



UNIVERSITY OF CAPE TOWN

FACULTY OF ENGINEERING AND THE BUILT ENVIRONMENT

Department of Civil Engineering

CIV4044S



Research Project

Title: The use of Neural Networks for traffic volume forecasting.

Prepared for: Prof. Mark Zuidgeest and Dr. Obiora Nnene

Prepared by: Norbert Matilya

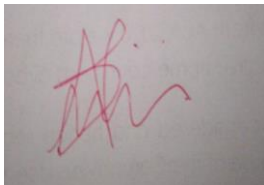
Plagiarism Declaration

1. I know that plagiarism is wrong. Plagiarism is to use another's work and to pretend that it is one's own.
2. I have used the Harvard Convention for citation and referencing. Each significant contribution to and quotation in this report from the work or works of other people has been attributed and has been cited and referenced.
3. This report is my own work.
4. I have not allowed and will not allow anyone to copy my work with the intension of passing it as his or her own work.

Name: Norbert Matilya

Student Number: MTLNOR002

Signature:

A handwritten signature in red ink, appearing to be 'Norbert Matilya', is shown on a white background.

Acknowledgments

This research proposal could not have been done without the assistance of the following individuals:

- **Prof. Mark Zuidgeest**, Professor at the University of Cape Town. His guidance on the viability of the importance of traffic volume forecasting was critical in my decision to conduct this research.
- **Dr. Obiora Nnene**, Postdoctoral Researcher at the University of Cape Town. His guidance on machine learning was important in conducting this research.
- **Derick Kazimoto**, his input during the preliminary stages of this research was helpful. He suggested possible frameworks and topics for this research.

Terms of Reference

On the 15th of July 2020, Norbert Matilya proposed an undergraduate research project to Prof Dr. Mark Zuidgeest and Dr. Obiora Nnene. This research would assess the feasibility of the use of Neural Networks for traffic flow predictions. Upon acceptance of this proposal, the final topic, 'Using Neural Networks to Forecast Traffic Volumes' was used as the research topic. This research aims to show that Neural Networks are a feasible modelling tool that can be used to forecast traffic volumes.

The following instructions were followed for this research project:

1. Conduct preliminary research on topic.
2. Conduct a literature review.
3. Obtain and sort the traffic volume data.
4. Create, implement and analyse Neural Network models that would be useful for this context.
5. Select and recommend the most accurate model.

The following deadlines will be observed throughout the research period:

Research proposal	21 st August 2020
Draft Research Report and Poster	25 th October 2020
Final Research Report and Poster	07 th December 2020

Abstract

A study done by Statistics South Africa showed that the most common transport mode for people going to work was private car (34.1%), then taxis (22.9%) (Statistics South Africa, 2018). This shows that a large majority of people depend on an efficient road transportation system to move from place to place. A good understanding of the movement of people will not only allow for better management of limited resources for the improvement of roads but could potentially allow for a better designed road system. Thus, being able to forecast traffic volumes accurately will make designing road transportation systems more efficient.

The availability of vehicle detection devices at strategic locations around the Western Cape Province, in South Africa, makes the collection of reliable and accurate traffic volume data possible. For this research project, an attempt was made to improve traffic volume forecasting. A variation of Machine Learning algorithm known as Neural Networks(NN) were used to forecast traffic volume values.

Neural networks are a more complex variation of machine learning algorithms. They go through large data in order to find complex patterns within the dataset (Ruder, 2016). Within the neural network framework, Recurrent Neural Networks(RNN) in particular have been used for traffic volume forecasting because of their ability to process data that is time dependant. Time series regression is a method used to predict future values based on time dependent data (MathWorks, 2020). Time series regression allows a model to notice trends, cycles, seasonal variations and irregular components within a dataset. Based on historical data learning outcomes from historical traffic data, time series prediction models can be used to predict traffic volumes adequately (Karami & Kashef, 2020). For this research Long Short Term Memory(LSTM) and Gated Recurrent Unit(GRU) algorithms were used to model traffic volume data.

For this research project, data collected by the aforementioned vehicle detection devices were used to train the models. The data had to be formatted and sorted to suit the format needed for it to be used by the model. In order to get the best model design an iterative process was adopted to find the parameters within the models that will improve the overall accuracy of the models. These parameters were optimized to maximise the forecasting ability of the models.

After analysis, the results show that both models were able to forecast traffic volume counts accurately. Of the two models, the GRU model was the more accurate model. This shows that Neural Networks models can be used to as a method to forecast traffic volume data.

Table of Contents

Plagiarism Declaration	i
Acknowledgments.....	iii
Terms of Reference	iv
Abstract	v
List of Figures	ix
List of Tables	ix
List of Acronyms	ix
Glossary	x
1 Introduction.....	1
1.1 Subject and Motivation	1
1.2 Background of the Study	1
1.3 Objectives.....	1
1.4 Scope and Limitations	2
1.5 Plan of Development	2
2 Literature review	4
2.1 Traffic volume forecasting.....	4
2.2 Neural Network(NN).....	4
2.2.1 Feed forward Neural Network.....	4
2.2.2 Backpropagation.....	6
2.2.3 Recurrent Neural Networks (RNN)	8
2.2.4 Time series for traffic volume prediction.....	9
2.3 Model Selection.....	9
2.3.1 Long Short Term Memory (LSTM)	9
2.3.2 Gated Recurrent Unit (GRU)	9
2.4 Model Implantation	9
2.4.1 Training Data split	9
2.4.2 Scaling.....	9
2.4.3 Dropout	10
2.4.4 Dense Layer.....	10
.....	10
2.4.5 Epochs.....	10

2.4.6	Training	10
2.5	Model Optimization.....	11
2.5.1	Stochastic Gradient Descent (SGD)	11
2.5.2	Root Mean Square Propagation (RMSProp).....	11
2.5.3	Adaptive Moment Estimation (Adam)	11
2.5.4	Regularization and Overfitting	11
2.6	Model Analysis	12
3	Methodology	13
3.1	Data Description.....	13
3.2	Site Selection.....	14
3.3	Data Cleaning	15
3.3.1	Organising the data	15
3.4	Model Analysis	16
3.4.1	Model testing.....	16
3.4.2	Model optimizer selection.....	16
3.4.3	Augmenting model architecture	16
3.5	Model Forecasting	17
4	Results	18
4.1	LSTM.....	18
4.1.1	Model architecture	18
4.1.2	Comparison with Test Data.....	19
4.1.3	Comparison Forecast Data	20
4.2	GRU	21
4.2.1	Model architecture	21
4.2.2	Comparison with Test Data.....	22
4.2.3	Data Forecasting	23
4.3	Model Comparison	24
5	Discussion.....	25
5.1	Overall Data composition	25
5.1.1	Data structure	25
5.1.2	Data Scaling	25
5.1.3	Missing Data.....	25
5.1.4	Train/Test Split.....	25
5.1.5	Timestep implementation.....	26
5.2	LSTM model performance.....	26

5.3 GRU model performance	26
5.4 Model Comparison	27
6 Conclusion	28
7 Recommendations	29
7.1 Data collection	29
7.2 Model selection	29
7.3 Potential implementations.....	30
8 Bibliography	31
Appendix A: Exit Level Outcomes (ELO).....	34
1. Problem solving.....	34
2. Investigations, experiments and data analysis	34
3. Professional and technical communication.....	34
4. Individual, team and multidisciplinary working	34
5. Independent learning ability	34
Appendix B: Code Used.....	35

List of Figures

Figure 1: Neuron Structure (Vieira, 2017)	5
Figure 2: Neural Network Structure (Vieira, 2017).....	6
Figure 3: Example of a Neural Network	7
Figure 4: Dense Layer Diagram	10
Figure 5: VDS 109 site location	15
Figure 6: Model forecasting algorithm illustration	17
Figure 7: LSTM model architecture.....	18
Figure 8: LSTM model prediction compared to test data	19
Figure 9: LSTM model forecast	20
Figure 10: GRU model architecture.....	21
Figure 11: GRU model prediction compared to test data.....	22
Figure 12: GRU model forecast.....	23
Figure 13: LSTM and GRU forecasting comparison.....	24
Figure 14: Timestep data.....	26

List of Tables

Table 1: Sample of Data recorded.....	13
Table 2: Location Data	14
Table 3: Sample of points collected at VDS 104	14
Table 4: Sample Data Calculations	16
Table 5: LSTM model addition details.....	18
Table 6: LSTM predictions compared to test data.....	19
Table 7: Error produced by LSTM model forecast	20
Table 8: GRU model addition details.....	21
Table 9: GRU predictions compared to test data	22
Table 10: Error produced by GRU model forecast	23
Table 11: Error comparison between GRU and LSTM model forecasting	24

List of Acronyms

Adam	Adaptive moment estimation
GRU	Gated Recurrent Unit
IB	In Bound
LSTM	Long Short Term Memory
MAE	Mean Average Error
MAPE	Mean Absolute Percentage Error
MSE	Mean Square Error
NN	Neural Network
OB	Out Bound
Relu	Rectified linear unit
RNN	Recurrent Neural Network
RMSE	Root Mean Square Error
RMSProp	Root Mean Square Propagation
SGD	Stochastic Gradient Descent
WC	Western Cape

Glossary

Activation Function	This is a mathematical function which augments the data. It is used to limit data point between specific values.
Bias	This is a constant that is added to the input of a node to shift the value of the activation function
Cleaning	The process of removing errors, duplicates and unnecessary data within a dataset.
Dropout	This an algorithm used to prevent overfitting.
Error	The difference between the values forecasted by the model and the actual values observed.
Gradient	The partial derivative of a specific weight with respect to the total error of the model.
Gated Recurrent Unit(GRU)	Is a Neural Network model used to make predictions. It is an updated version of an LSTM model.
Layer	A sequence of neurons that receives weighted inputs, and passes the inputs through an activation function.
Long-Short Term Memory(LSTM)	This is an updated version of a Recurrent Neural Network. It makes predictions, but does not loose the influence of weights in earlier hidden layers overtime.
Mean Absolute Error(MAE)	This is the average of all the absolute errors.
Neural Network	Neural networks are a subset of machine learning that is used to model more complex patterns in database.
Neuron	A data point in a neural network. It typically takes multiple input values and generates one output by putting the values through a series of mathematical formulas.
Overfitting	This is when data mimics the training data used too closely resulting in the model struggling to predict data which it has not 'seen' before.
Root Mean Square Error(RMSE)	This is an equation that is used to find the difference between the predictive values made by model, and the true values observed on the ground.
Testing data	A section of the dataset that is used to check the validity of the model's forecasting.
Time series regression	A statistical method that is used to predict future values based on time dependent data.
Traffic Volume	The number/count of vehicles in an area
Training	This is the process of a model finding a set of weights and biases that produce the most accurate predictions.
Weights	Randomly generated numbers that augment the values of the data input into it. They decide how much influence the input has on the prediction

1 Introduction

1.1 Subject and Motivation

With the availability of vehicle detection devices throughout the Western Cape that can record traffic volumes, it has become easier to gain access to accurate traffic volume data. With this data, reliable traffic planning can be done to improve the current road transportation system. For this research project, neural network models will be used to forecast future traffic volumes. Successful implementation of a model will allow transport engineers and town planners to improve the current traffic planning system because it will be easier to plan with access to reliable traffic volume predictions (Karami & Kashef, 2020).

