APS360: WEATHER FORECASTING

Andrew Radke
Student# 1008222237
andrew.radke@mail.utoronto.ca

Matthew Grech
Student# 1007997544
matthew.grech@mail.utoronto.ca

Mitchell Palermo
Student# 1008038589
mitchell.palermo@mail.utoronto.ca

Liza Abraham
Student# 1008008371
l.abraham@mail.utoronto.ca

ABSTRACT

This project progress report provides an update on a weather forecasting project utilizing deep learning, specifically a Recurrent Neural Network (RNN), with VARMAX serving as the baseline model. The report highlights the team's progress in data collection and cleaning, presenting relevant statistics and examples of the cleaned training samples. Furthermore, the baseline model, implemented as VARMAX, is described, and compared with the RNN model. Results from the baseline model are presented to assess its feasibility in achieving the project objectives. Challenges encountered during the project's execution are also discussed, providing insights into potential obstacles and considerations for future work.

The team's individual contributions and responsibilities have been carefully outlined to ensure efficient collaboration and progress. To facilitate effective communication, progress tracking, and code updates, the team has utilized project management software such as Gantt charts and debriefs. This has provided a way to facilitate team members to share updates, track tasks, and monitor the overall progress of the project.

—-Total Pages: 7

1 PROJECT DESCRIPTION

This project aims to leverage deep learning to make accurate weather forecasts, which can have significant implications in various domains such as agriculture, transportation, energy management, and personal planning. The goal of this project is to develop a deep learning model using Long Short-Term Memory (LSTM) to forecast different weather metrics given current weather data.

The motivation behind this project stems from the importance of accurate weather predictions in our daily lives. Weather conditions directly impact people's decisions, from choosing appropriate clothing for the day to planning outdoor activities and optimizing energy consumption. By accurately forecasting the temperature for the next hour, we can provide users with valuable information to make informed decisions.

Deep learning, specifically LSTM networks, is a suitable approach for this task due to its ability to capture temporal dependencies and patterns in sequential data. Weather data exhibits temporal characteristics, where the current temperature is influenced by the previous temperature readings and other related factors. LSTMs can effectively model these dependencies and capture long-term dependencies, making them a suitable choice for weather forecasting.

The input to our model consists of historical weather data, including variables such as temperature, humidity, wind speed, and atmospheric pressure, collected at regular time intervals. These inputs are used to predict the temperature for the next hour, which is the output of our model. Deep learning

models excel at learning complex patterns and relationships in large datasets, making them well-suited for generating accurate predictions from such inputs.

To minimize required reading and facilitate understanding, we have included a concise and easy-to-read diagram showcasing the input and output of our LSTM model. The diagram illustrates the flow of historical weather data as input to the LSTM, and the predicted temperature for the next hour as the output. This visual representation helps clarify the project's goals and the role of deep learning in generating weather predictions.

