LIGHTNING PREDICTION USING RECURRENT NEURAL NETWORKS

THESIS

Dominick V. Speranza, 2nd Lt, USAF

AFIT-ENS-MS-19-M-150

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

The views expressed in this document are those of the author and do not reflect the official policy or position of the United States Air Force, the United States Department of Defense or the United States Government. This material is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

AFIT-ENS-MS-19-M-150

LIGHTNING PREDICTION USING RECURRENT NEURAL NETWORKS

THESIS

Presented to the Faculty

Department of Operational Sciences

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

in Partial Fulfillment of the Requirements for the

Degree of Master of Science in Operations Research

Dominick V. Speranza,

2nd Lt, USAF

March 21, 2019

AFIT-ENS-MS-19-M-150

LIGHTNING PREDICTION USING RECURRENT NEURAL NETWORKS

THESIS

Dominick V. Speranza,
2nd Lt, USAF

Committee Membership:

Lt Col A. Geyer, Ph.D.
Advisor

R. Hill, Ph.D.
Reader

AFIT-ENS-MS-19-M-150

Abstract

Cape Canaveral Air Force Station (CCAFS), Kennedy Space Center (KSC), and Patrick Air Force Base (PAFB) all reside in the thunderstorm capital of the United States. According to the Florida Climate Center, these installations experience more thunderstorms per year than any other place in the United States. It is the mission of the 45th Weather Squadron to provide timely and accurate warnings of weather conditions such as lightning that pose a risk to assets and personnel CCAFS, KSC and PAFB.

To aid 45th Weather Squadron forecasters, a network of 30 Electric Field Mills (EFM) was installed in the area in and around CCAFS, KSC, and PAFB. EFMs record the electrification of the local atmosphere. Several efforts have been made over the years to find an optimal way to utilize the EFM network data to improve lightning prediction. These efforts approached the problem using atmospheric science as well as traditional statistical regression techniques with mixed results.

In this paper, hourly statistics were generated from the raw EFMs data set used in Hill [1]. Input variables were generated from surface observations from every station within 50 miles of CCAFS and then combined with the EFM statistics for the same time periods. This combined data set was used to create Long Short-term Memory (LSTM) Neural Networks designed to capture trends within the data for each observation. A variety of different LSTM model structures were created and trained to see which model structure performed best when predicting lightning around CCAFS, KSC, and PAFB. By utilizing design of experiments techniques, optimal parameters for the LSTM model structures are narrowed down providing a solid baseline for future endeavors in predicting lightning.

Acknowledgements

I would like to express my deep appreciation for the guidance and mentorship provided by my advisor Lt. Col. Andrew Geyer. Without his help, this project would not have been possible. I would also like to thank the 45th Weather Squadron for the continued support and funding throughout the project and follow on research. Without the hardware purchased for the experiment, this project would have never been completed. Additionally, I would like to thank Dr. Raymond Hill for his guidance and support as a mentor and reader of this thesis.

Finally, I would like to thank my parents, for the undying love and support throughout my life, and my siblings for setting high expectations which I strive to live up to!

Dominick V. Speranza

List of Tables

List of Figures

Figure Page

List of Abbreviations

Abbreviation	Page
Cape Canaveral Air Force Station (CCAFS)	1
Patrick Air Force Base (PAFB)	1
Kennedy Space Center (KSC)	2
Artificial Neural Networks (ANNs)	3
ground flash density (GFD)	5
Electric Field Mills (EFM)	6
Meteorological Aerodrome Reports (METARs)	7
Recurrent Neural Networks (RNN)	8
Predictive Mean Matching (PMM)	9
Long Short-term Memory Neural Network (LSTM)	12
Design of Experiments (DOE)	13
Visual Basic for Applications (VBA)	17
lightning detection and range (LDAR)	18

Table of Contents

LIGHTNING PREDICTION USING RECURRENT NEURAL NETWORKS

I. Introduction

On March 26, 1987, an Atlas/Centaur rocket was struck by lightning around 38 seconds into its flight. The lightning strike itself did not cause the spacecraft to explode. However, the electrical surge caused a failure in stability systems which caused an excessive angle of attack and destroyed the rocket [6]. This resulted in an investigation that showed the importance of lightning prediction when it comes to space shuttle and rocket launches.

Cape Canaveral Air Force Station (CCAFS) along with Patrick Air Force Base (PAFB) use their launch facilities for both publicly and privately funded space missions. The US Air Force, NASA, and privately owned businesses such as SpaceX regularly launch payloads into orbit. Large rockets are used to propel the cargo from CCAFS. The preparation and resources used in just planning for a launch are enormous. Postponing a launch can cost around $300,000 and can lead to other space launches also being delayed [7].

This chapter first provides a brief introduction into previous efforts at accurate lightning prediction around CCAFS. Next, the problem this research addresses is formally stated. There is a then discussion of the research questions developed to address the problem statement. The chapter concludes with an overview of the rest of the document.

1.1 Background

Many previous efforts to improve lightning prediction capabilities at the Kennedy Space Center (KSC) at CCAFS utilized regression on predictor variables. One such predictor introduced was the integrated precipitable water vapor data gathered from the GPS around KSC. In one study, four regressors were identified that better predicted the lightning for the 1999 lightning season. The addition of these new variables showed a 26.2% decrease in false alarms for a non-independent period of time and a 13.2% decrease for an independent time. The only issue with this model was the potentially long 12 hour window for making the prediction [8].

Other parameters have been used to create models for lighting prediction. These parameters focused heavily on ground-to-cloud strikes and the parameters that might be related to them based on prior research into the physics of lightning. The three parameters were lightning peak current, ground flash density, and keraunic level. The lightning peak current is the location where the lightning is observed. The ground flash density is the number of lightning strikes. The keraunic level is lightning observations based on hearing the thunder after the lightning bolt is observed. These data were gathered in Brazil, Malaysia, and Colombia. All of the sites are tropical locations, which tend to have larger amounts of extreme lightning storms. The goal of the study was to make a comparison between tropical regions and temperate regions. It found that tropical regions tend to have larger ground flash density than temperate regions [9].

Another difficulty in trying to predict lightning is that the methods and techniques developed to predict the localized weather in one region tend to not work for other regions [10]. Every area of study is only able to generalize for future weather in the specific area. Very rarely are the results able to be generalized to other regions [11]. This leads to the problem investigated in this research by taking a look at the lightning prediction around CCAFS.

1.2 Problem Statement

The 45th Weather Squadron seeks to better predict lightning around CCAFS. This is necessary to both avoid lightning striking the rockets as well as reducing costly false alarms which cause launch delays. As several different clients use the CCAFS launch pad to launch payloads into orbit, this problem has an impact on many entities.

1.3 Research Question

To address this problem, two research questions are addressed:

1. Which variables can be used as regressors to better predict lightning strikes around CCAFS? This is addressed with time-series data gathered around CCAFS.
2. Can an improved model be formed to better predict lightning at CCAFS/KSC? Artificial Neural Networks (ANNs) are developed with the R programming language to manage, build, and tests the models.

The primary motivation for this thesis is to build a model to better predict lightning for the Cape Canaveral area using specific regressor variables. This can potentially save the companies using the launch sites thousands of dollars, keeping the launches on schedule as best as possible.

1.4 Organization of the Thesis

The next chapter features a review of current literature regarding use of ANNs for lightning prediction. This review builds a lexicon for the later discussion of methodology and results. Next, the methodology is discussed to include: description of the raw data, pre-processing of the raw data, description of the model structures, determining optimal model design given a time constraint, and implementation of the experiment. Following the methodology, the key results and findings are shown to illustrate the usefulness of experimental models. Finally, key findings are presented along with proposals for potential future work.

II. Literature Review

2.1 Introduction

This chapter provides a brief review of literature related to weather prediction and artificial neural networks. First, there is a discussion of various multivariate techniques that are currently being used for weather prediction. Next, a description of what CCAFS currently does for lightning prediction is provided. Lastly, the basics of ANNs and the packages used for this research are discussed.

2.2 Multivariate Techniques for Lightning Prediction

Lightning storms in dry climates can often result in wildfires. Storms in populated areas can result in damage to power and telecommunications, human injuries/fatalities, and airport disruptions. An initial look into various multivariate techniques applied to lightning prediction has aided in improved warnings in areas of Australia [12]. Of the multivariate techniques used, logistic regression performed the best for lightning prediction accuracy. The other techniques used in Bates *et al.* [12] were discriminant analysis, principal component analysis, classification and regression trees, and random forests. Note that for Bates *et al.* [12], artificial neural networks were not used. These methods were still a large improvement from the methods using only climatological values compared to in their study. [12].

Recent studies conducted in Colombia show how lightning warning systems can be used to better manage the risk involved with being in a high ground flash density (GFD) area. The Colombia study portrays the GFD in high valued areas along with providing a risk measurement [2]. These results are in Table 1.

Case	***GFD (Flashes/ km^2 year)***	***People at risk***	***Exposure time(h)***	***R1***	***R2***
Oil Facility	8	80	2000	9.9E-03	986
Stadium	26	40000	832	1.1E-03	108
Mine	33	200	8760	4.0E-02	4032
Airport	16	100	8760	1.1E-03	112
Military Base	6	300	8760	1.4E-03	140

Table 1. Output for Risk Areas in Colombia. [2]

The stadium, the mine, and the airport listed in Table 1 are in the three largest GFD areas. This insight can be used to best decide where to build important high population density areas to minimize the risk of having potential damage due to lightning strikes. Additionally, Tovar *et al.* [2] shows how a thunderstorm warning system could help reduce this risk to human life significantly in lesser developed areas in Colombia.

One way to construct a lightning warning system such as the one used in Colombia is to use Electric Field Mills (EFM). These mills gather data of electrostatic potential in thunderstorms. These mills have shown their ability to detect lightning starting to form in the clouds. They are commonly used in research to predict adverse weather conditions in other areas [13]. In fact, Hill [1] used EFM data from CCAFS/KSC in a previous effort to improve lightning prediction there.

2.3 Current Model/Data Used

The 45th Weather Squadron is in charge of issuing lightning warnings for the CCAFS and Patrick AFB. Currently, they use lightning circles that are roughly 5 nautical miles in diameter. There are a total of 10 circles throughout the area, several of which overlap. Lightning circles are circles drawn around a center point where if lightning is detected anywhere within the circle, the whole area goes on a lightning warning. If lightning is sighted or predicted in these areas, the base alarms will go off and flights and space launches are delayed. Efforts have been made to improve the warning system so that there is less overlap while still maintaining the same level of safety. The improvements reduced the total number of lightning circles which reduced the number of overlaps. Additionally, a streamlined lightning warning process made issuing warnings much easier. This allowed for more focus and effort to be given to the lightning prediction rather than the lightning warnings themselves [14].

Meteorological Aerodrome Reports (METARs) are frequently used in weather prediction by the 45th Weather Squadron, as well. METARs are surface weather observations that capture a wide range of variables from wind direction and speed to dew point temperature. There are 10 weather stations located within a 50nm radius from KSC that the 45th Weather Squadron uses when making predictions. Data points are not captured continuously but rather at the top of every hour or when significant changes to the weather occur. Some of the weather stations do not run 24 hours a day which can make using the data problematic when using time-series data analytic techniques [14].

2.4 Neural Networks

ANNs are "black-box" methods that are meant to simulate connections made in an animal's brain so that the algorithm is able to learn when more information is presented [15]. Starting with the raw data, the usable components (dependent variables) are broken out and regressed onto the hidden nodes which process the signals prior to reaching the output node. The algorithm finds weights for each of these nodes to best align the input to the output. There can be multiple hidden nodes in each layer and multiple layers with in a single ANN (Figure 1).

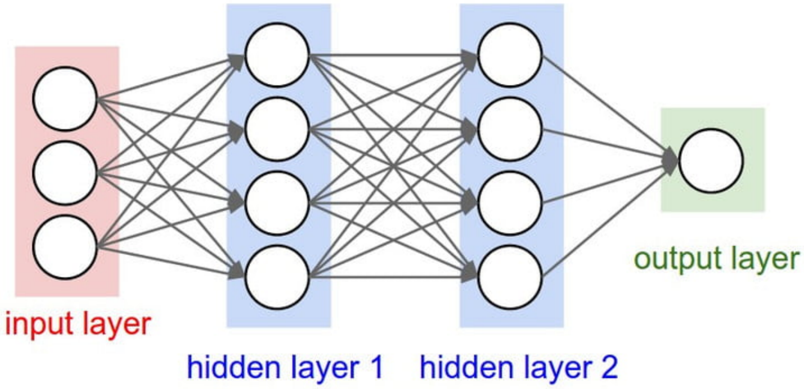


Figure 1. Basics of an Artificial Neural Network [3]

A basic ANN only allows for information to travel one way (towards the output). Recurrent Neural Networks (RNN) allow for information to travel in both directions using loops. This allows for modeling exceedingly more complex problem which is necessary for modeling lightning weather prediction [15].

Not all data is necessarily good for RNNs. Time-series data tends to work better than fixed time data. The data made available by the 45th Weather Squadron is time-series data with several different variables. The time between the data collection differs for each variable. Since all data must be on the same time scale, data manipulation was necessary for the R programming. If data that is on different time scales is fed into an RNN, the algorithms will not function properly and the RNN will not train properly.

Using R Programming

The programming language used for the analysis in this research is R. R is a free, open-source programming language mainly meant for statistical computations and graphical representations for model building [16]. Given enough computing power, R is able to handle large datasets relatively easily. Lantz [15] discusses the R packages involved with the machine learning techniques in the package called “keras”. The package primarily deals with deep learning neural networks along with visual representations for the models developed using neural networks [3].

Several packages are available in R that are related to data that has time components [17]. Using the “tidyverse” and “lubridate” packages, time-series data is able to be manipulated to fractions of a second which is helpful with getting all time series data on the same time steps [18, 19]. The data can be manipulated to get all the data onto the same time scale. This is done by either averaging data between increments of time-series data or using regression techniques to add data into the time-series data. This puts the data in the proper format to use in RNNs [17].

For time-series data and recurrent neural networks, there must be a continuous string of data at equal time steps. This means that there cannot be any holes in the data. The package “MICE” uses other observations and variables around the missing data points in order to impute the missing data point. This is done through a process of Predictive Mean Matching (PMM) [20]. PMM uses surrounding data to fill in the missing data. The imputed value is randomly selected from among the observed surrounding values. This ensures that the imputed values are plausible, which makes the data more appropriate than using regression methods to smooth the surrounding values to estimate the missing value [20]. Imputation can potentially biases the analysis. However, this work did not involve a significant amount of missing data. Thus, the remainder of the analysis uses the imputed data values to provide a complete dataset.

Weather Prediction

Trying to predict weather using multivariate techniques is not a new field of study. A study in 1998 used time-series data to produce an ANN that outperformed normal linear regression when predicting precipitation over a 6 hour period [4]. The model did particularly well in predicting the amount of precipitation as the amount of precipitation increased. Figure 2 is an illustration of the ANN model results outperforming other weather predicting models used in forecasting [4].