1.2 Background of the Study

All the resources needed for a person to live is probably not where they stay. In order to get all the resources to sustain ourselves, people need to move from place to place to achieve different goals. A study done by Statistics South Africa showed that the most common transport mode for people going to work was private car (34.1%), then taxis (22.9%) (Statistics South Africa, 2018). This shows that a large majority of people depend on an efficient road transportation system to be able to move from place to place. A good understanding of the movement of people will not only allow for a better management of limited resources for the improvement of roads but will also allow for a better designed road system. Thus, being able to forecast traffic volumes will make it easier to design transportation systems efficiently and effectively.

In recent years the improvement of computing power, machine learning and artificial intelligence has rapidly developed within the transportation field (Changxi, et al., 2019). This surge in the use of advanced computing has allowed for experimentation with different algorithms within the machine learning framework. With this surge of use, it has been found that neural networks are a viable way of forecasting traffic flow (Fu, et al., 2016). This research will be centred on finding out how accurate some of these models are and if they can be used in South Africa.

1.3 Objectives

The objective of this research project is to use data that was collected by vehicle detection devices to create Neural Network models that can forecast traffic volume counts. The following procedure was used during the research process to accomplish this objective:

1. Conduct literature review on the use of machine learning for traffic volume prediction.
2. Obtain hourly volume data for the Western Cape region.
3. Analyse and sort this data.

4. Build Neural Network models to forecast future traffic volumes.
5. Place traffic volume data into the model and run it.
6. Analyse the results.

1.4 Scope and Limitations

- Over the past few years, neural networks have improved in complexity and quality. Even with these improvements neural networks cannot forecast data with 100% accuracy.
- With limited computing processing power, there is only so much data I was able to process, thus limiting the scope of the predictions.
- The data obtained for this research had hourly volume traffic increments. For more accurate predictions, more refined data would be needed e.g. traffic volume per minute.
- The time for the completion of this study was nine weeks, which was limited.

1.5 Plan of Development

This research project begins with acquiring the necessary data that will be needed for analysis. The analysis of the data will give a better understanding of how to make a model that will best represents the data. The data should then be sorted and refined so that only the relevant data will be used within the model.

The model's architecture will then be created. The model made will be able to carefully and accurately analyse the data, learn the patterns within the data, and then forecast future data based on the data it has trained on. The true values will then be compared to the output of the model. These values will be compared and if there is a large discrepancy between them, certain parameters within the model will be changed in order to produce the most accurate outputs.

To adequately describe the entire research process, this report has been drafted. The report is subdivided into different chapters as shown below:

- Chapter 1: Introduction – This section describes the subject and motivation, background to the study, objectives and scope and limitations of the research.
- Chapter 2: Literature Review – This outlines how machine learning is a growing part of the transportation field. It then describes neural networks in detail, how they work, why they can be used in the context of traffic volume forecasting and which models best fit this context.

- Chapter 3: Methodology – This section details the entire procedure that was used to make the models. From sorting and cleaning the data, to making and analysing the models.
- Chapter 4: Results – The traffic forecast output from the models will be compared to the actual traffic volumes.
- Chapter 5: Discussion – The results from the previous section will be analysed and explained in detail.
- Chapter 6: Conclusion – A summary of the entire research process will be given.
- Chapter 7: Recommendations – Suggestions for how this research project can be improved upon will be shared.

The appendices will have information that is not in the main report, but are essential to the research project.

2 Literature review

2.1 Traffic volume forecasting

Traffic forecasting has always been an important aspect of traffic optimization and thus there have been many attempts to predict traffic data (Fu, et al., 2016). In recent years the improvement of computing power, machine learning and artificial intelligence has rapidly developed within the transportation field (Changxi, et al., 2019). This surge in the use of advanced computing has allowed for experimentation with different algorithms within the machine learning framework.

Over the past few years Neural Networks(NN) have been used to build models for predictions. Within the neural network framework, Recurrent Neural Networks(RNN) in particular have been used because of their ability to process data that is time dependant. A method that is used to predict future values based on time dependent data is called Time series regression (MathWorks, 2020). Time series regression allows a model to notice trends, cycles, seasonal variations and irregular components within a dataset. Based on historical data learning outcomes from historical traffic data, time series models can predict traffic volumes adequately (Karami & Kashef, 2020).

2.2 Neural Network(NN)

A Neural Network(NN) is a subset of the wider machine learning framework. Machine learning models go through large datasets and learns the patterns that the data has, this allows for the prediction of future values within the dataset based on ‘unseen’ data (Google Developers, 2020). The primary aim is to allow the algorithm to find patterns within the dataset without the continuous intervention of a human. This allows for analysis of large datasets without monotonous human intervention.

Neural networks are a more complex variation of machine learning. They go through large datasets in order to find complex patterns within them (Ruder, 2016). The neural network architecture is roughly modelled on the human brain (Brownlee, 2019). There are multiple processing neurons that are densely interconnected. The neurons are structured into layers with each layer finding different patterns within the dataset. Traditional neural networks take inputs and move them through each layer sequentially until processing is completed.

2.2.1 Feed forward Neural Network

The NNs will follow a feedforward structure. In a feedforward NN, the information travels linearly. The data travels through the input neurons, then through hidden layer(s) and finally goes through the output neuron (McGonagle, et al., 2020). Between each layer there are weights and biases, which are arbitrary values that are used to find the trends within a database. A layer is a sequence of neurons that receives weighted inputs, and passes them through an activation function. An activation function is a mathematical function which augments data input into it. It is used to limit data point between certain parameters (Burkov, 2019). Common activation functions are the sigmoid, tanh and Rectified linear unit (Relu) functions.

$$relu(x) = \begin{cases} x, & x > 0 \\ 0, & x \leq 0 \end{cases}$$

$$Sigmoid(x) = \frac{1}{1 + e^{-x}}$$

$$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

The data within a dataset is split into neurons in the input layer depending on the attribute of the dataset e.g. speed, count etc. Attributes in the input layer are given an arbitrary value and are arranged in a matrix. This data then moves from the input layer to a hidden layer. As the arbitrary values moves between the layers there are weights which affect the data. The weights are arranged in a matrix and are multiplied with the arbitrary inputs given to them. A bias is then added to this function to form a weighted sum. The weighted sum then travels to the hidden layer(s). Once at the hidden layer the weighted sums pass through the neurons within the hidden layer. The neurons in a hidden layer represent an activation function. An activation function suppresses the weighted sum into a function. This is then output to the next layer.

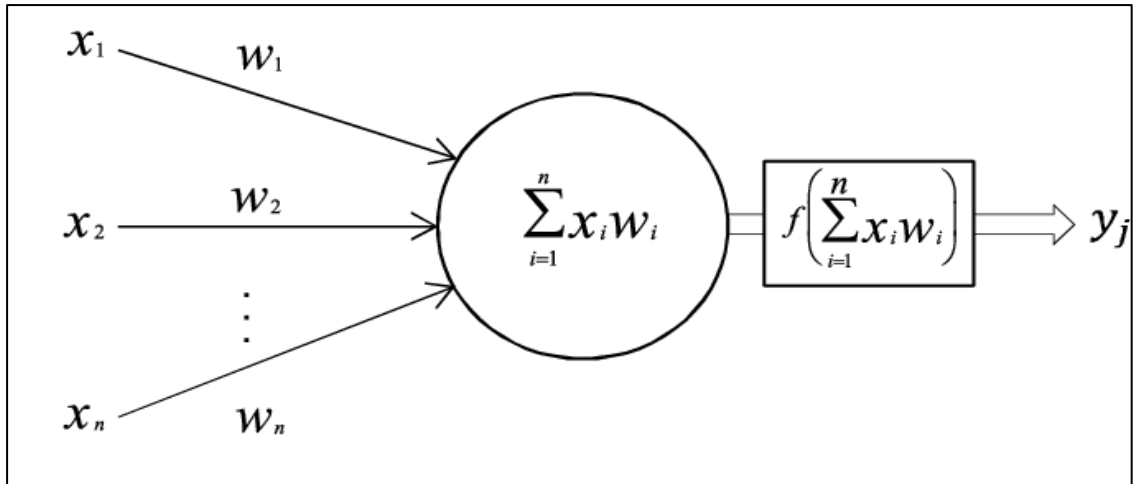


Figure 1: Neuron Structure (Vieira, 2017)

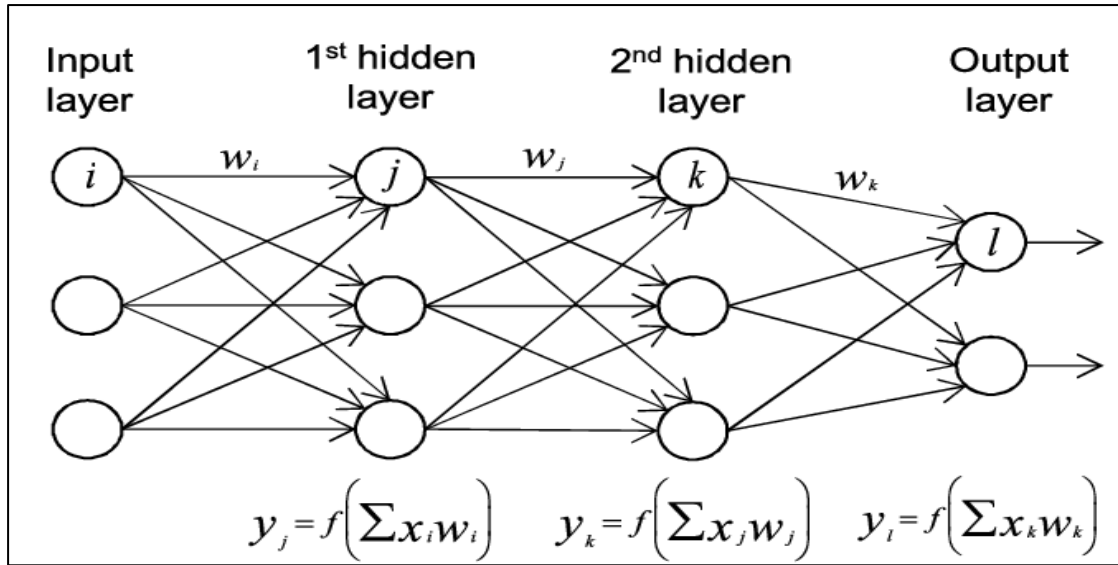


Figure 2: Neural Network Structure (Vieira, 2017)

x	Input data
w	Weight
y	Output
f	Activation Function

This process is repeated between each hidden layer until the values reach the final layer. Once at the final layer the output is compared to the expected outcome. The difference between the expected outcome and the model output is the error or cost.

