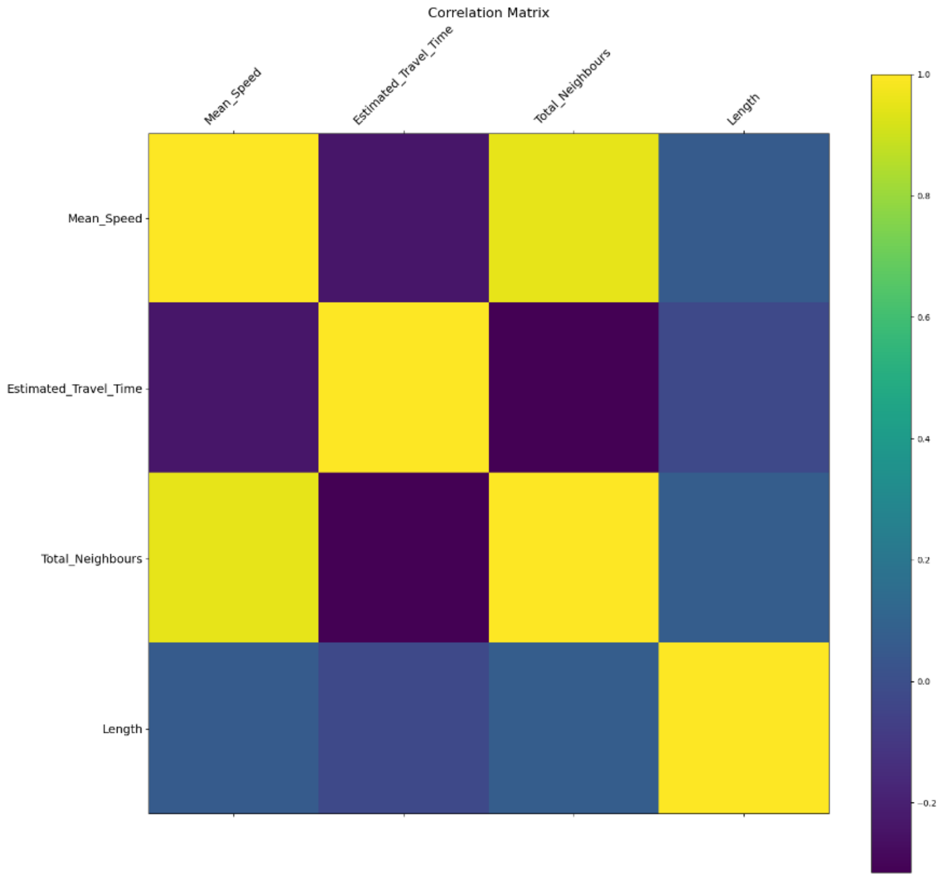
Appendix D

ClassificationGenerator.py, labels pollution data.

