Short-Term Traffic Forecasting for a Smart Satellite Communications System

by

Damon E. Jones

M.S. Operations Research
Southern Methodist University, 2012

B.S. Operations Research
United States Air Force Academy, 2008

SUBMITTED TO THE SYSTEM DESIGN AND MANAGEMENT PROGRAM IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE IN ENGINEERING AND MANAGEMENT
AT THE
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 2020

Signature of Author____
Damon E. Jones
MIT System and Design Management Program
May 24, 2019

Certified by____
Prof. Edward F. Crawley
Professor, Aeronautics and Astronautics
Ford Professor of Engineering
Thesis Supervisor

Accepted by____
Joan Rubin
Executive Director, System Design and Management Program

This page is intentionally left blank

Short-Term Traffic Forecasting for a Smart Satellite Communications System

by

Damon E. Jones

Submitted to the System Design and Management Program in Partial Fulfillment of the Requirements for the Degree of Master of Science in Engineering and Management

Abstract

Satellite communications systems are undergoing a modernization to efficient capacity allocation from a traditional "bent pipe" or static allocation. One challenge to address with a more precise usage of satellite resources is the change in user terminal traffic during a complete cycle of the system: collecting data, generating a constellation setting solution, transmitting the new solution to each satellite and executing changes to the satellite's parameters. As the system's cycle time grows the user's desired data rate changes causing an optimized solution based on an erroneous traffic model. This thesis proposes a comparison of single user models using a gradient boosting algorithm, and a multi-user model using Long-Short Term Memory neural networks (LSTM) or Gated Recurrent Unit neural networks (GRU) to forecast terminal traffic. Each algorithm was tuned using a two-stage design of experiments process consisting of a fractional screening design to identify impactful hyper-parameters and a central composite design to find optimal model settings. During a holdout period, the mean absolute percentage error using a 15-minute lag was 10.7% with a standard deviation of 2.6% over a month of forecasting. Networks using a GRU layer and tuned with Random Search had the best average performance with an error of 9.6% and standard deviation of 2.6%, outperforming the best found XG Boost models with an error of 9.9% and standard deviation of 3.5%.

Thesis Supervisor: Edward F. Crawley
Professor of Aeronautics and Astronautics
Ford Professor of Engineering

This page is intentionally left blank

Acknowledgements

First and foremost, I want to thank my wife Megan and daughter Isla. You were willing to leave our great life in Charlotte and embrace an uncertain one in Boston. I'm sure we will talk about Westgate for the rest of our lives. Without your unceasing support this entire adventure to MIT wouldn't be possible and I'll find a way to make it up to you.

Thank you to my parents, Katy and Mike, for always pushing me to learn and pursue my dreams. You've always did everything you could to make sure that I had these amazing opportunities. I hope I can copy you guys and give Isla the same level of support.

Thank you to Markus and Juanjo for bringing me into such a high functioning team in the lab. I learned more from the three of you than any coursework. I know you'll go on to do amazing things and revolutionize your industry.

Finally, thank you to Ed and Bruce for bringing me into this project and advising me along the way. You've created an environment that pushes learning and experimentation with the safety net of your expertise that I am sad to leave.

This page is intentionally left blank

TABLE OF CONTENTS

List of Figures

List of Tables

1. Introduction

This thesis demonstrates the performance of two machine learning algorithms, as well as using Design of Experiments and Random Search for parameter tuning to produce short term traffic forecasting for flexible, autonomous communications satellites.

1.1. Static Allocation for Satellites

The world is growing increasingly more connected and the need for internet is rapidly spreading across the globe. According to Statista, today there are 3.5 billion smartphone users or roughly 45% of the total population, and the number of users has grown by 40% since 2016 [1]. Facebook alone reports 2.5 billion monthly users in Q4 2019 [2]. In the U.S., the % of adults who do not use the internet declined from 48% in 2000 to just 10% in 2019 [3]. Data usage is also increasing per person in the U.S. For example, a report from iGR determined that over 95% of the 190 GB per month in U.S. home traffic is video [4]. Streaming video is available in higher and higher resolutions which require more and more data; an estimate of Youtube data usage ranges from 90 Mb per hour for 144p to 2.7 GB per hour for 720p and up to 23 GB per hour for 4k resolution [5].

The demand for internet traffic, defined by million exabytes per month, is projected to grow exponentially from 33.45 to 98.64 in 2023 [4]. These forecasts do not include the impacts of COVID-19 where the CEO of MainStreaming said "Telecom Italia has seen a 90% increase in traffic alone since the lockdown was put into place, driven by people trying to work from home" [5].

While traditionally the majority of demand was satisfied through cable, DSL and now fiber, in 2009 10% of home internet users connect through a satellite [6]. Due to the increasing demand for data hungry applications (video streaming for TV & conferences and file sharing) the stage is set to deliver service to people who are unable to utilize terrestrial based internet providers.In the U.S. roughly 20% of the population live in "rural" counties [4] and are less likely to have access to the broadband infrastructure that exists in urban communities. fast and affordable satellite broadband internet is ideal to service these types of customers. In addition to populations without traditional internet access, SES also provides satellite broadband connections to VIP and Commercial aircraft, cruise ships and yachts and multiple government entities that are uniquely denied access to a standard connection [5]. Morgan Stanley sized the 2020 global market for satellite consumer broadband at $7.3 billion. However with more people demanding a broadband connection in locations that are traditionally unserved, Morgan Stanley estimates that by 2040 the market will grow to $94.8 billion [6].

The satellite communications industry is responding to demand by first launching more satellites into orbit [12] and improving existing technology. Guerster's analysis touches on the technological improvements in the satcpm industry "such as digital communication payloads, advanced modulation, multi-beam antennas, and advanced manufacturing are used by the new generation of communication satellites" that improve overall performance and reduce access costs. Guerster suggests that bent-pipe satellites, with a fixed spatial distribution of capacity and static resource allocation, will not accurately meet a changing demand and the market will shift to newer satellites using "flexible communication payloads in the form of digital processors, multi-beam antennas, and flexible amplifiers capable of adapting to changing demand" [7].

1.2. Dynamic Resource Allocation in Other Industries

Other industries moved away from the bent-pipe style of static allocation; in companies like AirBnB, Uber or Tesla, users receive and pay for their exact usage. Longer term rentals of vehicles or homes leaves an asset unused for a large duration of time while the user is still financially responsible for any unused capacity. AirBnB allows a homeowner to lease out their home if they leave on a long vacation or have an empty unit in their home. With personally owned vehicles there is also a large amount of unused capacity. The owner of the car will use it to commute to work, go to the store or visit a restaurant but while not in use, the car sits idle in a parking lot. Uber allows car owners to offer taxi style rides, whenever they want. This reduces the usage downtime for car owners and reduces the need to own a car for people who only want to be a rider. Recently, Tesla announced a plan to allow owners to add their vehicles to a ride-sharing app of robotaxis [8]. As a robotaxi, the car enters autopilot mode and rents itself out in an Uber like fashion but without a driver. Instead of sitting idle, the car would be in use and generating revenue. In all three examples user's demand for these assets are cyclic with periods of high usage and periods of idle demand.

1.3. Flexible Satellite System

One approach to building communications capacity uses massive, high throughput GEO satellites like Echostar [15] and Viatsat [16]. OneWeb [17] and SpaceX [18] are two of the companies focused on low latency LEO constellations. SES is moving towards more dynamic allocation and resource optimization in order to service an increasing demand [10]. The static allocation of satellite beams and power is the equivalent of a long-term rental where a user sends and receives data at up to a specified rate at any time. For most users, who have unique peak usage times and lulls, this is an extremely inefficient use of resources.

1.3.1. O3B mPower

SES launched a new generation of communications satellites to address this shortfall and improve the efficiency to meet customer's demand [9]. SES's O3b mPower satellites have the capability to provide "flexible high-bandwidth, low-latency connectivity" to markets without traditional internet access. The MEO constellation will feature 7 satellites with electronically controllable phased array antennae. A phased array "is two or more antennas used together to provide some desired characteristic or feature not available with a single anenna" [20].Each satellite can rapidly change its configuration electronically to maximize the number of customers served their full demand by using the minimum required resources.. The frequency of each beam in the phased array will shift along the spectrum to minimized interference and provide higher data rates to high demand customers. The shape of each beam will change to group customers together efficiently in a single beam. If a beam is not required it can be turned off until the user demand is enough for it to be operational. When the user demand increases beyond what a single beam can provide through power or frequency changes, then another beam can move to that location and service the high demand customer. All of these changes can happen in almost real time.

1.3.2. Dynamic Resource Manager

Currently changes to power and frequency are manually generated and updated infrequently with changes to beam shape and placement occurring even less often. While the hardware for a dynamic system is in orbit, the software and supporting system lags behind. The work of Guerster and Luis [7] proposes the creation of a dynamic resource management (DRM) system which can implement the flexibility of the O3b mPower satellites. A proposed system architecture for the DRM is shown in Figure 1-1. Each satellite in the constellation reports its data rate usage for each terminal to a Centralized Controller. This controller consolidates the usage rates across the constellation and sends the data to a Demand Estimator (DE). The DE produces forecasted traffic for each terminal and sends the forecasted traffic model to the Real Time Engine (RTE). The RTE generates an optimal resource allocation that attempts to minimize unmet terminal demand and total power consumption [10]. Further work is currently underway to include beam frequency, beam shaping and beam placement. The optimal power, frequency, shape and beam placements are then sent back to the Central Controller and communicated to each satellite, which then changes configuration.

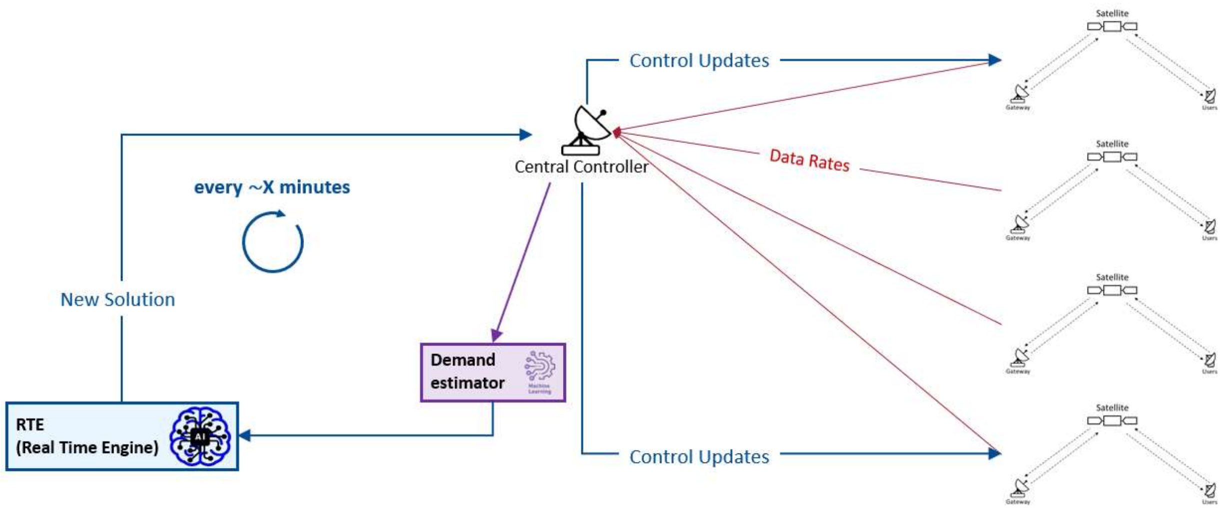


Figure 1 Dynamic Resource Management Architecture

1.3.3. System Cycle Rate

The cycle period of the DRM or solution delay is a critical system parameter that determines how often satellites must reconfigure, the number of minutes ahead the DE must forecast and the frequency of produced solutions from the RTE. The main contributors that increase the solution delay are the communication and compilation time of the Central Controller, prediction time of the DE models, the computation time required by the RTE to generate a new solution, the frequency of control messages sent to the satellites and the required time to reconfigure the constellation. Work by Garau-Luis [10] on a single GEO satellite, using only changes to power and frequency, show a significant reduction in power usage and zero unmet demand by a genetic algorithm generated solution within 100 seconds. Additional satellites and inclusion of beam size and placement will increase the computing time.

A faster cycle rate leads to a system that reacts to changes in traffic quicker, but also leaves less time for solution computation and consumes additional ground resources as more control messages are sent and time is spent reconfiguring. A longer cycle rate allows for a greater difference between current and future traffic. This means that the RTE computes a solution on an outdated traffic model which can lead to an overallocation of resources or unmet demand for the terminal. This thesis assumes a cycle rate of 15 minutes or 4 updates per hour. Additional cycle rates should be studied when the timing of the system is known.

1.3.4. Demand Estimator

Given a 15-minute solution delay a demand estimator needs to generate 15-minute ahead predictions that feed into the RTE. This work assumes that historical traffic for each terminal be compiled and available to generate up to date forecasts. It does not address RTE accuracy, timing or solution implementation. The DE trains a traffic forecast model using best found parameters on historical data for a terminal or set of terminals every 24 hours. After training completes, the model generates 15-minute ahead predictions using the most recent terminal usage available. Traffic forecasts generated are sent to the RTE which generates a global solution to efficiently meet the forecasted traffic. Each model actively predicts terminal traffic for a 24-hour period until a new model is trained and implemented.

1.4. General Objectives

This thesis has two main objectives: 1) to explore the impact of traffic patterns on DRM performance and 2) to demonstrate the potential DRM performance improvement by forecasting the traffic patterns. Without incorporating the DE into the system, any change in a user’s data rate over the 15-minute solution delay becomes either an overallocation of resources or unmet demand. Inclusion of a DE with better accuracy than a lagged data rate will improve the allocation performance of the DRM.

2. Literature Review

2.1. Traffic Forecasting

Demand & traffic estimation and the forecasting of time series data is not a new field of research. According to Grooijer and Hyndman [11], in the journals published by the International Institute of Forecasters from 1985 through 2005, nearly one-third of all papers were time-series forecasting. They chart the progress of time series forecasting by analyzing some historical approaches from simpler exponential smoothing, to ARIMA models with seasonality and multi-variate inputs and finally more complex algorithms like neural networks or bagging and boosting. Their conclusion in 2006 was that "with the availability of very large datasets and high powered computers, we expect [neural networks and boosting methods] to be an important area of research in the coming years". Another study conducted by Makridakis compared the accuracy of simple Machine Learning algorithms like a multi-layer Perceptron, CART, and Support Vector Regression to traditional statistical approaches like Auto-Regressive Integrated Moving Average (ARIMA) and Theta Models [23]. Their review of performance on over 1,000 monthly time series datasets showed that the statistical methods performed as well or better than the simple Machine Learning algorithms. However, this study did not include more complex algorithms, their tuning approaches or forecasting on higher resolution datasets (minutes). In fact, the author acknowledges that datasets of much longer length can lead to the Machine Learning algorithms to train more optimally. They also recommend deseasonalization of the data prior to application of any complex algorithm.

2.1.1. Long Term Forecasting

Past demand forecasting in satellite communication focused on efforts to predict far future demand, in the magnitude of years, to drive accurate creation of capacity. An effort by de Weck Et al.. [12] uses a Geometric Brownian Motion (GBM) model to estimate demand over a 15-year system lifetime. GBM is "is commonly used in the financial domain to model the price of a stock" but can be applied to broader time series modeling. The author uses a 20% drift constant and a 70% volatility constant to estimate demand every two years. A Monte Carlo simulation generates a range of demand scenarios, a series of estimates over a 15 year timespan, for which the author proposes flexible satellite architectures to more efficiently provide service for the demand in that scenario. This approach predicts the macro-style demand and attempts to address the total capacity issue. In the short-term capacity issues are typically handled through Time Division Multiple Access (TDMA) which statically allocates time slots on a beam to a terminal. This allows multiple users to share a single beam and improves efficiency. Additional work like the solution proposed by Yi Qian Et al. [13] shows that the system can dynamically provide fewer or additional time slots to optimize performance. This approach does not address a moving demand pattern.. Sharing beam access creates additional efficiencies, however computing an optimal, global solution for a full constellation requires computational time beyond the sub second cadence of this approach..

2.1.2. Other Industry Short Term Forecasting

Other industries recognize the need to use short term forecasting to maximize their capital utilization. Uber uses short term demand forecasts to optimize their driver allocation, set surge pricing and anticipate anomalies in usage. Laptev Et al.. [14] propose using a high dimensional Long Short Term Memory (LSTM) neural network to predict the number of completed trips in the subsequent few days. They show that the inclusion of relevant exogenous variables into this network reduces the forecast's average MAPE to 27% from 32% by the current proprietary method, which uses a univariate time-series estimate. In the utilities market, hourly demand forecasts are used to determine energy production levels and to ensure that the system has the capacity to store any unused energy. Hobbs Et al.. [15] showed that forecasts generated using Artificial Neural Networks (ANN) provided a 1.9% improvement over previous methods. They also concluded that this improvement equates to an $800,000 per year savings for each of the 19 utilities included in their studies. This type of demand forecasting and its application to efficient allocation is an ongoing field of research.

2.2. Theoretical background on two Machine Learning Algorithms

As predicted by Grooijer and Hyndman datasets and computational power increased drastically and popular algorithms like neural networks and boosting dominate the field. This thesis compares using XG Boosted Trees for single terminal demand forecasting to using Recurrent Neural Networks (RNN) for multi-terminal demand forecasting. Both algorithms have documented success in time series predictions.

2.2.1. XG Boosting for Time Series

In 2009, Ye and Keogh [16] introduced the idea of using time series shapelets, subsets of the entire time series, for classification. They showed that their approach “can be more interpretable, more accurate and significantly faster than state-of-the-art classifiers”. The primary example is classification of leaves which have slight deviations to their patterns and random holes or missing sections. Attempting to match the data of entire leaf to a class is hindered by the randomness in a single leaf. However, using shapelets reduces the impact of the noise within the leaf by matching against sections that were a purer representation of the class. Furthering the application, Ji Et al. [17] use a shapelet based time series XG Boost algorithm to measure classification accuracy of 12 different data sets from the UEA & UCR Time Series Classification Repository. They used shapelets in their feature engineering process and then built XG Boost models using a feature set that tried to avoid noisy data. Their results show that the XG Boost model had the highest average classification accuracy over traditional shapelet approaches.

Pavlyshenko [18] uses XG Boost models to accurately predict next day drug store sales using daily sales data from many stores, as well as exogenous variables like promotions, pricing and competitor’s behavior. He proposes stacking XG Boosted models by using their output as an input to a second layer of a single tree model, a linear model and a simple ANN to generate a weighted ensemble forecast. This stacked XG Boost approach yielded a 1% net improvement in error over the next best algorithm. Karakatsanis [19] compares the use of gradient boosting to seasonal ARIMA models and a Random Forest model, a bagging tree ensemble algorithm, in forecasting Short-term electricity load values. Their application of gradient boosting yielded a model with Mean Average Percentage Error (MAPE) of forecasts of 1.32%, compared to 2.62% for a seasonal ARIMA and 1.97% for a Random Forest. Their analysis included the use of an exogenous variable, air temperature, which was shown to be strongly correlated with electricity loads. They conclude that the ARIMA models were less able to take advantage of the exogenous variables than the two machine learning methods and suggest the inclusion of additional variable to further improve forecasts. Taieb and Hyndman [20] also use gradient boosting for energy load forecasting placing 5th of 105 teams in a 2012 Kaggle competition. Their work required the predictions of loads in 20 different zones each with its own historical demand and temperature data. They generated separate boosted models for each zone and combined the resulting forecasts instead of attempting to generate a single forecast for all the zones combined. Additionally, they created a separate model for each hour of the day to reduce the impact of highly seasonal energy usage.

2.2.2. Recurrent Neural Networks for Time Series

RNN usage for time series forecasting is very popular, making it nearly impossible to search the internet without running into a novel application of LSTM's. In addition to the previously mentioned efforts by Uber, Karim Et al.. [21] demonstrate the use of fully convolutional neural networks (FCN) in conjunction with LSTMs that outperform other network models on 85 different time series data sets. They use standard approaches for both the LSTM and FCN networks, concatenate each model's output, and pass that output to a single network layer with a softmax activation.. Qin Et al.. [22] use LSTM networks in a dual stage attention model to predict NASDAQ stock prices. They found their model to have improved accuracy over ARIMA and simpler RNN structures and conclude that their model "can adaptively select the most relevant [exogenous variables] but can also capture the long-term temporal dependencies of a time series". Che Et al.. [23] use Gated Recurrent Units (GRU) with trainable decay rates to classify medical time series data with missing values. They conclude that their new GRU-D can learn both from the patterns in the present data and from the patterns of where data is missing. Including a learnable decay vector causes their GRU-D to outperform standard GRU architectures on their datasets.