2.2.2 Backpropagation

In order to reduce the error, the weights in-between the layers of the neural network should be augmented. To do this the weights have to change in proportion to how much they affect the error. This is done by finding the partial derivative of each individual weight with respect to the average error of the model, this is called the gradient of that particular weight. As the model continues to run, the gradient of each weight will change, thus update the weights. As the weights are being updated, the total error/loss of the model reduces, thus the model is more accurate. The process of using the final error to change individual weights is called backpropagation (Burkov, 2019).

The backpropagation method is illustrated below:

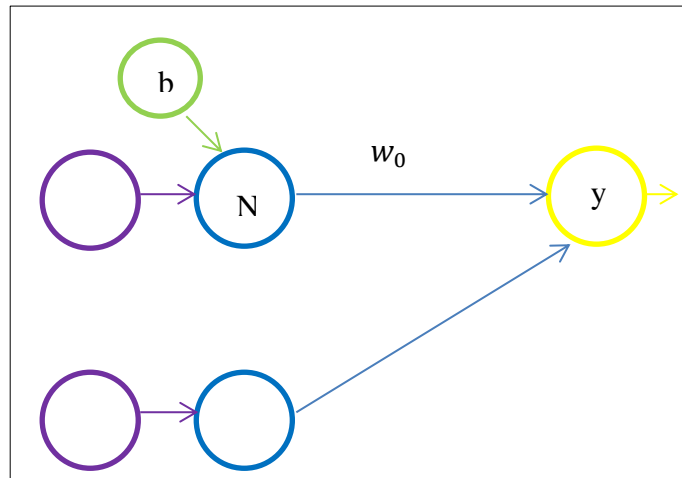


Figure 3: Example of a Neural Network

N	Neuron
w_0	Weight
b	Bias
y	Output of the Node
Purple Circle	Input Layer
Blue Circle	Hidden layer
Yellow Circle	Output Layer

- For the output node, the sigmoid function will be used as the Activation function(Af).

$$Af(x) = \frac{1}{1 + e^x}$$

- This is the error function. The target is the expected value, while the output is the prediction from the neural network model.

$$Error(x) = \frac{1}{2}(Target - Output)^2$$

$$y_i = N_{out} \times w_0 + b$$

y_i Weighted sum

N_{out} This is the output from the previous neuron

$$y_{out} = Af(y_i)$$

y_{out} The output of data as it moves out of a neuron within a layer

$$E(y_{out}) = \frac{1}{2}(Target - y_{out})^2$$

In order to get a factor to change the weight between the two layers, its partial derivative with respect to the total error is needed. This is called the gradient of the particular weight.

$$gradient = \frac{\partial E}{\partial w_0}$$

The formula contains the initial weight value(w_0) , but this is not a parameter within the error equation. Thus to get this partial derivative, the chain rule needs to be implemented.

$$\frac{\partial E}{\partial w_0} = \frac{\partial E}{\partial y_{out}} \times \frac{\partial y_{out}}{\partial y_i} \times \frac{\partial y_i}{\partial w_0}$$

- Updating the weight

$$w_{new} = w_o - \mu \frac{\partial E}{\partial w_0}$$

w_{new} Updated weight

μ Learning Rate

The learning rate is a scalar value that controls how much the weights are adjusted with respect to the gradient loss (Brownlee, 2019). Choosing a reliable rate for a model is important as a value that is too small will result in the model taking a long time to train, whereas a value that is too big will result in sub-optimal change in weights as they will be changing too fast.

The initial weight and basis within a neural networks are initially randomized, thus there should be a lot of variation with how rapidly the neurons learn within each layer. But after multiple tests, it has been found that gradient within the first few hidden layers tend to become too small as models with more complex layers are used (Nielsen, 2015). The lower the gradient, the smaller the changes in the weights for particular neuron within the model. This problem is known as vanishing gradient, and it prevalent in feed-forward neural networks.

2.2.3 Recurrent Neural Networks (RNN)

One of the main issues with traditional feed forward neural networks is that the previous data input will not affect the next data input. To solve this Recurrent Neural Networks(RNN) were created (Nielsen, 2015). A RNN uses internal memory cells to store the output of the previous layer. This gives the model the distinct advantage of being able to use aspects of every layer to form a prediction. This makes it useful for complex datasets (Nielsen, 2015).

2.2.4 Time series for traffic volume prediction

The time dependant nature of traffic counts makes RNNs a very good neural network to use to forecast traffic counts. However, its inability to not select important data to store makes it inefficient (Hochreiter, et al., 2009). To keep the advantages of a RNN and eliminate vanishing gradients, LSTM(Long Short Term Memory) were developed. A LSTM model is able to exploit the long – term dependency in traffic data and can find important features hidden in traffic volume data, which makes it a more useful forecasting model than a traditional RNN (Luo, et al., 2019) . GRU(Gated Recurrent Unit) is a variation of an LSTM model. It has most of its features but it is able to process data faster. This makes it an ideal model to use for traffic volume forecasting (Fu, et al., 2016).

2.3 Model Selection

2.3.1 Long Short Term Memory (LSTM)

Long Short Term Memory (LSTM) is a specially designed RNN model. LSTM memory units have the ability to determine when to remember or forget certain parts of the information. LSTMs attempt to truncate the gradient in order to minimize the error, but not loose important data that will be useful for the next layer (Hochreiter, et al., 2009). When the important information is saved in the memory units, the LSTM then finds the optimum way to efficiently find it in the next iteration of the model's training loop. This illustrates the ability for LSTMs to store historical data effectively overtime (Changxi, et al., 2019).

2.3.2 Gated Recurrent Unit (GRU)

Gated Recurrent Unit(GRU) is an RNN model that is a variation of the LSTM model. It inherits the ability to learn features, and effectively model long – distance dependencies. Its main advantage over LSTM is that is can process data faster ,while being able to effectively learn the important details to keep in its memory units (Changxi, et al., 2019).

2.4 Model Implantation

2.4.1 Training Data split

To efficiently make models based of a dataset, it best to not use all the data to train a model. If this is done, make preliminary tests based of the complete dataset will be inefficient. To avoid this, data that is used to build a model is split into training and testing data (Géron, 2017).

Training data will be placed into the model to train the model to make predictions. The testing data will subsequently be used to test the accuracy of the model. Once the model has completely trained, the testing data is fed into the model and outputs predictions. The predictions are then compared to the true values to assess the accuracy of the model. The difference between the testing data and the predicted value is the error. If the error is significant, then aspects of the model's architecture should be changed to minimise this error.

2.4.2 Scaling

Data scaling is a recommended pre-processing step when using neural networks. Unscaled input variables can result in slow or unstable learning process, and unscaled target variables tend to have vanishing gradient issues causing the learning process to fail (Brownlee, 2019).

2.4.3 Dropout

One of the key issues in creating models based of data is overfitting. Overfitting happens when a model matches the training data so closely that the model struggles with making good predictions with new data. One technique to mitigate this is the Dropout technique. Dropout regularization works by extracting a random selection of a given number of input values of a neural network. This prevents the model from exclusively learning all the data in the training set, thus reduces the probability of overfitting (Srivastava, et al., 2014).

2.4.4 Dense Layer

A dense layer is a NN layer that is deeply connected. This means that each neuron of the dense layer gets inputs from all neurons from the previous layer (Géron, 2017). Fully connected layers go to the next layer and combine to form a single output per neuron. This is used to take the information from multiple neurons and place them into a single neuron.

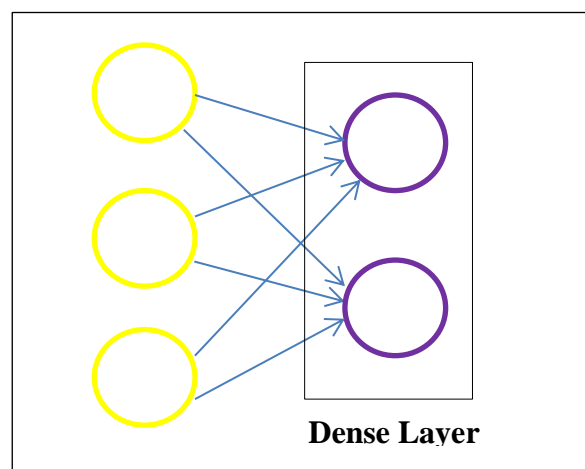


Figure 4: Dense Layer Diagram

2.4.5 Epochs

An epoch is the number of iterations that the model will go through to complete its training (Google Developers , 2020). The bigger the number of epochs, the longer the optimizer has to minimise the error. An epoch value that is too big will result in the model overfitting onto the data. A small epoch value will limit the time the model has to find the minimum error. The numbers of epochs vary depending on the situation.

2.4.6 Training

Training is the process of executing all the parameters within the model to find a prediction (Hardesty , 2017). The training dataset is input into the model. The training process can vary in time depending on how much data needs to be processed.

2.5 Model Optimization

The main aim of an optimization algorithm is to reduce the error/cost of a model by updating the model in response to its output. This is done by changing the weights and biases within the model in order to minimize the error (Ruder, 2016). There are two ways with which an optimizer's efficiency is assessed; speed of convergence to the minimum error and the model's accuracy once tested. The most frequently used optimizer algorithms used for neural networks are variations of the Stochastic Gradient Descent(SGD) algorithm known as Root Mean Square Propagation(RMSProp) and Adaptive moment estimation (Adam) (Burkov, 2019).