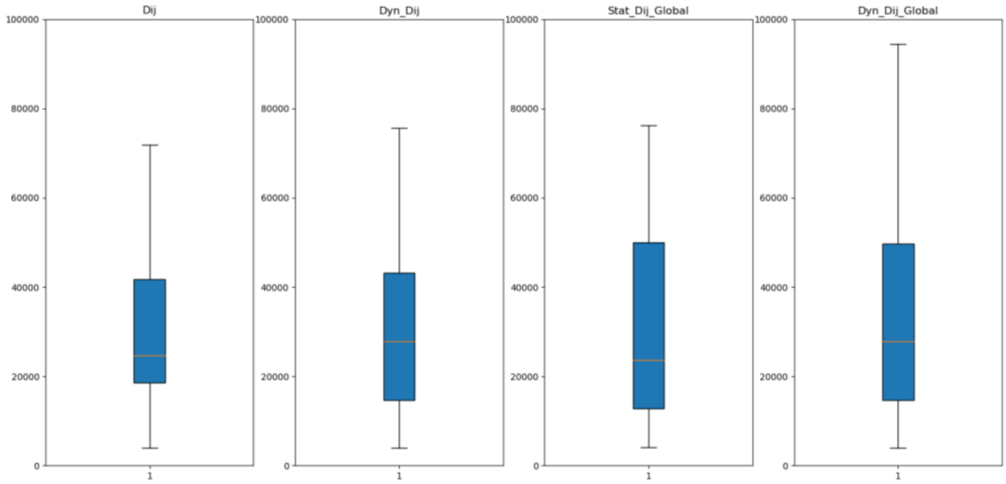
```
# Setup Directories
current_dir = Path(os.path.dirname(__file__))
if sys.platform == "linux" or sys.platform == "linux2":
    processed_dir = os.path.join(current_dir.parent, 'PROCESSED_CSV/')
    ranked_dir = os.path.join(current_dir.parent, 'RANKED_CSV/')
elif sys.platform == "win32":
    processed_dir = os.path.join(current_dir.parent, 'PROCESSED_CSV\\')
    ranked_dir = os.path.join(current_dir.parent, 'RANKED_CSV\\')

ranks = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12] # Possible Pollution Ranks
for fname in (os.listdir(processed_dir)):
    ranked_file_name = os.path.join(ranked_dir,
                                    (str(fname.replace("_processed.csv",
                                     "_ranked.csv"))))
    # New Ranked File
    with open(ranked_file_name, 'w', newline='') as ranked_file:
        writer = csv.writer(ranked_file)
        # Processed File
        with open(os.path.join(processed_dir, fname), 'r', newline='') as file:
            reader = csv.reader(file)
            data = list(reader)
            step = 0
            for row in data:
                if step == 0:
                    step += 1
                    new_row = ["Mean_Speed", "Estimated_Travel_Time",
                               "Traffic_Level", "Total_Neighbours",
                               "Length", "OverallRank"]
                    writer.writerow(new_row)
                else:
                    total_pollutants = (((1*float(row[2]))
                                      + (100*float(row[3])) + (100*float(row[4]))
                                      + (1000*float(row[5])) / (10000)))

                    rank_index = math.floor(total_pollutants/350)

                    if rank_index >= 12:
                        rank_index = 11

                    rank = ranks[rank_index]

                    new_row = [row[5], row[6], row[7], row[8], row[9],
                               rank]

                    writer.writerow(new_row)

        file.close()
    ranked_file.close()
```

Appendix E

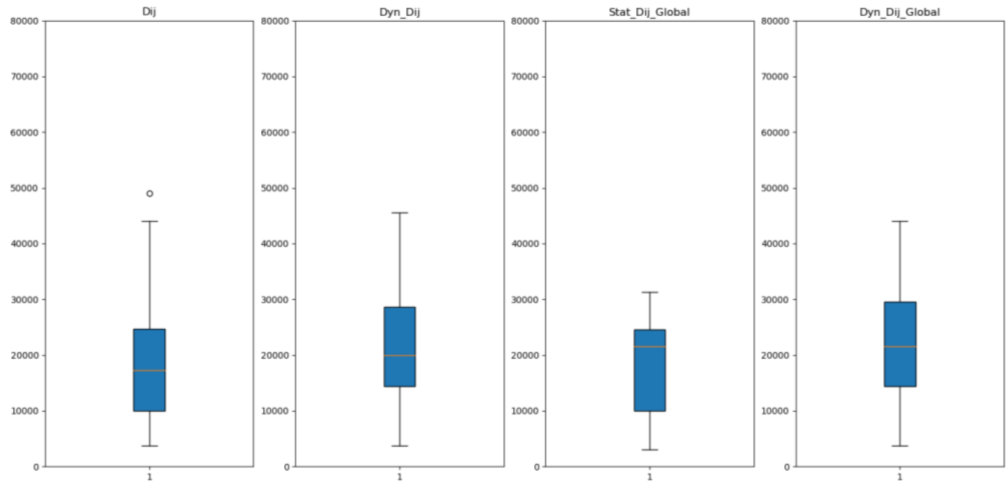
Correlation Matrix of Predictor Variables



Appendix F



Emission-Optimal (_Global) routing compared against Standard-Routing with 4 Pollution Labels



Emission-Optimal (_Global) routing compared against Standard-Routing with 12 Pollution Labels