2.3. Parameter Tuning

Machine learning algorithms require parameter tuning to improve their accuracy and generalization for forecasting. While an algorithm can learn the best settings, weights or tree decisions, a user must decide on numerous other parameters that determine how the algorithm operates. Simple algorithms like ARIMA or nearest neighbors have a few parameters to tune, but more complex algorithms can have hundreds of different parameters that impact the algorithm's performance. The algorithms selected for this thesis include three types of parameters to tune: data/feature parameters, network architecture parameters and algorithmic parameters. An example parameters for XG Boost includes how many days of training data to allow, the maximum depth of an individual tree, and the initial estimate of the model prior to building any trees. RNN examples include: the number of previous timesteps to include in each training example, whether to use an LSTM or GRU layer, and the percentage of dropout entering a dense layer. The best way to optimize these parameters is an ongoing field of research. The biggest challenge is that the large number of available parameters to test and the larger number of options or ranges for each parameter creates a search problem in a high dimensional space. Two of the easiest tuning approaches to implement are Grid Search and Random Search, while two more complex approaches are Evolutionary and Design of Experiments.

2.3.1. Grid Search

The traditional approach to parameter search is a grid search. Grid search creates and measures a model for every combination of parameter value and selects the combination that yields the best value. If F parameters are selected with L number of values to attempt in each parameter, then LF models are built using this approach. It is easy to implement, and the analysis is simple however the scaling quickly becomes computationally infeasible as L and F increase.

2.3.2. Random Search

Random Search is similar to grid search in that a number of parameters are selected to test and their domain of values is chosen. However, instead of a static combination of all parameters and potential values, Random Search randomly selects a value within the domain of every parameter for each trial. The work of Bergstra and Bengio [24] compares the results of tuning neural networks with grid search and random search. They conclude that random search finds solutions as good or better than grid search in a small fraction of computation time. Their logic is demonstrated in their example shown in Figure 2. Two parameters are measured at 3 levels. In grid search, only the three unique levels are tested, while in a random search 9 unique values of each parameter are evaluated. Given a situation where only one parameter is impactful to the Loss function and no interaction effect is present, this means that the random search is more likely to find a better solution because it tests more unique values of the important parameter. While this is an improvement over grid search, it can require many trials and does not yield statistical insights into the effects of each parameter.

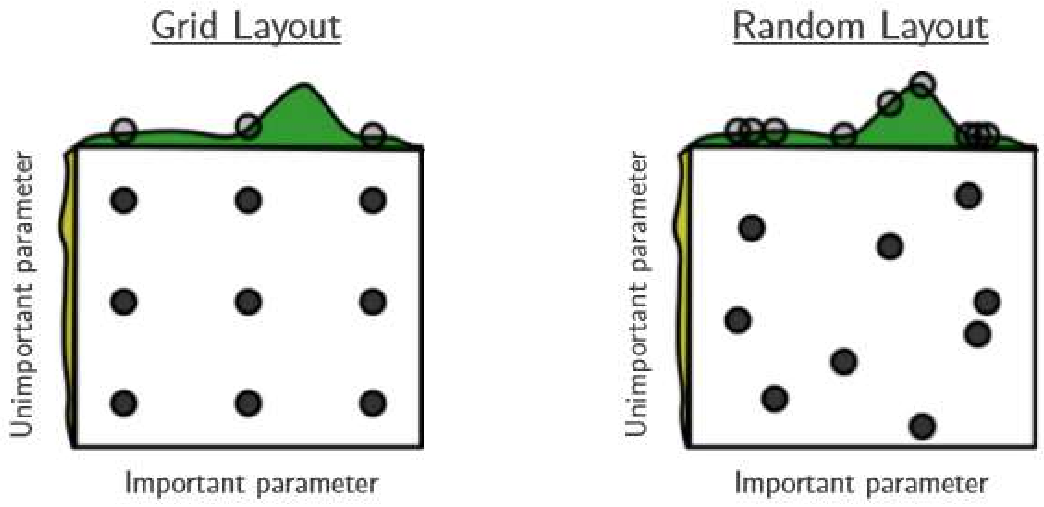


Figure 2 Bergstra and Bengio Example of Grid vs Random

2.3.3. Evolutionary

Evolutionary algorithms utilize a parameter set and domain of values like Random Search. Each trial is measured by its performance and its probability of contributing to the population of the next round increases as its performance increases. These approaches use randomness to change parameter values through mutation, changing a value of a parameter on a selected trial, or cross-mutation, swapping parameter values of two selected trials. The algorithm iteratively selects high performing trials and mutates them to find better and better performance. Bergstra Et al.. [25] improve on their random search approach by applying a gradient-free evolutionary algorithm to optimize continuous parameters. Their results yield superior performance on high dimensional (32 parameters) Random Search tuning for two different network examples with only 457 and 361 trials.

2.3.4. Design Of Experiments

Lujan-Moreno [26] proposes using a two staged design of experiments approach to parameter tuning, which first used a screening test to determine relevant parameters to tune a Random Forest model and then used a Response Surface Methodology (RSM) to finalize its tuning. This effort built a classification model to determine whether a person earned over $50,000 per year based on attributes like age, race and sex. The screening test found 5 of 7 factors to have statistically impactful main effects using only 24 combinations. 3 of those factors were selected for an RSM design using 15 combinations to test for interactions and non-linear effects. A single two-way interaction was found to be significant as well as one quadratic effect. On the neural network algorithm, Sukthomya [27] applied a single staged test to tune parameters of an ANN used to model a manufacturing process at an aircraft engine plant. The main effects of the 7 chosen parameters were measured and optimized to improve average error. However, this effort did not include RNN or architectural parameters beyond the number of dense layers and the number of neurons in each layer.

3. Data

3.1. Basic description

SES Satellites provided the dataset for this analysis. The data consists of forward and return data rates in Mbps for 7 terminals at 1-minute resolution. Over a 4-month span from June through September this represents 123 days of traffic data. No additional information is provided to the models to include user information, location or weather. The objective is to generate accurate 15 minute ahead forecasts of each terminal's forward data rate using its historical rate and its historical return rates as an exogenous variable.

3.2. Descriptive statistics

Basic statistics are shown in Table 1. The terminals range in demand from an average of 74.1 Mbps to 293.2 Mbps. The standard deviations and variance do not appear problematic however every terminal has a minimum value of 0. Present within this dataset are occurrences where traffic can drop to zero or near zero very quickly. Roughly 1% of entries for each terminal are less than 1 Mbps. An example of this is shown in Figure 3. The forward and return rates for Terminal 1 drop to zero from their normal rates. In this instance the terminal usage remained at zero for 6 minutes during the first drop and 39 minutes during the second. These drops could be caused by satellite outages either technical or due to weather, terminal outages or errors in/missing data. In order to remove these drops, every instance where the rate dropped by more than 50% in a single minute was replaced by the previous value. This causes traffic to remain artificially flat during instead of zero when data is missing but is reasonable because the DRM would continue to allocate resources to the terminal at the same level as the most recent information. Figure 3 also demonstrates the strong positive correlation between a terminal's forward and return data rates. This was present in each terminal.

Table 1 Descriptive Statistics of Terminals (Mbps)

		T1 FWD	T2 FWD	T3 FWD	T4 FWD	T5 FWD	T6 FWD	T7 FWD
Forward	Mean	255	226	74	100	293	114	72
	Standard Deviation	84	88	26	35	110	41	27
	Sample Variance	7032	7778	692	1226	12086	1683	721
	Minimum	0	0	0	0	0	0	0
	Maximum	457	483	122	199	577	207	138
		T1 RTN	T2 RTN	T3 RTN	T4 RTN	T5 RTN	T6 RTN	T7 RTN
Return	Mean	116	112	32	35	120	58	33
	Standard Deviation	38	44	12	14	46	21	14
	Sample Variance	1405	1929	141	187	2072	419	191
	Minimum	0	0	0	0	0	0	0
	Maximum	213	307	61	122	329	93	66

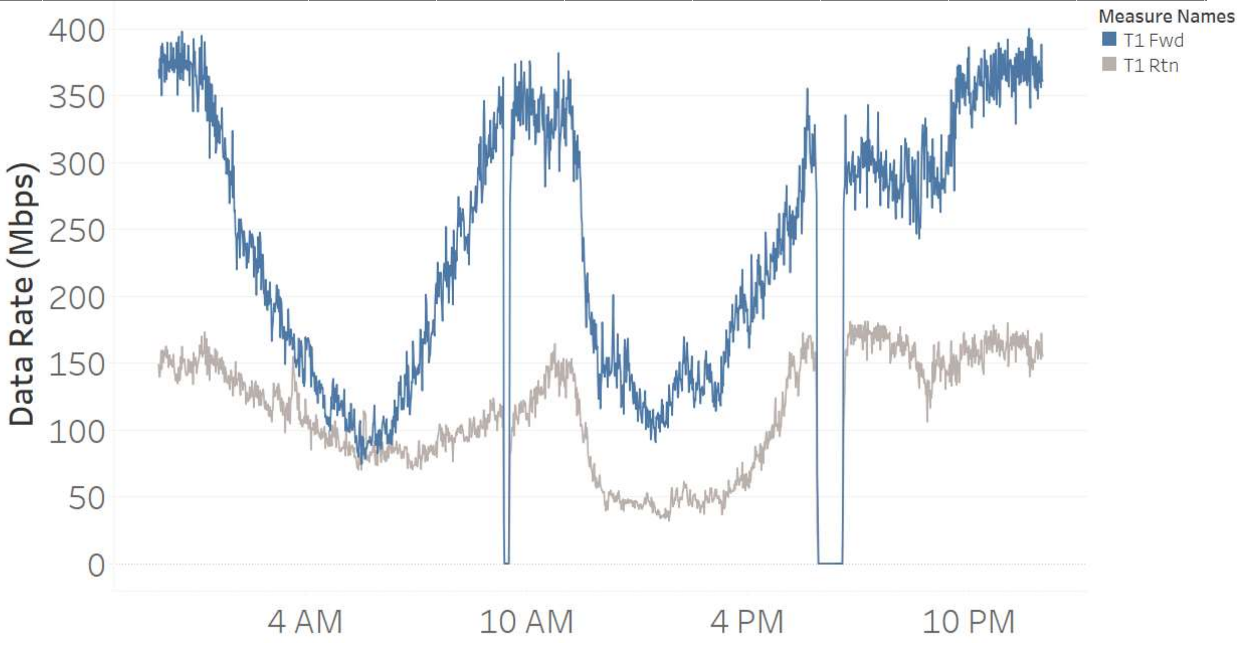


Figure 3 Terminal 1 Traffic Rates

3.3. Seasonality

The demand for satellite communication rate appears to have a strong daily seasonality. A potential seasonality shown by the pattern of Terminal 1 in Figure 4. Each terminal displays the same general tendency to have a repeating low demand time period however, the timing and shape of this low period changes from terminal to terminal and from day to day. Each algorithm had a Seasonality parameter available during tuning. This parameter creates an additive seasonal model using statsmodels package in python from the training data. Then, it subtracts the seasonal component from the dataset. The parameter options were: “None” – no seasonal model built, “Daily” – a seasonal model with period 1440 was built and subtracted, or “Weekly” – a seasonal model with period 10080 was built and subtracted.

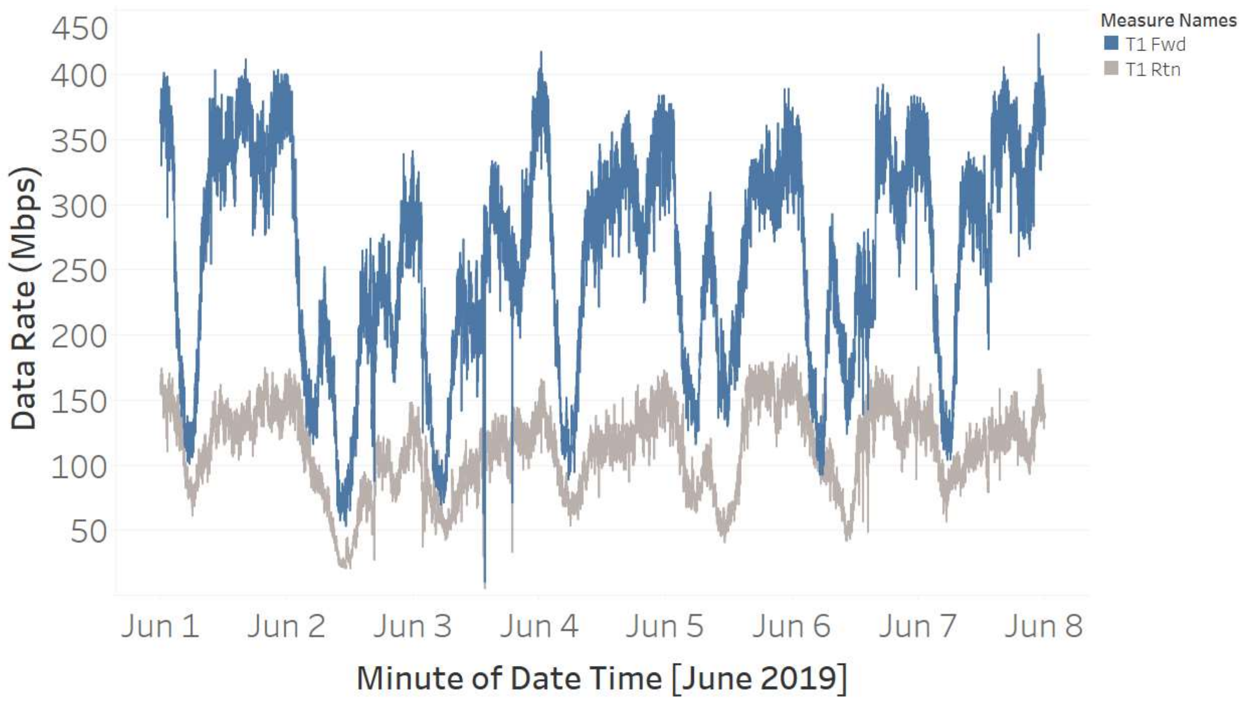


Figure 4 Terminal 1 Traffic Rates Full Week

3.4. Short-Term Variation

By visual inspection of Figure 3 or Figure 4, the short-term variation of traffic is noticeable. Figure 5 shows the Mean Absolute Percentage Error (MAPE) for each terminal when compared to its demand X minutes prior. The least noisy terminal’s change grows from 5.6% at a 1-minute lag to 8.6% at a 15-minute lag.

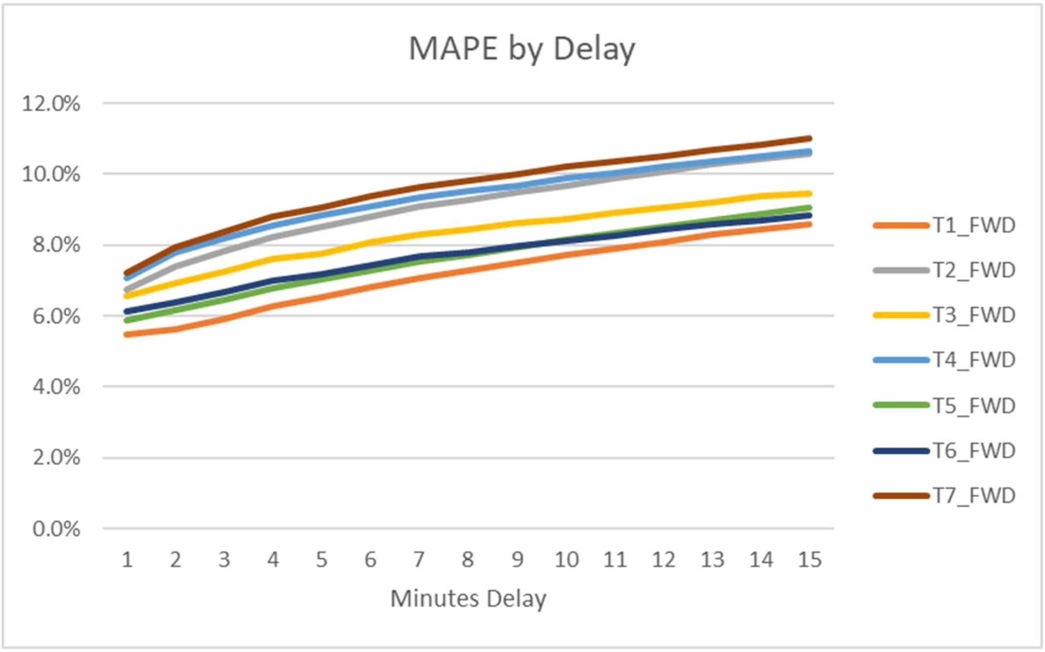


Figure 5 MAPE by Solution Delay

3.5. Autocorrelations

The Box/Jenkins [29] autocorrelation function (ACF) is the correlation between the traffic and lagged traffic. The plotted ACF for Terminal 1 is shown in Figure 6 over a full day. The data has a significant positive, declining correlation through the first 300 minutes before a slight negative correlation at near the half day mark. The correlation returns as the lags get closer to a full day. Each of the 7 terminals had the same pattern of ACF.

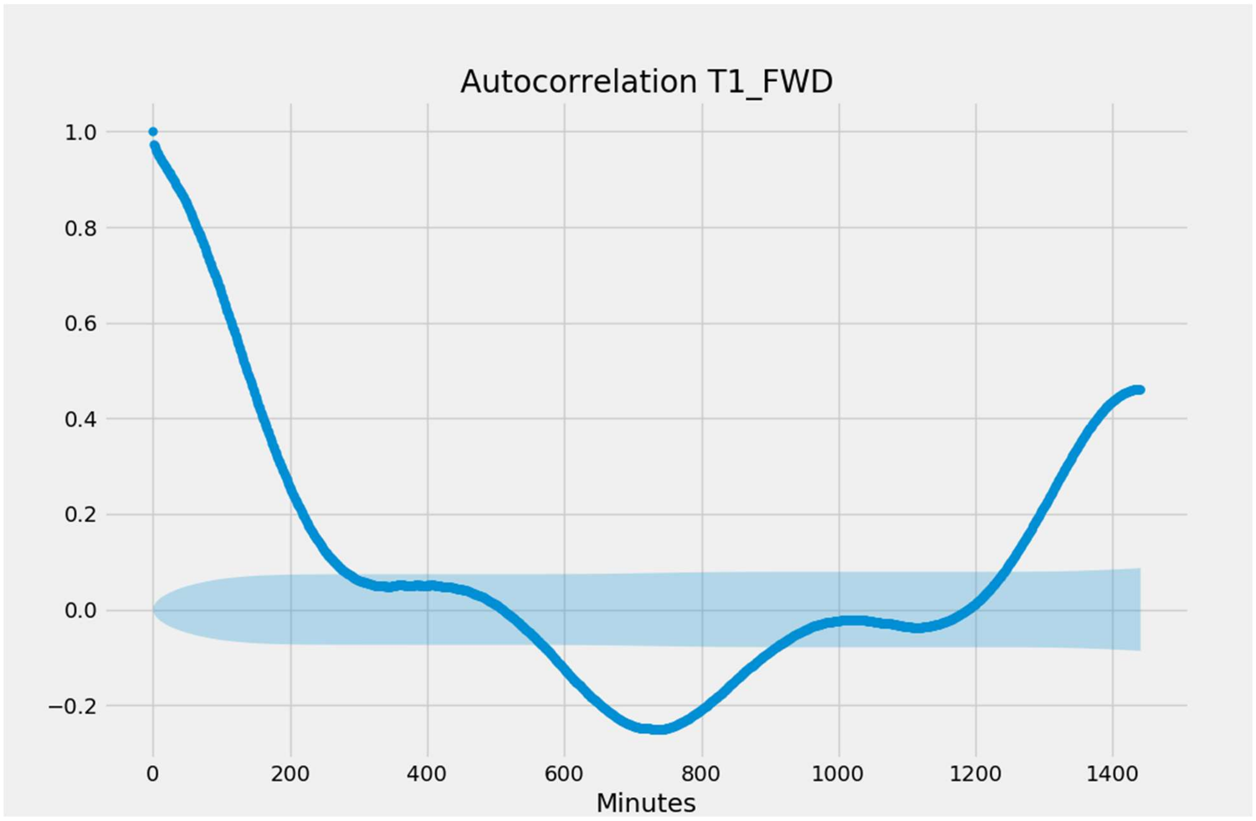


Figure 6 Terminal 1 Auto Correlation Plot

3.6. Partial Autocorrelations

A Partial AutoCorrelation Function (PACF) is the correlation of a future value and the residual values of the demand sequence and the expected correlation of all previous lags. Essentially, it gives insight into whether the next lagged value provides any new correlation/information into the most current value. The plotted PACF for terminal 1 is shown in Figure 7. The PACF shows positive correlations to residuals for up to 10 minutes. This is similar across all 7 terminals.