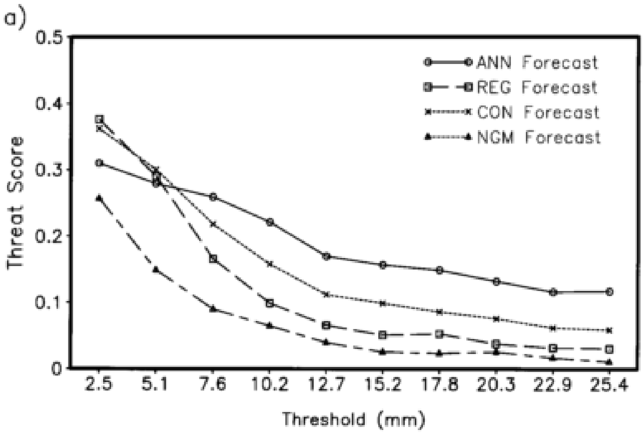
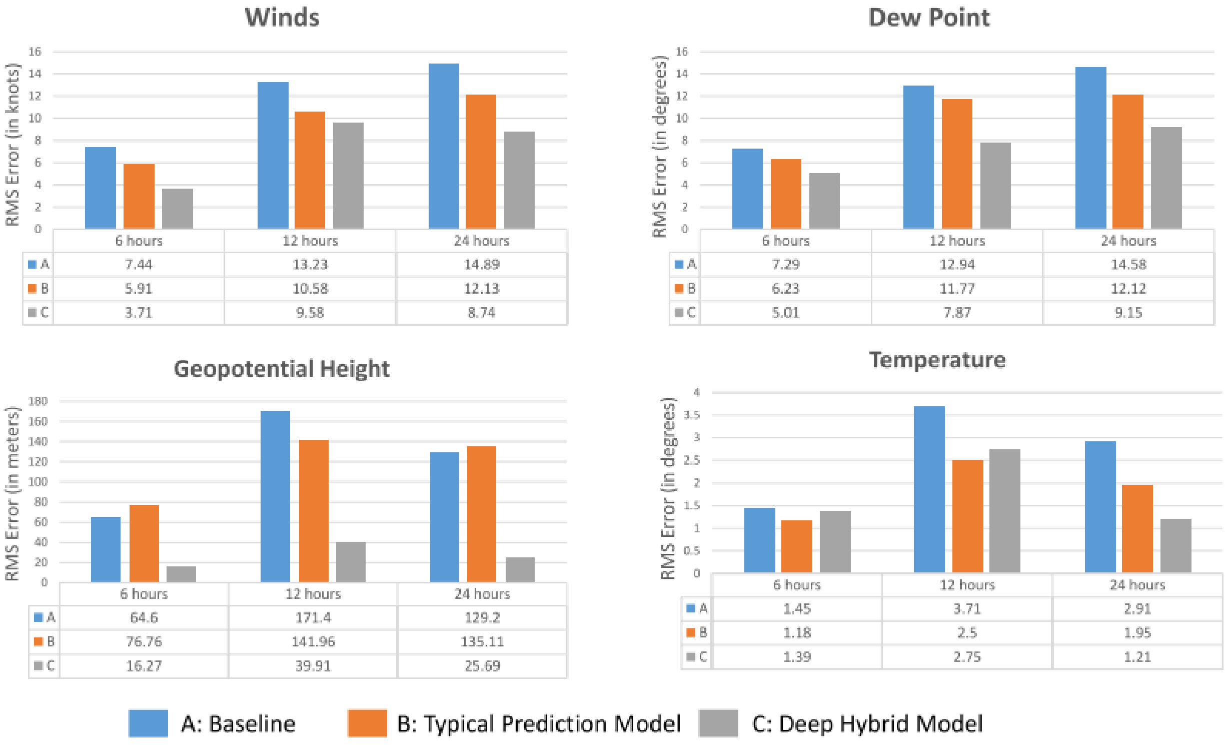


Figure 2. Output from the ANN and comparisons [4]

Other, more recent, efforts have also tried to use deep neural networks to better predict the weather. This is done by using a newer, data intensive method and combining spatial and time-series data. To see how well the model performs, a baseline model (basic weather predicting techniques), a static kernel method (commonly used), and the deep learning neural network were compared. In virtually all areas, the neural network hybrid model outperformed the baseline and the kernel common method (Figure 3).



Winds

	6 hours	12 hours	24 hours
A	7.44	13.23	14.89
B	5.91	10.58	12.13
C	3.71	9.58	8.74

Dew Point

	6 hours	12 hours	24 hours
A	7.29	12.94	14.58
B	6.23	11.77	12.12
C	5.01	7.87	9.15

Geopotential Height

	6 hours	12 hours	24 hours
A	64.6	171.4	129.2
B	76.76	141.96	135.11
C	16.27	39.91	25.69

Temperature

	6 hours	12 hours	24 hours
A	1.45	3.71	2.91
B	1.18	2.5	1.95
C	1.39	2.75	1.21

Figure 3. Comparing the Neural Network models with Other Models [5]

The hybrid model had lower error rates than the other two methods. This supports the potential in weather predition using ANNs. Everywhere except for short term temperature, the hybrid model had better errors than the other two methods [5]. This also shows the large potential in weather prediction using these ANNs.

Another study looked at trying several different types of neural networks in an attempt to predict temperature, wind speed, and humidity for all seasons of the year with data collected in Saskatchewan, Canada [21]. All the models developed made predictions for a 24 hour ahead forecast. Out of all the models, the artificial neural network models performed better in learning the data along with generalizing the data to make more accurate predictions [21]. This provides further motivation into using ANNs in weather patterns and predictions.

Lightning Prediction

Rather than look at weather in general, some articles and prior theses have examined using ANNs specifically related to lightning predictions. Hill [1] focused on the same research question as this work. However, that effort built a time-series dataset with fewer parameters as only EFM data from around CCAFS was available at the time. While the Hill [1] model has a lower probability of false detection, there is the potential for improvement by including other parameters when building the neural network.

Hill [1] also focused on lightning detection using a short time step. This is good for attempting to predict lightning for a specific prediction window in the near future. The drawback to this approach is that the amount of data being run through the neural networks in this model using such small time windows meant that training went very slow and the neural network structures needed to be much less complicated in order to compensate for the longer run times. As a result, while the data captured was sufficient, the neural networks themselves may not have been complex enough to capture the potentially complex trends in the data.

Long Short-term Memory Neural Networks

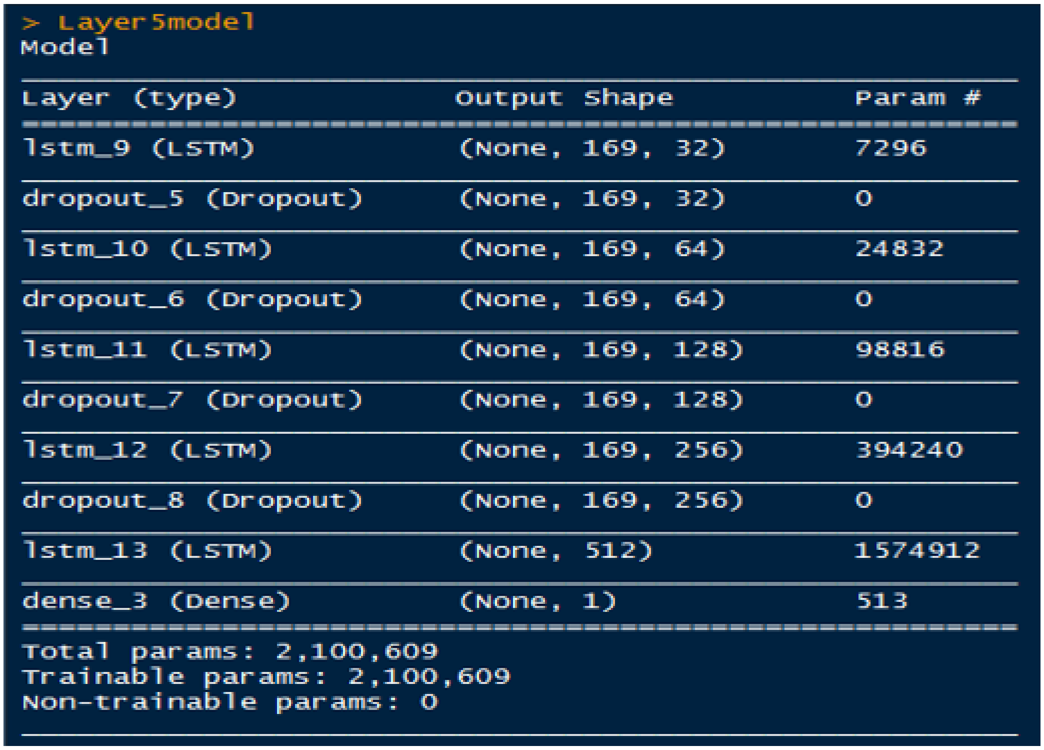
A specific type of neural network that is good at dealing with time-series data related to weather is a Long Short-term Memory Neural Network (LSTM). When LSTMs are used, a generator function is developed to parse through and extract the data needed based on steps (number of data points per hour), lookback (number of hours for the LSTM), and delay (number of hours to predict in the future). For example, with a steps value of 1, a lookback value of 4, and a delay of 2, the resulting dataset looked trends in the data in 4 hour increments in order to predict the next hour's target variable (in this case the target is a lightning occurrence). This allows for the data to be pushed through the network and capture trends and dependencies within the data over given hour periods.

There is no “rule of thumb” when it comes to creating neural networks [22]. Most results found from other papers are a result of creating robust models and guessing and checking different values for parameters to see which ones improved the accuracy. In order to find the best parameters, Bashiri [22] used Design of Experiments (DOE) to identify the optimum parameters. Other methods include the Taguchi method which tends to have the problem of having a discrete solution space and excludes any interactions amount parameters [22]. For this research, a DOE method is proposed and tested on parameters for a LSTM neural network. This is to find the best model with limited time to run the models.

Examining LSTM structures start with creating a basic single layer LSTM. From there, additional layers are added until the training data is performing at an acceptable level. This usually results in over-fitting, meaning the LSTM did not do a very good job at predicting on the validation data. Various techniques are available to reduce the tendency to over-fit the model which include increasing the amount of data, introducing dropout layers, and reducing complexity/ parameters with which the model is training. Since additional data was unobtainable, introducing dropout layers and changing the complexity of the model were the main source of fixing the over-fitting problem. A list of all the model structures trained is found in Appendix A. The best performing models are the topic of discussion and comparison in the remainder of the research.

Below are examples of three of the model structures developed and the rational behind why they were selected as candidates for CCAFS lightning prediction.

5 Layer LSTM - A dense complex LSTM can to capture complex trends but may over-fit the data. To combat over-fitting of the data, a 25% dropout is used after each layer. The overall model structure is shown in Figure 4.



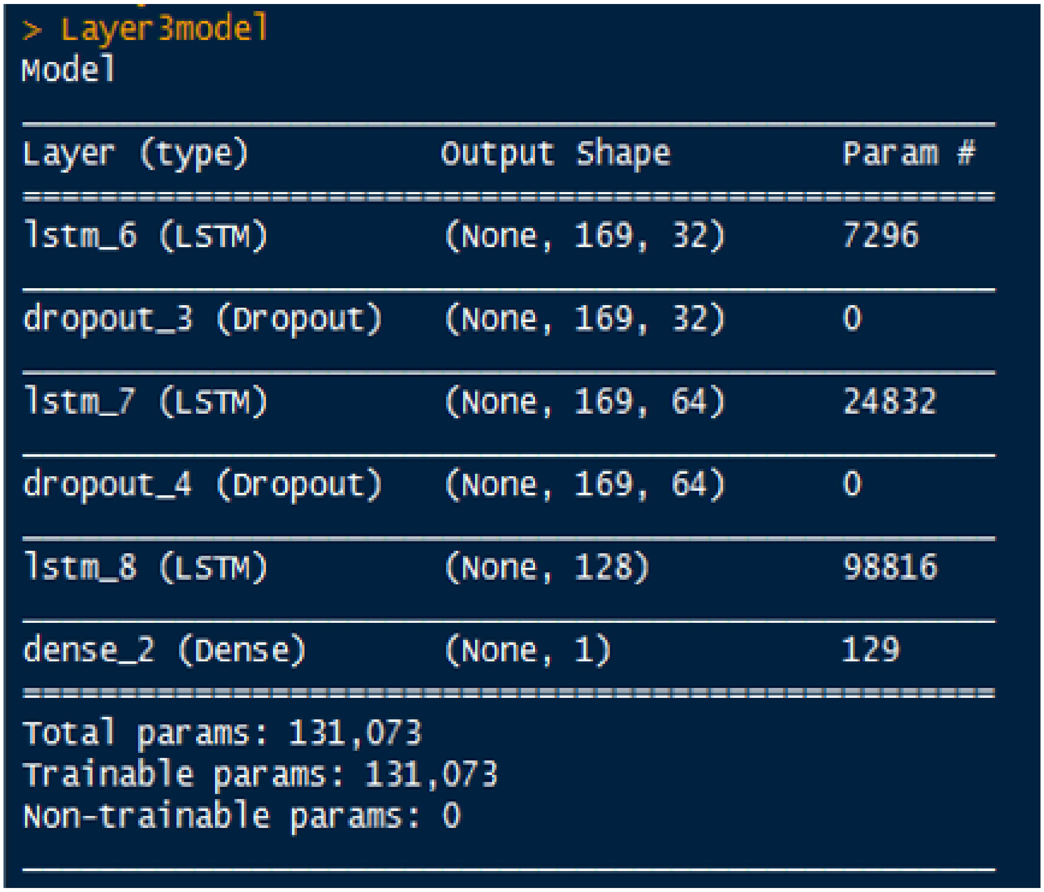
```
> Layer5model
Model
________________________________________________________________
Layer (type)                  Output Shape              Param #
================================================================
lstm_9 (LSTM)                 (None, 169, 32)           7296
________________________________________________________________
dropout_5 (Dropout)           (None, 169, 32)           0
________________________________________________________________
lstm_10 (LSTM)                (None, 169, 64)           24832
________________________________________________________________
dropout_6 (Dropout)           (None, 169, 64)           0
________________________________________________________________
lstm_11 (LSTM)                (None, 169, 128)          98816
________________________________________________________________
dropout_7 (Dropout)           (None, 169, 128)          0
________________________________________________________________
lstm_12 (LSTM)                (None, 169, 256)          394240
________________________________________________________________
dropout_8 (Dropout)           (None, 169, 256)          0
________________________________________________________________
lstm_13 (LSTM)                (None, 512)               1574912
________________________________________________________________
dense_3 (Dense)               (None, 1)                 513
================================================================
Total params: 2,100,609
Trainable params: 2,100,609
Non-trainable params: 0
________________________________________________________________
```

Figure 4. 5 Layer LSTM Model Structure

Rare Event LSTM - This has the same model structure as the 5 layer LSTM (figure 4) with slightly less dropout (to allow the model to learn better). The main difference in this model is that the model attempts to capture the rarity of lightning occurring. In the data, lightning does not occur 50% of the time. Rather, lightning occurs roughly 32% of the time. By changing the classification weights for lightning occurrence, the model can capture lightning as a rare event.

3 Layer LSTM - Similar to the 5 layer LSTM except without the last two LSTM layers. This model should be able to capture complex trends but perhaps not quite as complex as the 5 layer LSTM. The reason this model is added is due to having far fewer parameters than the 5 layer LSTM. This means that the model trains significantly fast than the 5 layer LSTM. Note that the first 3 layers of the 5 Layer LSTM

are the same as the first (and only) 3 layers in the 3 Layer LSTM model. This makes comparing the results between the two models much easier (Figure 5).



```
> Layer3model
Model
________________________________________________________________
Layer (type)               Output Shape              Param #
================================================================
lstm_6 (LSTM)              (None, 169, 32)           7296
________________________________________________________________
dropout_3 (Dropout)        (None, 169, 32)           0
________________________________________________________________
lstm_7 (LSTM)              (None, 169, 64)           24832
________________________________________________________________
dropout_4 (Dropout)        (None, 169, 64)           0
________________________________________________________________
lstm_8 (LSTM)              (None, 128)               98816
________________________________________________________________
dense_2 (Dense)            (None, 1)                 129
================================================================
Total params: 131,073
Trainable params: 131,073
Non-trainable params: 0
________________________________________________________________
```

Figure 5. 3 Layer LSTM Model Structure

Due to the nature of LSTMs and the amount of data pushing through the models, a considerable amount of time is needed in order to run each model. With limited time, optimal settings are needed to obtain the best results. DOE is used to help determine which models to run in order to make conclusions about the accuracy results [22].