2.5.1 Stochastic Gradient Descent (SGD)

Stochastic Gradient Descent(SGD) is the backbone to most optimizer algorithms. The algorithm attempts to find the minimum of the partial derivation of each individual weight with respect to the error (Burkov, 2019). But finding the minimum for each weight is feasible for a small dataset, but it will be time consuming for large datasets. To mitigate this the SGD selects a batch of inputs and calculates the minimum gradient for these specific weights throughout the runtime of the model. This results in a more sporadic method to reach the global minimum, but it eventually reaches a reasonable minimum relatively fast. The batch size is selected before the execution of the model.

2.5.2 Root Mean Square Propagation (RMSProp)

Root Mean Square Propagation(RMSProp) is a variation of SGD. The RMSProp algorithm has an adaptive learning rate instead of using a single scalar learning rate for every parameter. Every parameter gets a special learning rate controlled by a global learning rate (Howard & Guggen, 2020). The learning rate adapts based on the average of the recent magnitudes of the gradients of the weights (Brownlee, 2017). This allows for training to be faster as large learning rates are used for weights that need to change, while weights that do not need to change drastically are given smaller learning rates.

2.5.3 Adaptive Moment Estimation (Adam)

Adaptive Moment Estimation (Adam) is an optimizer algorithm that is based on the SGD and RMSProp algorithms. The algorithm computes adaptive learning rate for each parameter based on the average of the recent first and second moment magnitudes of the gradient values for the weights (Kingma & Ba, 2015).

2.5.4 Regularization and Overfitting

Regularization is a technique that reduces overfitting within the model. The effect of regularization is to make the model prefer to train with smaller weights (Brownlee, 2019). Large weights are only used if they improve the cost function. Regularization is a way of compromising on small weights while minimizing the cost function (Brownlee, 2019). Regularization techniques include dropout, L1 regularization, L2 regularization, adam etc.

2.6 Model Analysis

When the model has completed training, it is important to see how well the predictions compare to the real-world values. To make this comparison the Mean Absolute Percentage Error(MAPE) , Mean Average Error(MAE) and Root Mean Square Error(RMSE) will be used. MAPE considers both the error between predicted values and true value, but also the ratio between the true error and predicted values. The MAE is the magnitude of the difference between the true value and the model outcome. RMSE is sensitive to very big and small error results (Tang, et al., 2019). Evaluating the results of the error formulas will show how accurate the model is.

$$RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^N (Y_{t_i} - Y_p)^2}$$

$$MAE = \frac{1}{N} \sum_{i=1}^N |Y_{t_i} - Y_{p_i}|$$

$$MAPE = \frac{1}{N} \sum_{i=1}^N \frac{|Y_{t_i} - Y_{p_i}|}{Y_{t_i}}$$

RMSE	Root Mean Square Error
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
N	Number of training examples
Y _t	True Value
Y _p	Predicted Value
I	index of specific example

3 Methodology

3.1 Data Description

In order to create a model for a specific use case, it is important that the data being used is understood, because it will be easier to extract the useful information to make the model when all aspects of the dataset are understood (Géron, 2017). The data for this research was collected by vehicle detection devices that are located around the Western Cape that captured traffic data.

Table 1: Sample of Data recorded

Region	Device Location	Date	Hour	Class	Volume	Average Speed
WC	DS VDS 111 OB	07/09/2019	14	2	103	82.75
WC	VDS 215 OB	08/09/2019	1	3	2	104.13
WC	DS VDS 111 IB	07/09/2019	14	2	2	86.87
WC	DS VDS 130 IB	10/09/2019	22	1	243	108.36
WC	DS VDS 306 OB	11/09/2019	23	3	9	107.02

The data shown in table 1 is an extract from values captured in September 2019 by vehicle detection devices. It shows various features that can be used for analysis. Brief descriptions of the parameters are listed below:

1. Region – This is the province that the recording device is located in. Each device was placed in the Western Cape(WC) as shown in table 1.
2. Device Location – This is the location of the device when the recordings were taken. To find more detail of where the device is located a separate dataset was made available for use. A section of this dataset is shown in table 2.
 - The recording device was also able to differentiate the vehicle coming in and out of its field of view. This is shown with the IB(In Bound) and OB(Out Bound) suffix added to the location of the device. IB represents the number of vehicles that entered the devices field of view within the hour stated at that location. OB shows the vehicles recorded that left the devices field of view during the hour stated.
3. Date – This is the date the data were captured.
4. Hour – This is the hour of the day that the recordings took place. The values range from 0 to 23 to represent each hour in a day.
5. Class – Class represents the vehicle type that was counted during that specific hour of recording. The vehicles were split based of the length of the vehicle. The length was estimated by the recording device. Class 1 represented the vehicles with the least length e.g. cars. Class 3 represented the longest vehicles e.g. buses, trucks and class 2 were vehicles that fit in-between e.g. minibus taxis.

6. Volume – This represents the traffic volume recorded i.e. the number of vehicles counted.
7. Average Speed – This was the average speed all the vehicles that passed during that specific hour. It is measured in Kilometres per Hour (km/h).

Table 2: Location Data

<u>Select</u>	<u>Asset Code</u>	<u>Asset Desc.</u>	<u>Asset Status</u>	<u>Latitude</u>	<u>Longitude</u>	<u>PO Addr1</u>
1	VDS406-WC	VDS406	Active	-33.9415899	18.4584274	streetlight pole
2	VDS407-WC	VDS407	Active	-33.942175	18.458595	streetlight pole

In summary, each individual location had:

- 3 Classes recorded i.e. 1,2 and 3
- Two separate recordings for the IB and OB values.

Per hour for each day of the month

The example below shows all the values recorded at location DS VDS 104 on the 1st of September 2019 during hour 0 i.e. the first hour after midnight.

Table 3: Sample of points collected at VDS 104

Device Location	Date	Hour	Class	Volume	Speed
DS VDS 104 IB	01/09/19	0	1	477	75.4
DS VDS 104 IB	01/09/19	0	2	4	113.33
DS VDS 104 IB	01/09/19	0	3	3	98.6
DS VDS 104 OB	01/09/19	0	1	2	102.56
DS VDS 104 OB	01/09/19	0	2	1	67.73
DS VDS 104 OB	01/09/19	0	3	599	99.84

3.2 Site Selection

The decision to use time series regression models means that the model will predict future values based on the trends found in a training dataset that is time dependant. Thus only values relevant for forecasting should be used to train the model. This resulted in the model created being trained on values from a single location. This will ensure that a model created from these observed values will forecast future values that are relevant to the location.

To select a point for analysis, the location which had the most consistent recorded data was selected. Thus the point that recorded the most individual counts between the time period being analysed will be used for further analysis. From searching through the data, the device which had the most counts was VDS 109.

VDS 109 is located along the N1, north of Parow North. Located at -33.8875 18.5725 latitude and longitude respectively.



Figure 5: VDS 109 site location

From the available data, the data used to train the model will consist of volume counts recorded between 01/09/2019 and 29/02/2020. This data will be trained by both models and will be used to forecast values recorded during the first week of March 2020.

3.3 Data Cleaning

Data cleaning is the process of identifying incomplete, inaccurate, incorrect, irrelevant or missing data within the dataset (Rahman, 2019). Once these errors have been found the data is then modified, replacing or deleting irrelevant values.

3.3.1 Organising the data

The data when obtained has a lot of information, but it needs to be cleaned in order to create a model that will represent the dataset. The data points which will be used for the models are the total hourly volumes for each location regardless of vehicle class. Each individual location's data was sorted as stated in the steps below:

- The average speed was removed from the dataset as it will not be used for the model.
- The volumes of each class were totalled to represent the total number of vehicles that passed through that location during that specific hour.
- To make sure all the data for each data location represents the total number of vehicles that passes during the stated hour regardless of whether the vehicle is coming in or leaving the point, the sum of the IB and OB values were also taken for the location.

Table 4: Sample Data Calculations

Device Location	Date	Hour	Class	Volume	Class sum	Location sum
DS VDS 104 IB	01/09/19	0	1	477	484	1086
DS VDS 104 IB	01/09/19	0	2	4		
DS VDS 104 IB	01/09/19	0	3	3		
DS VDS 104 OB	01/09/19	0	1	2		
DS VDS 104 OB	01/09/19	0	2	1		
DS VDS 104 OB	01/09/19	0	3	599	602	

Table 4 shows the value which will represent VDS 104's volume recorded on the 1st of September 2019 during hour 0 i.e. 1086.

3.4 Model Analysis

Once the model has run, it is important to test its accuracy and make changes to ensure the models can produce the most accurate predictions.

3.4.1 Model testing

When the model's architecture is designed, the training dataset(80% of the total available dataset) will be placed into the model as its input and the model will the run. To test the accuracy of the model, the model will be made to predict the values within the testing data set(remaining 20% of available values). The MSE, RMSE and MAPE will be used to analyse the result.

3.4.2 Model optimizer selection

To minimize the training error and avoid local minimal points, optimizers are implemented into models (Fu, et al., 2016). All the models that were designed with either Adam or RMSprop algorithms used. Their prediction results compared with the testing dataset was compared and the model with the optimizer that performed better was chosen for forecasting.

3.4.3 Augmenting model architecture

After analysis of the MSE, RMSE and MAPE, if the error obtained is too high, then parameters within the model's architecture were changed until the error was minimised.

In order to get the model which bests fits the data, multiple iteration of the models were created. To find the model that can best fit the data, various aspects of the model can be changed. The following are some of the parameters of the model that can be changed to find the best model.

- Model optimizer i.e. RMSprop or Adam
- Number of model layers
- Type of each hidden layer i.e. Dense/ model(LSTM/GRU)
- Batch size
- Timestep (explained in chapter 5 section 5.1.5)

3.5 Model Forecasting

Once the model is trained, any set of data put input into the model will produce a single output that the model predicts will be the next value in the sequence the data that was input. To produce continuous predicted values the data input into the model needs to keep updating. To accomplish this, an algorithm that placed the predicted output onto the model input data will form a series of outputs that will be interpreted as the forecast data by model.