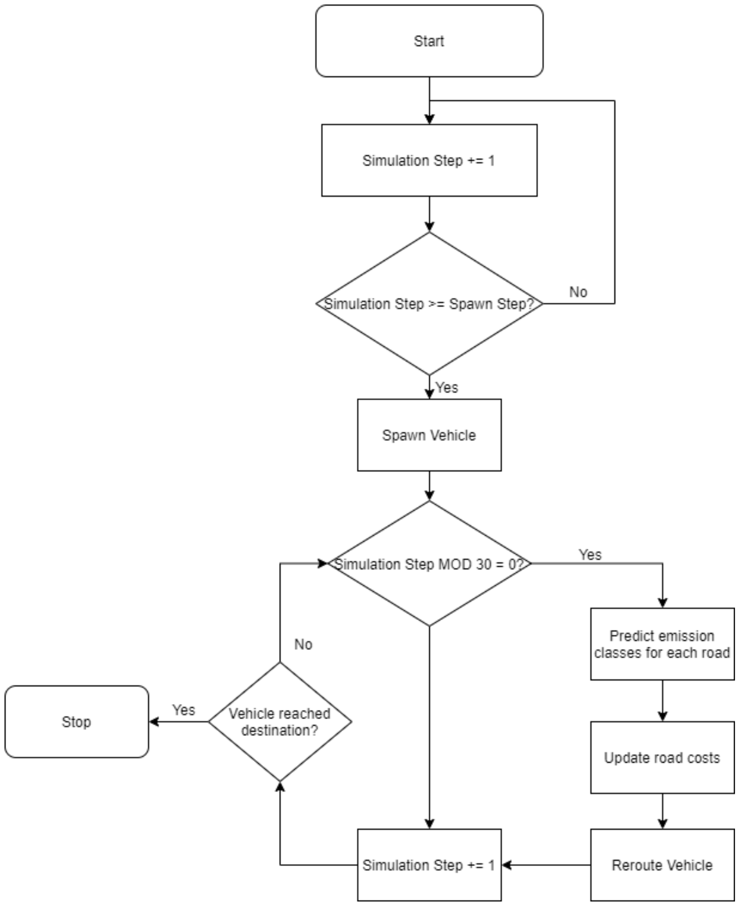
Appendix G

TrainNetwork.py, Feed-Forward Neural Network Implementation.

```
class LearningRateReducerCb(tf.keras.callbacks.Callback):
  def on_epoch_end(self, epoch, logs={}):
    prev_lr = self.model.optimizer.lr.read_value()
    new_lr = prev_lr * 0.95
    self.model.optimizer.lr.assign(new_lr)

def model(train_x, train_y, test_x, test_y, output_neurons, output_file_name):
    model = tf.keras.Sequential()
    model.add(Dense(64, input_dim=(train_x.shape[1]), activation='relu'))
    model.add(BatchNormalization())
    model.add(Dropout(0.2))
    model.add(Dense(32, activation='relu'))
    model.add(Dense(16, activation='relu'))
    model.add(Dense(output_neurons, activation='softmax'))
    opt = tf.keras.optimizers.Adam(learning_rate=0.001)
    loss = tf.keras.losses.SparseCategoricalCrossentropy(from_logits=False)
    model.compile(loss=loss, optimizer=opt, metrics=['accuracy'])

    early_stop = keras.callbacks.EarlyStopping(monitor='accuracy', min_delta=0,
                                               patience=15, verbose=0,
                                               mode='max', baseline=None)
    checkpoint = ModelCheckpoint(output_file_name, monitor='accuracy',
                                 mode='max', save_best_only=True)

    history = model.fit(train_x, train_y, epochs=10000, batch_size=75,
                        validation_data=(test_x, test_y), verbose=2,
                        shuffle=False, callbacks=[LearningRateReducerCb(),
                        early_stop, checkpoint])
    return history
```