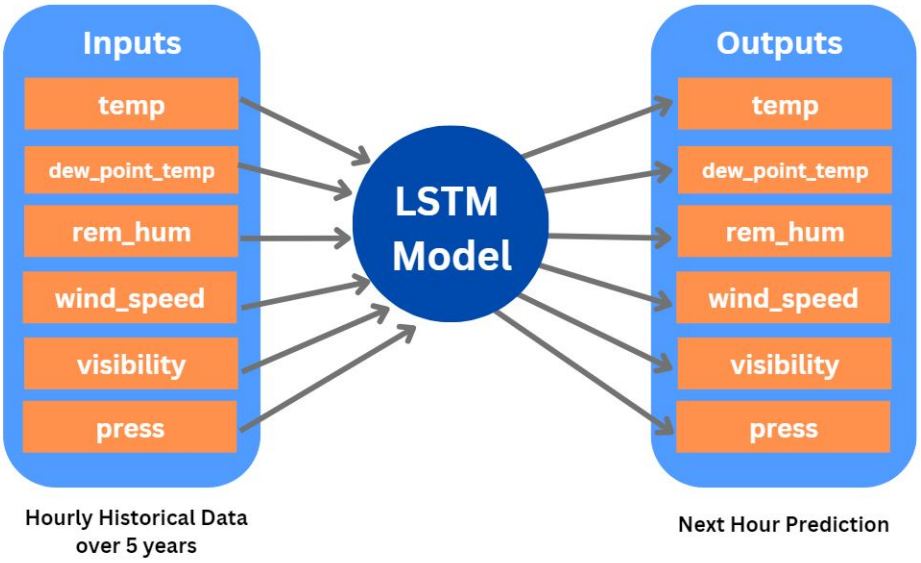

Figure 1: High-Level of Proposed Model

By developing an accurate temperature prediction model, we aim to pave the way for expanding our project to predict multiple weather metrics in the future. This work is not only interesting from a scientific perspective but also has practical applications that can benefit individuals, businesses, and society as a whole.

2 Individual Contributions and Responsibilities

The team has taken different methods and utilized different tools to ensure that the project stays on track to completion. To manage the progress of the high-level tasks of the project, the team is utilizing a Gantt Chart. We have identified key tasks and assigned one Task Owner to oversee the completion of that task as a whole.

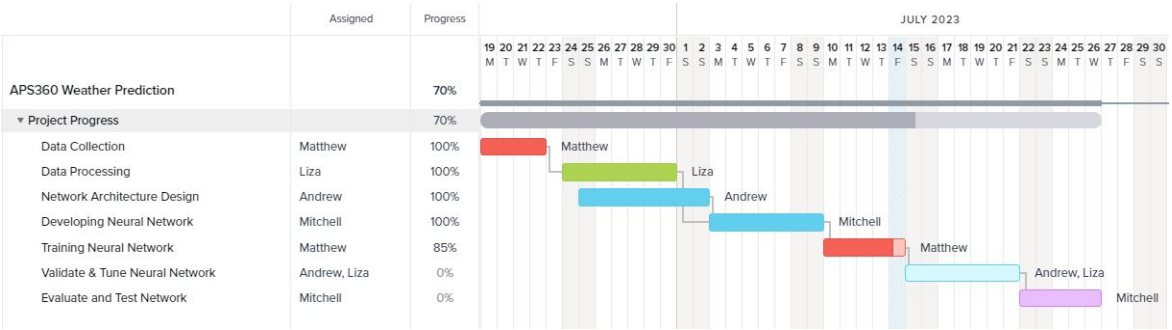

Figure 2: High-Level Tasks Gantt Chart

The Gantt was sufficient to provide a high-level timeline for project completion, however, there are various sub-tasks that needed to be tracked on a granular level. The team opted to use a Master Excel Sheet that tracked all tasks and any potential risks. Figure 3 shows the Tracker we have created.

Tasks	Task Owner	Assigned To	Supported By	Internal Deadline	Status	Risks?	Mitgation Plan
Data Colection	**Matthew**			**23/06/2023**	**Complete**		
Research Datasets		Matthew	Liza	20/06/2023	Complete		
Collect and Load Data to Drive		Liza	Matthew	23/06/2023	Complete	Liza has work conflict	Deadline delayed a day, no negative impact to overall schedule
Data Processing	**Liza**			**30/06/2023**	**Complete**		
Determine Relevant Metric		Mitchell	Liza	27/06/2023	Complete	Mitchell in Spain, working at 50% capactiy	Mitchell most of the tasks prior to deadline & trip
Write Script to Clean Data		Liza	Matthew	30/06/2023	Complete		
Network Architecture Design	**Andrew**			**03/07/2023**	**Complete**		
Research Baseline Models		Mitchell	Andrew	27/06/2023	Complete		
Research Primary Models		Mitchell	Matthew	27/06/2023	Complete		
Outline Baseline Architecture		Andrew	Mitchell	03/07/2023	Complete	Andrew has family emergency	Mitchell & Liza support task completetion
Outline Primary Architecutre		Matthew	Mitchell	03/07/2023	Complete		
Develop Neural Networks	**Mitchell**			**09/07/2023**	**Complete**		
Develop Baseline Network		Andrew	Liza	09/07/2023	Complete		
Develop Primary Network		Matthew	Mitchell	09/07/2023	Complete		
Train Neural Networks	**Matthew**			**14/07/2023**	**In-Progress**		
Train Baseline Model		Andrew	Liza	14/07/2023	Complete		
Train Primary Model		Mitchell	Matthew	14/07/2023	Complete		
Get Training Accuracy of Models		Liza	Matthew	14/07/2023	In-Progress		
Validate & Tune Neural Network	**Liza** **Andrew**			**21/07/2023**	**Not Started**		
Test Different Parameters to Maximize Accuracy		Matthew	Liza	19/07/2023	Not Started		
Select Best Model & Document Justification		Andrew	Matthew	21/07/2023	Not Started		
Evaluate and Test Network	**Andrew**			**26/07/2023**	**Not Started**		
Run Test dataset Through model		Mitchell	Andrew	24/07/2023	Not Started		
Report & Document Testing Accuracy		Liza	Matthew	26/07/2023	Not Started		