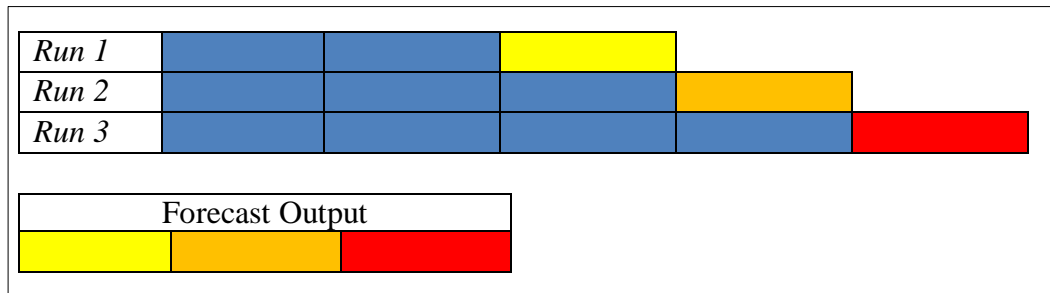


Figure 6: Model forecasting algorithm illustration

Figure 6 is an illustration of the forecasting algorithm. During Run 1, the data being added to the model (blue block) outputs a prediction i.e. yellow block. For the next iteration, Run 2, this output is added to the values being input to the model and now the previous yellow block is blue and the model goes through Run 2 producing another output i.e. orange block. This process will continue until the algorithm reaches the iteration limit set by a user prior to execution. For the illustration the yellow, orange and red values will be the forecasted values.

The trained models were used to forecast a week's worth of data i.e. between 01/03/2020 at [0] hour till 06/03/2020 at hour [23]. The result of this forecast will be compared with the actual values that were recorded by the vehicle detection device. The RMSE, MSE and MAPE formulas will be used to compare the results numerically and a line graph will compare the results visually. The conclusions to the research were based on these results.

The final iteration of the LSTM and GRU model will be compared to see which one produced the most accurate forecasted data. The model that produces the least error and mimics the actual values recorded best will be deemed as the more accurate model and will be recommended.

4 Results

4.1 LSTM

4.1.1 Model architecture

Model: "LSTM model"		
Layer (type)	Output Shape	Param #
=====		
Input_layer (LSTM)	(None, 100, 48)	9600
Hidden_layer_1 (LSTM)	(None, 100, 24)	7008
Hidden_layer_2 (LSTM)	(None, 24)	4704
Dropout (Dropout)	(None, 24)	0
Output_layer (Dense)	(None, 1)	25
=====		
Total params: 21,337		
Trainable params: 21,337		
Non-trainable params: 0		

Figure 7: LTSM model architecture

Table 5: LSTM model addition details

Added details	
Timestep	100
Batch Size	20
Optimizer	Adam
Dropout	20%
Epochs	100

4.1.2 Comparison with Test Data

Table 6: LSTM predictions compared to test data

Comparison with test data		
	Adam	RMSProp
RMS	783.94	708.74
MAE	3309.62	3220.23
MAPE	196.01	194.44

Figure 8 and 11 are snapshots of the transition between training and testing data. The ‘Predictions’ line represent the model output after training with the training dataset and the testing dataset being input into this model. The graphs illustrate the model predictions compared to the testing dataset which was reserved to check the accuracy of the model.

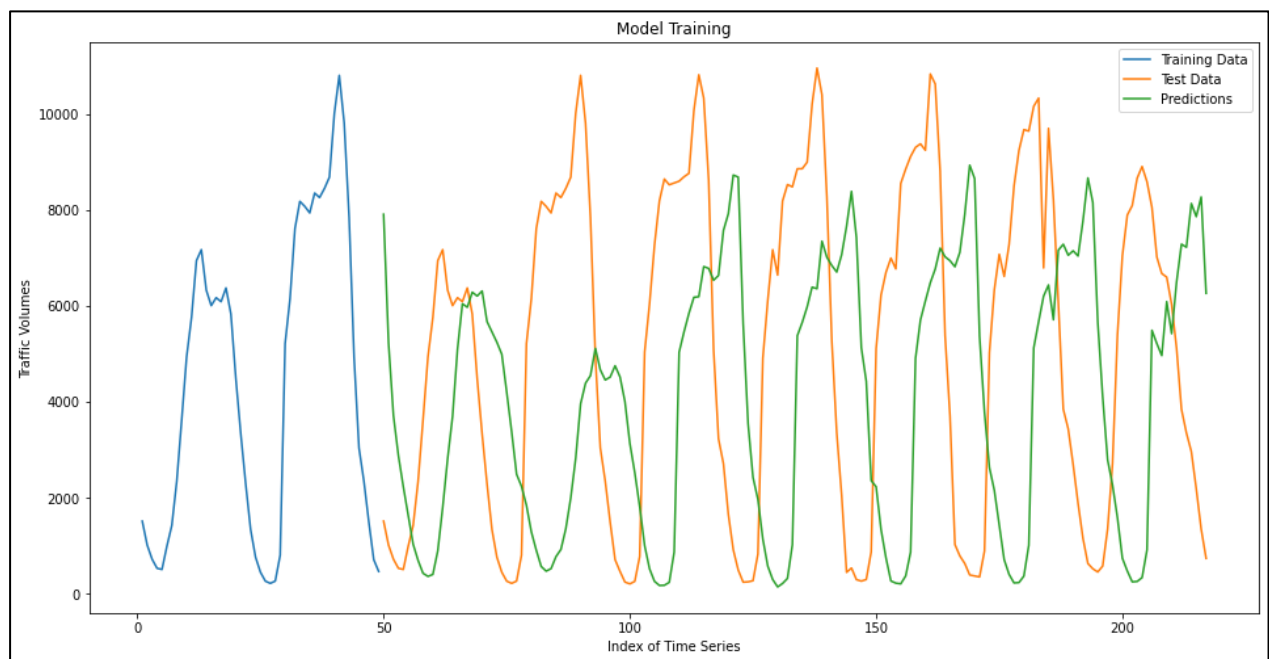


Figure 8: LSTM model prediction compared to test data

4.1.3 Comparison Forecast Data

Table 7: Error produced by LSTM model forecast

One week forecasting error	
RMS	560.8
MAE	952.38
MAPE	33.52

Figure 9 and 12 shows the result of the algorithm used for forecasting future values based of all the training and testing data. The model output(green) is the iterative outputs of the model.

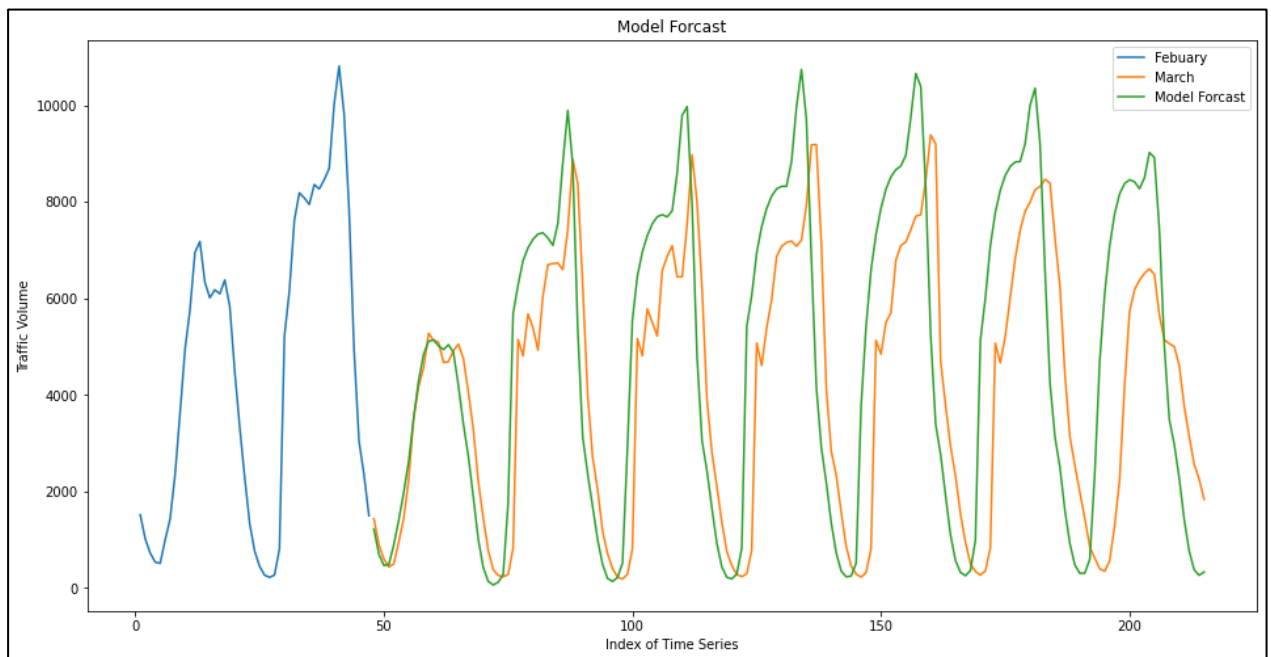


Figure 9: LSTM model forecast

4.2 GRU

4.2.1 Model architecture

Model: "GRU"		
Layer (type)	Output Shape	Param #
=====		
Input_layer (GRU)	(None, 100, 48)	7344
Hidden_layer_1 (GRU)	(None, 100, 24)	5328
Hidden_layer_2 (GRU)	(None, 24)	3600
Dropout (Dropout)	(None, 24)	0
Output_layer (Dense)	(None, 1)	25
=====		
Total params: 16,297		
Trainable params: 16,297		
Non-trainable params: 0		

Figure 10: GRU model architecture

Table 8: GRU model addition details

Added details	
Timestep	100
Batch Size	12
Optimizer	RMSprop
Dropout	20%
Epochs	100

4.2.2 Comparison with Test Data

Table 9: GRU predictions compared to test data

Comparison with test data		
	Adam	RMSProp
RMSE	708.74	572.52
MAE	3220.23	3317.64
MAPE	194.44	204.54