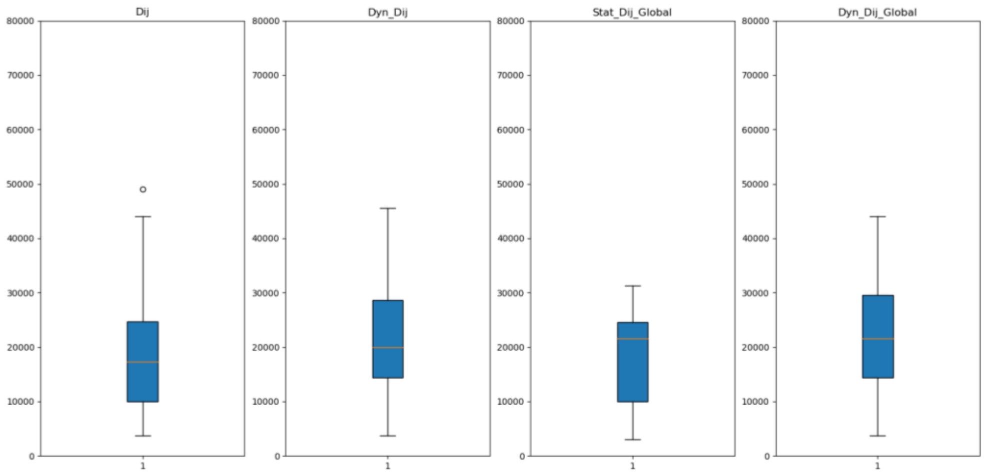
Appendix H

Dynamic Routing Design



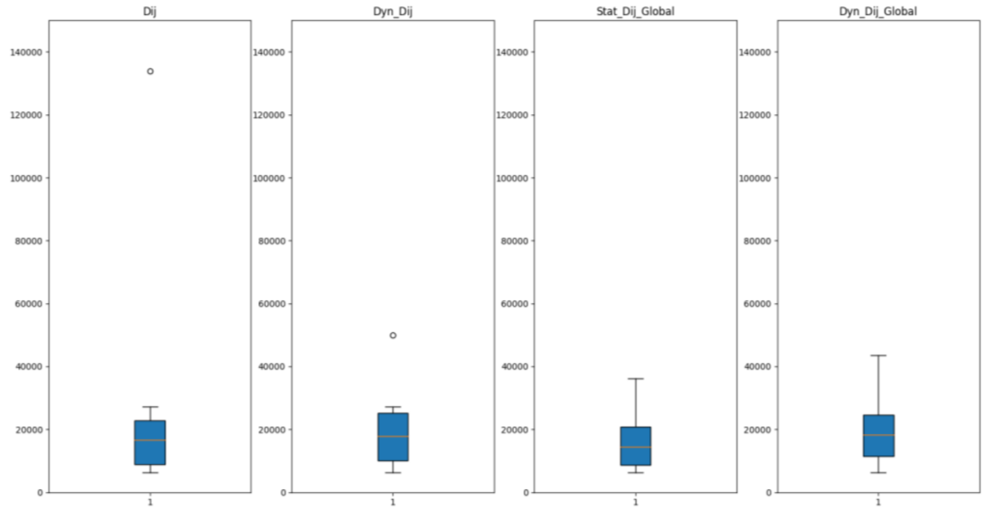
Appendix I

Boxplot analysis of Cost A weights against standard cost routing



Appendix J

Boxplot analysis of Cost B weights against standard cost routing.



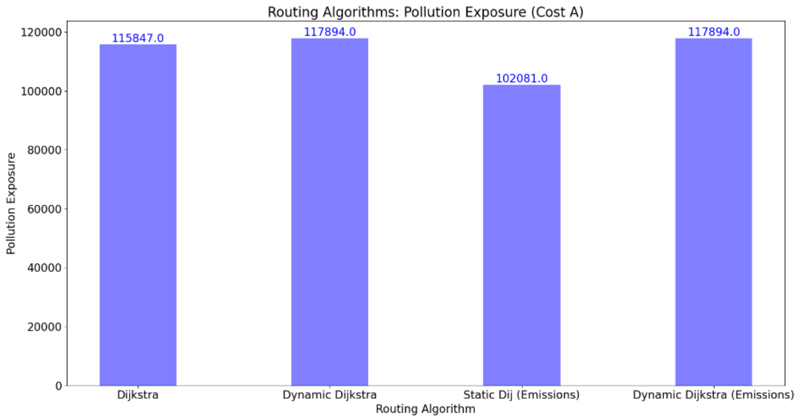
Appendix K

A visual comparison between the typical sized simulations used throughout the project and the larger sized simulation (map_11.osm) respectively.

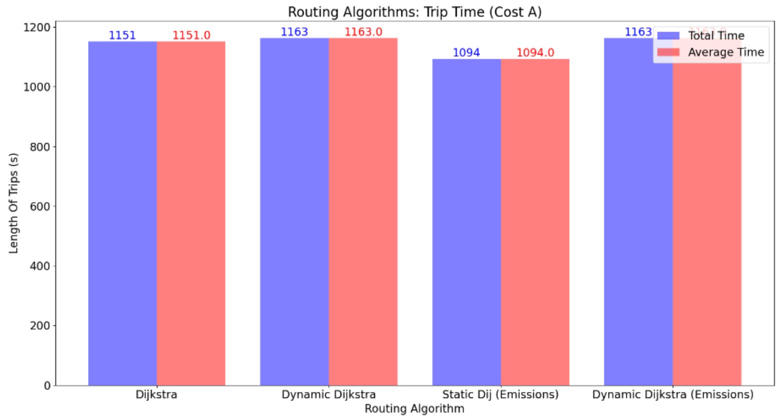
Appendix L

Results Analysis of the larger test simulation.