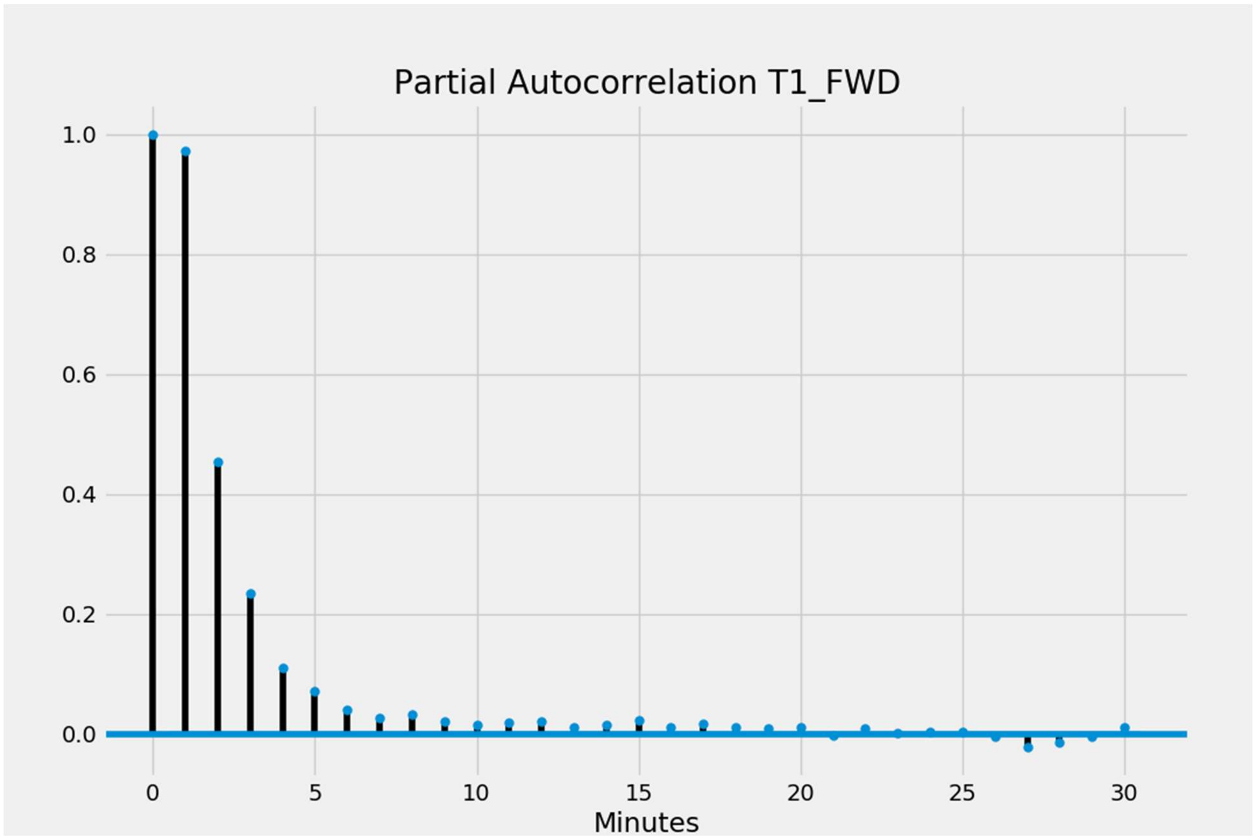


Figure 7 Terminal 1 Partial Autocorrelation Plot

3.7. Training/Validation/Holdout Split

6% of the total data is used as the validation set for all algorithms is a 7-day period from Aug 21st – Aug 27th (the 81st through 87th day). The training set is a variable during up to 75 full days prior to the 24-hour validation period for that model. The training set of subsequent models will contain the previous model's validation set when the model is retrained. This offers a 7-day cross-fold validation for each terminal to improve the generalization of the algorithms. The final 25% of the dataset or 31 days is used as a holdout period to test the generalization of the algorithms. This period is never used during testing or tuning.

3.8. Specific Objectives

Current satcom literature focuses on long-term demand forecasting or short-term forecasting of rain attenuation [28]. The author found no examples of short-term communications satellite traffic to aid in dynamic resource allocation. To the best knowledge of the author, there is also no literature showing a comparison of parameter tuning using design of experiments to other tuning methods for either RNN or XG Boosting. This thesis seeks to show that 1) RNN and XG Boost algorithms provide an improvement in short-term traffic forecasting and 2) parameter tuning using experimental design optimizes model performance over tuning done by random search.

4. Testing Methodology

4.1. Overview

This thesis proposes a two-stage design of experiments approach to tuning models using two algorithms: XG Boost and RNN. Before tuning, each algorithm produces a model at its default or the author's "best guess" settings. The first stage of parameter tuning used a screening test for each algorithm to find important parameters with a statistically significant effect on MAPE. The second stage used a Response Surface Method (RSM) test to find optimal settings for important parameters found in the first stage. After conducting each stage, the best parameters found were used to generate models and track the contribution of each stage. A random search of parameters was also conducted using the same number of total runs as DOE approach. The performance of all models was also compared with using the 15-minute lagged traffic as a forecast. The full list of final models compared is shown in Table 2.

Table 2 Models to Compare

Model #	**Algorithm**	**Version**
1	Baseline: Lag	15 minutes
2	XG Boost	Default Parameters
3	XG Boost	Screening
4	XG Boost	RSM
5	XG Boost	Random Parameters
6	RNN	Default
7	RNN	Screening
8	RNN	RSM
9	RNN	Random

4.1.1. Response Metric

Each model is scored by its Mean Absolute Percentage Error (MAPE) for each terminal. The below calculation is used for each terminal, over the validation period t = {0…n}, using the A actual traffic and the F forecasted traffic. The lagged MAPE uses the most recent available traffic rate where $F_t = A_{t-15}$.

$$MAPE(F_x) = \frac{1}{n}\sum_{t=0}^{n}\left[\frac{A_t - F_t}{A_t}\right]$$

4.1.2. Stage 1 of 2: Screening

The purpose of this stage is to identify which parameters are impactful and which are not. 11 parameters that the author felt were most important were selected and a small number of combinations of these parameters at high and low settings were used to generate models. Through deliberate selection of which combinations to run it is possible to generate statistically relevant estimations of the main effects of each parameter. A linear regression is performed to estimate the coefficients of all relevant parameter's main effects and create a single equation that represents the MAPE of a model given its parameters.

4.1.3. Stage 2 of 2: Response Surface Method

Parameters with significant main effects or effects deemed necessary to test further were included in the RSM Stage. The RSM used is a Central Composite design using a Full Factorial 2-level test with center and axial points (giving a 5-level test). The results are analyzed in the same manner as the Screening Stage and an optimal setting was found using optimization on the linear regression model of the main, interaction and quadratic effects. Final models were created using these parameter settings and evaluated by MAPE.

4.1.4. Random Search

A Random Search was conducted using the same 11 parameters and the domain established during the Screening Stage. Continuous parameters were randomly set to any value between the low and high settings tested. Discrete parameters randomly select either the low of high value in the Screening Stage. Each algorithm runs the same number of total trials as the Screening and RSM stages combined. This provides equal computing power access for each tuning approach and allows for a fairer comparison.

4.1.5. Holdout Performance

The parameter settings from each stage and the best performing parameters from Random Search were determined based on performance during a 7-day validation period. The performance of these 4 settings, Default, Screening, RSM, and Random, is measured over a 31-day holdout period. Because this data was not used in any parameter tuning it is the final measurement of performance for each algorithm and is the best representation of the algorithm's ability to generalize to new data.

4.2. XG Boosting

4.2.1. How it works

Gradient Boosting creates an ensemble of weak models, typically a Classification and Regression Tree, by iteratively adding weak models that improve on the overall Loss function. In 1996 Freund and Schapire [30] introduced AdaBoost, the predecessor to gradient boosting. AdaBoost also creates an ensemble of weak classifiers to improve the overall error rate. This method iteratively places more and more importance on training point that are incorrectly classified by the ensemble. New weak classifiers are selected by their weighted error rates (larger weights on misclassified training points), are given a voting power and added to the ensemble. This method was built upon by Breiman in 1997 [31] who introduced the idea of treating the boosting algorithm as an optimization of a Loss function.

In 1999, Friedman [32] created the first gradient boosting algorithm, which instead of reweighting training points finds a function F that minimizes the Loss function of F and the target y. The strong learner function F is defined as the summation of the output of n weak learners h times a constant gamma.

$$F = \min_{F} L(y, F(x))$$

$$F(x) = \sum_{i=1}^{n} \gamma_i h_i(x) + constant$$

A fundamental difference between AdaBoost and Gradient Boosting is that the weak classifiers are trained on the "pseudo-residuals" of the target and the current F as opposed to weighted training points. The new weak learner h is then trained on the dataset (xi, rin) as opposed to (xi, yi).

$$r_{in} = \left[\frac{\partial L(\, y_i, F(x_i))}{\partial F(x_i)}\right]$$

Once the new weak learner is trained, the constant gamma is computed by finding a value that results in the minimum overall Loss. Finally the new weak learner is added to the strong learner F. This process is repeated until a certain number of weak learners are created, the newest weak learner does not improve the Loss function enough or some other exit criteria.

$$\gamma_n = \min_{\gamma} L(y_i, F_{n-1}(x_i) \; + \; \gamma h_n(x_i))$$
$$F_n(x) = F_{n-1}(x) + \; \gamma_n h_n(x)$$

Each round n adds a new weak learner which predicts the residual error of the target and the strong learner resulting in a strong learner which very accurately fits the training data. Gradient Boosted Trees uses the above method and only produces smaller CART trees as weak learners. The algorithm concludes and returns a strong model, the ensemble of CART trees, when it has built the maximum number of trees or cannot find a new tree to add that improves the strong model beyond a given threshold.

4.2.2. Pre-Processing Steps

Before training a XG Boost model the dataset goes through several steps of feature engineering, transformations or controls. Some of the pre-processing steps, Remove Seasonality and Create Day Lags, are considered model parameters and can either be optional or can change in their operation.

1. Remove Drops – Replaces declined traffic of 50% or greater with the prior value
2. Pick Users – The algorithm is built to predict a single terminal’s traffic and the inputs are limited to only that terminal’s prior traffic and return data rates. Other terminal information is dropped from the dataset.
3. Normalization – The traffic and return rates are normalized based on the entire history prior to the validation set by subtracting the mean rate and dividing by the standard deviation of that sample.
4. (Parameter-Removable) Remove Seasonality – An additive seasonal decomposition is subtracted from the terminals traffic and return rates. If set to “None” this step is not executed.

5. (Parameter) Create Lags – Creates features using the lagged traffic and return values of that terminal. The number of lagged features is a variable parameter of the model.
6. (Parameter-Removable) Create Day_Lags – Creates features using the lagged traffic from the same time as the target but on the prior day. The number of prior day values included as features is a variable parameter of the model. If set to 0, this step is not executed.
7. Create Date Categorical Features – Generates features from the date time index. Minute and Hour are left as numerical features while "day of the week" and "month" are one-hot encoded categorical features.
8. Train/Validation Split – Creates a training set of 75 days prior to the validation set by removing all current rates and leaving only the lagged variables at every time step. Creates a target set using only the current rates for every time step. Validation sets are created using the same method but for a set 24-hour period.

4.2.3. Training

An XG Boost model is trained using the given parameters for that run using Mean Squared Error (MSE) as its loss function. Predictions are generated using the lagged values of the validation set. Seasonality is replaced if a seasonal model was used in pre-processing and the predictions are then unnormalized.

4.2.4. Parameters to tune

11 parameters were selected for tuning.

- Lags: The number of previous minutes included in each timestep. A lag of 60 would include the past hour's traffic to forecast the next value or a lag of 5 would only have the previous 5 minutes. The author selected a default value of 15 minutes which covers the duration of significant partial autocorrelations for each terminal.
- Day_Lags: This parameter decides how many previous day's values from the same time as the target to include as features. The author selected a default value of 7 days which gives the model an example from each day of the week.

- Seasonality: Whether a seasonal decomposition is applied to the normalized data rates before training the boosting model. The potential seasonal models will be additive with periods of 1 day or 1 week. The seasonality is calculated using the statsmodel python package with 60 days of training data. After a model is trained, the seasonality is reapplied to the resulting predictions for analysis.
- Depth: The maximum depth allowed for any weak learner tree. A higher value allows trees that are more complex, making them better able to predict the target, but also more prone to overfitting. The default setting is 6.
- Gamma: The minimum reduction on the Loss function to create a new leaf on a tree. A higher value will prevent more complex trees from forming if they only contribute minimal benefit to the overall Loss. The default setting is 0.
- Eta: The dampening learning rate that reduces the impact of each new tree. This helps to slow the learning and to allow future trees to contribute more to the overall model. The default setting is .3.
- Subsample: The ratio of total training data available when creating a new weak learner tree. This is a bagging technique that randomly samples a subset of the training data at every iteration. The new tree is built only from that subset of data. A lower value can reduce overfitting. The default setting is 1.
- Colsample_bytree: The ratio of features available when constructing a new weak learner tree. This limits the available features to a randomly selected subset at every iteration. A lower value can reduce overfitting. The default setting is 1.
- Number of Estimators: The maximum number of weak learner trees that can be added to the ensemble. The default setting is 100.
- Basebias: The initial estimate made for every point at iteration 0. The first set of residuals is computed by subtracting this basebias from the target values. The default setting is .5.
- Return Rate Drop: Whether to remove the return data rate from the dataset and make it unavailable as an exogenous variable for training. The author selected a default value of False, to use the Return Rate as an exogenous variable.

4.3. Recurrent Neural Networks

4.3.1. How it works

RNN's are a type of Artificial Neural Network based on the work of Rumelhart [33] which maintains a hidden vector state to learn sequences. By sequentially stacking RNN nodes one after another, the hidden vector state serves as a second input to a new node. This allows the node to make more accurate predictions based on the current input and historical insights passed from previous nodes. LSTM networks are a more complex version of RNN from Hochreiter and Schmidhuber [34]. The full process is skillfully described by Graves [35] and an example is depicted in Figure 8. At every timestep each LSTM cell has a hidden vector state h similar to the RNN and also a memory vector state m. Each cell makes use of 3 gates: the "forget" gate – which allows the cell to remove previous information that is no longer useful, the "update" gate – which determines what new information is important to add to the current state and an "output" gate – which determines what new information is important to add to the current state to become the next output. The cell also uses a hyperbolic tangent activation for both the hidden and memory state before passing their outputs to the next stage. The equations for each timestep are shown below using W to represent the weight matrices, I to represent the projection matrices, x as input and σ to represent the sigmoid function.

$$\mathrm{Gate}^{\mathrm{Update}} = \sigma(\mathbf{W}^{u}\mathbf{h}_{t-1} + \mathbf{I}^{u}\mathbf{x}_{t})$$
$$Gate^{Forget} = \sigma(\boldsymbol{W}^{f}\boldsymbol{h}_{t-1} + \boldsymbol{I}^{f}\boldsymbol{x}_{t})$$
$$\mathrm{Gate}^{\mathrm{Output}} = \sigma(\mathbf{W}^{o}\mathbf{h}_{t-1} + \mathbf{I}^{o}\mathbf{x}_{t})$$
$$\mathbf{m}_{t} = G^{f} * \mathbf{m}_{t-1} + G^{u} * \tanh(\mathbf{W}^{c}\mathbf{h}_{t-1} + \mathbf{I}^{c}\mathbf{x}_{t})$$
$$\mathbf{h}_{t} = \tanh(G^{o} * \mathbf{m}_{t})$$

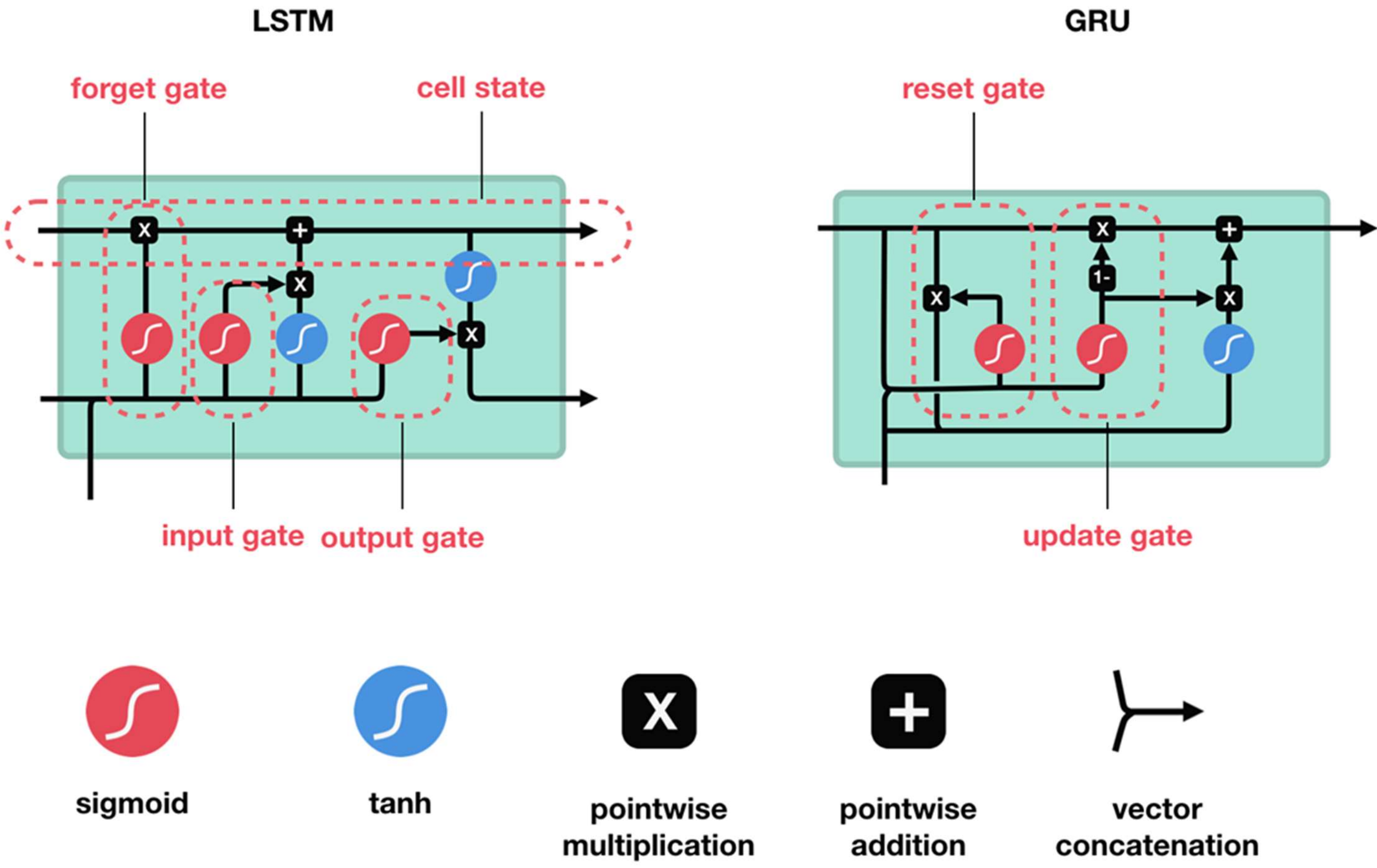


Figure 8 LSTM & GRU Architectures

4.3.2. Pre-Processing Steps

Before training a RNN model the dataset goes through several steps of feature engineering, transformations or controls. Some of the pre-processing steps, Drop Return Rates and Remove Seasonality, are considered model parameters and can either be optional or can change in their operation.

1. Remove Drops – Replaces declined traffic of 50% or greater with the prior value
2. (Parameter – Removeable) Drop Return Rates – This parameter gives the option to remove the return data rates as an exogenous variable from the model. If True, the model is based on the traffic only.
3. Normalization – The traffic and return rates are normalized based on the entire history prior to the validation set by subtracting the mean rate and dividing by the standard deviation of that sample.

4. (Parameter-Removable) Remove Seasonality – An additive seasonal decomposition is subtracted from the terminals traffic and return rates. If set to "None" this step is not executed.
5. Train/Validation Split – Creates a training set of 75 days prior to the 1-day validation set.
6. (Parameter) Create Lookback Tensors – Creates training and validation tensors by generating sequences of "lookback" length of prior traffic. This parameter controls how far back the algorithm can see for each timestep.

4.3.3. Training

An RNN model is trained using the given parameters for that run using Mean Squared Error (MSE) as its loss function. Predictions are generated using the lagged values of the validation set. Seasonality is replaced if a seasonal model was used in pre-processing and the predictions are then unnormalized.

4.3.4. Parameters to tune

11 parameters were selected for tuning to include parameters that determine the architecture of the RNN. A representation of the potential network architectures is shown in Figure 8.

4.3.4.1. Network structure parameters:

- First Layer: Whether to use a LSTM or a GRU in the first layer.
- Stateful: When training, does the final state of RNN nodes become the initial state of the next epoch.
- Dense Nodes: How many nodes are in the dense layer?
- RNN Nodes: How many nodes are in the RNN layer?
- Extra RNN layer: Whether to include an extra LSTM layer? If included, this causes the first layer to be stacked and the hidden states at each timestep are returned to the cell.
- Extra Dense layer: Whether to include an extra Dense layer?