2.5 Conclusion

Much research is currently investigating ways to better predict weather phenomenon. In areas like CCAFS, lightning specifically is an expensive and potentially disastrous nuisance. Prior research done by Hill [1] and others have shown some success in predicting lightning each with their own drawbacks and problems. Taking these into consideration, improvements are made to further address the research questions and predict lightning around CCAFS so that launches can be planned and run more smoothly.

III. Methodology

3.1 Introduction

This chapter discusses the several parts used to set up the analysis. First, a look at the software used in data cleaning and analysis will be explained in order to allow future research to be duplicated. Next, a description of the datasets will explain where the inputs came from for the neural network. Finally, detailed explanations of the neural networks themselves will show how the results were obtained.

3.2 Software and Data Pre-processing

Data pre-processing used R, Visual Basic for Applications (VBA) /Excel, and Matlab (Appendix D). METAR data obtained from the 14th Weather Squadron (AAC) at Asheville, NC was in a CSV file written in METAR code from ten different weather stations located around KSC. This encompasses all weather stations within a 50nm radius from KSC. Matlab code generated the following variables from each of the weather stations: wind direction, wind speed, visibility, fog (binary), rain (binary), rain rate (rain intensity), cloud height, cloud cover, altimeter, sea level pressure, temperature, and dew-point temperature. The data were collected once per hour or if any significant event that happened at a specific location. If a significant event happened, not all locations would take new readings. To get all of the data on the same time scale, the dataset was reduced using VBA/Excel so that each hour had only a single data point. Each variable with multiple values per hour were compressed using the average (for wind direction), the max value (for wind speed, wind gust, fog, rain, thunderstorm, rain rate, cloud cover, and dew point), or the min (for visibility, cloud height, altimeter, sea level pressure, and temperature). Even with the data compressed, all of the stations had large amounts of missing values. To start, some of the stations were missing nearly half of their data because the station was only active during specific hours. As there is no way to obtain this data, these variables were removed from the dataset. All variables missing more than 1% of the data were removed. This left 4 locations with a total of 48 variables for analysis. VBA/Excel filled in the remaining 1% of missing data using linear regression.

The other chunk of data used for the analysis came from the EFMs. This is the same dataset used in Hill [1]. The data came from 30 different locations and recorded every two hundredths of a second for the months May-September during the years 2013-2016. To extract the data, R code was written to extract the data into a format that is more easily processed. The executable used to decompress the .dat files for R processing came from the NASA website [23]. See Appendix D for the code used to extract the compressed .dat files.

Because the analysis and the neural network are predicted in hourly increments, the field mill data was compressed from every two hundredths of a second to the minimum, maximum, average, and standard deviation for each hour. This allows the model to capture any abnormalities within each hour in order to predict if there will be a lightning strike in the following hour. This would provide PAFB enough time to make any necessary precautions. The total number of additional variables this added to the model was $30 \cdot 4 = 120$ (for 30 locations and 4 variables per location.

The last bit of data used in the model was the lightning detection and range (LDAR) dataset. This data was easily read into R. The data presented gave the exact time that the lightning was detected and the number of meters from the center of the KSC given in X (East/West), Y (North/South), and Z (altitude). The range in distances around KSC spanned for hundreds of kilometers in all directions. Obviously, KSC will not shut down if a lightning strike happened hundreds of kilometers away. So when determining if a lightning strike happened in a given hour of time, an imaginary box was created around the epicenter of KSC. The box was 41.7 kilometers wide (east/west) and 87 kilometers long (north/south). This encompasses the entirety of CCAFS along with having a 5 mile buffer around the entire base. KSC on CCAFS launches craft through the atmosphere. Therefore, the lightning detected at any elevation was included in the dataset. The data extracted was the date (year, month, day, hour) and a binary variable indicating a lightning occurred (1,0). In total, there were 4894 hours with lightning strikes out of the 14880 total hours in the dataset giving a lightning occurrence rate of 32.89%.

Of these 14880 hours, there were several missing data points with no reasonable way to get access to the missing data. The package "MICE" in R imputed data points for missing gaps (See Chapter II) [20]. MICE filled in the missing data points so that the dataset is complete for the neural networks. MICE imputed the data points using PMM as discussed in Chapter II.

For the analysis, training and validation split the data 80/20 to ensure more generalized results for the model. The model trained on 11824 observations and was validated using 2957 observations. These values slightly differ based on the parameters chosen for the LSTM.

The final complete dataset has 14880 observations with 168 variables per observation. This excludes lightning occurrence which is retained as the output variable.

3.3 Building the Neural Networks

With the time-series data complete, neural networks were made to provide an optimal structure for learning. In order to produce the most robust results possible for the analysis, looking at different parameters is important for each of the models built. The analysis looked at a variety of different lookback values ranging from 12-48 hours in order to capture the trends in the data for each time-step. After basic testing of the model structures referenced at the end of section 2.3, three model structure are used in the comparison: 5 Layer LSTM, RareEvent LSTM, and 3 Layer LSTM. The other models found in Appendix A were dismissed due to underwhelming performance when compared to the three models chosen.

3.4 Which Parameters to Run?

Table 2 lists the possible values for lookback and delay examined in testing. [24].

Possible Parameter Values	
Lookback	12,18,24,30,36,42,48
Delay	3,6,9,12,15,18,21,24

Table 2. Possible Parameter Values

This leads to a total of 56 combinations of possible runs for a full factorial design. Due to the time constraint of the project, 26 runs were chosen in a 1/2 fractional factorial design with D-optimality. The optimality criteria for D-optimality is one that maximizes the determinant of $(X' \cdot X)$. The result minimizes the generalized variance of the parameter estimates for the experiment. The output the JMP produced for the DOE runs are located in Table 3.

Combinations		**Combinations**		**Combinations**		**Combinations**	
Delay	Lookback	Delay	Lookback	Delay	Lookback	Delay	Lookback
6	12	3	12	12	18	6	48
9	18	3	30	12	24	21	48
18	24	21	42	6	36	6	24
12	30	18	30	9	48	3	24
21	24	12	36	18	36	12	18
24	18	24	48	18	48	6	18
						21	12
						24	36

Table 3. Fractional Factorial Parameter Test

3.5 Determining the Level of Success

Establishing a baseline allows for a more accurate comparison. When dealing with a binary output, a bad prediction is 50%. That is if each outcome is equally likely similar to a coin flip. If a dataset is unbalanced in anyway (which can be seen by just looking at the list of binary outputs), predicting better than 50% is very simple. Just always predict the most frequent outcome. For this data, there are lightning strikes in 35% of the hours that data was collected. Guessing there will not be a lightning strike for every hour yields a predicting accuracy of 65%. This is far better than 50% but it is in no way informative. This can be taken a step further with the introduction of the time-series. For example, to develop a baseline for temperature prediction in an area, it is generally accepted that the temperature 24 hours before the present time will roughly be the temperature at the present time. This is known as persistence. This adds no real information aside from the time-series nature of the data. By doing this with the lightning data, the new baseline model rises up to 70%. This day-before baseline is used to assess the model's utility as a lightning predictor for CCAFS. Neural Networks that produce a result better than the 70% baseline are beneficial in predicting lightning strikes around CCAFS.

3.6 Conclusion

Although the construction of the dataset used for the neural networks was complex, the result was a dataset that is suitable for training LSTMs. By creating robust models and changing the parameters to allow for the best fit, a neural network was created that can better predict lightning strike around CCAFS. The baseline day-before metric will be used in determining the actual utility of the experiment using LSTMs to predict lightning. Chapter IV will delve deeper into the development of the best performing model, analysis of models performance, and discussion as to how well they address the problem statements.

IV. Discussion

4.1 Introduction

This chapter presents the results and analysis for the neural network models developed. First, model performance is examined. Next, comparisons are made to the persistence baseline along with other studies covered in previous chapters. Following that, an analysis of the results are addressed. This chapter concludes by addressing how this work addresses the initial problem statement and research questions.

4.2 Results

Differences in the Model Structures

Each of the LSTM models ran for 150 epochs to allow sufficient time to train the variables and achieve a high degree of accuracy. A single epoch is a single run through of all the data through the model. Typically, the loss and accuracy of the validation and training sets started to diverge roughly between 80-120 epochs with some models diverging sooner and some diverging later. Given enough time and epochs, the training data would eventually approach 100% accuracy due to how complex the models were. The accuracy for the validation set, however, would not continue to improve indefinitely and began to level off once the binary cross-entropy loss began to level off.

Figures 6-8 are plotted examples of each of the three model structures. Most of the models examined followed the same general trend for each of the different structures. The plotted examples below have the parameters of a 18 hour delay and a 36 hour lookback (Figure 6),(7),(8). Different amounts of dropouts for each model were used to try and minimize over-fitting the data. The 5-Layer model had more dropout than the 3-Layer model due to it's significantly more complex design. The 5-Layer model had more dropout to try and prevent the validation and training split in the binary cross-entropy loss.