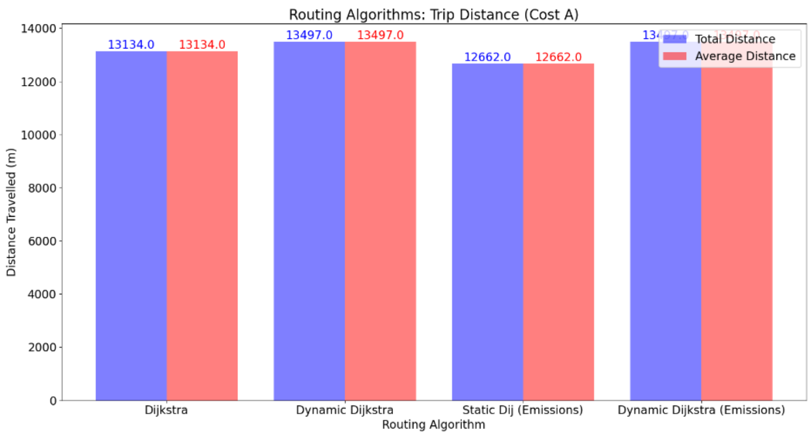
Average Pollution exposure:



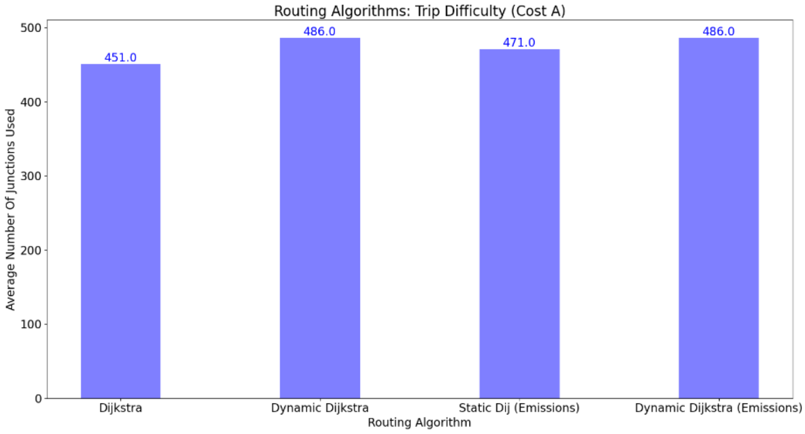
Trip Duration (s)



Trip Distance (m)



Trip Difficulty



Bibliography

[1] “Emissions of air pollutants from transport — European Environment Agency.” https://www.eea.europa.eu/data-and-maps/indicators/transport-emissions-of-air-pollutants-8/transport-emissions-of-air-pollutants-8 (accessed February 12, 2021).

[2] “Government takes historic step towards net-zero with end of sale of new petrol and diesel cars by 2030 - GOV.UK.” https://www.gov.uk/government/news/government-takes-historic-step-towards-net-zero-with-end-of-sale-of-new-petrol-and-diesel-cars-by-2030 (accessed May 24, 2021).

[3] “Predicting Air Pollution | IQAir.” https://www.iqair.com/blog/air-quality/can-air-pollution-be-predicted (accessed May 25, 2021).

[4] “Dispersal of air pollutants — European Environment Agency.” https://www.eea.europa.eu/publications/2599XXX/page005.html (accessed May 24, 2021).

[5] P. Kastner-Klein, R. Berkowicz, and E. J. Plate, “Journal of Environment and Pollution. Paper presented at the 5th Workshop on Harmonisation within Atmospheric Dispersion Modelling for Regulatory Purposes.”

[6] M. Kampa and E. Castanas, “Human health effects of air pollution,” *Environmental Pollution*, vol. 151, no. 2. Elsevier, pp. 362–367, Jan. 01, 2008, doi: 10.1016/j.envpol.2007.06.012.