4.3.4.2. Other parameters

- Lookback: The # of minutes of lagged traffic available at each timestep.
- Return Rate Drop: Whether to remove the return data rate from the dataset and make it unavailable as an exogenous variable for training.
- Seasonality: Whether a seasonal decomposition is applied to the normalized data rates before training the boosting model. The potential seasonal models will be additive with periods of 1 day or 1 week. The seasonality is calculated using the statsmodel python package with 60 days of training data. After a model is trained, the seasonality is reapplied to the resulting predictions for analysis.
- RNN Dropout: The % of recurrent cell weights excluded from updates during training.
- Dropout: The % of weights from the final RNN layer to the Dense layer that are excluded from updates during training.

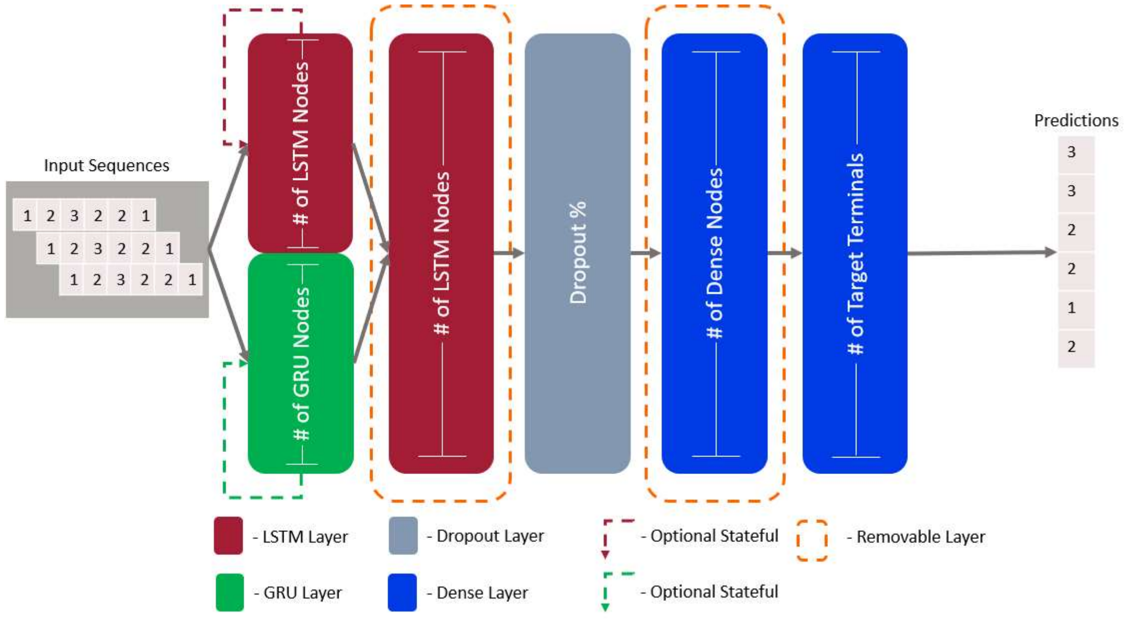


Figure 9 Recurrent Neural Network Flexible Architecture Visualization

4.4. Design of Experiments

DOE is a method of selecting runs and controlling experiments to more accurately and statistically determine the effects of multiple factors over their numerous levels or values. Runs are conducted under specific conditions where one or more of the factors change values and the response variable or result of the process is measured. Historical usage of DOE is most common in agriculture or manufacturing, where the settings of the production plant are factors and the quality of the product is the response variable. This thesis proposes a two staged approach for both XG Boost and RNN tuning using a Plackett-Burman screening design [48] followed by a Central Composite design.

4.4.1. Screening

11 factors are identified as parameters to tune for each algorithm. A full-factorial design would require every combination of these factors and levels be tested. This quickly becomes computationally infeasible with 7 terminals to test over a 7-day validation period. 2048 combinations of parameter settings would be created for each terminal and day. In total over 100k models would be trained and evaluated. If each model took one minute to train and evaluate, this process would take nearly 70 days of computation.

4.4.1.1. Fractional Factorial Designs

According to Montgomery's text "Design and Analysis of Experiments" [36, p. 320], "fractional factorial designs are among the most widely used types of designs for product and process design". Fractional designs can provide statistical insight to main effects and low order interactions by conduction a smaller fraction of trials required by a full-factorial design.

4.4.1.2. Notation

Fractional designs are typically notated in the following form Lk-p. L is the number of levels or values to test each factor. K is the number of factors to test. P is the number of independent generators or assignments of main effects to higher order effects. Standard approaches use 2 level designs to simplify the generation of a test matrix and would use a different method, like a response surface, to create tests with K > 2. This notation is useful also quickly determine the number of runs; a 25 design requires 32 runs while a 25-2 only requires 23 or 8 runs.

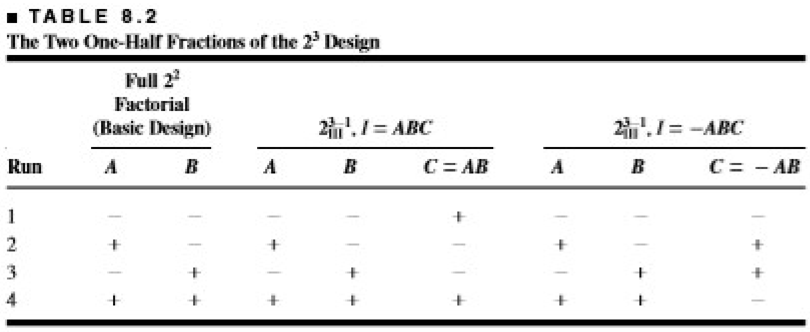
4.4.1.3. Aliasing

The key concept lies in combining the effects of higher order interactions with main effects or low order interactions. This process is called Aliasing. The simplest example starts with a 2^2 Design with Factors A and B at levels '+' and '-'. A full factorial design for these two factors has 4 trials with levels shown in the left side of Table 3. When a 3rd Factor is added to the experiment and the number of trials is held at 4, the design changes to a 23-1 where factor C is aliased with the interaction of AB. Montgomery shows the two possible settings for C where the levels of C are set at either AB or -AB. The main effect of changing the level of C is indistinguishable from the interaction of AB. This also causes the main effects of A and B to be aliased with the other interactions of BC and AC.

4.4.1.4. Resolution

A common metric to describe these designs is its Resolution, a generalization of the aliasing schemes. The example shown is a Resolution III design, which means that no main effects are alias with other main effects, but main effects are aliased with two factor interactions. A Resolution IV design has no main effects aliased with other main effects or any two factor interactions, but two factor interactions may be aliased with each other. A Resolution V design allows two factor interactions to be alias with three factor interactions.

Table 3 Fractional Design Example from Montgomery



■ TABLE 8.2
The Two One-Half Fractions of the 2^3 Design

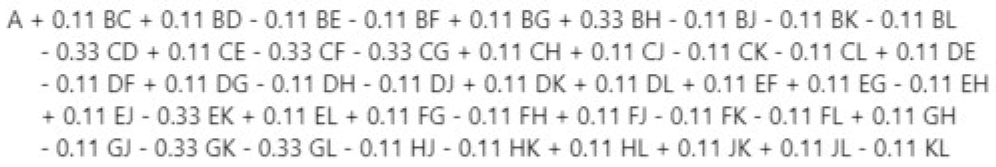
	Full 2^2 Factorial (Basic Design)		$2^{3-1}_{III}, I = ABC$			$2^{3-1}_{III}, I = -ABC$		
Run	A	B	A	B	$C = AB$	A	B	$C = -AB$
1	−	−	−	−	+	−	−	−
2	+	−	+	−	−	+	−	+
3	−	+	−	+	−	−	+	+
4	+	+	+	+	+	+	+	−

4.4.1.5. Plackett-Burman Design

In 1946 Plackett and Burman developed an approach to building fractional designs that ensured that for each pair of factors their level combinations will all be tested the same number of times. This creates a design with a more complex alias scheme than a standard fractional design but allows for more interesting approaches in analysis. These designs can project themselves very successfully into lower order fractional designs with higher resolution. In the example by Montgomery [36], a 12-run design of 11 factors can be folded into a 23 full factorial plus a 23-1 fractional design of just 3 factors. This allows the estimation of interaction effects of a smaller number of factors once non-important factors are identified and eliminated from the analysis. The trade off is a complex alias scheme where main effects are partially aliased with two factor interactions. For example, instead of A being fully aliased with CD in a 4-factor test, A is aliased with 1/3 BC + 1/3 BD + 1/3 CD.

4.4.1.6. Selected Design

This thesis uses a 36 run Plackett-Burman design with 11 factors at 2 levels. This design is considered resolution IV but has a complex aliasing scheme with each factor partially aliased with two factor interactions. The alias scheme for Factor A is shown in Figure 9. Ideally, multiple main effects for each algorithm will have no significant impact and the design can be projected into a higher resolution design to more accurately estimate the remaining factors and their interactions.



```
A + 0.11 BC + 0.11 BD - 0.11 BE - 0.11 BF + 0.11 BG + 0.33 BH - 0.11 BJ - 0.11 BK - 0.11 BL
   - 0.33 CD + 0.11 CE - 0.33 CF - 0.33 CG + 0.11 CH + 0.11 CJ - 0.11 CK - 0.11 CL + 0.11 DE
   - 0.11 DF + 0.11 DG - 0.11 DH - 0.11 DJ + 0.11 DK + 0.11 DL + 0.11 EF + 0.11 EG - 0.11 EH
   + 0.11 EJ - 0.33 EK + 0.11 EL + 0.11 FG - 0.11 FH + 0.11 FJ - 0.11 FK - 0.11 FL + 0.11 GH
   - 0.11 GJ - 0.33 GK - 0.33 GL - 0.11 HJ - 0.11 HK + 0.11 HL + 0.11 JK + 0.11 JL - 0.11 KL
```

Figure 10 Alias Scheme for Factor A in Screening

4.4.1.7. Blocking

These tests are conducted using 7 terminals and a 7-day cross fold validation approach. Each terminal and day introduce additional variability in forecasts. Some terminals may have a lower MAPE and are easier to forecast and some days may have less overall variation leading to a lower MAPE. It is important to ensure that the generated best settings generalize to all terminals over all days, so the test is replicated a full 49 times, once for each combination of terminal and day. These replications are blocked in the design, which means that their effects are measured and removed from the general error term but are not part of the overall model. George Box said, “block what you can, randomize what you cannot”. In this case, randomization has no impact on algorithm building because there is no overall learning curve occurring and models with initial random settings are already randomized.

4.4.1.8. Analysis

The response variable MAPE is analyzed using Minitab software. First an analysis of variance (ANOVA) is performed calculating the sum of squares (SS) for each factor, the squared difference between the actual value and the mean. An F-test, which is a ratio of the factor's SS to the total SS, yields a statistic to measure whether the error explained by that factor is statistically significant. The resulting p-value represents the probability of seeing the given result if there is no significant impact from that factor. A low p-value causes the rejection of the hypothesis that the factor has no effect. The software then generates a regression equation to estimate the coefficients (impacts) of each factor on the response variable. These values are used to select the optimal levels for each factor resulting in the "best found" solution in the Screening stage. If there are multiple main effects with high p-values, the analysis is re-run without those factors included. This results in the projection of the conducted lower resolution test into a higher resolution test.

4.4.1.9. Down-Selecting Parameters

Main effects or two-way interactions that are significant are considered for further testing. If the factor has only two settings (True/False), then the optimal level found in Screening is used and no further testing is done. For significant, continuous factors additional testing is required to measure non-linear effects or interactions.

4.4.1.10. Screening Test Values (36 runs)

The high and low settings for each parameter in XG Boost are shown in Table 4 and the same is shown for RNN in Table 5.

Table 4 XG Boost Screening Parameters

Boosting Variable	**Description**	**Low**	**High**
RTN Drop	Remove RTN values from training	FALSE	TRUE
Lags	# of minutes of prior values available	10	180
Day Lags	# of prior day values	0	7
Depth	Maximum Depth of Trees	4	8
Eta	Learning Rate	.2	.4
Gamma	Minimum benefit required for a new leaf	0	.1
Subsample	% of training data available to each new Tree	.8	1
Column Sample	% of factors available to each new Tree	.8	1
N_estimators	Maximum number of Trees built	50	250

Basebias	Initial estimated value for each point	0	1
Seasonality	Period of seasonal adjustment for each terminal	Daily	Weekly

Table 5 RNN Screening Parameters

RNN Variable	**Description**	**Low**	**High**
Lookback	# of minutes of prior values available	15	250
RNN Dropout	% of RNN cell weights to exclude	0	.2
RTN Drop	Remove RTN values from training	FALSE	TRUE
First Layer	Use a GRU or LSTM	GRU	LSTM
Stateful	Use final state as initial state for next epoch	FALSE	TRUE
Dense Nodes	# of Nodes in the dense layer	32	256
RNN Nodes	# of Nodes in LSTM/GRU layer	32	256
Extra LSTM	Add a second LSTM/GRU layer and make the first layer return/stacked	FALSE	TRUE
Extra Dense	Add an additional Dense layer after RNN	FALSE	TRUE
Dropout	% of weights from final RNN layer to exclude	0	.2
Seasonality	Period of seasonal adjustment for each terminal	Daily	Weekly

4.4.2. Response Surface Methodology

The second stage test was conducted using a subset of important factors found in the screening stage using a RSM for both XG Boost and RNN tuning. This test looks to measure non-linear effects (quadratic) and two-way interactions between factors. According to Montgomery [36] the RSM is “useful for modeling and analysis of problems in which a response of interest is influenced by several variables and the objective is to optimize this response”.

4.4.2.1. Central Composite Design

The Central Composite Design (CCD) is “the most popular class of designs used for fitting these models” Montgomery [36]. A typical CCD is comprised of a two level factorial design, a set of center point tests (using the median value for each factor) and a set of axial points (more extreme values beyond the two-level values). Figure 10 from the Montgomery text shows example level settings of a 3 factor test. The cube is the factorial design with a center at the origin and axial points extending beyond the volume of the cube.

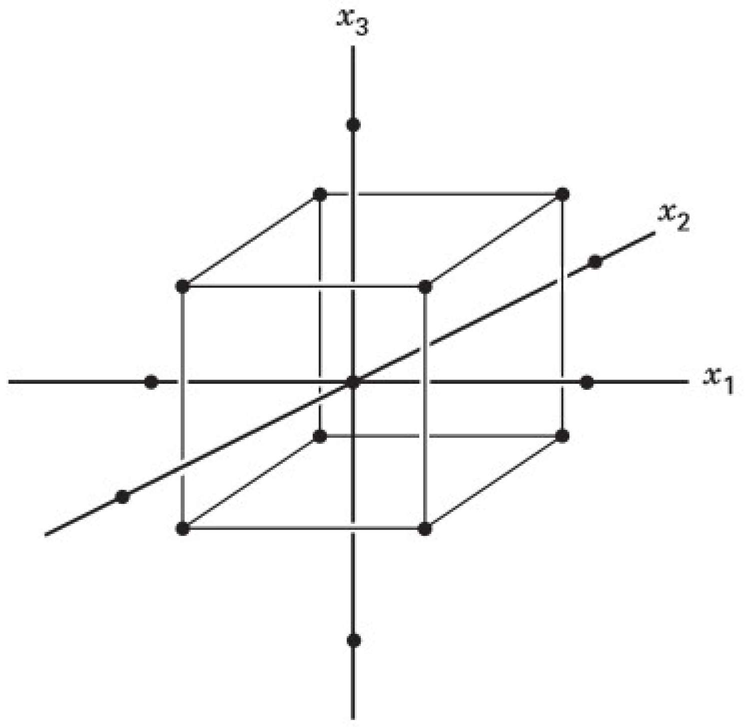


Figure 11 Montgomery Central Compsite Design Spacial Example

4.4.2.2. Selected Design

The XG Boost design had 5 continuous factors to test. The result is 52 total runs: 25 runs for a full two-level factorial, 10 replications at the center and 10 axial points. The RNN design had 3 selected factors, lookback, RNN Nodes and first layer(categoric). The design has 26 total runs: 23 runs for a full two-level factorial, 10 replications at the center and 8 axial points.

4.4.2.3. Analysis

The analysis was conducted in the same manner as the Screening Stage, except that two-way interactions and quadratic terms were included in the model. Contour plots of the interactions and non-linear effects were generated and useful in understanding the relationship of all factors. An optimization of the resulting regression equation provides the “best found” settings for the RSM stage.

4.4.2.4. RSM Test Values (52 runs/26 runs)

The axial values for XG Boost are shown as Low and High in Table 6. The axial values for the RNN are shown as Low and High in Table 7. These are the parameters that warranted further testing in the RSM stage and in some cases their ranges expanded.

Table 6 Boost Central Composite Parameters (52 runs)

Boosting Variable	**Description**	**Low**	**High**
Depth	Maximum Depth of Trees	4	8
Eta	Learning Rate	0	.4
Subsample	% of training data available to each new Tree	.7	1
Column Sample	% of factors available to each new Tree	.7	1
N_estimators	Maximum number of Trees built	50	250

Table 7 RNN Central Composite Parameters (26 runs)

RNN Variable	**Description**	**Low**	**High**
Lookback	# of minutes of prior values available	5	180
RNN Nodes	# of Nodes in LSTM/GRU layer	32	256
First Layer	Use a GRU or LSTM	GRU	LSTM

4.4.3. Random Search

In order to compare the benefit of the DOE process and establish a baseline for tuning benefits a Random Search was also conducted. Each algorithm used all 11 of its factors and randomly selected a value between the low and high settings during the screening stage. 88 randomly generated trials were conducted with XG Boost and 62 randomly generated trials were conducted with RNN. Models were built in the same fashion as DOE, where the XG Boost produces a single model per terminal, the RNN produces a single model for every terminal and both algorithms were tested across the 7-day validation period. The trial with the lowest average MAPE over the validation period was selected for holdout testing.

5. Results

5.1. Default Settings

The baseline comparison for each algorithm performance is against the MAPE of a simple 15-minute lag. It is also important to measure the accuracy gains of algorithm tuning, so the default settings of each parameter or a best guess were used to generate the default models. All settings are found in Table 8. The average MAPE using a 15-minute lag was 10.3% averaged across all terminals and validation days. Applying either algorithm at default settings was an improvement to 9.5% for XG Boost and 9.2% for RNN. Figure 11 shows the average MAPE over 7 days of each algorithm at default settings for each of the 7 terminals. Terminal 7 proved the most challenging to predict and was the only instance where either algorithm failed (XG Boost – 12.0%) to outperform the lag (Lag – 11.9%). Figure 12 shows an example of previous value forecasts vs an RNN forecast. The previous value repeats the pattern that occurred 15 minutes prior, while the RNN forecasting uses the prior examples to learn what values to expect.