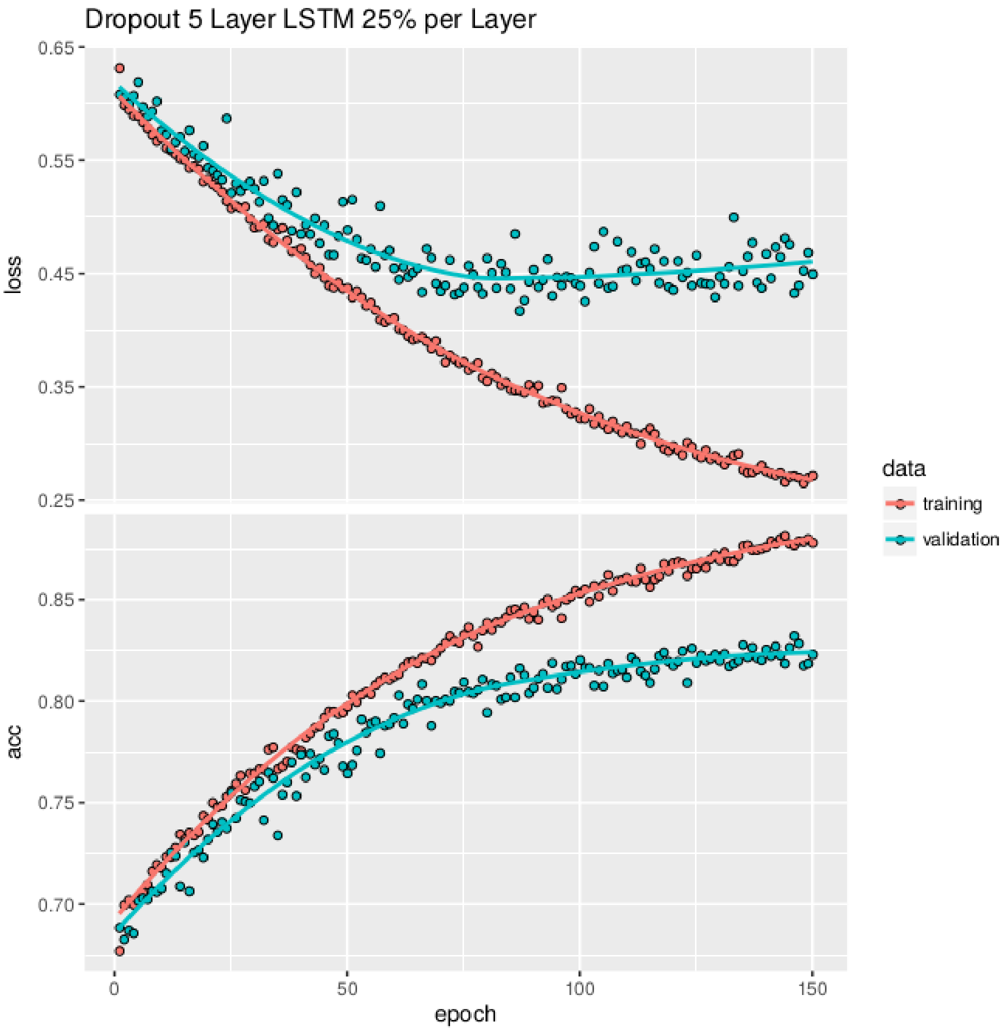


Figure 6. 5-Layer LSTM Result for Delay = 18, Lookback = 36

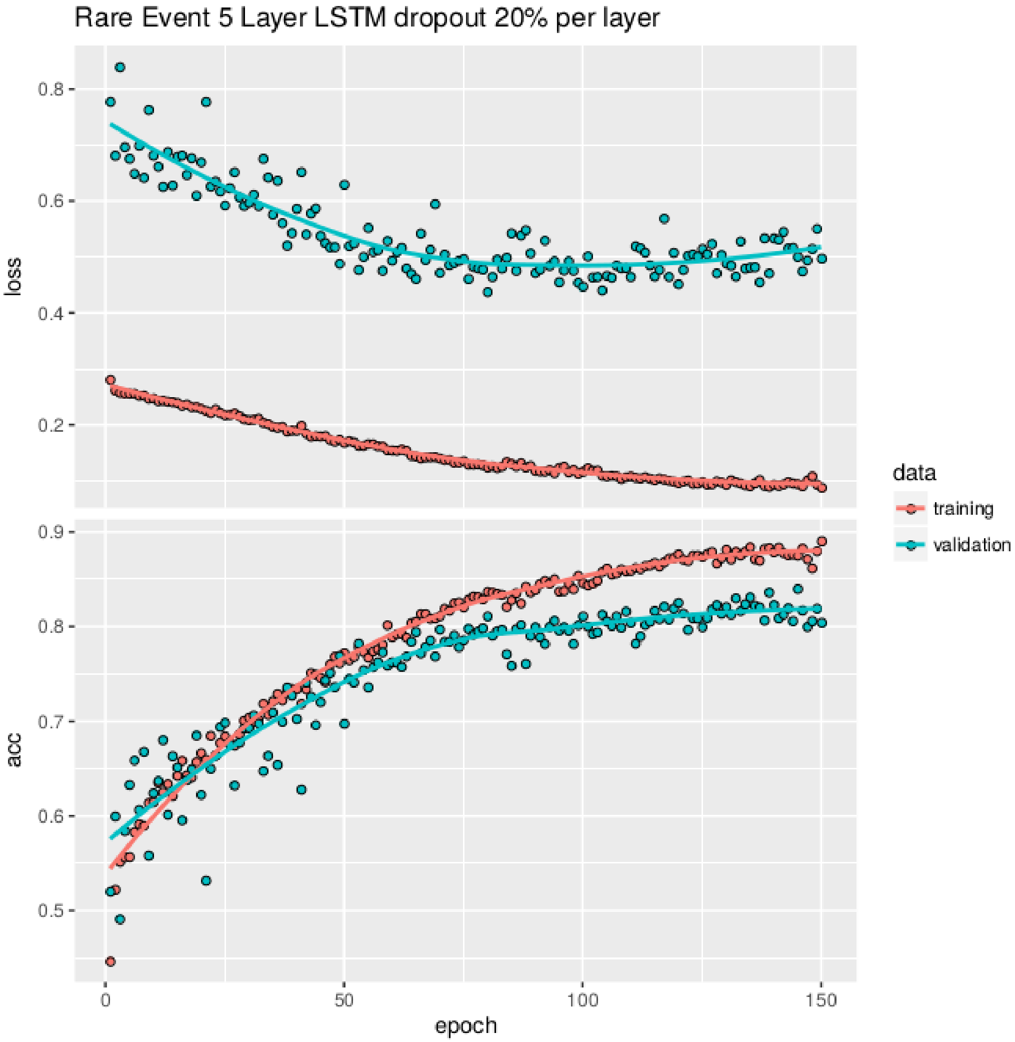


Figure 7. Rare Event LSTM Result for Delay = 18, Lookback = 36

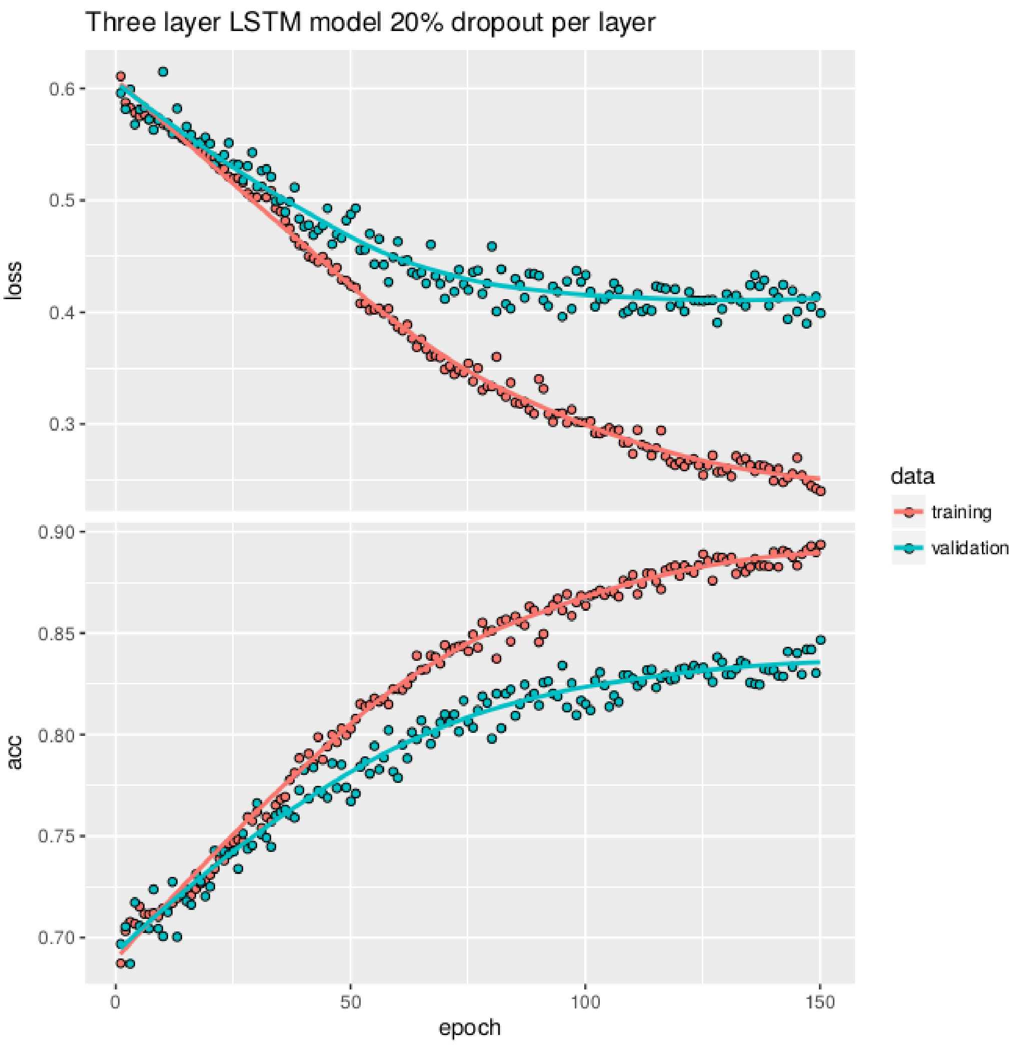


Figure 8. 3-Layer LSTM Result for Delay = 18, Lookback = 36

Notice that most models begin their validation accuracy right around the baseline mentioned in the previous chapter (right around 70%) but all of the models see a noticeable improvement as epochs increase. In general, most of the models run with differing parameters followed the same trends as these with different final accuracy and level off points.

For nearly all parameters run in the various models, the 5-Layer LSTM (Figure 6) produced the lowest binary cross-entropy loss and the greatest accuracy. This result is not too surprising as the model structure was far more complex than the

3-Layer LSTM. However, due to the increase in complexity, the 5-Layer LSTM took significantly longer to train than the 3-Layer LSTM. The highest accuracy achieved with this model structure produced a validation accuracy of 84.66%.

The Rare Event LSTM (Figure 7) had the same number of parameters as the 5-Layer LSTM. The only difference was that the Rare Event LSTM model attempted to capture the fact that lightning does not occur 50% of the time. While this model did better than the baseline 70%, it performed the worst of the three model structures on average. Additionally, the models with a smaller lookback parameter ended up having a large amount of variance in the validation set as the model was training instead of the tightly clustered trends found in the 5-Layer and 3-Layer models. This make the results less significant than the other two model structures.

Even though the 3-Layer LSTM (Figure 8) had far fewer parameters than the 5-Layer LSTM (Figure 6), it's performance was very close to the 5-Layer LSTM. Typically, the 3-Layer LSTM was only a percentage point or two lower than the 5-Layer model, but ran several times faster when training the model.

The 5-Layer LSTM performed the best compared to the other two model structures. Therefore, it is logical to choose the 5-Layer LSTM for a more in-depth analysis. In general, similar trends followed for each of the other model structures.

Parameter Tuning

As mentioned in Chapter 3.4, a fractional factorial design of experiment was run and the results used to measure the effect of delay and lookback on model accuracy and loss for the model types. The results for the 5-Layer LSTM model structure for each of the parameter pairs in the fractional factorial design are illustrated in Table 4. The results for the Rare Event LSTM and 3-Layer LSTM are in Appendix C.

Results for 5-Layer LSTM Model for each set of parameters

Delay	Lookback	Validation Accuracy (Final Epoch)	Validation Loss
6	12	74.94	0.5458
9	18	78.01	0.4947
18	24	78.19	0.4811
12	30	83.66	0.4262
21	24	80.49	0.4822
24	18	78.53	0.4767
3	12	73.53	0.5727
3	30	82.9	0.489
21	42	83.19	0.4347
18	30	81.69	0.4566
12	36	82.89	0.4277
24	48	83.6	0.4308
12	18	76.41	0.5227
12	24	81.35	0.4484
6	36	83.23	0.4544
9	48	84.3	0.4359
18	36	82.34	0.4503
18	48	**84.66**	0.4688
6	48	83.21	0.4552
21	48	83.91	**0.4149**
6	24	80.11	0.5039
3	24	78.55	0.5235
12	18	76.41	0.5227
6	18	75.55	0.5291
21	12	77.11	0.5118
24	36	83.23	0.4478

Table 4. Results for All 5-Layer LSTM Runs

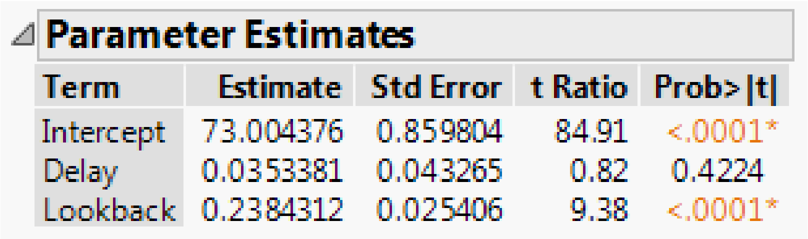
The runs with the best accuracy for the final epoch and the run with the lowest binary cross-entropy loss are underlined in Table 4. Both of these runs had a lookback value of 48 hours. This makes sense as more lookback increases the amount of data being looked at for each output, capturing more complex trends. The trade-off with having a larger lookback is that the models will take longer to initially run and train. For example, the models with 48 hour lookback took roughly twice as long to train and required significantly more memory than the models with 24 hours of lookback. The delay parameter had no bearing on how long the models took to train as no additional data is required. The delay simply looked at a an output for a different time step.

Parameter Analysis

With the results from Table 4, regression analysis was conduced to determine which parameters had a bearing on model accuracy. Both the delay and lookback are treated as continuous variables. JMP software created a least squares design with the delay and lookback parameters. This method finds a line of best fit by minimizing the sums of squares created by the regression formula. The parameter estimates, residuals, and plots are all derived from the minimization of the sum of squares for the line of best fit. The data used in the least squares analysis is located in Table 4.

Validation Accuracy and Loss for the Final Epoch - Linear Model

The linear model derived from the results of the designed experiment led to some interesting results. The parameter estimates for delay show that there is no statistically significant effect on accuracy. However, the lookback parameter showed a statistically significant effect. This estimate was a positive coefficient meaning that the more lookback used in building the models, the better the accuracy got. This makes sense as the run with the best result of 84.66% featured a lookback parameter of 48 (Figure 9).



Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Intercept	73.004376	0.859804	84.91	<.0001*
Delay	0.0353381	0.043265	0.82	0.4224
Lookback	0.2384312	0.025406	9.38	<.0001*

Figure 9. Parameter Estimates for Validation Accuracy for the Final Epoch

The residual plot for the data Figure 10 yielded concerning results. For results and parameter estimates to be valid, the residuals must show a linear relationship with a mean of 0 and slope of 0. The residual plot shown in Figure 10 does not seem to be linear. The residuals plotted look quadratic in nature meaning a quadratic term for lookback may be missing from the model.

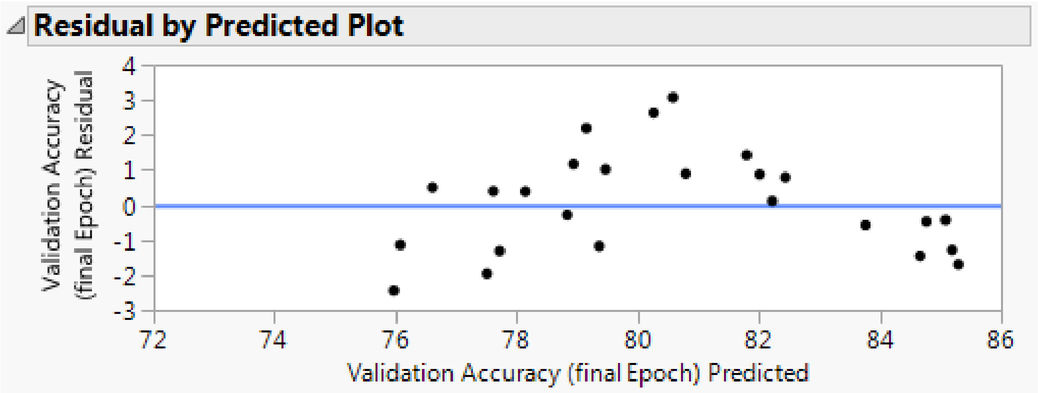


Figure 10. Residual Plot for Validation Accuracy for the Final Epoch

Similar residual plot assumptions seemed to be violated (Figure 11) when looking at the validation loss for the final epoch of the models. These abnormal residuals require additional analysis to gain a better understanding of the data.

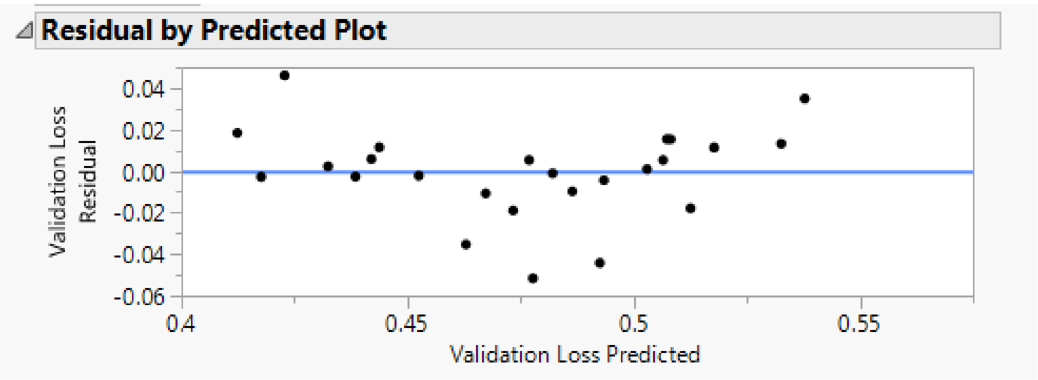
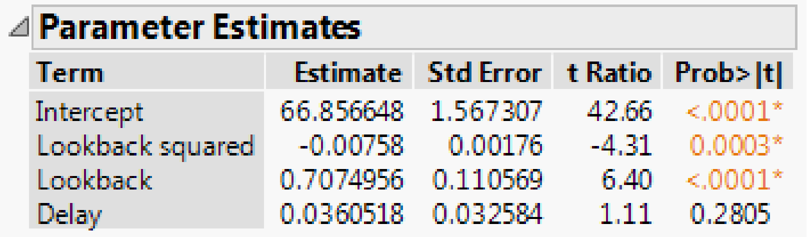


Figure 11. Residual Plot for Validation Loss for the Final Epoch

Validation Accuracy for the Final Epoch - Quadratic Model

Because the lookback value in the linear model showed signs of significance, an additional parameter was created to capture the possibility of a quadratic interaction when determining accuracy and loss. A new parameter equal to the lookback values squared was added to the model.

Looking at the parameter estimates in the quadratic model, both the linear and quadratic terms for lookback showed signs of significance with a low p-value. The linear parameter estimate remained positive meaning that the longer the lookback, the greater the accuracy of the model. However, the parameter estimate for the quadratic lookback term was negative. This suggests that while the accuracy is increasing as lookback increases, there are diminishing returns as to how much the accuracy will increase as the lookback increases. The delay parameter was still insignificant meaning that increasing the delay parameter had little to no effect on the accuracy of the model (Figure 12).



Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Intercept	66.856648	1.567307	42.66	<.0001*
Lookback squared	-0.00758	0.00176	-4.31	0.0003*
Lookback	0.7074956	0.110569	6.40	<.0001*
Delay	0.0360518	0.032584	1.11	0.2805

Figure 12. Residual Plot for Validation Accuracy for the Final Epoch in the Quadratic Model

While the linear model had problems when plotting the residuals, these problems were fixed when the quadratic parameter was introduced. There are no additional problems seen within the residual plot meaning that the results meet the assumptions of the regression model (Figure 13).

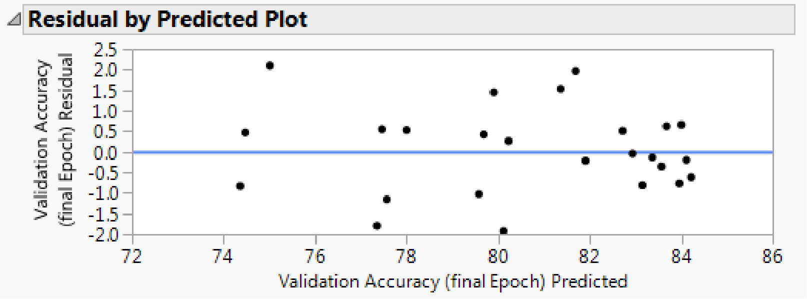
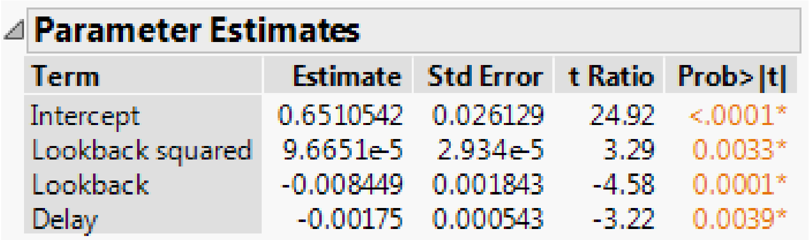


Figure 13. Residual Plot for Validation Accuracy for the Final Epoch in the Quadratic Model

Validation Loss for the Final Epoch - Quadratic Model

The validation loss output from the results showed different results than the validation accuracy. For the linear model in the validation loss, similar quadratic trends were seen. The parameter estimates for lookback were negative meaning that the greater the lookback, the less loss. This makes sense as the lowest loss value occurred with a lookback of 48 hours. The quadratic term was also significant and positive in nature. This is similar to the validation accuracy models showing the potential for diminishing returns as lookback increases. The most surprising result from the quadratic model showed a statistically significant result for the delay parameter. The delay parameter showed a negative coefficient. While this result was not as significant as the lookback parameters, the negative is a cause for concern. This is the opposite result that would be expected from increasing delay. In general, as the delay increases, the validation accuracy and loss are expected to decrease and increase respectively. This is because it is supposed to be more difficult to predict weather events further into the future (Figure 14).



Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Intercept	0.6510542	0.026129	24.92	<.0001*
Lookback squared	9.6651e-5	2.934e-5	3.29	0.0033*
Lookback	-0.008449	0.001843	-4.58	0.0001*
Delay	-0.00175	0.000543	-3.22	0.0039*

Figure 14. Residual Plot for Validation Loss for the Final Epoch in the Quadratic Model

As before, introduction of the quadratic parameter ensured the resulting model met the linear model assumptions (Figure 15).

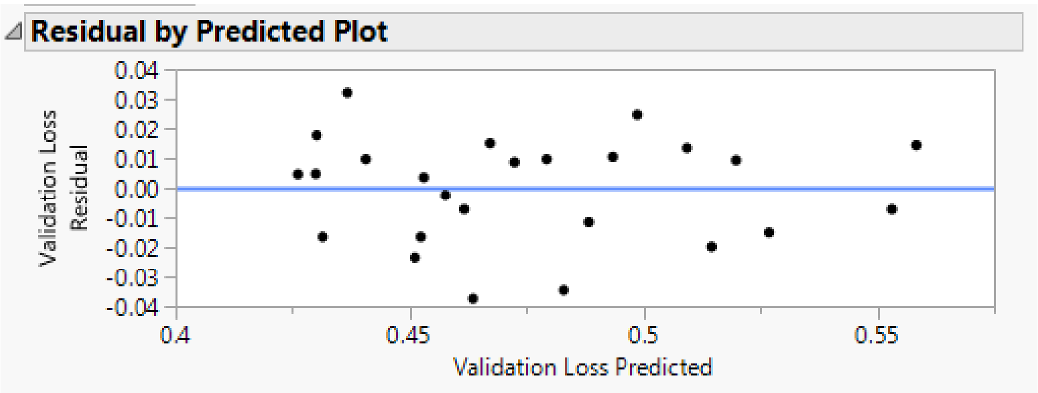
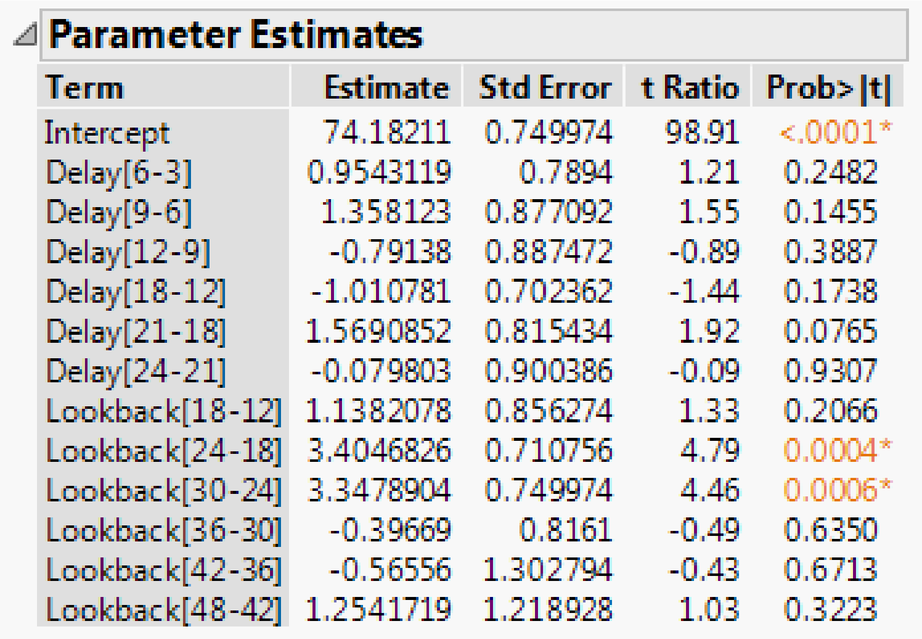


Figure 15. Residual Plot for Validation Loss for the Final Epoch in the Quadratic Model

A Closer Look

Grouping the lookback and delay parameters into ordinal groupings and running another least squares analysis is an effective way to gain more insight. The resulting parameter estimates (Figure 16) show a positive increase in accuracy when changing the lookback parameter from 18 hours to 24 hours and 24 hours to 30 hours. The change from 30 hours to 36 hours did not show any significance. This suggests diminishing returns from increasing the lookback value may begin around the 30 hour mark. While increasing past 30 hours may produce better results, the increase may be less than the initial increases between 12-30 hours.



Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Intercept	74.18211	0.749974	98.91	<.0001*
Delay[6-3]	0.9543119	0.7894	1.21	0.2482
Delay[9-6]	1.358123	0.877092	1.55	0.1455
Delay[12-9]	-0.79138	0.887472	-0.89	0.3887
Delay[18-12]	-1.010781	0.702362	-1.44	0.1738
Delay[21-18]	1.5690852	0.815434	1.92	0.0765
Delay[24-21]	-0.079803	0.900386	-0.09	0.9307
Lookback[18-12]	1.1382078	0.856274	1.33	0.2066
Lookback[24-18]	3.4046826	0.710756	4.79	0.0004*
Lookback[30-24]	3.3478904	0.749974	4.46	0.0006*
Lookback[36-30]	-0.39669	0.8161	-0.49	0.6350
Lookback[42-36]	-0.56556	1.302794	-0.43	0.6713
Lookback[48-42]	1.2541719	1.218928	1.03	0.3223

Figure 16. Ordinal Parameter Estimates for Accuracy

To see other plots and information gathered from the analysis, see Appendix B.

4.3 Experimental Results

The results from the experiment were only half as expected. For both the validation accuracy and validation loss, it was expected that increasing the amount of lookback would increase accuracy and decrease the loss but would have diminishing returns at some point. Diminishing returns became evident around 30 hours of lookback meaning that adding additional data to the model by increasing the lookback further than 30 hours did not significantly improve the models. The delay parameter showed the most suprising result. Trying to predict further away from the data should result in worse results. Surprisingly, for validation loss, the opposite was true. This is a cause for concern in addressing the research question.

The experiment showed non-intuitive results for the changing of delays between the various models. Even though the results showed an increase of around 10% for accuracy when compared to the baseline of persistence (roughly 70%), there may have been other factors contributing to earth's natural daily cycle. By examining the original dataset, more information was gathered about which hours of the day lightning generally occurs. For the most part, the lightning variable in the dataset occurred primarily in the afternoon/evening hours and occurred noticeably less in the night/morning hours. This might serve as an explanation as to why increasing the delay did not significantly affect the results for accuracy and had a non-intuitive effect on the loss. This realization serves as an example as to why further research is required to produce better results that can be used by the 45th Weather Squadron. Weather in general is diurnal meaning that it cycles daily. This diurnal pattern is seen in figure (17). The majority of the lightning occurs around CCAFS during the hours of 1400-midnight with a significant spike in the late afternoon/early evening. The initial assumption of persistence may be slightly skewed due to this diurnal pattern causing the baseline result to actually be better than roughly 70% previously stated.

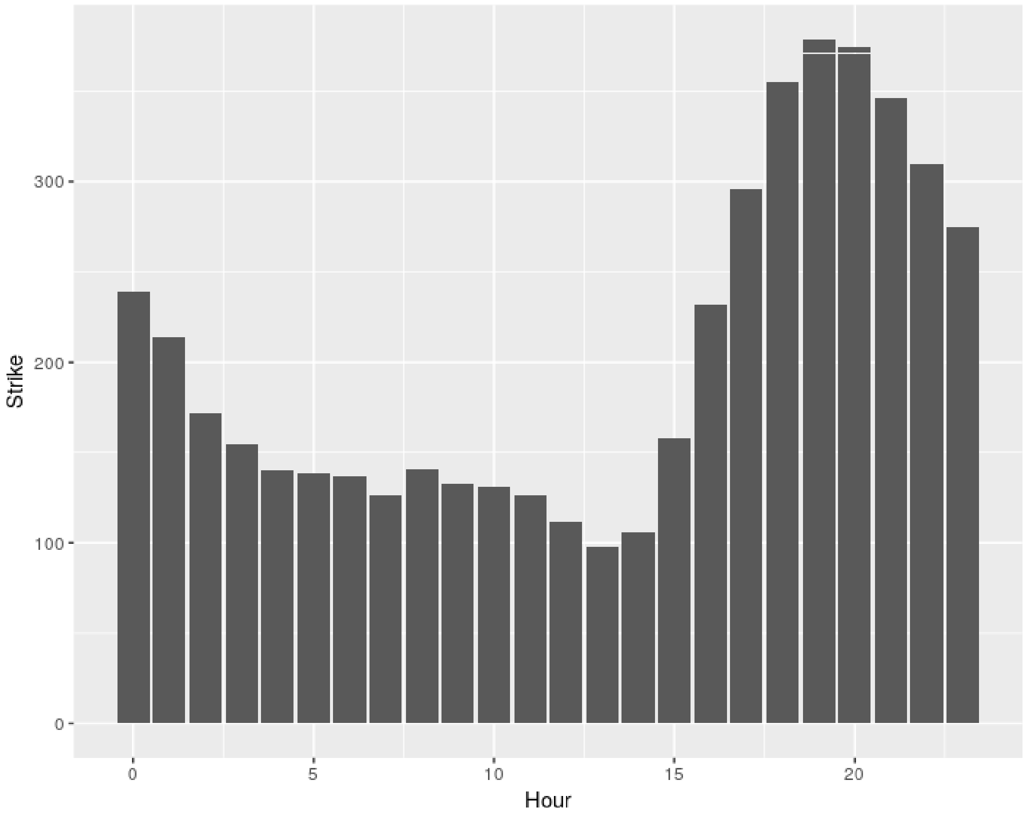


Figure 17. Lightning Count by the Hour

The results from lookback showed that increasing lookback past 30 hours showed diminishing returns. Increasing lookback when performing the experiment was computationally expensive causing the models to take significantly longer to train. With these time-series data, it is easy to modify the lookback but is still vitally important in choosing a good length of lookback so to not waist computational time. These results for the effect of lookback can be used in future research when examining the data as looking back more than 30 hours should only be looked at if the experiment has enough time to run.

4.4 Conclusion

The overall success of the experiment is illustrated in the examination of the lookback along with the failing result of the delay parameter. The 5-Layer and 3-Layer models showed relatively similar results even though the 3-Layer models took significantly less time to train (even if more epochs were run). This showed that a potential change to model structure may end up helping future research in determining which models to run to make the time most efficient. Sample code for what was actually run in the experiment in R can be found in Appendix C. Note that this is code for just one set of parameters. Similar structures were run for all of the other sets of parameters in the models.

V. Conclusion/Future Work

5.1 Introduction

This chapter is a brief discussion of an overview of the results, shortcomings of the analysis done for lightning detection, what could have been improved upon given more time, and follow-on thesis level work.

5.2 Overview of Results

The analysis for this research stemmed from the research questions presented in Chapter I:

1. Which variables can be used as regressors to better predict lightning strikes around Cape Canaveral?
2. Can an improved model be formed to better predict lightning at CCAFS/KSC?

The analysis provided insight into both of these questions although follow-on research is required for more definitive answers.

Through the use of LSTM structures, EFM data, and surface observations, models were made to predict lightning at a maximum of 84% accuracy. This provided significantly greater results than the day-before baseline which came in around 70% accurate. The 5 Layer LSTM model structure with a lookback of 48 hours achieved the best result. Upon further analysis into the results, the delay parameter showed little to no significance when predicting lightning. This may be caused by the diurnal pattern as shown in Figure 17. Nevertheless, with a robust analysis on the lookback parameter, the analysis shows that there is diminishing returns on increasing the lookback past roughly 30 hours. This result can be used in follow-on research in order to focus more closely on the delay parameter perhaps trying to use much shorter time window.

5.3 Improving the Models

Neural Networks are a growing field of study that have great potential to be very powerful tool in machine learning. However, Neural Networks are still a “black box” technique. This means that once the data has been inputted into the model to be trained, the algorithms and math happening in the back ground of the training grows increasingly complex. As shown, several of the models run for this analysis had upwards of two million parameters that were being trained when trying to figure out how to predict lightning around PAFB. Due to this complexity, it proves to be increasingly difficult to figure out the most optimal structure to train the data on. Additionally, if an optimal structure was found, there would be no way of knowing if that truly was the optimal design or if tweaking one of the input parameters would improve the model. Because of this, to gain more insight into the data, the models presented in the analysis could be more finely tuned. Time constraints and the size/complexity of the various types of models prevented a more finely tuned analysis. There are an infinite number of possible model structures that could be tested to see if they outperform the models in this analysis. Nevertheless, this analysis provided enough insight to show that there is a great potential with neural networks in order to more accurately predict lightning strike around CCAFS.

5.4 Follow-on Ideas

Potential follow-on research could take two approaches to the problem: create better models for the existing data or try and get more data for the models to train and validate on. Once the data was initially compressed down to one hour increments, there were only 14880 total observations. The limited observations stemmed from surface observations occurring once per hour at most locations. To remain consistent with the surface observations, the EFM data was compressed down into 1 hour increments. Additionally, the surface observation from all of the weather stations showed relatively incomplete data as discussed in the Chapter III. If follow-on research is able to get complete and more frequent data from these surface observations, the field mill data could be compressed to smaller increments and the training data could expand greatly. Just changing the surface observation data to occur every 30 min rather than every hour would double the amount of data that would be used in training and validation. While this would not promise better accuracy results, it would make the users of the neural networks more confident in their output.

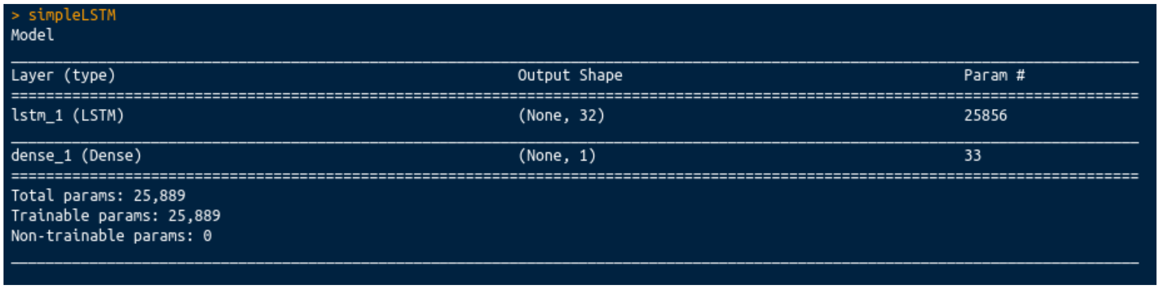
Other follow-on research ideas include looking at much shorter time steps in order get better prediction accuracy for a shorter time window. The 45th Weather Squadron suggested that, along with the shorter time window, a look at the interaction between the different field mills may yield interesting results. This would add a spacial component to the models which also might benefit from adding in some convolutional layers.

5.5 Conclusion

This research serves as a baseline for follow-on research done on the topic of lightning prediction around CCAFS. While the results may not have been as expected (particularly with the delay parameter), the methodology used to obtain the results will serve as a good stepping off point for future work. This research also dived into developing model structures that may potentially be useful with different data being inputted and trained.