[7] “Pollution | Threats | WWF.” https://www.worldwildlife.org/threats/pollution (accessed May 19, 2021).

[8] H. Mayer, “Air pollution in cities,” in *Atmospheric Environment*, Oct. 1999, vol. 33, no. 24–25, pp. 4029–4037, doi: 10.1016/S1352-2310(99)00144-2.

[9] “What are the causes of air pollution in inner cities - and how bad is it?” https://gsttcharity-uk.shorthandstories.com/thecausesofairpollutionininnercities/index.html (accessed May 25, 2021).

[10] “Air pollution.” https://www.who.int/health-topics/air-pollution#tab=tab_1 (accessed March 12, 2021).

[11] J. C. Wakefield, *HPA-CHaPD-004 A Toxicological Review of the Products of Combustion*. 2010.

[12] I. Manisalidis, E. Stavropoulou, A. Stavropoulos, and E. Bezirtzoglou, “Environmental and Health Impacts of Air Pollution: A Review,” *Frontiers in Public Health*, vol. 8. Frontiers Media S.A., p. 14, Feb. 20, 2020, doi: 10.3389/fpubh.2020.00014.

[13] “Carbon monoxide poisoning - NHS.” https://www.nhs.uk/conditions/carbon-monoxide-poisoning/ (accessed May 6, 2021).

[14] “Emissions of air pollutants in the UK – Nitrogen oxides (NOx) - GOV.UK.” https://www.gov.uk/government/statistics/emissions-of-air-pollutants/emissions-of-air-pollutants-in-the-uk-nitrogen-oxides-nox (accessed May 20, 2021).

[15] “Basic Information about NO2 | Nitrogen Dioxide (NO2) Pollution | US EPA.” https://www.epa.gov/no2-pollution/basic-information-about-no2 (accessed May 21, 2021).

[16] W. Health Organization and R. Office for Europe, “Review of evidence on health aspects of air pollution-REVIHAAP Project Technical Report,” 2013. Accessed: May 27, 2021. [Online]. Available at: https://www.euro.who.int/en/health-topics/environment-and-health/air-quality/publications/2013/review-of-evidence-on-health-aspects-of-air-pollution-revihaap-project-final-technical-report

[17] P. Kastner-Klein, R. Berkowicz, and E. J. Plate, “Modelling of vehicle-induced turbulence in air pollution studies for streets,” *Int. J. Environ. Pollut.*, vol. 14, no. 1–6, pp. 496–507, 2000, doi: 10.1504/ijep.2000.000573.

[18] U. Epa, O. of Transportation, A. Quality, and S. Division, “Frequently Asked Questions Frequently Asked Questions Near Roadway Air Pollution and Health: Frequently Asked Questions,” 2014.

[19] M. Cepeda *et al.*, “Levels of ambient air pollution according to mode of transport: a systematic review,” *Lancet Public Heal.*, vol. 2, no. 1, pp. e23–e34, Jan. 2017, doi: 10.1016/S2468-2667(16)30021-4.

[20] P. Kumar, I. Rivas, A. P. Singh, V. J. Ganesh, M. Ananya, and H. C. Frey, “Dynamics of coarse and fine particle exposure in transport microenvironments,” *npj Clim. Atmos. Sci.*, vol. 1, no. 1, p. 11, Dec. 2018, doi: 10.1038/s41612-018-0023-y.

[21] A. Azid, H. Juahir, T. Latif, S. M. Zain, and M. R. Osman, “Feed-Forward Artificial Neural Network Model for Air Pollutant Index Prediction in the Southern Region of Peninsular Malaysia,” *J. Environ. Prot. (Irvine,. Calif).*, 2005, doi: 10.4236/jep.2013.412A1001.

[22] A. Kurt, B. Gulbagci, F. Karaca, and O. Alagha, “An online air pollution forecasting system using neural networks,” *Environ. Int.*, vol. 34, no. 5, pp. 592–598, Jul. 2008, doi: 10.1016/j.envint.2007.12.020.

[23] Y. Tsai, Y. Zeng and Y. Chang, "Air Pollution Forecasting Using RNN with LSTM," 2018 IEEE 16th Intl Conf on Dependable, Autonomic and Secure Computing, 16th Intl Conf on Pervasive Intelligence and Computing, 4th Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress(DASC/PiCom/DataCom/CyberSciTech), 2018, pp. 1074-1079, doi: 10.1109/DASC/PiCom/DataCom/CyberSciTec.2018.00178.