Table 8 XG Boost & RNN Default Settings

XG Boost Variable	XG Boost Default	RNN Variable	RNN Default
RTN Drop	FALSE	Lookback	15
Lags	15	RNN Dropout	0
Day Lags	7	RTN Drop	TRUE
Depth	6	First Layer	LSTM
Eta	.3	Stateful	FALSE
Gamma	0	Dense Nodes	128
Subsample	1	RNN Nodes	128
Column Sample	1	Extra LSTM	TRUE
N_estimators	100	Extra Dense	TRUE
Basebias	.5	Dropout	.2
Seasonality	Weekly	Seasonality	Weekly

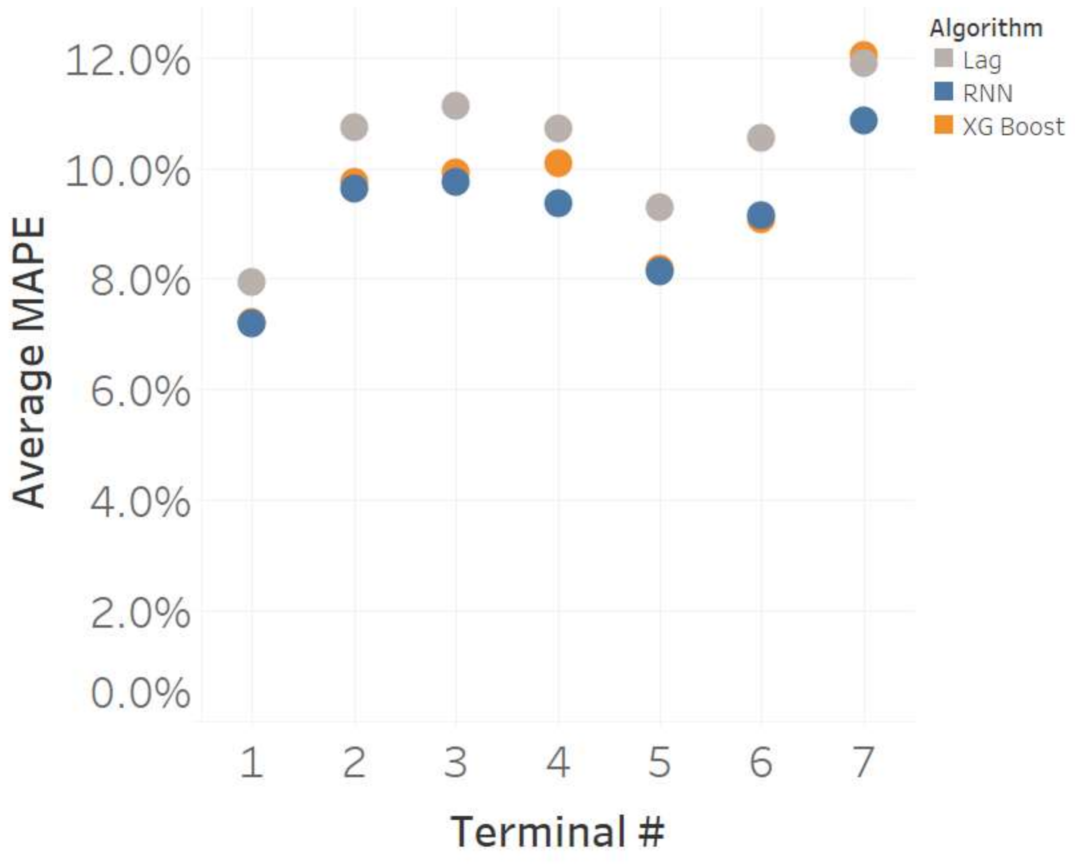


Figure 12 Default Model Performance by Terminal

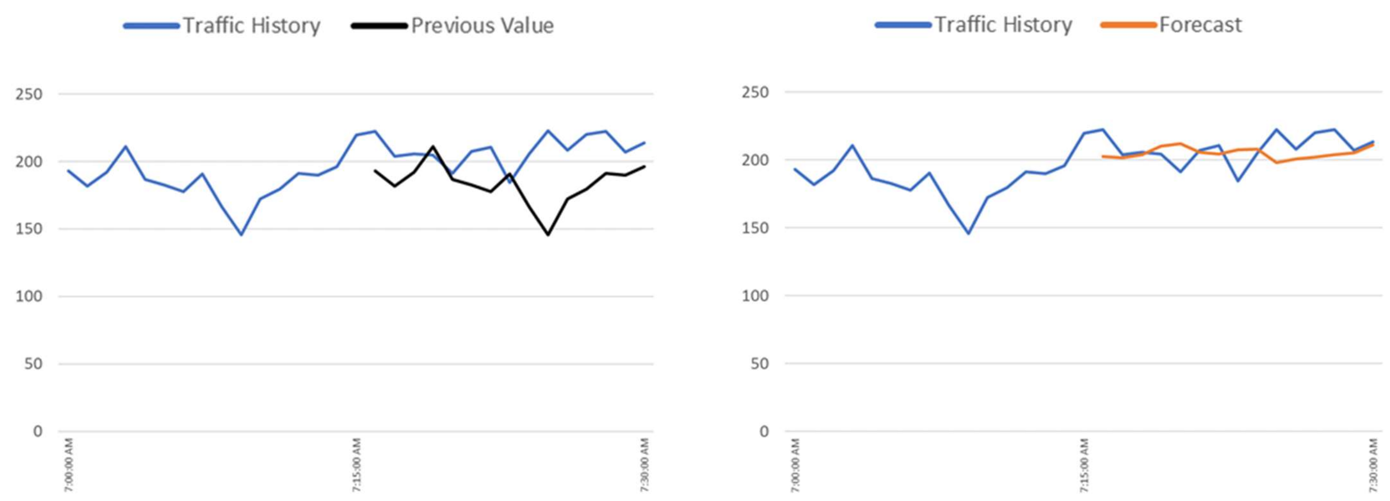


Figure 13 15-Minute Ahead Forecasts

5.2. Screening Results

5.2.1. XG Boost Screen

36 trials or 1764 total models are analyzed using MAPE as the response variable. Analysis of Variance (ANOVA) results in Table 9 show that 9 of the 11 factors have significant main effects. Only lags and daylags had p-values above .05. The results had a total sum of squares (SS) of .608 and the resulting model only accounted for .544 which gives the model an R2 of .89. Interestingly the vast majority of attributed model SS was to the blocked features: day and terminal. This means that which terminal and on what day has a much larger impact on the forecast's accuracy than the parameters of the algorithm. Each main effect is considered significant if the p-value from the F-Test in the ANOVA table is below .05. The linear effect of each significant factor is shown in Figure 12 with the optimal value of that parameter as the lowest MAPE value.

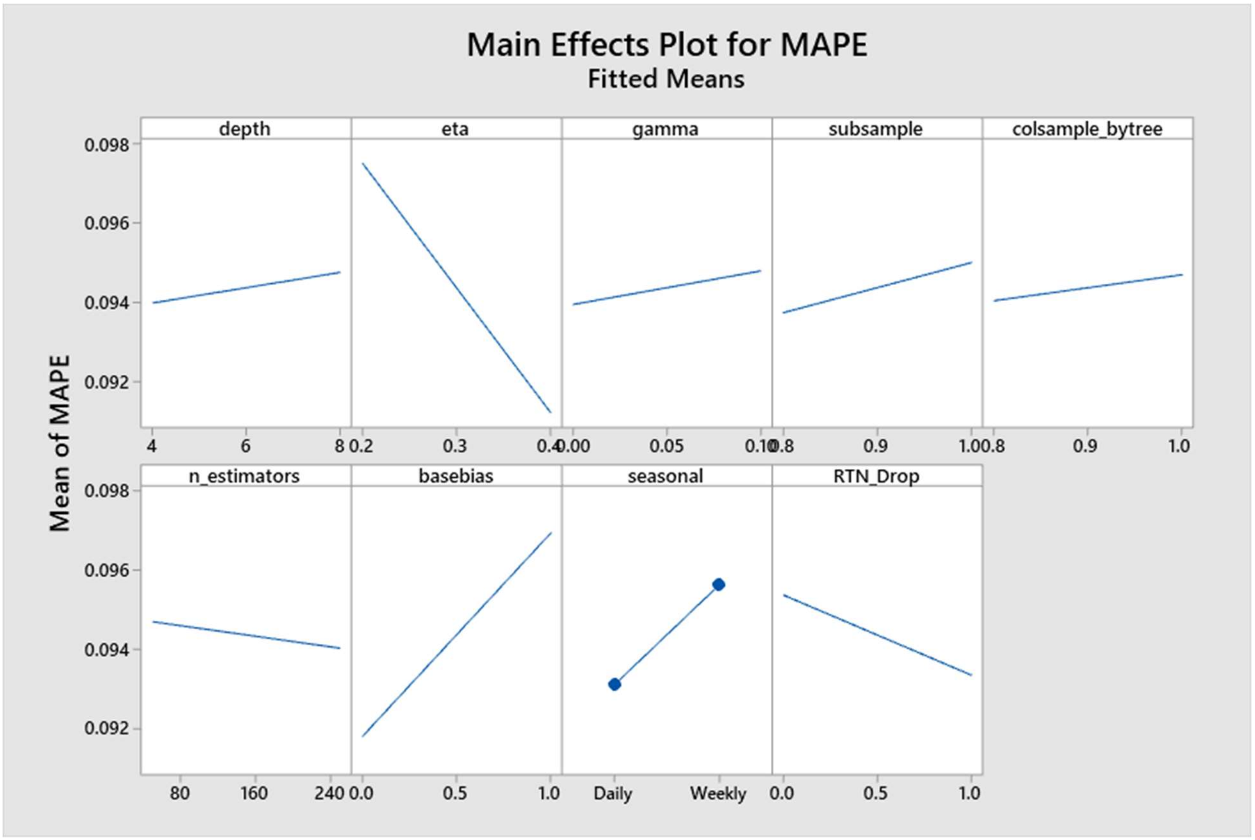
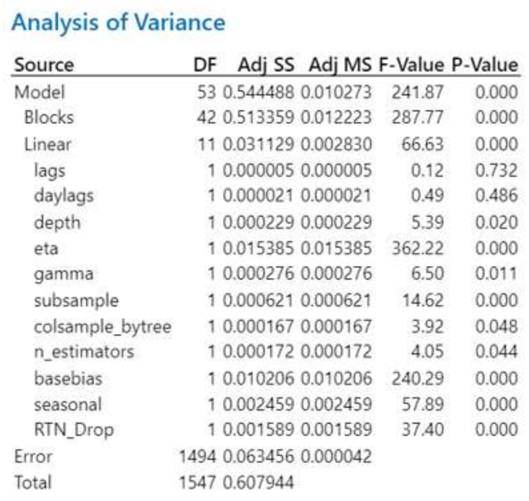


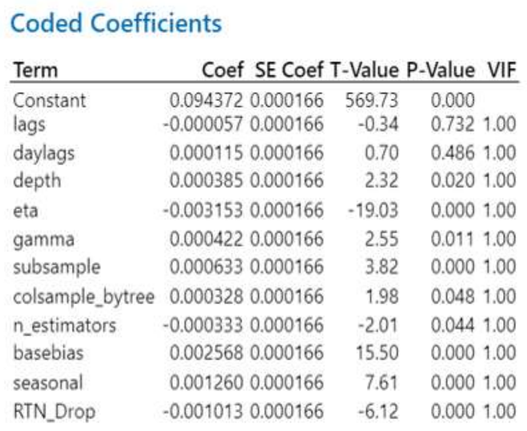
Figure 14 XG Boost Main Effects and Optimal Values

Table 9 XG Boost Screening ANOVA and Regression Coefficients



Analysis of Variance

Source	DF	Adj SS	Adj MS	F-Value	P-Value
Model	53	0.544488	0.010273	241.87	0.000
Blocks	42	0.513359	0.012223	287.77	0.000
Linear	11	0.031129	0.002830	66.63	0.000
lags	1	0.000005	0.000005	0.12	0.732
daylags	1	0.000021	0.000021	0.49	0.486
depth	1	0.000229	0.000229	5.39	0.020
eta	1	0.015385	0.015385	362.22	0.000
gamma	1	0.000276	0.000276	6.50	0.011
subsample	1	0.000621	0.000621	14.62	0.000
colsample_bytree	1	0.000167	0.000167	3.92	0.048
n_estimators	1	0.000172	0.000172	4.05	0.044
basebias	1	0.010206	0.010206	240.29	0.000
seasonal	1	0.002459	0.002459	57.89	0.000
RTN_Drop	1	0.001589	0.001589	37.40	0.000
Error	1494	0.063456	0.000042		
Total	1547	0.607944			



Coded Coefficients

Term	Coef	SE Coef	T-Value	P-Value	VIF
Constant	0.094372	0.000166	569.73	0.000	
lags	-0.000057	0.000166	-0.34	0.732	1.00
daylags	0.000115	0.000166	0.70	0.486	1.00
depth	0.000385	0.000166	2.32	0.020	1.00
eta	-0.003153	0.000166	-19.03	0.000	1.00
gamma	0.000422	0.000166	2.55	0.011	1.00
subsample	0.000633	0.000166	3.82	0.000	1.00
colsample_bytree	0.000328	0.000166	1.98	0.048	1.00
n_estimators	-0.000333	0.000166	-2.01	0.044	1.00
basebias	0.002568	0.000166	15.50	0.000	1.00
seasonal	0.001260	0.000166	7.61	0.000	1.00
RTN_Drop	-0.001013	0.000166	-6.12	0.000	1.00

5.2.1.1. Non-Impactful Factors

The parameters number of lags and number of prior day lags did not show a statistically significant main effect. The coefficient on lags is slightly negative, which means that increasing the number of minutes the algorithm can access results in a lower MAPE. The coefficient on daylags is slightly positive, which means that less day lags result in a lower MAPE. While neither coefficient is statistically significant, it does represent the unbiased estimator, and so the higher lag value and the lower day lag value are used for future tests and performance models.

5.2.1.2. Impactful Binary Factors

Of the 9 impactful effects, 2 are binary variables which only have categoric settings. The seasonal parameter tested a daily and a weekly seasonality correction. This showed a significant main effect with the daily option superior to weekly. RTN Drop is a True/False variable. When this parameter is True, the return data rate is dropped and not included in the model as an exogenous data stream. This showed a significant main effect with the option to drop the return data rate preferred and did not find performance improvement by including the return data rate in the model. The values of both these variables are held at daily and True for future tests and performance models.

5.2.1.3. Impactful Factors with set values

2 of the remaining 7 continuous variables had impactful main effects but do not warrant further testing. Gamma showed a significant main effect with the lower value of 0 being preferred. 0 is the default value of gamma, so this value is held, and no further testing of gamma is performed. Basebias showed a significant main effect as well. The preferred value is 0. Because the data is normalized, it centers on 0 so an initial estimate of 0 would be closest to an unbiased estimation.. Given more time and computing power, this factor would be included in further testing, however it is held at 0 and no further testing is done on basebias.

5.2.1.4. Impactful Factors

The remaining 5 factors: depth, eta, subsample, column sample and number of estimators had impactful main effects. None of their computed optimal values for this problem were the default settings of the algorithm or values that would cause the parameter to do nothing; ex. a value of 1 for column sample means that there is no subset sampling. These factors continue to the RSM stage for further testing and optimization.

5.2.1.5. XG Boost Screening Lessons Learned

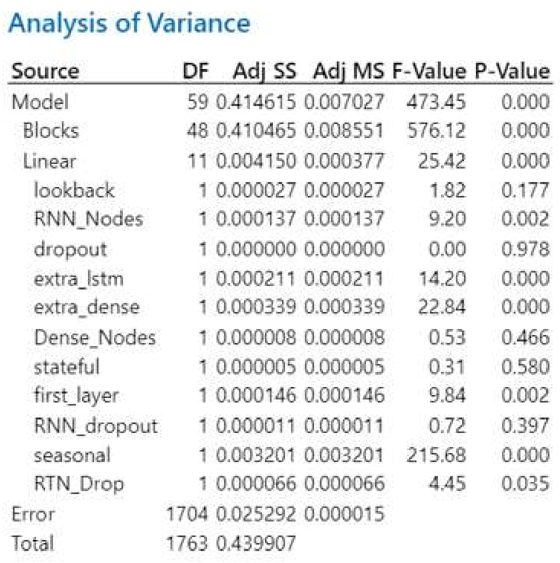
The test showed that the number of lags given to the model did not significantly impact its performance. This could suggest that the model only needs 10 minutes of history which corresponds with the partial autocorrelation of the traffic data. The results also suggest that there is no correlation between the previous day's traffic and the current day. This could be due to shifts in the daily patterns of the customers (ex. an aircraft takes off 10 minutes later than it did the previous day) or that correcting for a daily seasonal pattern removed the correlation from the previous day. The results strongly suggest that the return traffic rate is not a useful exogenous variable in the model.

The model parameters suggest that generalization is challenging. A lower depth builds more simple trees that are much less exacting (a depth 4 tree can have 16 end leaves while a depth 8 tree can have 256 leaves). A higher value of eta directly scales down the learning speed by limiting the impact of new trees. This higher value helps to prevent overfitting. Both subsample and column sample improved performance suggesting that there could be outliers in the data that can cause overfitting.

5.2.2. RNN Screen

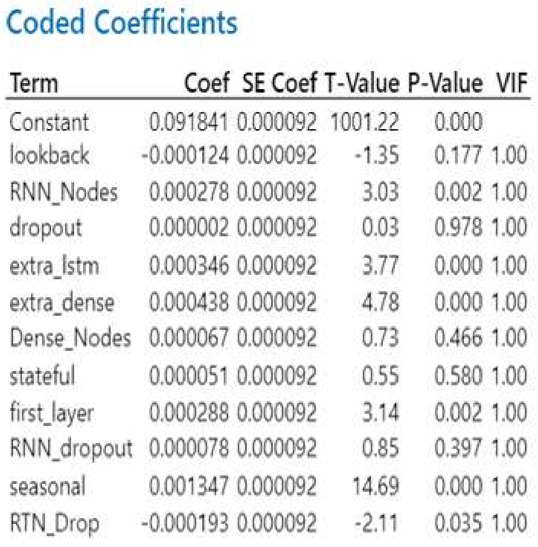
36 trials or 252 total models are analyzed using MAPE as the response variable. ANOVA shows that 6 of the 11 factors have significant main effects. Dropout, the number of dense nodes, lookback, statefulness and RNN dropout had p-values above .05. The results had a total SS of .440 and the resulting model only accounted for .415 which gives the model an R2 of .94. Similar to XG Boost the vast majority of attributed model SS was to the Blocks (day/terminal). This means that which terminal and on what day has a much larger impact on the forecast's accuracy than the parameters of the algorithm.

Table 10 RNN Screening ANOVA and Regression Coefficients



Analysis of Variance

Source	DF	Adj SS	Adj MS	F-Value	P-Value
Model	59	0.414615	0.007027	473.45	0.000
Blocks	48	0.410465	0.008551	576.12	0.000
Linear	11	0.004150	0.000377	25.42	0.000
lookback	1	0.000027	0.000027	1.82	0.177
RNN_Nodes	1	0.000137	0.000137	9.20	0.002
dropout	1	0.000000	0.000000	0.00	0.978
extra_lstm	1	0.000211	0.000211	14.20	0.000
extra_dense	1	0.000339	0.000339	22.84	0.000
Dense_Nodes	1	0.000008	0.000008	0.53	0.466
stateful	1	0.000005	0.000005	0.31	0.580
first_layer	1	0.000146	0.000146	9.84	0.002
RNN_dropout	1	0.000011	0.000011	0.72	0.397
seasonal	1	0.003201	0.003201	215.68	0.000
RTN_Drop	1	0.000066	0.000066	4.45	0.035
Error	1704	0.025292	0.000015		
Total	1763	0.439907			



Coded Coefficients

Term	Coef	SE Coef	T-Value	P-Value	VIF
Constant	0.091841	0.000092	1001.22	0.000	
lookback	-0.000124	0.000092	-1.35	0.177	1.00
RNN_Nodes	0.000278	0.000092	3.03	0.002	1.00
dropout	0.000002	0.000092	0.03	0.978	1.00
extra_lstm	0.000346	0.000092	3.77	0.000	1.00
extra_dense	0.000438	0.000092	4.78	0.000	1.00
Dense_Nodes	0.000067	0.000092	0.73	0.466	1.00
stateful	0.000051	0.000092	0.55	0.580	1.00
first_layer	0.000288	0.000092	3.14	0.002	1.00
RNN_dropout	0.000078	0.000092	0.85	0.397	1.00
seasonal	0.001347	0.000092	14.69	0.000	1.00
RTN_Drop	-0.000193	0.000092	-2.11	0.035	1.00

5.2.2.1. Non-Impactful Factors

Dropout had a p-value of near 1 and showed a rounded SS of 0. With a regression coefficient of near 0, this factor does not warrant further testing and is set to .2 (the default dropout value). The number of dense nodes did not have a significant main effect and warrants no further testing. The regression coefficient is slightly positive, so a lower value is preferred. Additionally, more nodes in the dense layer increases total training time, so the lower value is selected. The statefulness of the first RNN layer did not have a significant main effect and warrants no further testing. It's regression coefficient is slightly positive which means a value of False is preferred. RNN dropout did not have a significant main effect and warrants no further testing. Its regression coefficient is slightly positive, so a lower value is preferred. While lookback's p-value was not below .05, it is not as high as these other 4 factors. It is also one of only 2 remaining continuous factors and does warrant further examination.

5.2.2.2. Impactful Factors with set values

Extra LSTM layer, Extra Dense layer, return rate drop, and seasonality each had significant main effects and are binary variables that do not warrant further testing. With positive coefficients, neither of the extra layers was preferred. The daily seasonality model outperformed the weekly and it was beneficial to remove the return data rate from the input. These values are set for all further testing at the lower values in Figure 13.

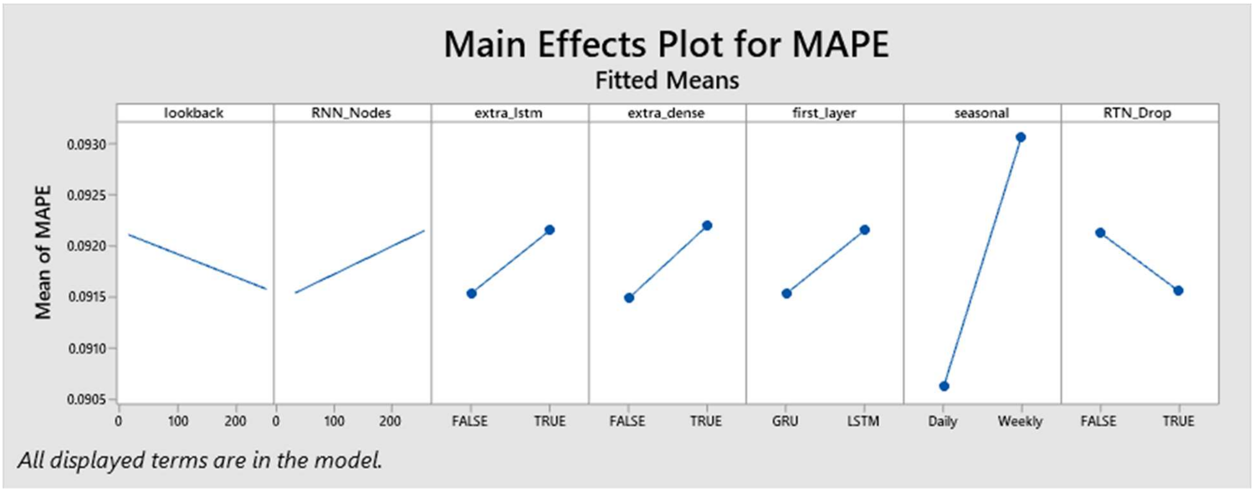


Figure 15 RNN Main Effects and Optimal Values

5.2.2.3. Other Impactful Factors

While GRU's outperformed LSTM's this is a fundamental architectural decision that deserves further exploration. While not statistically significant and heavily aliased with other effects, the two-way interactions are interesting to observe. In Figure 14, there is a possible interaction between the first layer and lookback and the first layer and RNN nodes. The LSTM seems to benefit more from having a higher lookback, while the GRU seems to perform better with less nodes. This screening test was not sufficient to accurately estimate these potential effects, so they are explored further in the next stage. The number of RNN nodes had a significant main effect with the smaller number of nodes preferred. This factor requires additional testing to optimize.