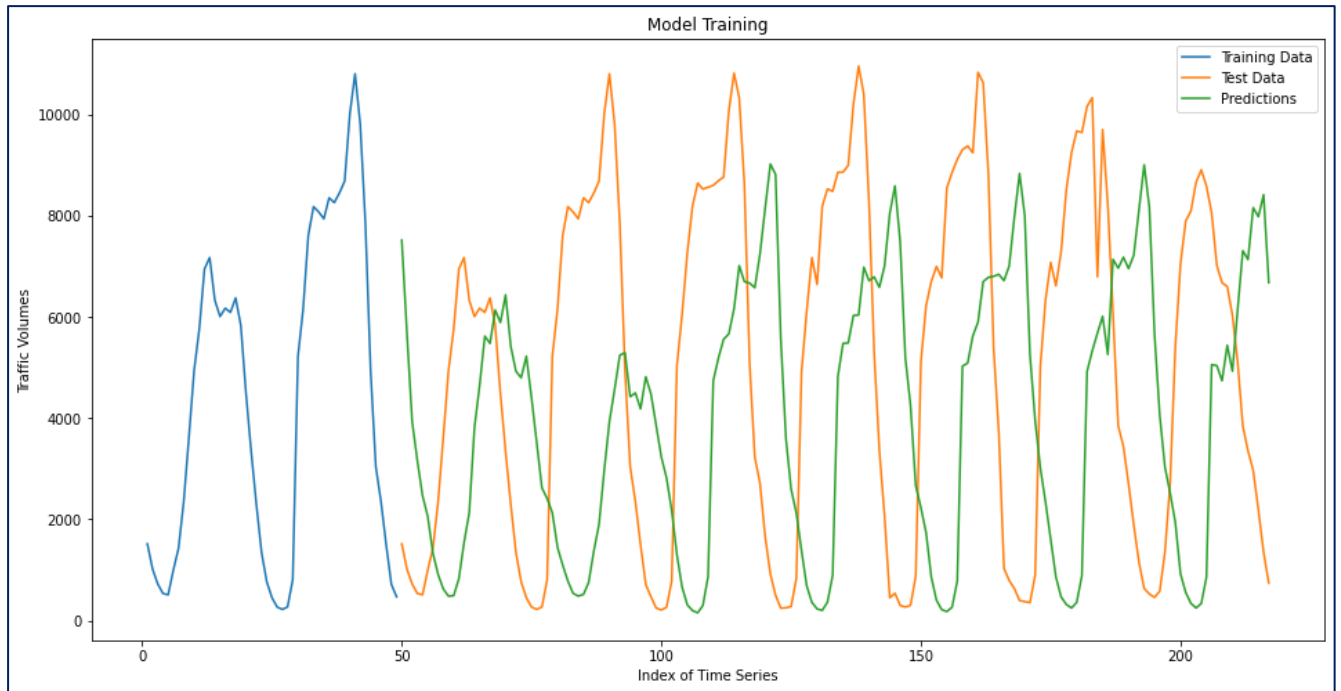


Figure 11: GRU model prediction compared to test data

4.2.3 Data Forecasting

Table 10: Error produced by GRU model forecast

One week forecasting error	
RMS	204.8
MAE	652.9
MAPE	23.4

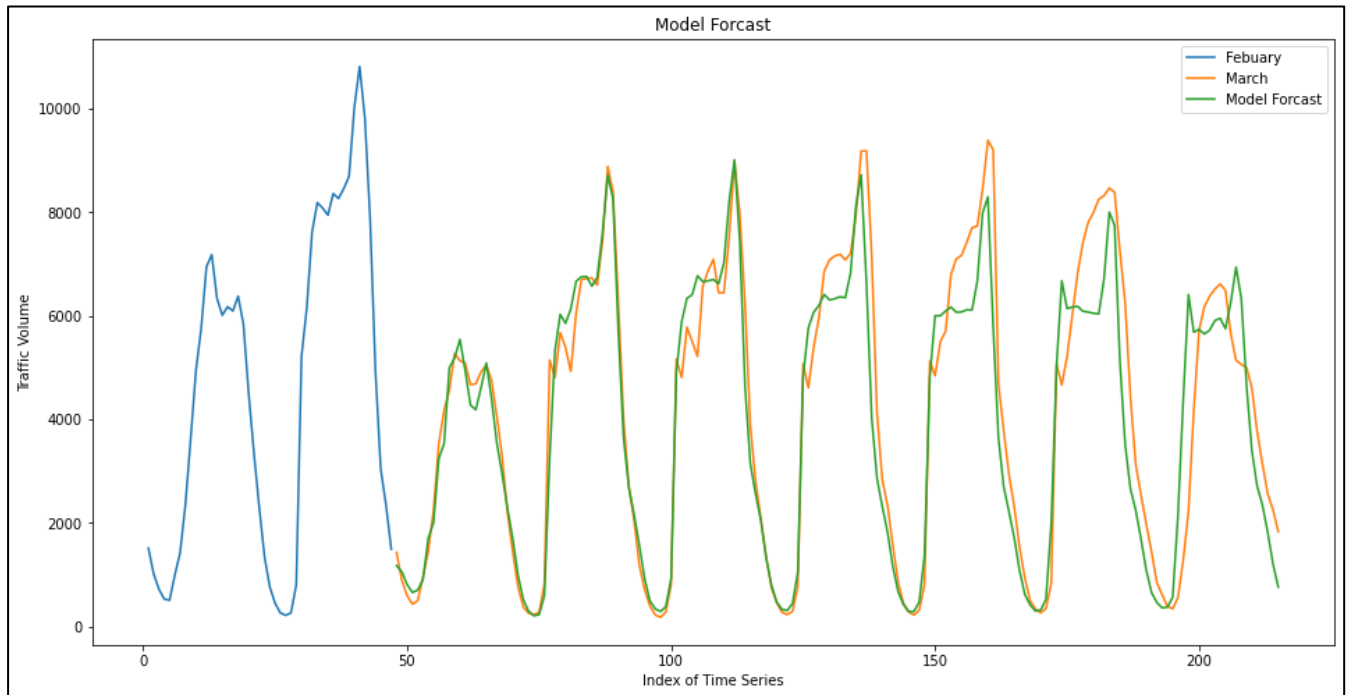


Figure 12: GRU model forecast

4.3 Model Comparison

Table 11: Error comparison between GRU and LSTM model forecasting

One week error		
	LSTM	GRU
RMS	560.8	204.8
MAE	952.38	652.9
MAPE	33.52	23.4

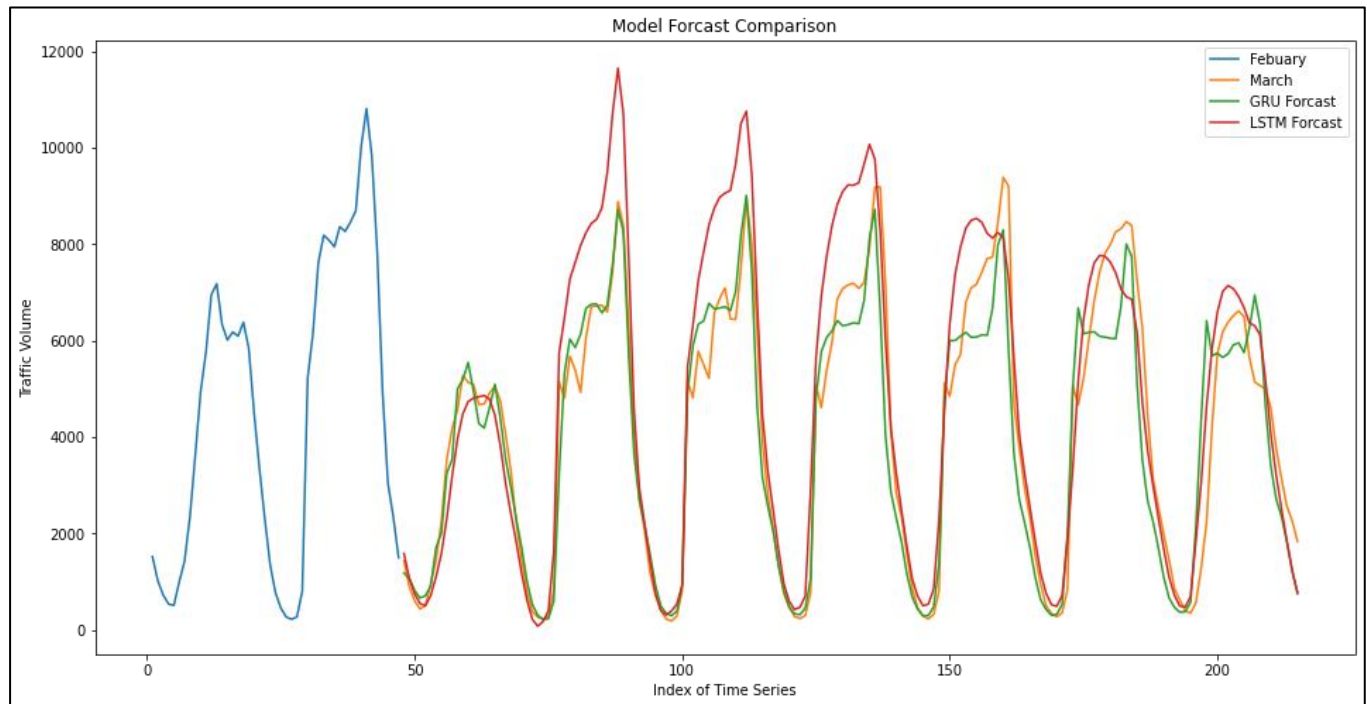


Figure 13: LSTM and GRU forecasting comparison

5 Discussion

5.1 Overall Data composition

5.1.1 Data structure

The only data from the original dataset that was used for the models was the traffic volumes, date and hour. The volumes were arranged in order beforehand to make sure they were in chronological order when being input into the model. This data was placed in a one – dimensional array because the models can only accept inputs in an array format.

5.1.2 Data Scaling

In order to scale the data, the MinMaxScaler algorithm will be used. This algorithm scales each individual data point between a range of values (scikit-learn developers, 2007). The dataset will be scaled between 0 and 1. This will scale the data within an acceptable range. The limit of 1 is normally used for scaling (Vieira, 2017) and no value predicted by the model should be below 0, thus 0 the lower limit.

$$Xi = \frac{X - \min(x)}{\max(x) - \min(x)}$$

X_i	Scaled Value
X	Initial / Unscaled Value
$\min(x)$	The smallest value within the unscaled dataset
$\max(x)$	The largest value within the unscaled dataset

5.1.3 Missing Data

The vehicle detection devices used to record the traffic volumes were able to constantly record traffic volumes, but some data points were missing. Considering the time period for the training data, there were 4341 counts recorded of a possible 4368 counts for device VDS 109. Thus 99.4% of all possible counts were recorded by the detection device. There are some missing values, but were deemed to not be enough to misrepresent the traffic flow trend. The data used to make the model would consist of all the counts recorded within the period used to train the data i.e. 01/09/2019 and 29/02/2020.

$$\text{Max count} = 24 \times (30 + 31 + 30 + 31 + 31 + 29) = 4368 \text{ hourly count}$$

24	Hours each day
30+31+30+31+31+29	Total number of days of month from September[2019] till February[2020]

5.1.4 Train/Test Split

To avoid the model only learning the dataset which was given to it, part of the dataset needs to be offset to be used testing. From dataset, 20% of the data will be set aside for testing. This is the standard percentage for most machine learning models (Géron, 2017). The remaining 80% of the dataset will be used to train the model.