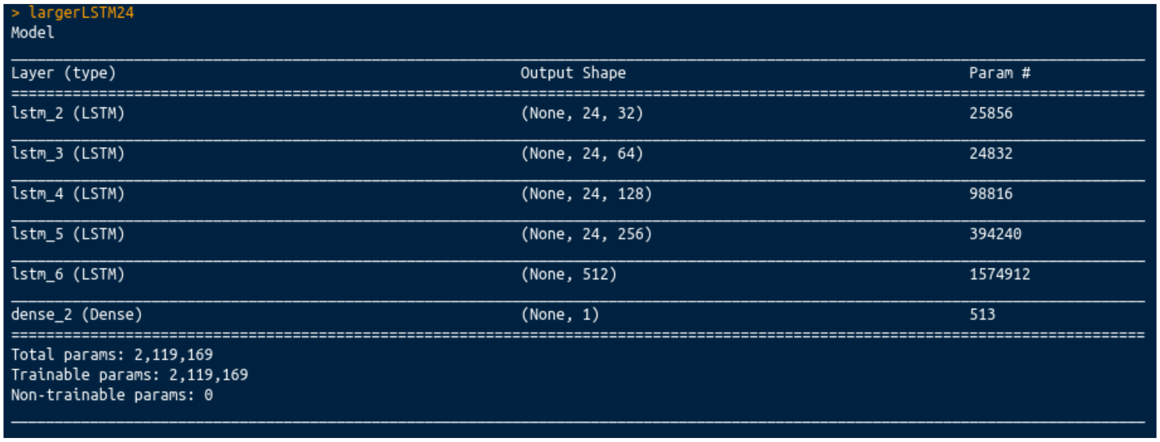
Appendices

Appendix A



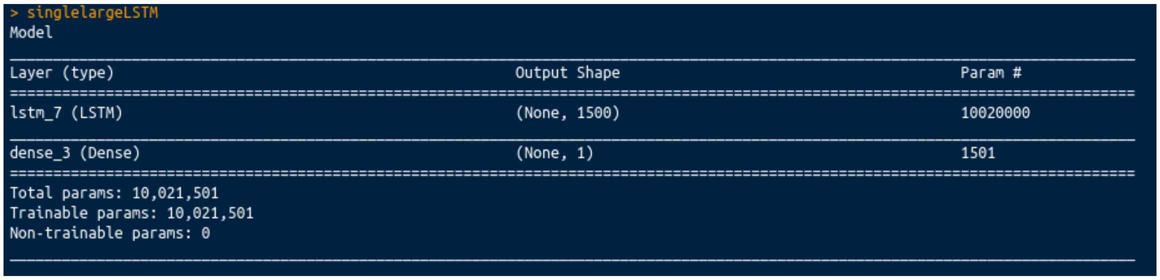
```
> simpleLSTM
Model
________________________________________________________________________________________________
Layer (type)                              Output Shape                         Param #
================================================================================================
lstm_1 (LSTM)                             (None, 32)                           25856
________________________________________________________________________________________________
dense_1 (Dense)                           (None, 1)                            33
================================================================================================
Total params: 25,889
Trainable params: 25,889
Non-trainable params: 0
________________________________________________________________________________________________
```

Figure 18. Single Layer LSTM



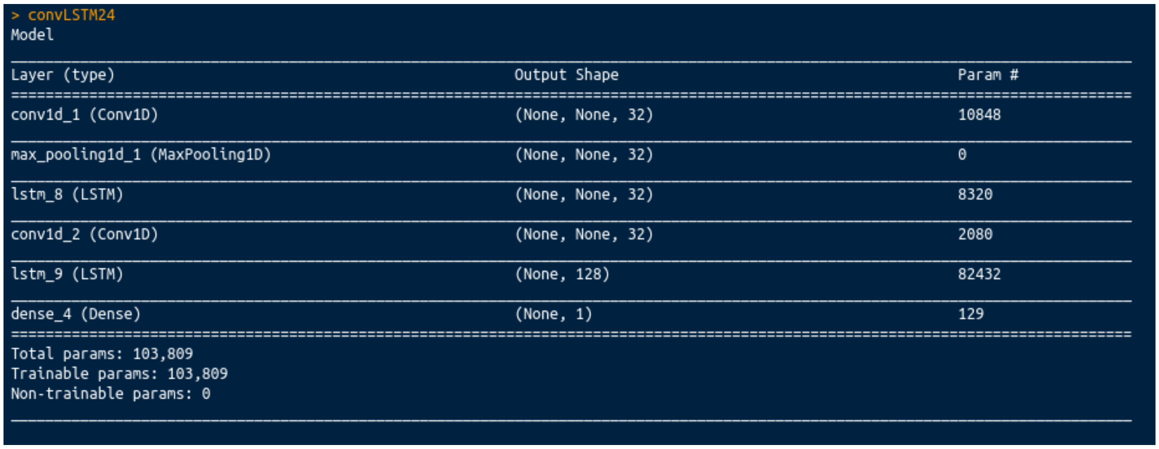
```
> largerLSTM24
Model
________________________________________________________________________________________________
Layer (type)                              Output Shape                         Param #
================================================================================================
lstm_2 (LSTM)                             (None, 24, 32)                       25856
________________________________________________________________________________________________
lstm_3 (LSTM)                             (None, 24, 64)                       24832
________________________________________________________________________________________________
lstm_4 (LSTM)                             (None, 24, 128)                      98816
________________________________________________________________________________________________
lstm_5 (LSTM)                             (None, 24, 256)                      394240
________________________________________________________________________________________________
lstm_6 (LSTM)                             (None, 512)                          1574912
________________________________________________________________________________________________
dense_2 (Dense)                           (None, 1)                            513
================================================================================================
Total params: 2,119,169
Trainable params: 2,119,169
Non-trainable params: 0
________________________________________________________________________________________________
```

Figure 19. 5 Layer No Dropout LSTM



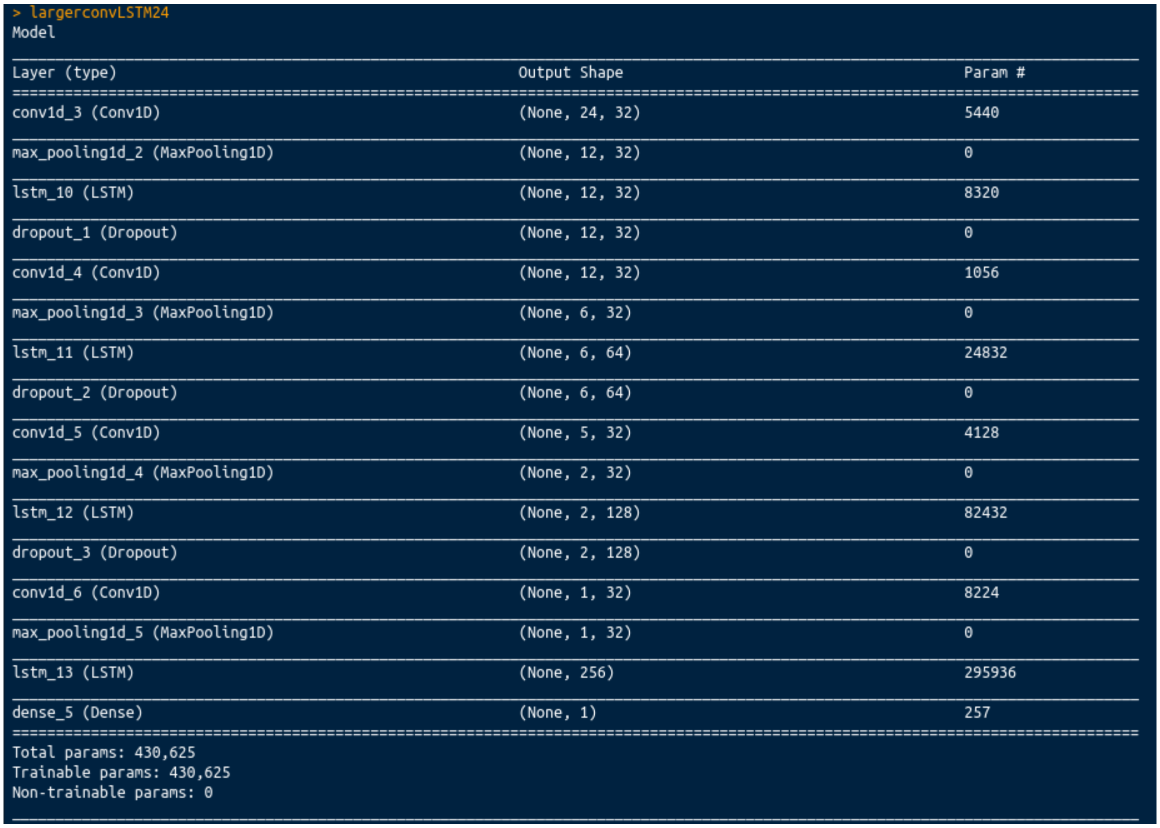
```
> singlelargeLSTM
Model
________________________________________________________________________________________________
Layer (type)                              Output Shape                         Param #
================================================================================================
lstm_7 (LSTM)                             (None, 1500)                         10020000
________________________________________________________________________________________________
dense_3 (Dense)                           (None, 1)                            1501
================================================================================================
Total params: 10,021,501
Trainable params: 10,021,501
Non-trainable params: 0
________________________________________________________________________________________________
```

Figure 20. Large Single Layer LSTM



```
> convLSTM24
Model
________________________________________________________________________________________________
Layer (type)                              Output Shape                          Param #
================================================================================================
conv1d_1 (Conv1D)                         (None, None, 32)                      10848
________________________________________________________________________________________________
max_pooling1d_1 (MaxPooling1D)            (None, None, 32)                      0
________________________________________________________________________________________________
lstm_8 (LSTM)                             (None, None, 32)                      8320
________________________________________________________________________________________________
conv1d_2 (Conv1D)                         (None, None, 32)                      2080
________________________________________________________________________________________________
lstm_9 (LSTM)                             (None, 128)                           82432
________________________________________________________________________________________________
dense_4 (Dense)                           (None, 1)                             129
================================================================================================
Total params: 103,809
Trainable params: 103,809
Non-trainable params: 0
________________________________________________________________________________________________
```

Figure 21. Small Convolutional LSTM



```
> largerconvLSTM24
Model
________________________________________________________________________________________________
Layer (type)                              Output Shape                          Param #
================================================================================================
conv1d_3 (Conv1D)                         (None, 24, 32)                        5440
________________________________________________________________________________________________
max_pooling1d_2 (MaxPooling1D)            (None, 12, 32)                        0
________________________________________________________________________________________________
lstm_10 (LSTM)                            (None, 12, 32)                        8320
________________________________________________________________________________________________
dropout_1 (Dropout)                       (None, 12, 32)                        0
________________________________________________________________________________________________
conv1d_4 (Conv1D)                         (None, 12, 32)                        1056
________________________________________________________________________________________________
max_pooling1d_3 (MaxPooling1D)            (None, 6, 32)                         0
________________________________________________________________________________________________
lstm_11 (LSTM)                            (None, 6, 64)                         24832
________________________________________________________________________________________________
dropout_2 (Dropout)                       (None, 6, 64)                         0
________________________________________________________________________________________________
conv1d_5 (Conv1D)                         (None, 5, 32)                         4128
________________________________________________________________________________________________
max_pooling1d_4 (MaxPooling1D)            (None, 2, 32)                         0
________________________________________________________________________________________________
lstm_12 (LSTM)                            (None, 2, 128)                        82432
________________________________________________________________________________________________
dropout_3 (Dropout)                       (None, 2, 128)                        0
________________________________________________________________________________________________
conv1d_6 (Conv1D)                         (None, 1, 32)                         8224
________________________________________________________________________________________________
max_pooling1d_5 (MaxPooling1D)            (None, 1, 32)                         0
________________________________________________________________________________________________
lstm_13 (LSTM)                            (None, 256)                           295936
________________________________________________________________________________________________
dense_5 (Dense)                           (None, 1)                             257
================================================================================================
Total params: 430,625
Trainable params: 430,625
Non-trainable params: 0
________________________________________________________________________________________________
```