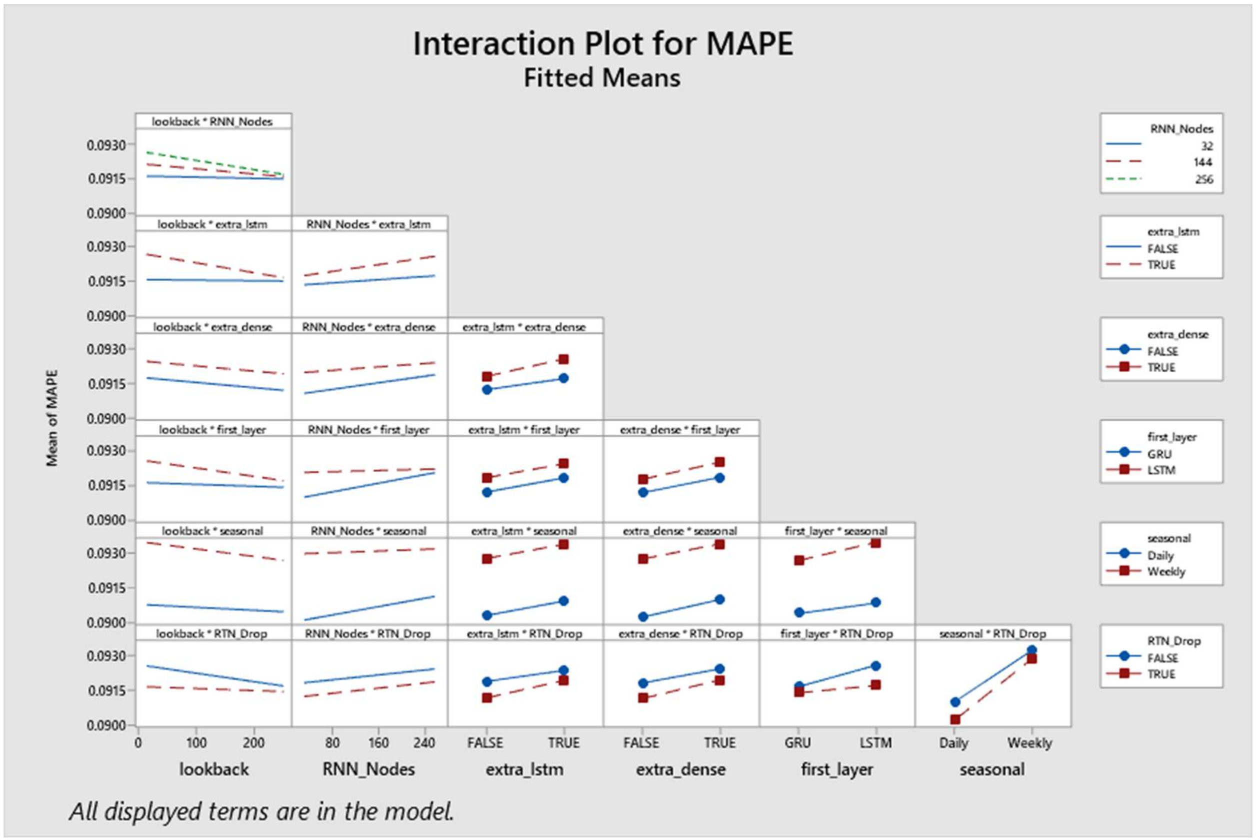


Figure 16 RNN Screening Interaction Plots

5.2.2.4. RNN Screening Lessons Learned

Similar to XG Boost the RNN results show that a daily seasonality improves the model and the return traffic rate is not a helpful exogenous variable. The most significant parameters found were binary and categoric (extra layers, first layer type, seasonality and return drop) which limits further testing unless those parameters have a second level interaction or a non-linear effect. Further testing is used to try and identify these more complex interactions and effects. The most interesting potential interaction found is the relationship between the number of nodes in the RNN layer and the type of layer. The results show that the number of nodes impacts GRU performance which could be due to the GRU having a simpler architecture and generalizing to the data better with less nodes. This relationship isn't found for the LSTM.

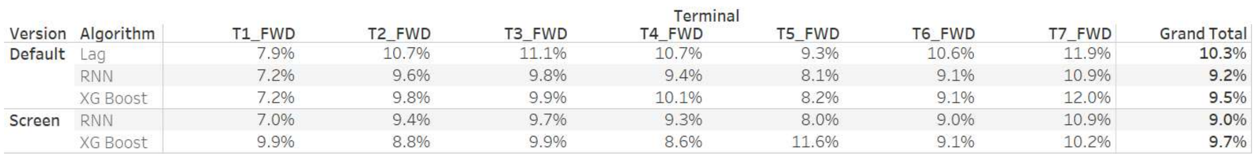
5.2.3. Best found

The "best found" settings after screening for both XG Boost and RNN are in Table 11 and the resulting MAPE by terminal for the validation period is in Table 12. The XG Boost regressed in performance from the default values from 9.5% to 9.7%. In terminals 1 and 5 the algorithm increased its MAPE by over 2%. The RNN improved overall from a 9.2% to a 9.0% average MAPE and either improved or had steady performance for all terminals.

Table 11 Screening Stage "Best Found" Parameters

XG Boost Variable	XG Boost Screening	RNN Variable	RNN Screening
RTN Drop	TRUE	Lookback	15
Lags	180	RNN Dropout	0
Day Lags	0	RTN Drop	TRUE
Depth	4	First Layer	GRU
Eta	.4	Stateful	FALSE
Gamma	0	Dense Nodes	32
Subsample	.8	RNN Nodes	32
Column Sample	1	Extra LSTM	FALSE
N_estimators	50	Extra Dense	FALSE
Basebias	0	Dropout	.2
Seasonality	Daily	Seasonality	Daily

Table 12 Screening "Best Found" Performance



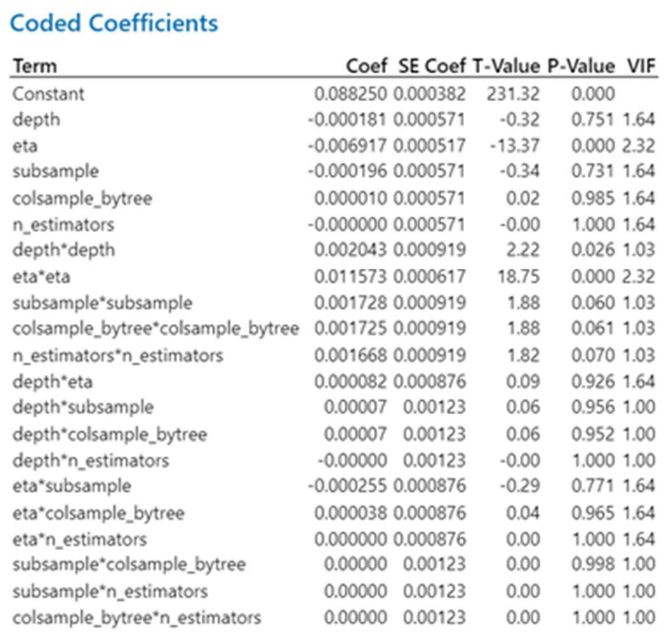
		Terminal							
Version	Algorithm	T1_FWD	T2_FWD	T3_FWD	T4_FWD	T5_FWD	T6_FWD	T7_FWD	Grand Total
Default	Lag	7.9%	10.7%	11.1%	10.7%	9.3%	10.6%	11.9%	**10.3%**
	RNN	7.2%	9.6%	9.8%	9.4%	8.1%	9.1%	10.9%	**9.2%**
	XG Boost	7.2%	9.8%	9.9%	10.1%	8.2%	9.1%	12.0%	**9.5%**
Screen	RNN	7.0%	9.4%	9.7%	9.3%	8.0%	9.0%	10.9%	**9.0%**
	XG Boost	9.9%	8.8%	9.9%	8.6%	11.6%	9.1%	10.2%	**9.7%**

5.3. Response Surface Results

5.3.1. XG Boost CCD

This test was 52 total trials and generated 2548 models. This test was 52 total trials and generated 2548 models. As shown in Table 13, of the 5 factors tested only eta's main effect and quadratic term and depth's quadratic term showed a statistically significant effect on MAPE. Although not statically significant at a .05 level, the other 3 factors possibly have a quadratic effect. All 5 factor's non-linear effects are shown in Figure 15 and a local minimum MAPE near the center of the design space. No two-way interactions were significant. The model R^2 was .91. This improvement over the .89 R^2 for the screening test could be due to allowing parameters to have a quadratic effect.

Table 13 XG Boost CCD Non-Linear Effects



Coded Coefficients

Term	Coef	SE Coef	T-Value	P-Value	VIF
Constant	0.088250	0.000382	231.32	0.000	
depth	-0.000181	0.000571	-0.32	0.751	1.64
eta	-0.006917	0.000517	-13.37	0.000	2.32
subsample	-0.000196	0.000571	-0.34	0.731	1.64
colsample_bytree	0.000010	0.000571	0.02	0.985	1.64
n_estimators	-0.000000	0.000571	-0.00	1.000	1.64
depth*depth	0.002043	0.000919	2.22	0.026	1.03
eta*eta	0.011573	0.000617	18.75	0.000	2.32
subsample*subsample	0.001728	0.000919	1.88	0.060	1.03
colsample_bytree*colsample_bytree	0.001725	0.000919	1.88	0.061	1.03
n_estimators*n_estimators	0.001668	0.000919	1.82	0.070	1.03
depth*eta	0.000082	0.000876	0.09	0.926	1.64
depth*subsample	0.00007	0.00123	0.06	0.956	1.00
depth*colsample_bytree	0.00007	0.00123	0.06	0.952	1.00
depth*n_estimators	-0.00000	0.00123	-0.00	1.000	1.00
eta*subsample	-0.000255	0.000876	-0.29	0.771	1.64
eta*colsample_bytree	0.000038	0.000876	0.04	0.965	1.64
eta*n_estimators	0.000000	0.000876	0.00	1.000	1.64
subsample*colsample_bytree	0.00000	0.00123	0.00	0.998	1.00
subsample*n_estimators	0.00000	0.00123	0.00	1.000	1.00
colsample_bytree*n_estimators	0.00000	0.00123	0.00	1.000	1.00

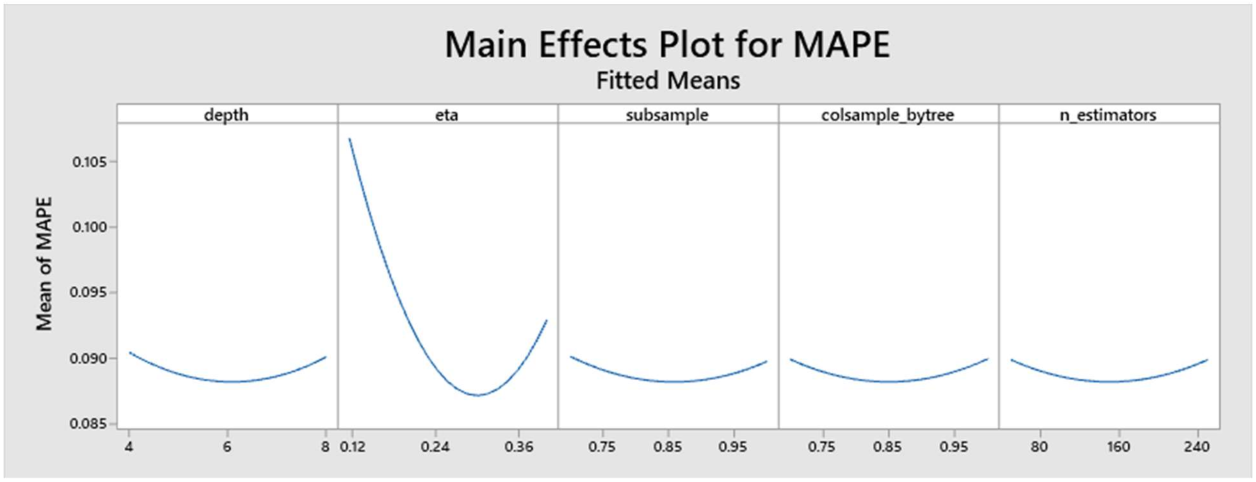


Figure 17 XG Boost CCD Non-Linear Effects

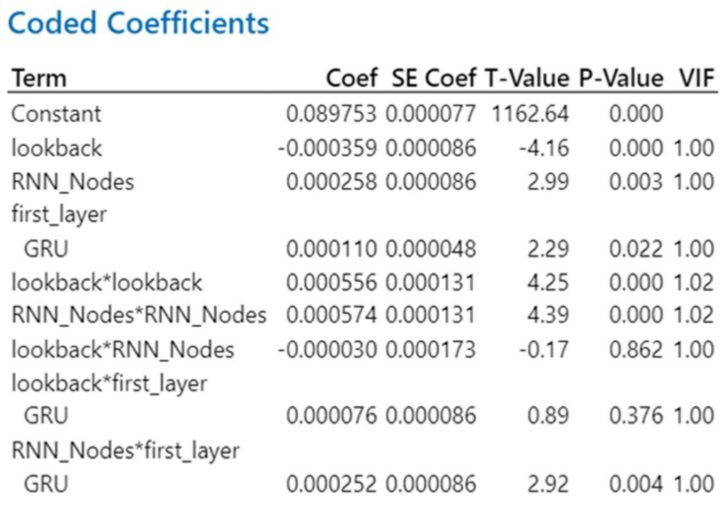
5.3.1.1. XG Boost CCD Lessons Learned

The strong linear effect of depth found in the screening stage is not significant in the CCD results and is replaced by a quadratic effect. The optimal depth value found is actually the algorithm's default depth of 6. This result confirmed the linear effect of eta found in the screening and added a quadratic effect as well. The final three parameters also replaced their linear effect with a quadratic effect. Generalization is still a challenge for the algorithm, but the parameters which slow learning are tuned down slightly by the results of this test.

5.3.2. RNN CCD

This test required 26 total trials and generated 182 models. As shown in Table 14 all 3 main effects were significant, the quadratic terms of lookback and RNN nodes were significant and the interaction between RNN Nodes and the first layer was significant. The model R2 was .99.

Table 14 RNN CCD Regression Coefficients



Coded Coefficients

Term	Coef	SE Coef	T-Value	P-Value	VIF
Constant	0.089753	0.000077	1162.64	0.000	
lookback	-0.000359	0.000086	-4.16	0.000	1.00
RNN_Nodes	0.000258	0.000086	2.99	0.003	1.00
first_layer					
GRU	0.000110	0.000048	2.29	0.022	1.00
lookback*lookback	0.000556	0.000131	4.25	0.000	1.02
RNN_Nodes*RNN_Nodes	0.000574	0.000131	4.39	0.000	1.02
lookback*RNN_Nodes	-0.000030	0.000173	-0.17	0.862	1.00
lookback*first_layer					
GRU	0.000076	0.000086	0.89	0.376	1.00
RNN_Nodes*first_layer					
GRU	0.000252	0.000086	2.92	0.004	1.00

5.3.2.1. RNN Non-Linear Effects

Lookback had a statistically significant quadratic term. The minimum of its effect shown in Figure 16 is at 127 minutes. Similarly RNN nodes had a significant quadratic term with a minimum at 144. The combined contour map of these two factors is shown in Figure 17 where the lighter region represents the combined minimum MAPE area for both parameters.

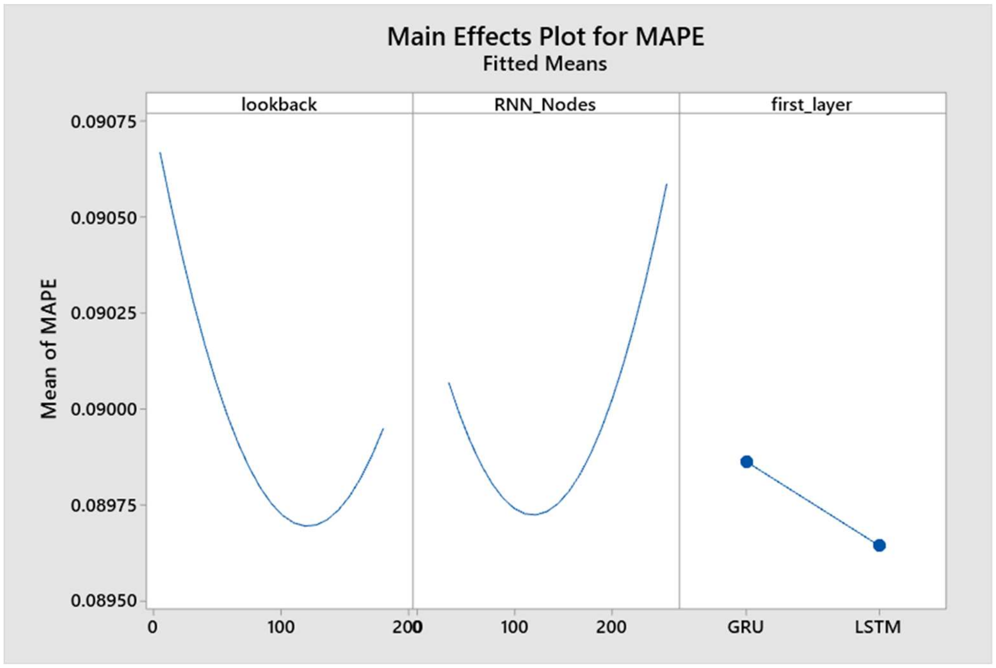


Figure 18 RNN CCD Non-Linear Effects

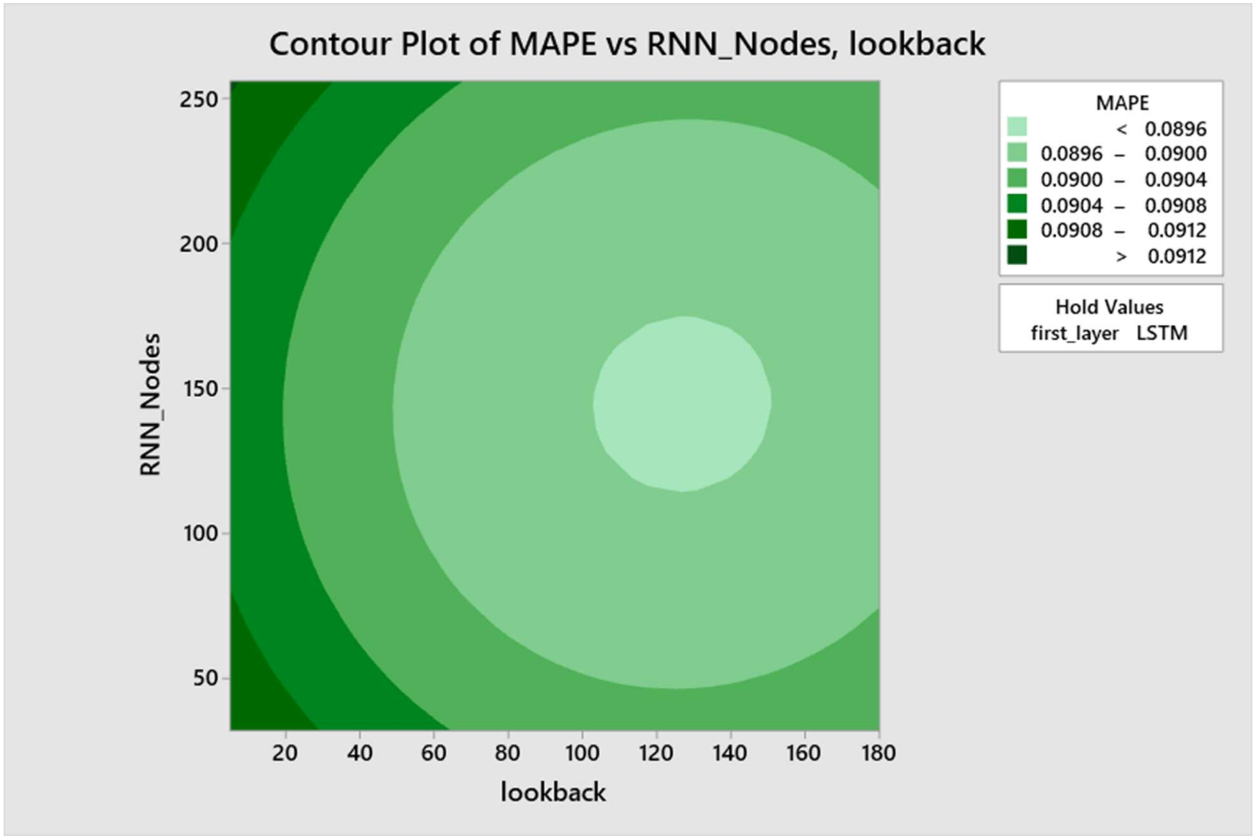


Figure 19 RNN CCD Contour Plot

5.3.2.2. RNN Two-Way Interactions

RNN nodes and the first layer of the network had a statistically significant interaction visible in Figure 18. The LSTM had a lower minimum MAPE that required a higher number of RNN nodes, while the GRU's minimum was overall higher but required a smaller number of nodes. Intuitively this makes sense because the LSTM is a more complex architecture and could require more paths through the network to better fit the data.