- Test Data – 20% of entire dataset
- Training Data – 80% of the dataset

5.1.5 Timestep implementation

Timestep is data in a sequence that is grouped together to check the accuracy of the model as the model is training (Olah, 2015).

The following is an example of a sequence of data within a section of the training dataset as shown in figure 14. These values are in chronological order.

For the example below, the timestep is 168 values



Figure 14: Timestep data

- Yellow: Timestep

While the model is running it will train with the first 168 values i.e. timestep, it will then use these 168 values to predict the 169th value. It will make a prediction and this will be compared to the actual value i.e. the blue block. The model will continue to do this with the next set of 168 values i.e. timestep. This process will be done with the entire training dataset.

5.2 LSTM model performance

Figure 7 shows the final iteration of the LSTM model architecture. The optimizer which produced the model with the least error when compared to the test values was the adam optimizer. This model was used for forecasting.

When the model was used to forecasting, it was able to mimic the trends observed by the actual recorded data as shown in figure 9. This shows that the model was able to forecast when traffic counts were going to increase or decrease at points throughout the timeline. This resulted in a model that can forecast the time when potential maximum and minimum traffic volumes could occur during the day.

The overall accuracy of the model was satisfactory but from figure 9, it is clear that the model's values were generally larger than the true values. This could be because the method used to scale the data was unsuitable. A reduced scaling limit could yield better results e.g. 0.8 instead of 1.

With the forecasted data being able to mimic the trends of the actual recorded values and it producing volume counts that were comparable to the actual data captured, the LSTM model's overall performance shows that it could be used for traffic volume forecasting.

5.3 GRU model performance

After several iterations, figure 10 shows the GRU model architecture that produced the most accurate results. The optimizer used was the RMSprop since it outperformed the adam optimizer as shown in table 9. This model subsequently used for forecasting.

As shown in figure 12, the overall ability for the model to mimic the trends of the data was excellent. It was nearly able to predict when the maximum and minimum values would occur.

This model could be used to predict when the traffic will have its highest and lowest traffic flows because it was able to forecast when points throughout the day would have the most or least traffic.

The model was able to forecast accurate values. From figure 12, it is evident that the model was able to produce results that were close to the actual values observed by the vehicle detection device. This is confirmed by the relatively small error produced by the model as shown in table 10.

The GRU model was able to forecast values that both mimicked the trends of the observed values and produce results that were accurate. Its ability to do both at a high level shows that this model can be used for traffic volume forecasting.

5.4 Model Comparison

Overall the GRU model performed better than the LSTM model. It produces smaller errors when comparing the results to test data and forecasted more accurate values. This could be because the model is able to process data faster than an LSTM model (Described in section 2.3.2). It was able to go through the dataset faster when training results in it learning the data trends better because over the same number of epochs the GRU model can learn more. This produced a model that was able to forecast data better than the LSTM model.

6 Conclusion

This research project was undertaken to ascertain the potential use of Neural Networks for traffic volumes predictions. Due to the availability of vehicle detection devices the ability to access accurate traffic data is possible. An improved way of forecasting can make traffic planning more accurate resulting in better long term city planning.

In an attempt to improve forecasting, Neural Networks were used because of their ability to find complex patterns within a dataset. Within a large enough dataset, a Neural Network model will be able to find and learn the trends within said dataset and predict values when a new dataset is input into the model after training. Traffic data is time sensitive, thus when making predictions, the outputs will have to follow an hourly format thus a variation of Neural Networks that can predict data for time sensitive data was implemented. The models were based of Recurrent Neural Networks, with an LSTM and GRU model being implemented.

When forecasting Neural Networks, a single output value based on the patterns learnt from the data input into it is produced. Thus any point that is not directly relevant to the point of interest will skew the model towards a wrong result. When forecasting data for a site only its previous values can be used to train a model to forecast future traffic volumes for that location. To give a model enough data to learn patterns the location that had the most values recorded between 01/09/2019 and 29/02/2019 was used as input to train the models. The models accuracy was determined by the error produced by the model. The error is the difference between the values recorded by the vehicle detection device and the model's output.

After analysis, the GRU model performed better than the LSTM model. As discussed in section 2.3.2, this is probably because of its ability to process data faster than an LSTM model resulting in the model being able to learn the patterns within the training data more accurately.

In conclusion the objectives detailed in Chapter 1 section 1.3 have been achieved. The model was not able to cover the entire Western Cape area because a time dependency model can only use relevant values, but it was still able to use values recorded within the Western Cape. This shows that there is potential for wide scale use of Neural Networks for traffic volume forecasting not only for the Western Cape, but potentially any place that has a record of traffic volume data.

This research can be used as a point of reference for future Machine Learning projects within the transportation field. With accurate traffic predictions, better traffic planning can be done and better traffic systems can be designed to better suit the population.

7 Recommendations

The recommendations for this research project will be split between data collection, model selection and implementation suggestions.

7.1 Data collection

- Increase the number of counting devices. With the increase of the number of counting devices, more data will be available for the model to train with. More data for the models to train on will result in more accurate models being produced.
- The data should be collected more frequently. If the counting device could record data at more regular intervals i.e. every 30 minutes, the data collected will more accurately represent the vehicle counts at that location. An example of this use would be analysis for peak hour values. To better represent data during this time, counts every fifteen minutes during peak hours can be used to make models that better forecast the peak hour traffic volumes.
- Improve the vehicle tracking capabilities of the vehicle detection devices. An improved vehicle detection system will allow for models that can forecast counts for a specific vehicles or vehicle types. An example of this would be a device that can differentiate the volume counts for minibus taxis, this would allow for models that predict the number of minibus taxis that will be at a specific spot at a given time. This model can be used for planning public transportation systems.

7.2 Model selection

- Use different models to observe how they will perform with the given data. Examples of possible models are Auto Regressive Integrated Moving Average (ARIMA), decision tress, xG boost algorithm, regression algorithm etc.
- More research should be done on the viability of the use of ensemble modelling. Ensemble modelling is the process of using different model combinations to predict an outcome (Kotu & Bala, 2015).An example would be combing an LSTM and regression model to make a single prediction.
- It is recommended that future research be done on creating a model that can forecast the average speed of vehicles within the Western Cape.

7.3 Potential implementations

- This is the first step in implementing Machine Learning in transportation systems within the Western Cape. Future studies need to be done to try assess the potential implementation of Machine Learning models that work with real time data. Data collection devises would be recording data near traffic lights and collecting the data to implement into a model that changes the traffic signals to better manage traffic loads.
- Additional research into the potential implementation of Reinforcement Learning within traffic forecasting should be done.

8 Bibliography

Brownlee, J., 2017. Gentle Introduction to the Adam Optimization Algorithm for Deep Learning. *Machine Learning Mastery*.

Brownlee, J., 2019. *Machine Learning Mastery*. [Online] Available at: <https://machinelearningmastery.com/how-to-improve-neural-network-stability-and-modeling-performance-with-data-scaling/#:~:text=Data%20scaling%20is%20a%20recommended,value%20input%20and%20output%20variables>. [Accessed 17 August 2020].

Burkov, A., 2019. In: *The Hundred-Page Machine Learning Book* . Quebec: Andriy Burkov.

Changxi, M., Guowen, D. & Xuecai, X. D., 2019. Short-Term Traffic Flow Prediction Method. *IEEE Access* , Volume 7, pp. 143025-143033.

Department of Health, Republic of South Afrika, 2020. *sacoronavirus.co.za*. [Online] Available at: <https://sacoronavirus.co.za/2020/08/17/update-on-covid-19-17th-august-2020/> [Accessed 18 August 2020].

Fu, R., Zhang, Z. & Li, L., 2016. Using LSTM and GRU Neural Network Methods for Traffic Flow Prediction. p. 2.

Géron, A., 2017. *Hands-On Machine Learning with Scikit-Learn and TensorFlow*. 1st ed. United States of America: O'Reilly Media.

Google Developers , 2020. *developers.google .com*. [Online] Available at: <https://developers.google.com/machine-learning/glossary#m> [Accessed 17 August 2020].

Hardesty , L., 2017. *news.mit.edu*. [Online] Available at: <https://news.mit.edu/2017/explained-neural-networks-deep-learning-0414> [Accessed 17 October 2020].

Hinton, G., Srivastava, N. & Swersky, K., 2018. Overview of mini-batch gradient descent. In: s.l.:Coursera .

Hochreiter, S., Bengio, Y., Frasconi, P. & Schmidhuber, J., 2009. Gradient Flow in Recurrent Nets: the. In: Munchen, Germany: s.n.

Howard, J. & Gugger, S., 2020. In: *Deep Learning for Coders with Fastai and PyTorch: AI Applications Without a PhD*. California: O'Reilly Media .

Karami, Z. & Kashef, R., 2020. Smart transportation planning: Data, models, and algorithms. *Transportation Engineering 2*.

Kingma, D. & Ba, L. J., 2015. *ADAM: A method for stochastic optimization*. Toronto: International Conference on Learning Representations.

Kotu, V. & Bala, D., 2015. *Predictive Analytics and Data Mining*. Santa Clara: s.n.

Luo, X., Li, D., Yang, Y. & Zhang, S., 2019. Spatiotemporal Traffic Flow Prediction with KNN and LSTM. *Journal of Advanced Transportation*, Volume 2019, pp. 2-7.

MathWorks, 2020. *Describe relationships and make predictions from time series data*. [Online]

Available at: <https://www.mathworks.com/discovery/time-series-regression.html#:~:text=Time%20series%20regression%20is%20a,of%20dynamics%20from%20relevant%20predictors.&text=Time%20series%20regression%20is%20commonly,%2C%20financial%2C%20and%20biological%20systems>.
[Accessed 2 December 2020].

McGonagle, J., García, J. A. & Mollick, S., 2020. <https://brilliant.org/>. [Online]
Available at: <https://brilliant.org/wiki/feedforward-neural-networks/>
[Accessed 29 November 2020].

Nielsen, M., 2015. Neural Networks and Deep Learning. In: *Neural Networks and Deep Learning*. s.l.:Online.

Olah, C., 2015. colah.github.io/. [Online]
Available at: <http://colah.github.io/posts/2015-08-Understanding-LSTMs/>
[Accessed 15 October 2020].

Rahman, A. U., 2019. towardsdatascience.com. [Online]
Available at: <https://towardsdatascience.com/what-is-data-cleaning-how-to-process-data-for-analytics-and-machine-learning-modeling-c2afcf4fbf45#:~:text=Data%20Cleaning%20means%20the%20process,of%20the%20basic%20data%20science>.
[Accessed 21 October 2020].