Figure 3: Master Tracking Sheet

The sheet defines our internal deadlines, the task owner, who is assigned to that task, and who will be supporting that task.

- **Task Owner**: delegates the work in their sections and monitors the progress to ensure completion of tasks by the deadline (the same person that is identified in the Gantt)
- **Assigned To**: the person that is assigned to that task is responsible for the completion of that task. They are also responsible for raising any risks when it comes to task completion.
- **Supported By**: This person will help support whoever is completing that task. They will provide feedback or additional help where needed, and mitigate risk if that arises.
- **Risks and Migration Plan**: When any risks arose, a migration plan was determined within 24 hours to ensure that the project did not fall off track.

In addition to the team continuously updating the Master Tracker on a day-to-day basis, the team held a virtual meeting every Monday at 6pm EST to share updates, brainstorm ideas, and communicate blockers or risks. During these weekly team meetings, the Gantt chart was also updated accordingly to ensure the project was on track at a high level.

In regards to updated shared code, each team member has a local copy of the Colab Notebook and works in their local copy. If someone is ready to commit their code to our shared notebook, they message the team Discord to ensure no one else is currently updating the main notebook. All other members are notified on the team Discord when a change is made so they can see and work with the latest version.

Overall, the team is satisfied with the work distribution and happy to see that the project is on track to completion. All members provide support to other members if their tasks are completed early. All members have also been good at identifying risks or blockers that arise, and all members have been communicating those risks with the greater team in a timely manner.

3 Notable Contribution

3.1 Data Processing

The section will be broken down into three different steps to explain the data processing completed for this project.

1. Data Collection: The team collected 5 years of hourly temperature data from the Government of Canada's historical data portal [1]. This has been collected from one exact location (Long: -79.37, Lat: 43.86) over the course of years. The data was provided in a .CSV file format, each file containing a month's worth of hourly temperature data. This resulted in 43,848 rows, each representing a single day of unique data. The input file also consisted of 30 unique fields/columns. The schema of the input data is presented in Figure 4.

Input Data Schema	
Field Name	**Datatype**
Longitude (x)	string
Latitude (y)	string
Station Name	string
Climate ID	string
Date/Time (LST)	string
Year	string
Month	string
Day	string
Time (LST)	string
Temp (°C)	string
Temp Flag	string
Dew Point Temp (°C)	string
Dew Point Temp Flag	string
Rel Hum (%)	string
Rel Hum Flag	string

Input Data Schema Con't	
Field Name	**Datatype**
Precip. Amount (mm)	string
Precip. Amount Flag	string
Wind Dir (10s deg)	string
Wind Dir Flag	string
Wind Spd (km/h)	string
Wind Spd Flag	string
Visibility (km)	string
Visibility Flag	string
Stn Press (kPa)	string
Stn Press Flag	string
Hmdx	string
Hmdx Flag	string
Wind Chill	string
Wind Chill Flag	string
Weather	string

KEY FIELDS

Figure 4: Input Data Schema

The team determined that too much irrelevant data was provided, and therefore only certain data needed to be collected. Additionally, the data was all currently of string datatype, which was inaccurate for certain fields.