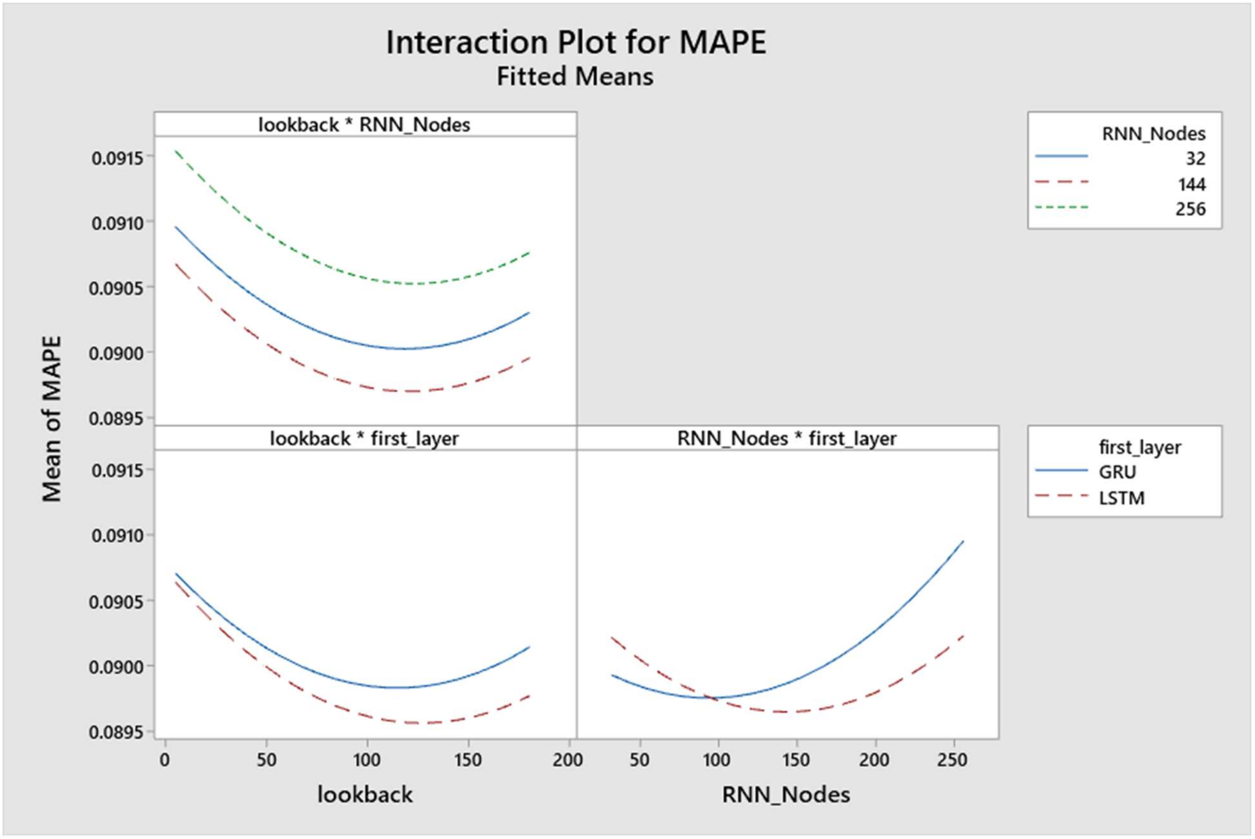


Figure 20 RNN CCD Interaction Plot

5.3.2.3. RNN CCD Lessons Learned

A more powerful test with less factors seems to reveal more about these three parameters. Lookback's linear effect is now significant and its quadratic effect is significant. The optimal lookback value of 127 minutes gives the model a long sequence to utilize. Because the algorithm uses full sequences to learn it makes sense that the lookback value is more similar to the autocorrelation vs the partial autocorrelation. The optimal lookback is shorter than the significant correlation (around 300 minutes) which could be due to the algorithm learning from correlations between terminals and not requiring additional lookback.

The interaction between the first layer type and the number of nodes is confirmed in this test. The more complex architecture of the LSTM improves performance when enough nodes are available for it to learn. While not statistically significant, the interaction between the first layer type and lookback follows the same logical pattern. The LSTM makes use of a longer lookback more effectively than the GRU.

5.3.3. Best found

The "best found" settings after RSM for both XG Boost and RNN are in Table 15 and the resulting MAPE by terminal for the validation period is in Table 18. The XG Boost improved in performance from the screening values from 9.7% to 8.9%. Performance on every terminal improved from screening but terminals 1 and 5 still have a higher MAPE than the default settings. The RNN improved from the screening stage very slightly from a 9.02% to an 8.98% average MAPE.

Table 15 RSM Stage "Best Found" Parameters

XG Boost Variable	XG Boost RSM	RNN Variable	RNN RSM
RTN Drop	TRUE	Lookback	127
Lags	180	RNN Dropout	0
Day Lags	0	RTN Drop	TRUE
Depth	6	First Layer	LSTM
Eta	.3	Stateful	FALSE
Gamma	0	Dense Nodes	32
Subsample	.86	RNN Nodes	144
Column Sample	.84	Extra LSTM	FALSE
N_estimators	150	Extra Dense	FALSE
Basebias	0	Dropout	.2
Seasonality	Daily	Seasonality	Daily

5.4. Random Search Results

5.4.1. XG Boost

88 unique parameter settings were randomly generated and tested over the validation period. The average MAPE of all 88 combinations was 9.3% and the best performing trial was 8.9%. The settings for the best performing trial are shown in Table 16. Also in Table 16, is a comparison of the top 25% of trial's parameters to all 62 trial parameters. For continuous factors the average value of each group is shown and for categoric factors the percentage with the same

value as the "best found" in RSM is shown. Every trial in the Top 25% used a Daily seasonality. No other factors seem to be overrepresented in the Top 25% of trials.

Table 16 XG Boost Random Search Results

Parameter	Evaluation	Top 25%	All Trials	"Best Found"
RTN Drop	% TRUE	50%	45%	TRUE
Lags	Average Value	94	91	138
Day Lags	Average Value	3	3	4
Depth	Average Value	6	6	6
Eta	Average Value	0.35	0.30	.36
Gamma	Average Value	0.06	0.05	.05
Subsample	Average Value	0.90	0.91	.88
Column Sample	Average Value	0.90	0.90	.95
N_estimators	Average Value	144	143	218
Basebias	Average Value	.45	.46	.25
Seasonality	% Daily	100%	50%	Daily
Average MAPE	Average Value	9.0%	9.3%	8.9%

5.4.2. RNN

62 unique parameter settings were randomly generated and tested over the validation period. The average MAPE of all 62 combinations was 9.2% and the best performing trial was 8.9%. The settings for the best performing trial are shown in Table 17. Also in Table 17, is a comparison of the top 25% of trial's parameters to all 62 trial parameters. For continuous factors the average value of each group is shown and for categoric factors the percentage with the same value as the "best found" in RSM is shown. A value of FALSE is overrepresented in the Top 25% for both the extra LSTM and extra Dense layers. Every trial in the Top 25% used a Daily seasonality. These results seem to confirm the RNN Screening results for main effects of these factors.

Table 17 RNN Random Search Results

Parameter	Evaluation	Top 25%	All Trials	"Best Found"
lookback	Average Value	147	142	146
RNN_Nodes	Average Value	133	141	42
dropout	Average Value	0.12	0.09	.01
extra_lstm	% FALSE	81%	53%	FALSE
extra_dense	% FALSE	75%	50%	FALSE
Dense_Nodes	Average Value	143	149	52
stateful	% FALSE	69%	50%	TRUE

first_layer	% LSTM	44%	48%	GRU
RNN_dropout	Average Value	0.10	0.11	.04
seasonal	% Daily	100%	52%	Daily
RTN_Drop	% TRUE	50%	60%	TRUE
Average of MAPE	Average Value	9.0%	9.2%	8.9%

5.5. "Best Found" Settings Validation Period Results

The parameter settings for each stage's best found values, the default parameters and the random search's best found settings are in Tables 18 and 19.

Table 18 XG Boost Best Found Parameter Values

XG Boost Variable	Default	Screening	RSM	Random
RTN Drop	FALSE	TRUE	TRUE	TRUE
Lags	15	180	180	138
Day Lags	7	0	0	0.4
Depth	6	4	6	6
Eta	0.3	0.4	0.3	0.36
Gamma	0	0	0	0.05
Subsample	1	0.8	0.86	0.88
Column Sample	1	1	0.84	0.95
N_estimators	100	50	150	218
Basebias	0.5	0	0	0.25
Seasonality	Weekly	Daily	Daily	Daily

Table 19 RNN Best Found Parameter Values

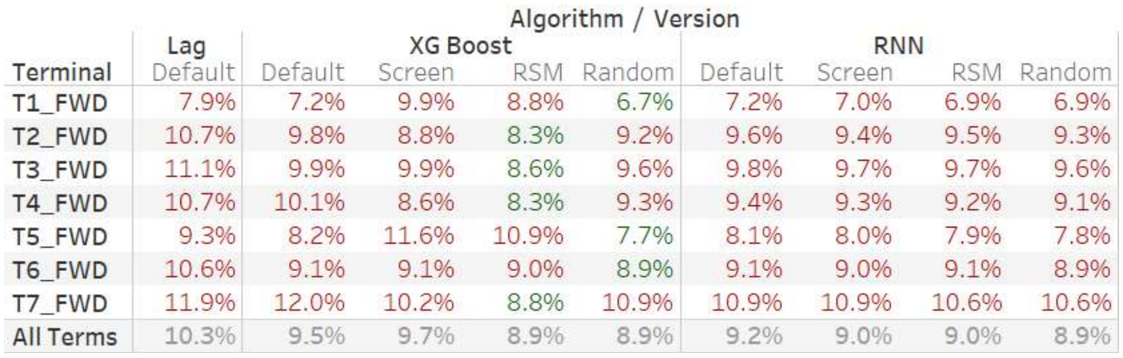
RNN Variable	Default	Screening	RSM	Random
Lookback	15	15	127	146
RNN Dropout	0	0	0	0.04
RTN Drop	TRUE	TRUE	TRUE	TRUE
First Layer	LSTM	GRU	LSTM	GRU
Stateful	FALSE	FALSE	FALSE	TRUE
Dense Nodes	128	32	32	52

RNN Nodes	128	32	144	42
Extra LSTM	TRUE	FALSE	FALSE	FALSE
Extra Dense	TRUE	FALSE	FALSE	FALSE
Dropout	0.2	0.2	0.2	0.01
Seasonality	Weekly	Daily	Daily	Daily

5.5.1. Best Performance by Terminal

Table 20 shows the average MAPE results over the 7-day validation period for each algorithm's best parameters after each stage. Highlighted in green is the algorithm and stage with the lowest average MAPE. XG Boost had the lowest average MAPE for every terminal. The RSM parameters were best for 4 terminals and the Random parameters were best for the other 3 terminals. There is a large disparity between the performance of these two parameter settings for XG Boost by terminal. When RSM is accurate, the Random settings are a full percentage point higher. The reserve is typically true with over 3% disparity between RSM and Random for terminal #5.

Table 20 Validation Period MAPE by Terminal & Algorithm



	Algorithm / Version								
	Lag	XG Boost				RNN			
Terminal	Default	Default	Screen	RSM	Random	Default	Screen	RSM	Random
T1_FWD	7.9%	7.2%	9.9%	8.8%	6.7%	7.2%	7.0%	6.9%	6.9%
T2_FWD	10.7%	9.8%	8.8%	8.3%	9.2%	9.6%	9.4%	9.5%	9.3%
T3_FWD	11.1%	9.9%	9.9%	8.6%	9.6%	9.8%	9.7%	9.7%	9.6%
T4_FWD	10.7%	10.1%	8.6%	8.3%	9.3%	9.4%	9.3%	9.2%	9.1%
T5_FWD	9.3%	8.2%	11.6%	10.9%	7.7%	8.1%	8.0%	7.9%	7.8%
T6_FWD	10.6%	9.1%	9.1%	9.0%	8.9%	9.1%	9.0%	9.1%	8.9%
T7_FWD	11.9%	12.0%	10.2%	8.8%	10.9%	10.9%	10.9%	10.6%	10.6%
All Terms	10.3%	9.5%	9.7%	8.9%	8.9%	9.2%	9.0%	9.0%	8.9%

5.5.2. Parameter Tuning Impacts

XG Boost benefitted the most from parameter tuning and lowered its average MAPE over all terminals from 9.5% to 8.9%. The RNN also benefitted from parameter optimization but showed improvement from the Default (9.2%) to Screening (9.0%) and minimal improvement from the RSM (9.0%). The Random parameters for RNN outperformed Screen and RSM on every terminal.

5.6. Holdout Performance

The previously untouched 31-day holdout period data provides an opportunity to test each best found setting for generalization to estimating traffic for each terminal.

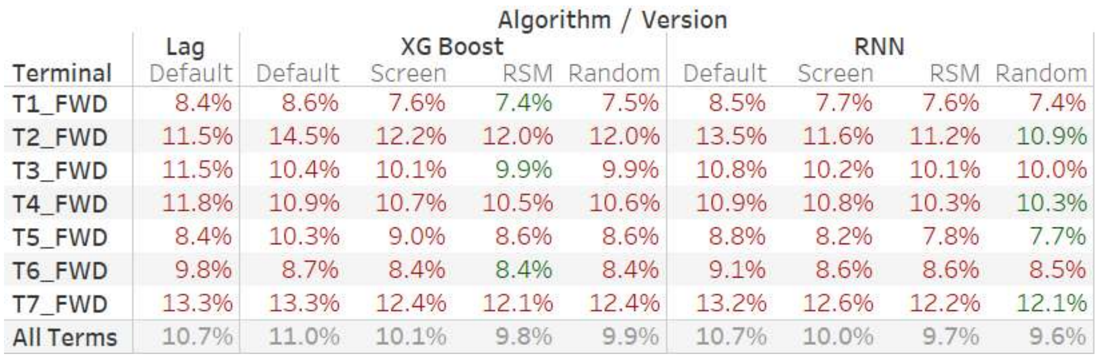
5.6.1. Best Performance by Terminal on Holdout

RNNs performed better over the holdout period than XG Boost. When comparing the best performing parameter settings, the RNN performed better than XG Boost on 4 of 7 terminals. Additionally, on the terminals where XG Boost had a lower MAPE, the gap between the two algorithms was very small (within 0.1%). The gap between the algorithms when the RNN had a lower MAPE was similarly small for 2 terminals but was nearly a full percentage point for the remaining two. This caused the overall average MAPE of the RNN (9.6%) to be slightly lower than XG Boost (9.8%).

5.6.2. Parameter Tuning Generalization

The impact of parameter tuning is more obvious in the holdout results. XG Boost improved its performance on the holdout period at every stage, while it did not show improvement in the Screening stage for the validation period. The magnitude of overall average improvement also increased and from Default to RSM the algorithm reduced MAPE by 1.2%. RNN also improved its performance on the holdout period at every stage. The tuning benefits of the RNN from Default to RSM are five times larger on the holdout period than the Validation period. The Random Search parameters performed just as well or slightly better than the RSM tuning; a result that is very similar to the Validation period.

Table 21 Holdout MAPE by Terminal & Algorithm



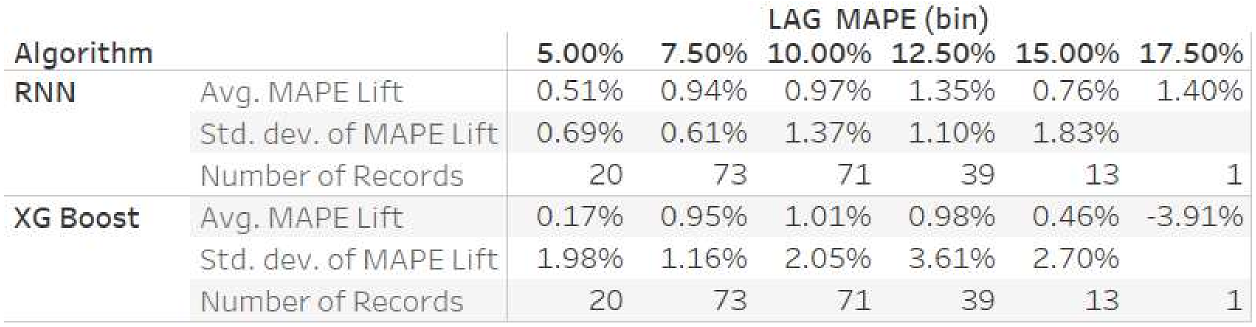
	Algorithm / Version								
	Lag	XG Boost				RNN			
Terminal	Default	Default	Screen	RSM	Random	Default	Screen	RSM	Random
T1_FWD	8.4%	8.6%	7.6%	7.4%	7.5%	8.5%	7.7%	7.6%	7.4%
T2_FWD	11.5%	14.5%	12.2%	12.0%	12.0%	13.5%	11.6%	11.2%	10.9%
T3_FWD	11.5%	10.4%	10.1%	9.9%	9.9%	10.8%	10.2%	10.1%	10.0%
T4_FWD	11.8%	10.9%	10.7%	10.5%	10.6%	10.9%	10.8%	10.3%	10.3%
T5_FWD	8.4%	10.3%	9.0%	8.6%	8.6%	8.8%	8.2%	7.8%	7.7%
T6_FWD	9.8%	8.7%	8.4%	8.4%	8.4%	9.1%	8.6%	8.6%	8.5%
T7_FWD	13.3%	13.3%	12.4%	12.1%	12.4%	13.2%	12.6%	12.2%	12.1%
All Terms	10.7%	11.0%	10.1%	9.8%	9.9%	10.7%	10.0%	9.7%	9.6%

5.6.3. Algorithm Lift Noise

MAPE Lift is defined as the difference between the Lag MAPE and the algorithm's MAPE for that day and terminal and is used to show the benefit of an algorithm over the baseline Lag. Using this lift value reduces the variance from each day and terminal, allowing comparison of algorithm performance. During the holdout period, the Random RNN had a more consistent performance than the RSM XG Boost. Table 22 shows the average MAPE Lift and standard deviation for days and terminal combinations separated by the average Lag MAPE. The number of records, one record for each terminal each day, is shown for each bucket. When the Lag MAPE is below 5%, the RNN on average outperforms XG Boost and has a much lower standard deviation for these 20 observations. In each bucket of Lag MAPE, the RNN has a lower standard deviation and a higher or very equivalent average lift. One hypothesis for this performance is the utilization of "forget gates" in the LSTM. The algorithm learns which values to discard from learning and can ignore information from some outliers.

An example of this is found in Terminal #2's traffic decline at the start of the holdout period. The traffic is consistent until the 2nd day of the holdout. There it drops significantly and then establishes a new pattern. The difference between the lagged MAPE and the average MAPE for the best performing RNN and XG Boost are shown in Figure 19. Performance is shown as MAPE lift over lag for ease of comparing different day's performances. When the traffic declines in the 2nd day of the holdout, the XG Boost model has an average MAPE up to 20% larger than the pure lag. During the same period, the RNN peaks at less than 4% larger than lag. Both algorithms eventually return to a state of outperforming the lagged traffic, but the RNN suffers less performance degradation when traffic is in flux.