Ruder, S., 2016. *An overview of gradient descent optimization algorithms*. [Online]
Available at: <https://ml-cheatsheet.readthedocs.io/en>
[Accessed 18 October 2020].

scikit-learn developers, 2007. scikit-learn.org. [Online]
Available at: <https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html>
[Accessed 25 September 2020].

Srivastava, N. et al., 2014. Dropout: A Simple Way to Prevent Neural Networks from. *Journal of Machine Learning Research*, Volume 15, pp. 2-8.

Statistics South Africa , 2018. *General Household Survey*, Pretoria : Statistics South Africa .

Tang, J., Li , L., Hu, Z. & Liu, F., 2019. Short-Term Traffic Flow Prediction Considering Spatio-Temporal Correlation. *IEEE Access*, Volume 7, pp. 101011 - 101016.

Vieira, S., 2017. www.researchgate.net. [Online]
Available at: https://www.researchgate.net/figure/a-The-building-block-of-deep-neural-networks-artificial-neuron-or-node-Each-input-x_f
[Accessed 15 August 2020].

Wikipedia, 2020.
Available at:
[Accessed 18 August 2020].

Western Cape. [Online]
https://en.wikipedia.org/wiki/Western_Cape

Appendix A: Exit Level Outcomes (ELO)

1. Problem solving

An improvement in the ability to accurately forecast traffic volumes make planning future traffic management systems easier and can be used in the context of city planning. For this research neural networks were used to assess their viability in the context of traffic analysis. The objective of this research was to find out how accurate a neural network would be if it had to forecast traffic volumes.

2. Investigations, experiments and data analysis

To find out if neural networks can be used for traffic volume forecasting, an initial literature review had to be conducted to determine if neural networks could be used for this context. Access to accurate and reliable data had to be made available for use because without it an accurate model cannot be created. After concluding that neural networks can be used in this context based on the literature review, the best model had to be selected from the entire neural network framework. With reliable data and models that could be used for forecasting, two separate models were created and their ability to predict traffic volume data were compared.

3. Professional and technical communication

- A detailed research report was written to explain the entire research process. This report contained explanations of critical concepts and detailed descriptions of how the research was conducted.
- A brief abstract is available within the report that gives an overview of the entire research project.
- To better explain some of the concepts that are critical when trying to understand neural networks, a series of tables, flowcharts and diagrams were created to make understanding these concepts easier.
- A poster was created to illustrate the project in a way that is easy to understand for the general public.

4. Individual, team and multidisciplinary working

As of writing this research project I am not in South Africa. In order to communicate with my supervisors, meetings on Microsoft Teams were conducted occasionally throughout the research process. Communication via email was maintained throughout the project.

5. Independent learning ability

Three months before the start of this project I used online platforms like YouTube, LinkedIn Learning and Google Colab to learn how to write machine learning code using python. The lessons learnt during this time were instrumental in my ability to code neural networks. The use of neural networks was a major part of this research project.

Appendix B: Code Used

[https://colab.research.google.com/drive/16l7KdQ_9moZHlrTmimaMtlJoJp0bKC9W#printM](https://colab.research.google.com/drive/16l7KdQ_9moZHlrTmimaMtlJoJp0bKC9W#printMode=true)
ode=true 1/9

#Import relevant python libraries

```
import tensorflow as tf
import pandas as pd
from pandas import ExcelWriter
from pandas import ExcelFile
import numpy as np
from tensorflow import keras
from sklearn.preprocessing import MinMaxScaler
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, LSTM, GRU
from tensorflow.keras.preprocessing.sequence import TimeseriesGenerator
from tensorflow.keras.models import load_model
import matplotlib.pyplot as plt
```

#Obtain the data

```
data= pd.read_csv('Combined_109.csv')
```

#Get the volume data from the dataset

```
target = data[['Count']]
training_data = target.values
```

#scaling data

```
scaler = MinMaxScaler(feature_range=(0,1))
scaler.fit(training_data)
scaled_data = scaler.transform(training_data)
```

#Get the amount of data that will be used as the training dataset

```
train_len= int(len(scaled_data) * 0.8)
```

#Creating the training dataset

```
split_data = scaled_data[0:train_len]
x_train = [ ]
y_train = [ ]
time_step = 100
for i in range(time_step, len(split_data)):
    x_train.append(split_data[i-time_step:i,0]) #The first timestep values go into
    y_train.append(split_data[i,0])
```

#Convert the values to a 1-dimentional array

```
x_train = np.array(x_train)
y_train = np.array(y_train)
x_train = np.reshape(x_train,(x_train.shape[0],x_train.shape[1],1))
```

#Build the model

```
model = Sequential()
model.add(LSTM(48,input_shape=(x_train.shape[1],1),return_sequences=True))
model.add(LSTM(24,activation='relu', return_sequences=True, name = 'Hidden_layer_1'))
model.add(LSTM(24, name = 'Hidden_layer_2'))
model.add(keras.layers.Dropout(0.2, name = 'Dropout'))
model.add(Dense(1, name= 'Output_layer'))
```

#Compile the model

```
model.compile(optimizer = 'RMSprop', loss = 'mae' )
```

#Run the model

```
model.fit(x_train , y_train, batch_size=20 , epochs= 100, verbose = 1)
```

Testing the data

#Obtain the Testing dataset

```
test_data = scaled_data[: -train_len]
test_data = np.array(test_data)
test_data = np.reshape(test_data,(test_data.shape[0],test_data.shape[1],1))
true_test = target[: -train_len]
true_train = target[train_len:]
```

#Format the Test data

```
test_data = scaled_data[train_len-time_step,: ]
x_test = [ ]
y_test = target[: -train_len]
for i in range(time_step,len(test_data)):
    x_test.append(test_data[i-time_step:i,0])
```

#Convert the data into a 1-dimentional array

```
x_test = np.array(x_test)
x_test = np.reshape(x_test,(x_test.shape[0],x_test.shape[1],1))
```

#Get predictions based of the test dataset being entered into the model

```
predictions = model.predict(x_test)
```

#Convert from scaled to unscaled data

```
predictions = scaler.inverse_transform(predictions)
```

#Get the error

```
true_test = y_test
error = true_test - predictions
```

#RMSE

```
print("Root mean Squared")
rmse = np.sqrt(np.mean(error)**2)
print('{} '.format(rmse))
```

#MSE

```
print('Mean Absolue Error')
MAE = np.mean(np.abs(error))
print('{} '.format(MAE))
```

#MAPE

```
print('Mean Absolue Percentage Error')
MAPE = np.mean(np.abs((error)/true_test)) * 100
print('{} %'.format(MAPE))
```

Forecasting the data

#Get entire scaled dataset

```
val_data = scaled_data
val_data = np.array(val_data)
```

#Forecasting algorithm

```
new_output = [ ]
i = 0 # initialise the values
#test data should be np.array(sscaled_data)
np_list = val_data # taking the testing data in the shape [2000,1,1]
while i<350: #Set a limit
    y_input = np.reshape(np_list,(1,np_list.shape[0],1)) # Reshape the allow output
    y_out = model.predict(y_input, verbose=0) # predict
    y_np = np.array(y_out) #Convert the sinle output to numpy array
    np_list = np.append(np_list , y_np) # Add it to the end of the list
    new_output.extend(y_out.tolist()) # Add the output to a list
    i=i+1 #Increments
```

#Get all of the forecasted values

```
test_pred= scaler.inverse_transform(new_output)
```

Test the forecasted values

#Get the actual values recorded by the vehicle detection device during March 2020

```
march_109 = pd.read_csv('March_109.csv')
march_count = march_109[['Count']]
```

#Plot the forecasted values and the true values to compare them visually

```
plt.plot( range(len(test_pred[:168])),test_pred[:168], label = 'GRU Model Prediction')
plt.plot( range(len(march_count[:168])),march_count[:168], label = 'Actual Recorded Values')
plt.legend()
plt.show()
```


#Use formulas to compare the results numerically

```
predictions = test_pred[:168]  
true_test = march_count[:168]
```

#Getting the error

```
error = true_test - predictions
```

#RMSE

```
print("Root mean Squared")  
rmse = np.sqrt(np.mean(error)**2)  
print('{}' .format(rmse))
```

#MAE

```
print('Mean Absolue Error')  
MAE = np.mean(np.abs(error))  
print('{}' .format(MAE))
```

#MAPE

```
print('Mean Absolue Percentage Error')  
MAPE = np.mean(np.abs((error)/true_test)) * 100  
print('{}%' .format(MAPE))
```

ETHICS APPLICATION FORM



Please Note:


Any person planning to undertake research in the Faculty of Engineering and the Built Environment (EBE) at the University of Cape Town is required to complete this form **before** collecting or analysing data. The objective of submitting this application *prior* to embarking on research is to ensure that the highest ethical standards in research, conducted under the auspices of the EBE Faculty, are met. Please ensure that you have read, and understood the **EBE Ethics in Research Handbook** (available from the UCT EBE, Research Ethics website) prior to completing this application form: <http://www.ebe.uct.ac.za/ebe/research/ethics1>

APPLICANT'S DETAILS		
Name of principal researcher, student or external applicant		Norbert Matilya
Department		Civil Engineering
Preferred email address of applicant:		MTLNOR002@myuct.ac.za
If Student	Your Degree: e.g., MSc, PhD, etc.	BSc Civil Engineering
	Credit Value of 60/120/180/360 etc.	48
	Name of Supervisor (if supervised):	Prof. Dr. Mark Zuidgeest
If this is a research contract, indicate the source of funding/sponsorship		
Project Title		Using Neural Networks to forecast traffic volumes

I hereby undertake to carry out my research in such a way that:

- there is no apparent legal objection to the nature or the method of research; and
- the research will not compromise staff or students or the other responsibilities of the University;
- the stated objective will be achieved, and the findings will have a high degree of validity;
- limitations and alternative interpretations will be considered;
- the findings could be subject to peer review and publicly available; and
- I will comply with the conventions of copyright and avoid any practice that would constitute plagiarism.

APPLICATION BY	Full name	Signature	Date
Principal Researcher/	Norbert Matilya		20/08/2020
Student/External applicant			
SUPPORTED BY	Full name	Signature	Date
Supervisor (where applicable)	MHP ZUIDGEEST		20/08/2020

APPROVED BY	Full name	Signature	Date
HOD (or delegated nominee) Final authority for all applicants who have answered NO to all questions in Section 1; and for all Undergraduate research (Including Honours).	DAVID IKUMI		02/09/2020
Chair: Faculty EIR Committee For applicants other than undergraduate students who have answered YES to any of the questions in Section 1.			