2. Data Transformation: The team decided the best approach to transform our data was to use the PySpark framework. First, all the rows from the .CSV files, located in our team drive, were loaded into a Spark Dataframe using Spark's read function. Next, the required fields were chosen and the Dataframe was redefined to only select the required columns. Finally, the data was mapped to the team-defined naming and datatype schema, as seen in Figure 5. The fields were renamed with the .withColumnRenamed() function and the fields were then cast to their required datatype. Please view Step 3 in the Colab Notebook to review the Python code for this section. The team discovered that a total of 69 rows, out of the 43,848 rows were completely null. Since this only represented 0.15% of the data, the team decided to remove these rows from the dataset. The new number of rows is 43, 779.

Old Schema		New Schema	
Key Fields	**DataType**	**Field Name**	**DataType**
Temp (°C)	string	temp	double
Dew Point Temp (°C)	string	dew_point_temp	double
Rel Hum (%)	string	rem_hum	int
Wind Spd (km/h)	string	wind_speed	int
Visibility (km)	string	visibility	double
Stn Press (kPa)	string	press	double

Figure 5: Data Schema Mapping

3. Splitting the Dataset and Plan for Testing: The cleaned Dataframe was split into three datasets, with approximately 70% training, 15% validation and 15% testing. The number of entities/rows in training is 30,645, in validation is 6567, and 6567 in testing. These datasets were then converted into PyTorch datasets that can be easily used to train the model - this was done using Pytorch's DataLoader class. The testing dataset will not be touched until we finally evaluate our model.

The primary challenge the team faced was determining which fields we required and how much relevant data we need. The team wanted to ensure that all data was relevant to our goal so that the model does not waste computational power, memory, and time due to irrelevant data. Additionally, ensuring we only have relevant fields can allow the model to focus on the most important features and avoid unnecessary noise. This can lead to better performance and better accuracy.

3.2 Baseline Model

The baseline model for temperature prediction is implemented using the VARMAX (Vector Autoregressive Moving Average with Exogenous Variables) algorithm. VARMAX is a time series model that can capture the dependencies and dynamics between multiple variables. It leverages both past temperature data and dependent variables such as historical weather patterns to make future temperature predictions.

The VARMAX model was selected as the baseline model due to its simplicity, ease of implementation, and ability to handle multivariate time series data. The model only requires that you tune the order and offers a reasonable starting point for comparison with more complex neural network models.

The performance of the baseline VARMAX model will be compared with our primary neural network model to assess the feasibility of achieving the project objectives. The primary neural network model is expected to outperform the baseline model by leveraging its ability to capture complex patterns, sequences, and nonlinear relationships in the data.

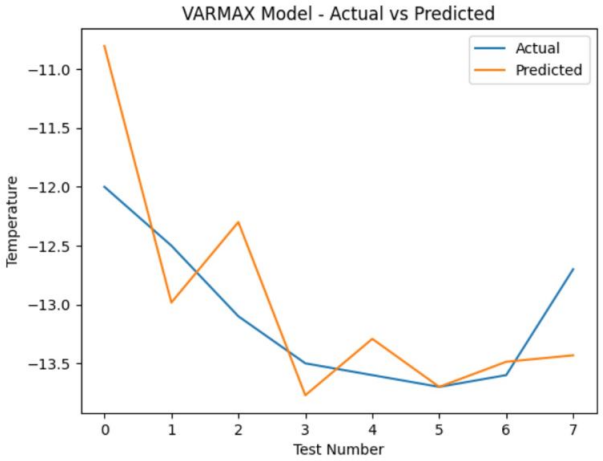

Figure 6: Snippet of baseline model performance on test data

Quantitatively, the model has a mean squared error of 2.18. This shows that it can accurately predict the future temperature within a degree of error, making a good baseline to improve upon with deep learning. Qualitatively, the VARMAX model shows a good understanding of the general trends in temperature variations and provides reliable forecasts for selected time periods. Figure 6 above shows the model being able to predict a dip in temperature. However, it may face challenges in capturing more nuanced and complex patterns in the data, which the primary neural network model aims to address.