Table 22 MAPE Lift by Average of Lagged MAPE



		LAG MAPE (bin)					
Algorithm		5.00%	7.50%	10.00%	12.50%	15.00%	17.50%
RNN	Avg. MAPE Lift	0.51%	0.94%	0.97%	1.35%	0.76%	1.40%
	Std. dev. of MAPE Lift	0.69%	0.61%	1.37%	1.10%	1.83%	
	Number of Records	20	73	71	39	13	1
XG Boost	Avg. MAPE Lift	0.17%	0.95%	1.01%	0.98%	0.46%	-3.91%
	Std. dev. of MAPE Lift	1.98%	1.16%	2.05%	3.61%	2.70%	
	Number of Records	20	73	71	39	13	1

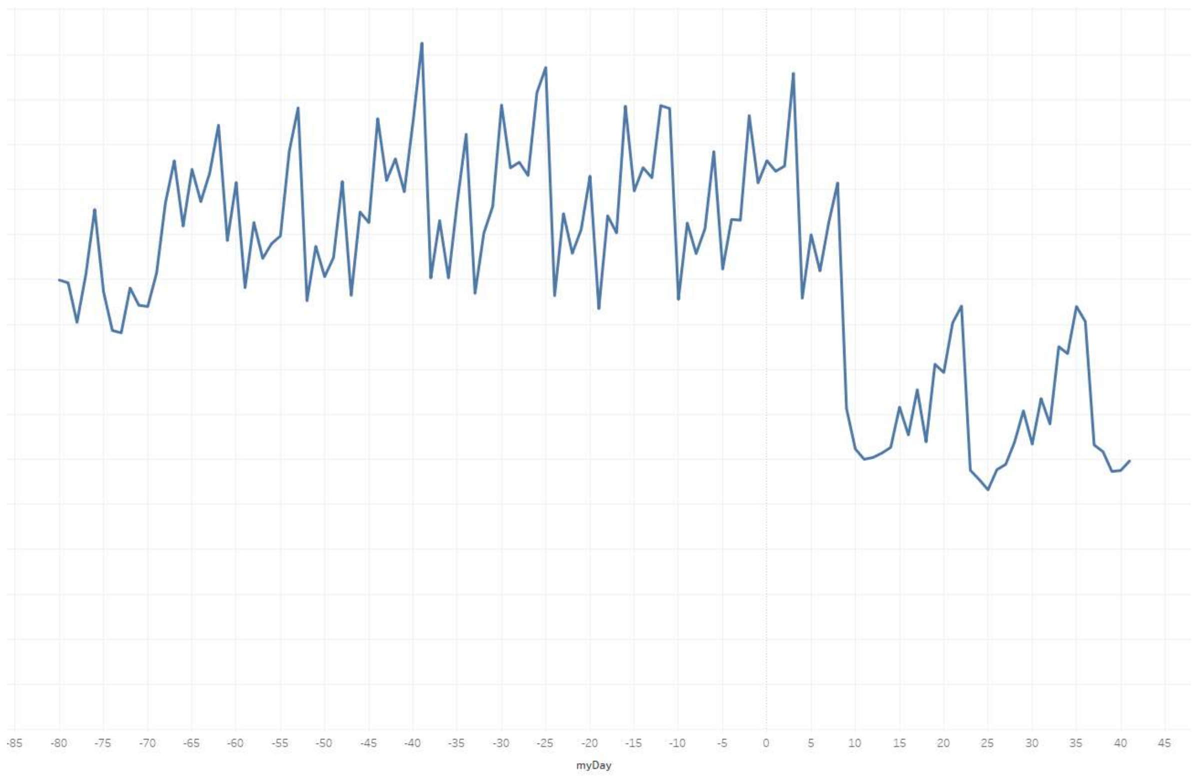


Figure 21 Terminal 2 Daily Average Traffic

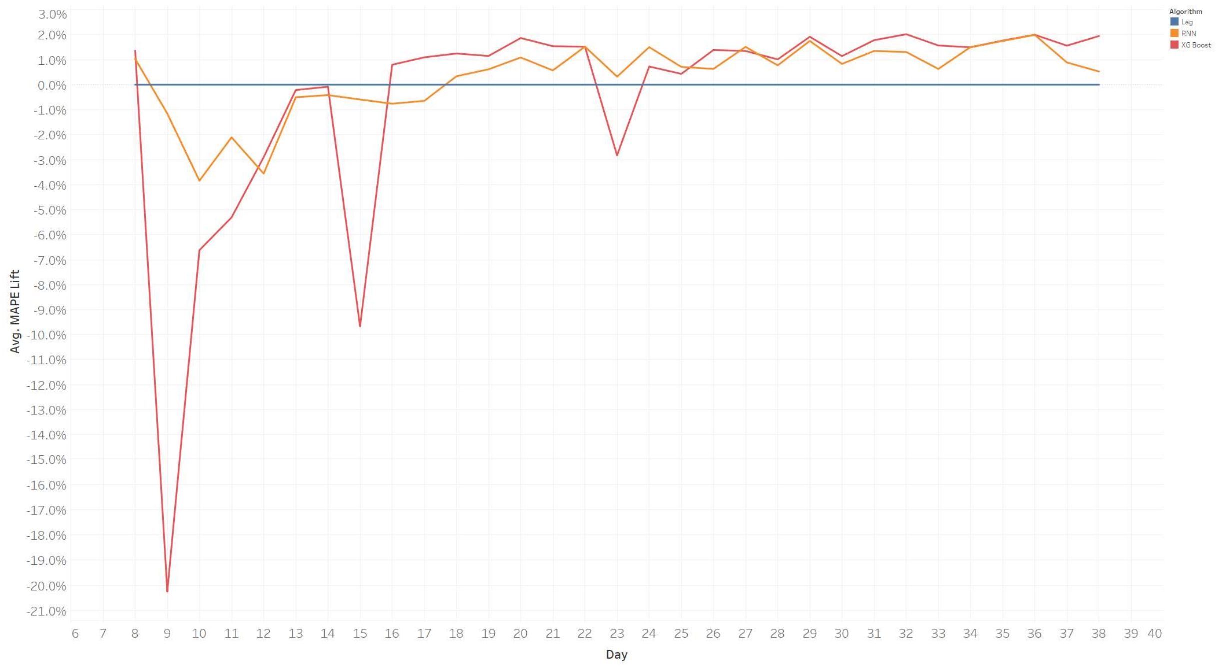


Figure 22 Terminal 2 Best Algorithm Lift Over Lag

6. Conclusion

This thesis showed the improvement of more sophisticated time-series forecasting algorithms for short term traffic estimation in a state-of-the-art satellite communications system. When implementing a Dynamic Resource Management system to take advantage of flexible communications satellites it is desirable to reduce the error introduced by the system cycle rate and changing traffic. Short term traffic forecasting is more accurate than using the most recent available traffic data. XG Boost models and RNN models provided approximately a 1% improvement over using the most recent traffic when solution delay is set to 15 minutes. This equates to a 1% more efficient allocation system or the ability to serve an additional 1% of user traffic.

6.1. Which tuning worked the best

Using design of experiments in a two staged approach with a screening test followed by a response surface test to optimize final parameters reduced the average MAPE for each algorithm at each stage. If both algorithms were only built with the default settings, neither would outperform using the previous traffic data and the entire system would not require a Traffic Estimator. Going through a tuning process improved the models produced by each algorithm to a performance level that makes a Traffic Estimator worthwhile. Random Search also provided the same benefit in terms of performance gain over the Default settings. Using the same number of runs, a random search produced a parameter setting that was either as good or better than the DOE approach.

6.1.1. Benefits of Random Search

Random Search is very easy to implement. The coding requirement and statistical knowledge is minimal. It is easy to test a larger number of factors or test factors over a larger range. Because there is no symmetry or orthogonality requirements in trials, adding a new factor or increasing the range during testing does not diminish the value of already concluded trials. The main drawback to this approach is a lack of statistical explanation of why one parameter setting is better than another.

6.1.2. Benefits of Design of Experiments

DOE provides a structured method of testing algorithm parameters and then yields statistically relevant impacts of each factor. These impacts allow the user to explain why certain parameter values are included in the final algorithm. It also provides insightful information for interactions between factors that is easily missed when conducting random search. This approach does require additional knowledge and extra steps in setting up which trials to test and to evaluate the results.

6.2. Which algorithm worked the best

The RNN using a GRU resulted in the lower average MAPE and better robustness over the holdout period. The algorithm performance could improve with the addition of more terminals and relevant exogenous data. While the RNN is more resilient to changes in the traffic patterns than XG Boost, it was still outperformed by the lagged value during some of these periods. An implementation of this system should track model performance and include a way to switch off the RNN and revert to a simpler algorithm during inconsistent periods of traffic.

6.3. Future Work

There is still room for improvement in this application of forecasting and more efficiency to gain through estimation, though this work can serve as a baseline evaluation of expected performance. This thesis also compared two strategies of parameter tuning: Design of Experiments and Random Search. Neither approach strongly dominated the other in this example, but there could be other approaches that would perform as well or better. This is an on-going field of research that will get more and more attention as model's are squeezed for every bit of performance possible. Future works should use multiple approaches for tuning to compare the value of each. Simply conducting one tuning approach and quoting its benefit does not further this field.

6.3.1. More Terminals

RNN algorithms allow for cross terminal learning and benefit. A sequence of traffic seen in Terminal #1 can influence the estimation in Terminal #2 and improve its forecast. This work had 7 total terminals. A full constellation of O3-b mPower satellites could have hundreds to thousands of terminals active at any time. The addition of more terminals to the model should cause an improvement in overall performance. If computing power and time is prohibitive, it could also be possible to cluster similar, or more beneficial terminals, together in a model and create a single model for each cluster within the constellation. Cluster generation techniques based on traffic levels, usage patterns, location or terminal type should be explored or a gradient based technique could be used to add the most beneficial terminals to a cluster until a computational limit is reached.

6.3.2. Weather/Flight Data

The return data rate seemed to have minimal benefit in forecasting the forward traffic rate. However, there could be other exogenous variables that would have a stronger impact on model performance. The weather at the terminal location is the most direct example. Typically satellite communications are degraded due to rain attenuation. The inclusion of a weather model into the traffic data could aid by forecasting when rain would be present and would lower the usage of that terminal. If the terminal is an aircraft, the scheduled take off and landing of that plane would also be beneficial to the model. More exogenous variables should be explored in these types of models by SES or in an academic setting if the anonymity of the users can be protected.

6.3.3. Unmet Traffic

This thesis assumed that the error from a low forecast, where a user is given a lower data rate than they trafficked, is equivalent to a high forecast, where a user is given more capacity then they needed. This assumption may change depending on the situation. Some users may need to have their full traffic met at any time, while some may be satisfied with their full traffic going unmet. One approach to this problem could be to use an asymmetrical loss function in the traffic estimator. This would penalize either type of error more than the other and cause the forecast to try and minimize the more important error type.

6.3.4. Integration with RTE

The traffic estimator should be simulated with the RTE allocation software to test for any interaction effects of these two systems. There may be constraints on either system that prevent the other from optimal performance. For example, if a terminal has a large increase in traffic forecasted, but the RTE has a limitation on the change to power from cycle to cycle then the overall traffic could go unmet. Future work could include a multi-step forecast to identify these upcoming large changes and to smooth them out over time to accommodate the limitations of the physical system.

1. Bibliography

[1] S. O'Dea, "Number of smartphone users worldwide from 2016 to 2021," Statista, 28 Feb 2020. [Online]. Available: https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/. [Accessed 2020].

[2] J. Constine, "Facebook hits 2.5B users in Q4 but shares sink from slow profits," Techcrunch, 29 Jan 2020. [Online]. Available: https://techcrunch.com/2020/01/29/facebook-earnings-q4-2019/.

[3] M. Anderson, A. Perrin, J. Jiang and M. Kumar, "10% of Americans don't use the internet. Who are they?," *Factank,* 2019.

[4] J. Engebretson, "iGR: Average Monthly Broadband usage is 190 Gigabytes Monthly Per Household," telecompetitor, September 2016. [Online]. Available: https://www.telecompetitor.com/igr-average-monthly-broadband-usage-is-190-gigabytes-monthly-per-household/.

[5] J. Hindy, "How much data does YouTube actually use?," Android Authority, June 2019. [Online]. Available: https://www.androidauthority.com/how-much-data-does-youtube-use-964560/.

[6] S. O'Dea, "Estimated internet traffic in the United States from 2018 to 2023," Statista, October 2018. [Online]. Available: https://www.statista.com/statistics/216335/data-usage-per-month-in-the-us-by-age/.

[7] M. Beech, "COVID-19 Pushes Up Internet use 70% And Streaming more than 12%," Forbes, 25 March 2020. [Online]. Available: https://www.forbes.com/sites/markbeech/2020/03/25/covid-19-pushes-up-internet-use-70-streaming-more-than-12-first-figures-reveal/#7ee85d3f3104.

[8] J. Horrigan, "Broadband Adoption and use in America," *Federal Communications Commission,* 2010.

[9] United States Census Bureau, "New Census Data Show Differences Between Urban and Rural Populations," CB16-210, 2016.

[10] SES, "Unleashing the Potential of an Empowered World with the Launch of O3b mPOWER," 11 Sep 2017. [Online]. Available: https://www.ses.com/sites/default/files/2017-09/170908_SES%20Launches%20O3b%20mPOWER_FINAL.pdf.

[11] Morgan Stanley, "Space: Investment Implications of the Final Frontier," 2017.

[12] T. Ryan-Mosley, E. Winick and K. Kakaes, "The number of satellites orbiting Earth could quintuple in the next decade," MIT Technology Review, June 2019. [Online]. Available: https://www.technologyreview.com/2019/06/26/755/satellite-constellations-orbiting-earth-quintuple/.

[13] M. Guerster, J. J. Garau-Luis, E. Crawley and B. Cameron, "Problem representation of dynamic resource allocation for flexible high throughput satellites," in *IEEE*, 2019.

[14] K. Krorsec, "Tesla plans to launch a robotaxi network in 2020," Techcrunch, 22 Apr 2019. [Online]. Available: https://techcrunch.com/2019/04/22/tesla-plans-to-launch-a-robotaxi-network-in-2020/.

[15] C. Henry, "EchoStar buys Jupiter-3 "ultra high density satellite" from SSL," Spacenews, August 2017. [Online]. Available: https://spacenews.com/echostar-buys-jupiter-3-ultra-high-density-satellite-from-ssl/.

[16] ViaSat, "Meet the World's Most Advanced Telecom Satellite," Aviation Week, April 2017. [Online]. Available: https://aviationweek.com/aerospace/connected-aerospace/meet-worlds-most-advanced-telecom-satellite.

[17] OneWeb, "OneWeb Techonology," OneWeb, July 2019. [Online]. Available: https://www.oneweb.world/technology.

[18] L. Grush, "SpaceX just launched two of its space internet satellites-the first of nearly 12,000," The Verge, Feb 2018. [Online]. Available: https://www.theverge.com/2018/2/15/17016208/spacex-falcon-9-launch-starlink-microsat-2a-2b-paz-watch-live.

[19] SES, "SES Selects SpaceX to Launch Groundbreaking O3b mPOWER MEO Communications System," SES, 9 Sep 2019. [Online]. Available: https://www.ses.com/press-release/ses-selects-spacex-launch-groundbreaking-o3b-mpower-meo-communications-system.

[20] L. Frenzel, "How Phased Array Antennas Work," *Nuts and Volts,* September 2018.

[21] J. J. Garau-Luis, M. Guerster, I. del Portillo, E. Crawley and B. Cameron, "Deep Reinforcement Learning Architecture for Continuous Power Allocation in High Throughput Satellites," *EESS,* 2019.

[22] J. G. Gooijer and R. Hyndman, "25 years of time series forecasting," *International Journal of Forecasting 22,* pp. 443-473, 2006.

[23] S. Makridakis, E. Spiliotis and V. Assimakopoulos, "Statsitcal and Machine Learning forecasting methods: Concerns and ways forward," *PLOS,* March 2018.

[24] O. de Weck, R. de Neufville and M. Chaize, "Staged Deployment of Communications Satellite Constellations in Low Earth Orbit," *Journal of Aerospace Computing, Information, and Communication,* 2004.

[25] Y. Qian, R. Hu, H. Abu-Amara and P. Maveddat, "Performance Evaluations on Bandwidth on Demand Algorithm for a High Capacity Multimedia Satellite Network," in *IEEE*, 2000.

[26] N. Laptev, J. Yosinski, L. E. Li and S. Smyl, "Time-series Extreme Event Forecasting with Neural Networks at Uber," in *ICML Time Series Workshop*, Sydney, 2017.

[27] B. Hobbs, U. Helman, S. Jitprapaikulsarn, S. Konda and D. Maratukulam, "Artificial neural networks for short-term energy forecasting: Accuracy and economic value," *Neurocomputing,* pp. 71-84, 1998.

[28] L. Ye and E. Keogh, Time Series Shapelets: A New Primitive for Data Mining, Riverside: UC Riverside, 2009.

[29] C. Ji, X. Zou, Y. Hu, S. Liu, L. Lyu and X. Zheng, "XG-SF: An XGBoost Classifier Based on Shapelet Features for Time Series Classification," *Procedia Computer Science,* pp. 24-28, 2019.

[30] B. Pavlyshenko, "Machine-Learning Models for Sales Time Series Forecasting," *Data,* 2019.

[31] S. Papadopoulos and I. Karakatsanis, "Short-term Electricity Load Forecasting using Time Series and Ensemble Learning Methods," in *IEEE*, 2015.

[32] S. Taieb and R. Hyndman, "A gradient boosting approach to the Kaggle load forecasting competition," *International Journal of Forecasting 30,* pp. 382-394, 2014.

[33] F. Karim, S. Majumdar, H. Darabi and S. Chen, "LSTM Fully Convolutional Networks for Time Series Classification," *IEEE,* 2018.

[34] Y. Qin, D. Song, H. Chen, W. Cheng, G. Jiang and G. Cottrell, "A Dual-Stage Attention-Based Recurrent Neural Network for Time Series Prediction," in *International Joint Conference on Artificial Intelligence*, 2017.

[35] Z. Che, S. Purushotham, K. Cho, D. Sontag and Y. Liu, "Recurrent Neural Networks for Multivariate Time Series with Missing Values," *Scientific Reports,* 2018.

[36] J. Bergstra, R. Bardenet, Y. Bengio and B. Kegl, "Algorithms for Hyper-Parameter Optimization," in *Advances in Neural Information Processing Systems*, 2011.

[37] J. Bergstra and Y. Bengio, "Random Search for Hyper-Parameter Optimization," *Journal of Machine Learning Research,* pp. 281-305, 2012.

[38] G. Lujan-Moreno, P. Howard, O. Rojas and D. Montgomery, Design of experiments and response surface methodology to tune machine learning hyperparameters, with a random forest case-study, Elsevier Ltd., 2018.

[39] W. Sukthomya and J. Tannock, "The optimisation of neural network parameters using Taguchi's design of experiments approach: an application in manufacturing process modelling," *Springer-Verlag London,* 2005.

[40] C. Kourogiorgas, A. Papafragkakis, A. Panagopoulos and S. Ventouras, "Long-Term and Short-Term Atmospheric Impairments Forecasting for High Throughput Satellite Communication System," in *IEEE*, London, 2018.

[41] G. Box, G. Jenkins and G. Reinsel, Time Series Analysis: Forecasting and Control, John Wiley & Sons Inc, 2008.

[42] Y. Freund and R. Schapire, "Experiments with a New Boosting Algorithm," *AT&T Research,* 1996.

[43] L. Breiman, "Arcing the Edge," University of California, Berkeley, 1997.

[44] J. Friedman, "Greedy Function Approximation: A Gradient Boosting Machine," in *IMS Reitz Lecture*, 1999.

[45] D. Rumelhart, G. Hinton and R. Williams, "Learning representations by back-propagating errors," *Nature,* 1986.

[46] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation,* pp. 1735-1780, 1997.

[47] A. Graves, Supervised Sequence Labelling with Recurrent Neural Networks, University of Toronto, 2012.

[48] D. C. Montgomery, Design and Analysis of Experiments, New Jersey: John Wiley & Sons Inc, 2006.

[49] M. Bashiri and A. Geranmayeh, "Tuning the parameters of an artificial neural network using central composite design and genetic algorithm," *Scientia Iranica,* pp. 1600-1608, 2011.

[50] D. Frey and N. Sudarsanam, An Adaptive One-Factor-at-a-Time Method for Robust Parameter Design: Comparison with Crossed Arrays via Case Studies, ASME, 2008.

[51] D. Frey, F. Engelhardt and E. Greitzer, "A role for "one-factor-at-a-time" experimentation in parameter design," *Research in Engineering Design,* 2003.

[52] M. Hamada and C. Wu, "Analysis of Designed Experiments with Complex Aliasing," *IIQP Research Report,* 1991.

[53] M. Neely, E. Modiano and C. Rohrs, "Power Allocation and Routing in Multibeam Satellites with Time-Varying Channels," in *IEEE*, 2003.

[54] D. Woods and S. Lewis, "Design of Experiments for Screening," *University of Southampton,* 2015.

[55] H. Chipman, M. Hamada and C. Wu, "A bayesian Variable-Selection Approach for Analyzing Designed Experiments with Complex Aliasing," *Technometrics,* pp. 372-381, 1997.

[56] D. Draguljinm, D. Woods, A. Dean, S. Lewis and J. Vine, "Screening Strategies in the Presence of Interactions," *Technometrics,* 2014.

[57] Stanley, Morgan, "Space: Investing in the Final Frontier," Morgan Stanley, 2 Jul 2019. [Online]. Available: https://www.morganstanley.com/ideas/investing-in-space.