[24] Arbel Nir, “How LSTM networks solve the problem of vanishing gradients | by Nir Arbel | DataDrivenInvestor,” Dec. 21, 2018. https://medium.datadriveninvestor.com/how-do-lstm-networks-solve-the-problem-of-vanishing-gradients-a6784971a577 (accessed Jun. 10, 2021).

[25] V. Pillac, M. Gendreau, C. Guéret, and A. L. Medaglia, “A review of dynamic vehicle routing problems,” *Eur. J. Oper. Res.*, vol. 225, no. 1, pp. 1–11, Feb. 2013, doi: 10.1016/j.ejor.2012.08.015.

[26] T. Cannon, “Development and Evaluation of Vehicle Routing Algorithms with the Aim of Reducing Vehicle Occupant Air Pollution Exposure”, Project Report from the Department of Computer Science at the University of York, 2019

[27] E. Ben-Elia *et al.*, “The combined effect of information and experience on drivers’ route-choice behavior,” *Transportation (Amst).*, vol. 35, pp. 165–177, 2008, doi: 10.1007/s11116-007-9143-7.

[28] P. A. Lopez *et al.*, “Microscopic Traffic Simulation using SUMO,” in *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, Dec. 2018, vol. 2018-November, pp. 2575–2582, doi: 10.1109/ITSC.2018.8569938.

[29] “Eclipse SUMO - Simulation of Urban MObility.” https://www.eclipse.org/sumo/ (accessed Jan. 11, 2021).

[30] “TraCI - SUMO Documentation.” https://sumo.dlr.de/docs/TraCI.html (accessed Jan. 11, 2021).

[31] “Vehicle licensing statistics: 2020 - GOV.UK.” https://www.gov.uk/government/statistics/vehicle-licensing-statistics-2020 (accessed Jun. 11, 2021).

[32] "HBEFA - Handbook Emission Factors for Road Transport", *Hbefa.net*, 2021. [Online]. Available: https://www.hbefa.net/e/index.html (accessed Jun. 6, 2021).

[33] “European emission standards - Wikipedia.” https://en.wikipedia.org/wiki/European_emission_standards (accessed Jun. 11, 2021).

[34] D. Krajzewicz, M. Behrisch, P. Wagner, R. Luz, and M. Krumnow, “Second generation of pollutant emission models for SUMO,” in *Lecture Notes in Control and Information Sciences*, 2015, vol. 13, pp. 203–221, doi: 10.1007/978-3-319-15024-6_12.

[35] S. Zeadally, R. Hunt, Y. S. Chen, A. Irwin, and A. Hassan, “Vehicular ad hoc networks (VANETS): Status, results, and challenges,” *Telecommun. Syst.*, vol. 50, no. 4, pp. 217–241, Aug. 2012, doi: 10.1007/s11235-010-9400-5.

[36] “The Popularity of Google Maps: Trends in Navigation Apps in 2018 | March 2021.” https://themanifest.com/mobile-apps/popularity-google-maps-trends-navigation-apps-2018 (accessed Jun. 11, 2021).

[37] “Driving Directions, Traffic Reports & Carpool Rideshares by Waze.” https://www.waze.com/en-GB/ (accessed Jun. 11, 2021).

[38] “OpenStreetMap.” https://www.openstreetmap.org/#map=7/54.910/-3.432 (accessed Jan. 25, 2021).

[39] “Planet OSM” https://planet.openstreetmap.org/ (accessed Jan. 25, 2021).

[40] “PEP 8 -- Style Guide for Python Code | Python.org.” https://www.python.org/dev/peps/pep-0008/ (accessed Jan. 15, 2021).

[41] “Keras: the Python deep learning API.” https://keras.io/ (accessed Mar. 5, 2021).

[42] “chrisblatchley/sumo-cuda: An implementation of the SUMO traffic simulation tool suite using CUDA for GPU speedup.” https://github.com/chrisblatchley/sumo-cuda (accessed Feb. 24, 2021).

[43] “tensorflow/tensorflow: An Open Source Machine Learning Framework for Everyone.” https://github.com/tensorflow/tensorflow (accessed Mar. 7, 2021).