Figure 22. Larger Convolutional LSTM

Appendix B

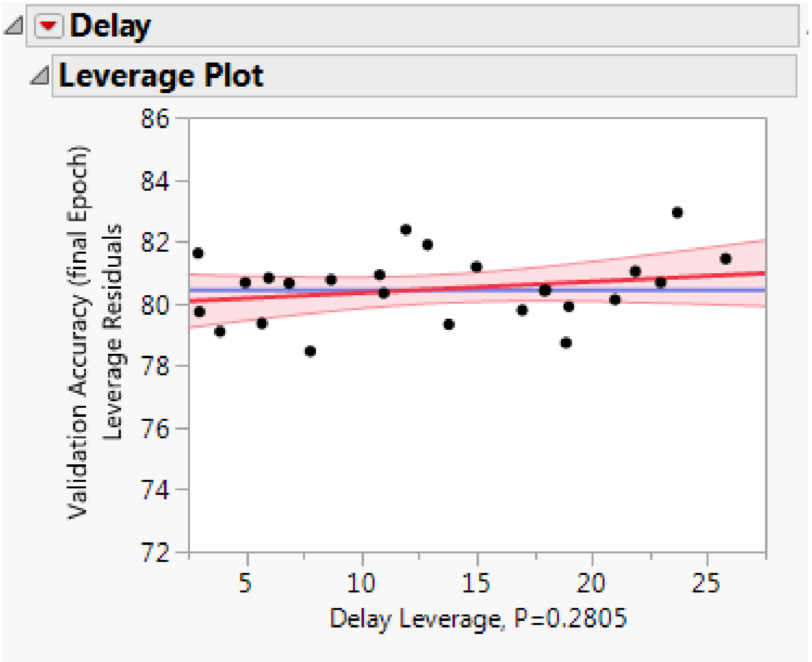


Figure 23. Leverage Plot for Lookback in Validation Accuracy

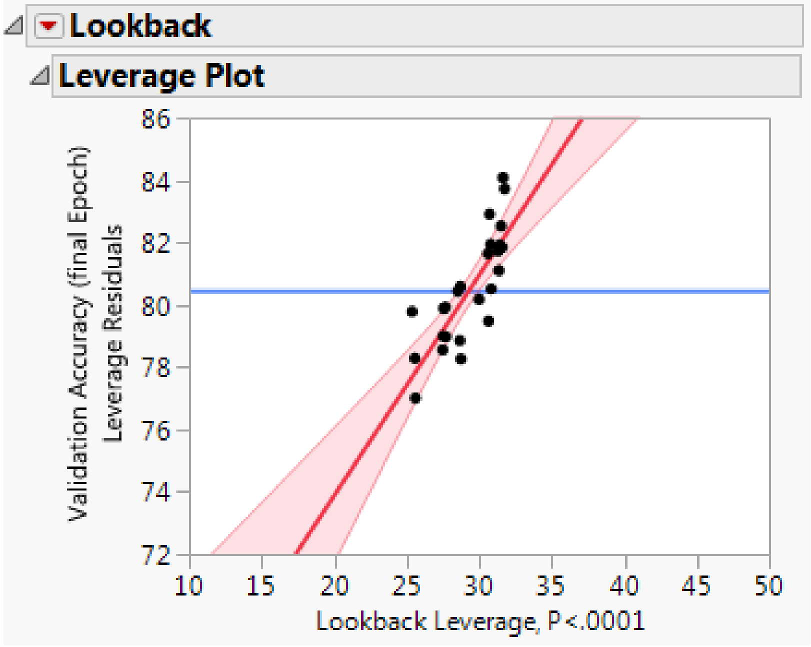


Figure 24. Leverage Plot for Lookback in Validation Accuracy

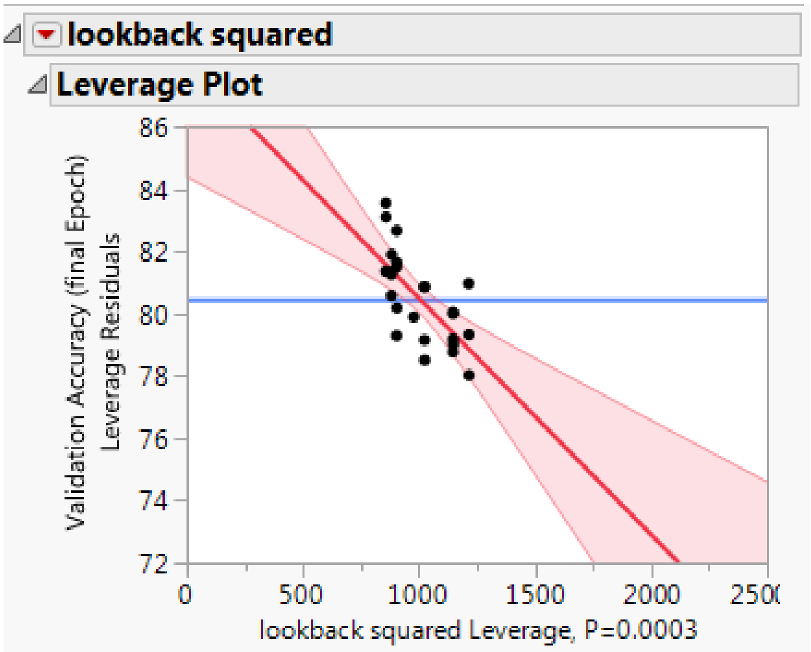


Figure 25. Leverage Plot for Lookback Squared in Validation Accuracy

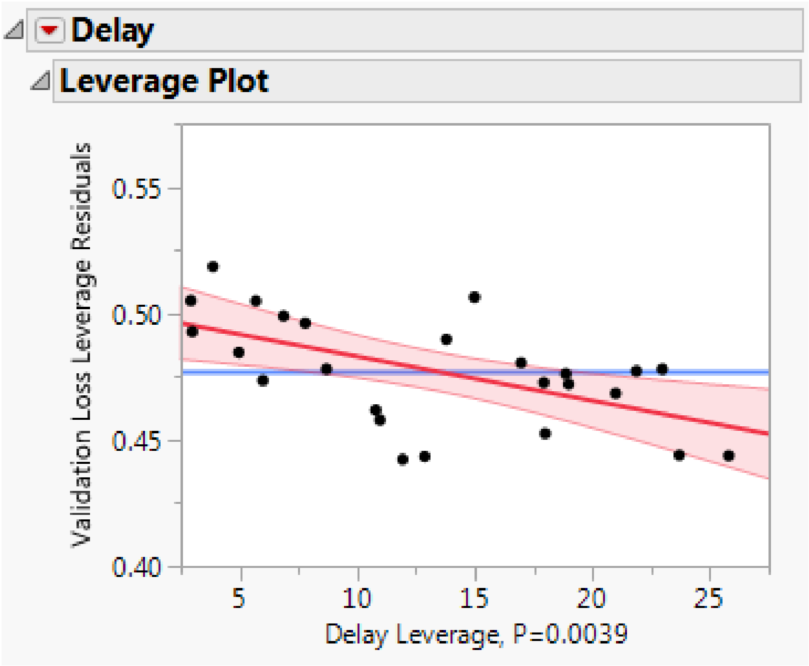


Figure 26. Leverage Plot for Delay in Validation Loss

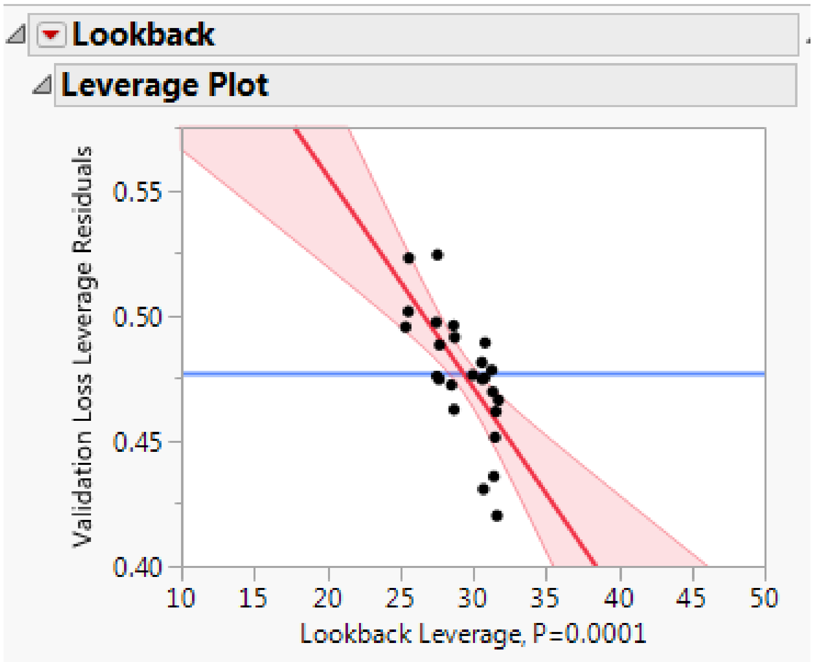


Figure 27. Leverage Plot for Lookback in Validation Loss

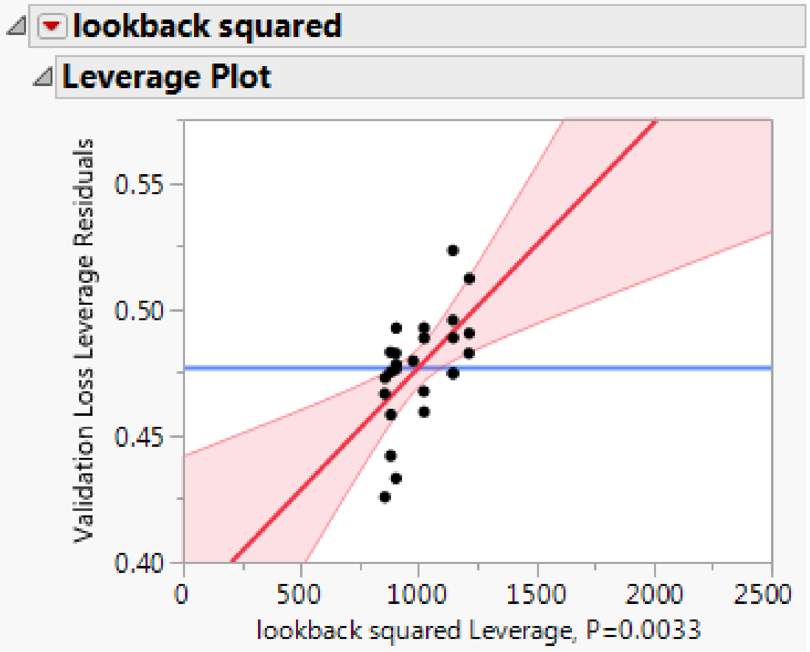


Figure 28. Leverage Plot for Lookback Squared in Validation Loss

Appendix C

Results for the Rare Events LSTM for Each Set of Parameters			
Delay	Lookback	Validation Accuracy (final Epoch)	Validation Loss
6	12	71.43%	0.6134
9	18	76.72%	0.5275
18	24	75.48%	0.5764
12	30	79.76%	0.5051
21	24	72.91%	0.5465
24	18	74.81%	0.5692
3	12	49.29%	0.6806
3	24	71.89%	0.5462
21	42	80.63%	0.4847
18	30	82.40%	0.4554
12	36	79.59%	0.548
24	48	82.20%	0.5023
12	18	72.42%	0.6379
12	24	77.16%	0.5522
6	36	82.04%	0.4667
9	48	81.00%	0.437
18	36	80.50%	0.4986
18	48	83.44%	0.4249
6	48	81.58%	0.4568
21	48	81.77%	0.4546
6	24	76.73%	0.5268
3	24	73.71%	0.6082
12	18	72.42%	0.6379
6	18	69.67%	0.6199
21	12	67.29%	0.6277
24	36	81.03%	0.5002

Table 5. Results for the RareEvents Model Structure

Results for 3 Layer LSTM Model for Each Set of Parameters			
Delay	Lookback	Validation Accuracy (final Epoch)	Validation Loss
6	12	74.82%	0.5670
9	18	77.92%	0.5213
18	24	79.31%	0.4798
12	30	82.12%	0.4532
21	24	79.22%	0.4783
24	18	75.82%	0.4767
3	12	73.94%	0.5775
3	24	80.65%	0.4710
21	42	81.53%	0.4214
18	30	79.21%	0.4368
12	36	81.87%	0.4195
24	48	82.40%	0.4212
12	18	75.23%	0.5133
12	24	80.89%	0.4447
6	36	81.26%	0.4386
9	48	83.68%	0.4309
18	36	80.15%	0.4328
18	48	83.02%	0.4557
6	48	83.51%	0.4576
21	48	83.60%	0.4224
6	24	80.21%	0.4997
3	24	79.67%	0.5230
12	18	77.83%	0.5011
6	18	76.09%	0.5295
21	12	76.13%	0.5040
24	36	82.73%	0.4438

Table 6. Results for the 3 Layer LSTM Model Structure

Appendix D

```
Option Explicit

Sub ChangeWindDir()
Dim row As Long
Dim col As Long
For row = 2 To 49531
    For col = 14 To 15
    If Worksheets("Sheet1").Cells(row, col).Value > 180 Then
        Worksheets("Sheet1").Cells(row, col).Value = Worksheets("Sheet1").Cells(row
            , col).Value - 360
    End If
    Next col
Next row

End Sub



Sub ReduceToHour()

'take min for visibility, cloud height, temperature, SLP, altimeter
'take average for wind direction
'take max wind speed, wind gust

Dim row As Long
Dim col As Long
Dim count As Long
Dim newVal As Double
Dim newCount As Long
Dim sum As Variant
Dim MaxVal As Long
Dim MinVal As Long
Dim numRows As Long

count = 1
newCount = 2
sum = 0
```

```
numRows = 49531



For col = 1 To 157

    'takes the average for specified columns.
    If col < 16 Then
        For row = 2 To numRows
            If Worksheets("Sheet1").Cells(row, 4) = Worksheets("Sheet1").Cells(row
                + 1, 4) Then
                count = count + 1
                If IsEmpty(Worksheets("Sheet1").Cells(row, col)) = True Then
                Else
                sum = sum + Worksheets("Sheet1").Cells(row, col).Value
                End If

            Else
                If IsEmpty(Worksheets("Sheet1").Cells(row, col)) = True Then
                newCount = newCount + 1

                Else
                sum = sum + Worksheets("Sheet1").Cells(row, col).Value
                newVal = sum / count
                Worksheets("Sheet2").Cells(newCount , col) = newVal
                newCount = newCount + 1
                sum = 0
                End If
                count = 1
                sum = 0
             End If
        Next row

     'takes the maximum over an hour for specified columns
    ElseIf col < 104 Then
        For row = 2 To numRows
            If Worksheets("Sheet1").Cells(row, 4) = Worksheets("Sheet1").Cells(row
                + 1, 4) Then
                count = count + 1
            Else
                If IsEmpty(Worksheets("Sheet1").Cells(row, col)) = True Then
                newCount = newCount + 1
                Else
                MaxVal = Application.WorksheetFunction.Max(Range(Worksheets("Sheet1
                    ").Cells(row - count + 1, col), Worksheets("Sheet1").Cells(row,
                     col)))
                Worksheets("Sheet2").Cells(newCount, col) = MaxVal
                newCount = newCount + 1
                End If
                count = 1
             End If
        Next row

'takes the minimum over an hour for specified columns.
    ElseIf col < 158 Then
        For row = 2 To numRows
            If Worksheets("Sheet1").Cells(row, 4) = Worksheets("Sheet1").Cells(row
                + 1, 4) Then
                count = count + 1
            Else
                If IsEmpty(Worksheets("Sheet1").Cells(row, col)) = True Then
                newCount = newCount + 1
                Else
                MinVal = Application.WorksheetFunction.Max(Range(Worksheets("Sheet1
                    ").Cells(row - count + 1, col), Worksheets("Sheet1").Cells(row,
                     col)))
                Worksheets("Sheet2").Cells(newCount, col) = MinVal
                newCount = newCount + 1
                End If
                count = 1
             End If
        Next row

    End If

newCount = 2
sum = 0
count = 1

Next col
```

```
End Sub



Sub FindLargeMissingData()

Dim row As Integer
Dim col As Integer
Dim count As Integer
Dim large As Integer
Dim i As Integer

count = 0
large = 0

For col = 5 To 88
    For row = 2 To 14212
        Do Until (IsEmpty(Worksheets("Sheet2").Cells(row, col)) = False)
            count = count + 1
            row = row + 1
            If row > 14212 Then
                Exit Sub
            End If
        Loop

        If count > 4 Then
            large = large + 1
            Worksheets("Sheet3").Cells(1, large + 1) = Worksheets("Sheet2").Cells
                (1, col)
            Worksheets("Sheet3").Cells(2, large + 1) = Worksheets("Sheet2").Cells(
                row, 1)
            Worksheets("Sheet3").Cells(3, large + 1) = Worksheets("Sheet2").Cells(
                row, 2)
            Worksheets("Sheet3").Cells(4, large + 1) = Worksheets("Sheet2").Cells(
                row, 3)
            For i = 1 To count
            Worksheets("Sheet3").Cells(i + 4, large + 1) = Worksheets("Sheet2").
                Cells(row - count + i, 4)
            Next i
            count = 0
        End If



    Next row
Next col
End Sub
```

```
library(doSNOW)
library(foreach)
library(parallel)
library(stringr)

# Start recording system time
start.time <- Sys.time()

# Set working directory to R file location
#this.dir <- getSrcDirectory(function(x) {x})
this.dir<-("D:/Hill Thesis/Thesis Data/RCode")
setwd(this.dir)


# PGD directory
setwd("../PGD")
PGD <- getwd()
PGD <- paste(PGD,"/trmm_pgd.exe",sep="")

# Output directory
setwd("../Unprocessed EFM Data")
outDir <- getwd()

# Input directory
setwd("../_EFM data (Original)")
inDir <- getwd()

# Gather list of zip files in inDir
zipFiles <- Sys.glob("*.zip")

# Set the number of clusters to the PC total - 1
cl<-makeCluster(detectCores() - 1)
registerDoSNOW(cl)

# Parallel loop through the zip files to process them
foreach(i=92:length(zipFiles)) %dopar% {

  library(stringr)
  outFile <- paste(outDir,"/",str_replace(zipFiles[i],".zip",""),sep="")
  dir.create(outFile, showWarnings = FALSE)
  unzip(zipFiles[i], files = NULL, list = FALSE, overwrite = TRUE,
        junkpaths = FALSE, exdir = outFile, unzip = "internal",
      setTimes = FALSE)
  subDirs<-list.dirs(path = outFile, full.names = TRUE, recursive = TRUE)

  if (length(subDirs)>0){
  for (j in 2:length(subDirs)) {
     setwd(subDirs[j])
     subZip <- Sys.glob("*.zip")
     if(length(subZip)>0){
     for (k in 1:length(subZip)) {
       unzip(subZip[k], files = NULL, list = FALSE, overwrite = TRUE,
             junkpaths = FALSE, exdir = ".", unzip = "internal",
```

```r
             setTimes = FALSE)
       # Delete zip file and keep dat file
       unlink(subZip[k], recursive = FALSE)
     }

     # copy the pgd program to the folder
     file.copy(PGD, getwd())
     subDat <- Sys.glob("*.dat")
     if(length(subDat)>0){
     for (k in 1:length(subDat)) {
       # Process dat file into RAW file

      # Process the RAW files in the command line
       system("trmm_pgd.exe", input = subDat[k], show.output.on.console = FALSE)

       # Delete dat file
       unlink(subDat[k], recursive = FALSE)
     }

     # Delete the local copy of the executable
     unlink("trmm_pgd.exe", recursive = FALSE)
     # Return to the root directory
     setwd(this.dir)
     }
     }
  }
  }
}

# Release the parallel cluster
stopCluster(cl)

# Calculate total run time
end.time <- Sys.time()
time.taken <- end.time - start.time
print(time.taken)
```

```r
#Example Code Used For Thesis
#this is the code written for a Delay of 18 and a Lookback of 36. All other sets of parameters followed similar structures.
#set up

library(data.table)
library(keras)
library(tensorflow)
library(mice)
library(VIM)
library(ggplot2)


setwd('/home/dom/Documents/DomThesis/RCode')
completeData<-readRDS('completeDataNoHolesImputed.rds')
setwd('/home/dom/Documents/DomThesis/ModelHistory')

#makes the generator function to get make the make data for LSTMs
weather_generator<-function(data,lookback,delay,min_index,max_index,shuffle = FALSE, batch_size = 128, steps = 1){
  if(is.null(max_index)) max_index <-nrow(data)-delay-1
  i<-min_index+lookback
  function(){
    if(shuffle){
      rows<-sample(c((min_index+lookback):max_index),size = batch_size)
    } else{
      if(i+batch_size>=max_index)
        i<<- min_index+lookback
      rows<-c(i:min(i+batch_size,max_index-delay))
      i<<- i+length(rows)
    }


    samples<-array(0,dim = c(length(rows),
                             lookback/steps,
                             dim(data)[[-1]]))
    targets<-array(0,dim = c(length(rows)))

    for (j in 1:length(rows)){
      indices<- seq(rows[[j]]-lookback,rows[[j]]-1)
      samples[j,,]<-data[indices,]
      targets[[j]]<-data[rows[[j]]+delay,dim(data)[[2]]]
    }

    list(samples,targets)
  }
}

#################################3
#D18L36
##################################


```

```
#now we can create the training, validation
#define our specs for the RNN
delay<-18        #trying to find the result for 18 hours ahead.
lookback<-36     #start with looking back 36 hours to gain the pattern
steps<-1         #each timestep is already in hours

batch_size<-3720-lookback



#sets the index for each of the years for the dataset
max_index2013<-3720
max_index2014<-3720+max_index2013
max_index2015<-3720+max_index2013*2

#changes the data in to a data.matrix
#also gets rid of the time data as we do not need it for analysis.
completeDataMatrix<-data.matrix(completeData[,-(1:4)])
#completeDataMatrix<-completeDataMatrix[,-123]

#creates each of the generators for the different years
gen2013_gen<-weather_generator(
  completeDataMatrix,
  lookback = lookback,
  delay = delay,
  min_index = 1,
  max_index = max_index2013,
  shuffle = FALSE,
  steps = steps,
  batch_size = batch_size
)


gen2014_gen<-weather_generator(
  completeDataMatrix,
  lookback = lookback,
  delay = delay,
  min_index = max_index2013+1,
  max_index = max_index2014,
  shuffle = FALSE,
  steps = steps,
  batch_size = batch_size
)


gen2015_gen<-weather_generator(
  completeDataMatrix,
  lookback = lookback,
  delay = delay,
  min_index = max_index2014+1,
  max_index = max_index2015,
  steps = steps,
  batch_size = batch_size,
  shuffle = FALSE
```

```
)

gen2016_gen<-weather_generator(
  completeDataMatrix,
  lookback = lookback,
  delay = delay,
  min_index = max_index2015+1,
  max_index = NULL,
  steps = steps,
  batch_size = batch_size,
  shuffle = FALSE
)

#extracts the data from the generators in a timeseries format.  looks at the previous lookback number of hours
gen2013 = gen2013_gen()
gen2014 = gen2014_gen()
gen2015 = gen2015_gen()
gen2016 = gen2016_gen()

#copies over the data into a more usiable form.
blankMatrix = array(0,batch_size*4*lookback*168-delay-1)
samples = array(blankMatrix,c(batch_size*4-delay-1,lookback,168))

#copies over the samples
for (i in 1:(batch_size-delay)){
  for (j in 1: lookback){
    for(k in 1: 168){
      samples[i,j,k]=gen2013[[1]][i,j,k]
    }
  }
}

for (i in 1:(batch_size-delay)){
  for (j in 1: lookback){
    for(k in 1: 168){
      samples[i+batch_size,j,k]=gen2014[[1]][i,j,k]
    }
  }
}


for (i in 1:(batch_size-delay)){
  for (j in 1: lookback){
    for(k in 1: 168){
      samples[i+batch_size*2,j,k]=gen2015[[1]][i,j,k]
    }
  }
}

for (i in 1:(batch_size-delay-1-delay)){
  for (j in 1: lookback){
    for(k in 1: 168){
      samples[i+batch_size*3,j,k]=gen2016[[1]][i,j,k]
    }
  }
}

#set up the targets
blankMatrix=array(0,batch_size*4-delay-1)
targets = array(blankMatrix,c(batch_size*4-delay-1))
for (i in 1: (batch_size-delay)){
  targets[i]=gen2013[[2]][i]
}
for (i in 1: (batch_size-delay)){
  targets[i+batch_size]=gen2014[[2]][i]
}
for (i in 1: (batch_size-delay)){
  targets[i+batch_size*2]=gen2015[[2]][i]
}
for (i in 1: (batch_size-delay-1-delay)){
  targets[i+batch_size*3]=gen2016[[2]][i]
}

#set up the training and validation sets.
set.seed(2019)
train_index<-sample(1:nrow(samples),.8*nrow(samples))

samples_train<-samples[train_index,,]
targets_train<-targets[train_index]

samples_val<-samples[-train_index,,]
targets_val<-targets[-train_index]


#########################################################
#5 Layer LSTM

moredropoutLSTM36<-keras_model_sequential()%>%
  layer_lstm(units = 32,input_shape = c(lookback,168),return_sequences = TRUE)%>%
  layer_dropout(.25)%>%
  layer_lstm(units = 64,return_sequences = TRUE)%>%
  layer_dropout(.25)%>%
  layer_lstm(units = 128,return_sequences = TRUE)%>%
  layer_dropout(.25)%>%
  layer_lstm(units = 256,return_sequences = TRUE)%>%
  layer_dropout(.25)%>%
  layer_lstm(units = 512)%>%
  layer_dense(units = 1,activation = "sigmoid")


moredropoutLSTM36%>% compile(
  optimizer = "rmsprop",
  loss = "binary_crossentropy",
  metrics = c("acc")
)


#we will train this model longer because of the drop out.
```

```r
moredropouthistoryD18L36<-moredropoutLSTM36%>%fit(
  samples_train,
  targets_train,
  epochs = 150,
  batch_size = 150,
  validation_data =  list(samples_val,targets_val)
)
plot(moredropouthistoryD18L36)+ggtitle("Dropout 5 Layer LSTM 25% per Layer")
saveRDS(layer3dropouthistoryD18L36,'Layer3_Delay18_Lookback36.rds')
######################################################

#####################################################
#rareEventLSTM

raredropoutLSTM36<-keras_model_sequential()%>%
  layer_lstm(units = 32,input_shape = c(lookback,168),return_sequences = TRUE)%>%
  layer_dropout(.2)%>%
  layer_lstm(units = 64,return_sequences = TRUE)%>%
  layer_dropout(.2)%>%
  layer_lstm(units = 128,return_sequences = TRUE)%>%
  layer_dropout(.2)%>%
  layer_lstm(units = 256,return_sequences = TRUE)%>%
  layer_dropout(.2)%>%
  layer_lstm(units = 512)%>%
  layer_dense(units = 1,activation = "sigmoid")

raredropoutLSTM36%>% compile(
  optimizer = "rmsprop",
  loss = "binary_crossentropy",
  metrics = c("acc")
)


raredropouthistoryD18L36<-raredropoutLSTM36%>%fit(
  samples_train,
  targets_train,
  epochs = 150,
  batch_size =  150,
  validation_data =  list(samples_val,targets_val),
  class_weight = list("0"=.25,"1"=.75)
)
plot(raredropouthistoryD18L36)+ggtitle("Rare Event 5 Layer LSTM dropout 20% per layer")
saveRDS(moredropouthistoryD18L36,'Layer5_Delay18_Lookback36.rds')

# #######################################################

#########################################################
#3 Layer LSTM

layer3dropoutLSTM36<-keras_model_sequential()%>%
  layer_lstm(units = 32,input_shape = c(lookback,168), return_sequences = TRUE)%>%
  layer_dropout(.2)%>%
  layer_lstm(units = 64,return_sequences = TRUE)%>%
  layer_dropout(.2)%>%
  layer_lstm(units =128)%>%
  layer_dense(units = 1,activation = "sigmoid")


layer3dropoutLSTM36%>% compile(
  optimizer = "rmsprop",
  loss = "binary_crossentropy",
  metrics = c("acc")
)




layer3dropouthistoryD18L36<-layer3dropoutLSTM36 %>%fit(
  samples_train,
  targets_train,
  epochs =  150,
  batch_size =  150,
  validation_data =  list(samples_val,targets_val)
)
plot(layer3dropouthistoryD18L36)+ggtitle('Three layer LSTM model 20% dropout per layer')
saveRDS(raredropouthistoryD18L36,'Rare5Layer_Delay18_Lookback36.rds')
##############################################

##############################################
# #get all the plots in the same spot
# # start pdf device
pdf(file='/home/dom/Documents/DomThesis/Delay18_Lookback36.pdf')

plot(moredropouthistoryD18L36)+ggtitle("Dropout 5 Layer LSTM 25% per Layer")

plot(raredropouthistoryD18L36)+ggtitle("Rare Event 5 Layer LSTM dropout 20% per layer")

plot(layer3dropouthistoryD18L36)+ggtitle('Three layer LSTM model 20% dropout per layer')

dev.off()
```

Bibliography

1. D. Hill, "Lightning Prediction Using Artifical Neural Networks and Electric Field Mill Data," tech. rep., Air Force Institute of Technology, 2018.

2. C. Tovar, D. Aranguren, J. López, J. Inampués, and H. Torres, "Lightning Risk Assessment and Thunderstorm Warning Systems," in *2014 International Conference on Lightning Protection, ICLP 2014*, pp. 1870–1874, 2014.

3. J. Allaire and F. Chollet, "keras: R Interface to 'Keras'," *Journal of Statistical Software*, vol. 63, no. 17, 2018.

4. R. J. Kuligowski and A. P. Barros, "Localized Precipitation Forecasts from a Numerical Weather Prediction Model Using Artificial Neural Networks," *Weather Forecasting*, vol. 13, pp. 1194–1204, 1998.

5. A. Grover, A. Kapoor, and E. Horvitz, "A Deep Hybrid Model for Weather Forecasting," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '15*, pp. 379–386, 2015.

6. H. J. Christian, V. Mazur, B. D. Fisher, L. H. Ruhnke, K. E. Crouch, and R. P. Perala, "The Atlas/Centaur Lightning Strike Incident," *Journal of Geophysical Research: Atmospheres*, vol. 94, no. D11, pp. 13169–13177, 1989.

7. C. Canright, "Lightning and Launches." https://www.nasa.gov/audience/foreducators/9%0A-12/features/F Lightning and Launches 9 12.html, 2001.

8. R. A. Mazany, S. Businger, S. I. Gutman, and W. Roeder, "A Lightning Prediction Index that Utilizes GPS Integrated Precipitable Water Vapor*," *Weather and Forecasting*, vol. 17, pp. 1034–1047, 2002.

9. H. Torres, E. Perez, C. Younes, D. Aranguren, J. Montaña, and J. Herrera, "Contribution to Lightning Parameters Study Based on Some American Tropical Regions Observations," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 8, no. 8, pp. 4086–4092, 2015.

10. S. Olsen, "Forecasting Lightning Initiation Utilizing Dual-Polarization Radar Parameters over Washington, D.C.," Tech. Rep. March, Air Force Institute of Technology, 2018.

11. N. Holden, "Forecasting Lightning Cessation Using Dual-Polarization Radar and Lightning Mapping Array near Washington, D.C.," tech. rep., Air Force Institute of Technology, 2018.

12. B. C. Bates, A. J. Dowdy, and R. E. Chandler, "Lightning Prediction for Australia using Multivariate Analyses of Large-scale Atmospheric Variables," *Journal of Applied Meteorology and Climatology*, vol. 57, pp. 525–534, 2018.

13. D. Aranguren, J. Inampués, H. Torres, J. López, and E. Pérez, "Operational Analysis of Electric Field Mills as Lightning Warning Systems in Colombia," *2012 31st International Conference on Lightning Protection, ICLP 2012*, vol. 31, no. 2, pp. 51–57, 2012.

14. W. P. Roeder, M. McNamara, M. McAleenan, K. A. Winters, L. M. Maier, and L. L. Huddleston, "The 2014 upgrade to the lightning warning areas used by the 45th Weather Squadron," *18th Conference on Aviation, Range, and Aerospace Meteorology*, 2017.

15. B. Lantz, "Machine Learning with R," pp. 219–259, 2015.

16. R. Ihaka and R. Gentleman, "R: A Language and Environment for Statistical Computing," *R Core Team*, 2018.

17. H. Wickham and G. Grolemund, *R for Data Science: Important, Tidy, Transform, Visualize, and Model Data*. Sebastopol,CA: O'Reilly Media, Inc., first ed., 2016.

18. H. Wickham, "tidyverse: Easily Install and Load the 'Tidyverse'," *Journal of Statistical Software*, vol. 59, no. 10, pp. 1–22, 2015.

19. G. Grolemund and H. Wickham, "Dates and Time Made Easy with lubridate," *Journal of Statistical Software*, vol. 40, no. 3, pp. 1–25, 2011.

20. S. van Buuren and K. Groothuis-Oudshoorn, "mice: Multivariate Imputation by Chained Equations in R," *Journal of Statistical Software*, vol. 45, no. 3, pp. 1–67, 2011.

21. I. Maqsood, M. Khan, and A. Abraham, "An Ensemble of Neural Networks for Weather Forecasting," *Neural Computing and Applications*, vol. 13, pp. 112–122, 2004.

22. M. Bashiri and A. Farshbaf Geranmayeh, "Tuning the parameters of an artificial neural network using central composite design and genetic algorithm," *Scientia Iranica*, vol. 18, no. 6, pp. 1600–1608, 2011.

23. L. Huddleston, "Spaceport Weather Archive." https://kscwxarchive.ksc.nasa.gov/, 2018.

24. J. Goodnight, "JMP pro 13.0," *Statistical Analysis System - SAS*, vol. 13.0, 2016.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

1. REPORT DATE *(DD–MM–YYYY)*	**2. REPORT TYPE**	**3. DATES COVERED** *(From — To)*
21–03–2019	Master's Thesis	Oct 2017–March 2019

4. TITLE AND SUBTITLE

LIGHTNING PREDICTION USING
RECURRENT NEURAL NETWORKS

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Dominick V. Speranza III

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT-ENS-MS-19-M-150

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

45th Weather Squadron
Morrell Operations Center
Cape Canaveral, FL 32920
Comm: 321-853-8484 DSN: 467-8484
Email: 45wsdor@us.af.mil

10. SPONSOR/MONITOR'S ACRONYM(S)

45th Weather Squadron

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

DISTRIBUTION STATEMENT A:
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

This material is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

14. ABSTRACT

In this paper, hourly statistics were generated from the raw EFMs data set used in Hill [1]. Input variables were generated from surface observations from every station within 50 miles of CCAFS and then combined with the EFM statistics for the same time periods. This combined data set was used to create Long Short-term Memory (LSTM) Neural Networks designed to capture trends within the data for each observation. A variety of different LSTM model structures were created and trained to see which model structure performed best when predicting lightning around CCAFS, KSC, and PAFB. By utilizing design of experiments techniques, optimal parameters for the LSTM model structures are narrowed down providing a solid baseline for future endeavors in predicting lightning.

15. SUBJECT TERMS

LaTeX,Thesis,typesetting

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			Lt. Col. A. Geyer, PHD
U	U	U	U	76	**19b. TELEPHONE NUMBER** *(include area code)* (937) 255-3636, x4630; dominick.speranza@afit.edu