One challenge faced with creating the baseline model was the nuances of the weather data. The temperature being a non-linear phenomenon would often cause the VARMAX model to blow up and make very inaccurate predictions (figure 7). These predictions had to be filtered out to preserve the accuracy of the model and will hopefully be fixed with a deep-learning model.

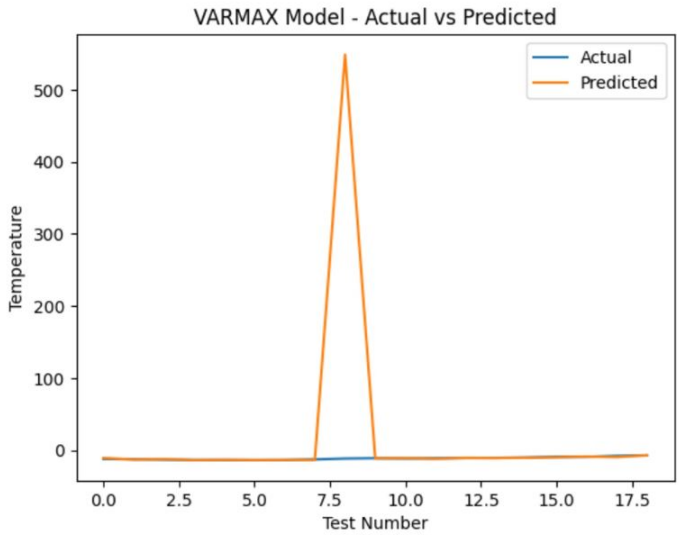

Figure 7: Inaccurate baseline model prediction

3.3 Primary Model

The primary model for this multivariate time series forecasting problem was chosen to be a Long-Short-Term-Memory(LSTM). This choice was made as LSTM networks are most equipped for time series problems and their ability to learn long-term patterns and data interactions over many time steps. Research has provided support to this choice as many articles publish their findings through experiments that the LSTM model has outperformed others.

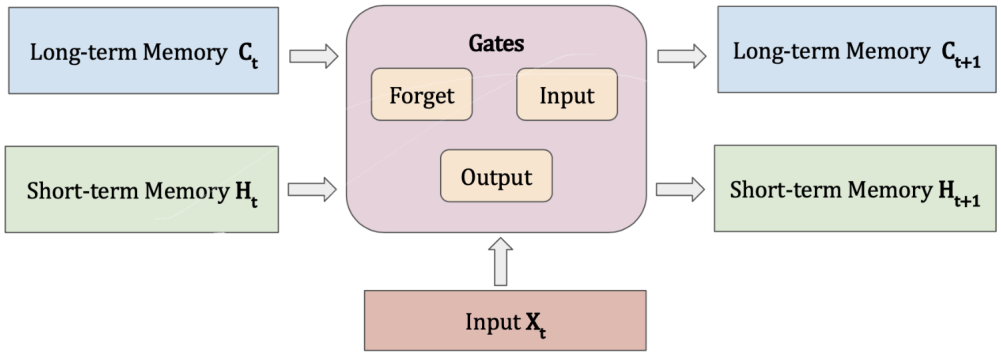

Figure 8: Diagram of Model Architecture [6]

Within our LSTM we have made several specifications, such as choosing the Adam optimizer as it is considered the industry standard[2]. Additionally, we chose to use a Root Mean Sum Error(RMSE) in our cost as it aids to ease of interpretation by being in the unit of probability [3]. Further, we determined the ideal capacity for our model by testing the amount of inputs our model would consume and we settled that the most ideal capacity would be satisfied with 3 layers and 90 nodes in each. This is similar to studies that have developed similar style models, who achieved similar results and used a similar architecture but with different metrics and targets [4]. Now our specific model choice of an LSTM would make it predisposed to over-fit to its data, so we also worked to include a drop out of 10% and a regressive drop out of 20% within each layer to help the network learn deeper relationships within the data [5].

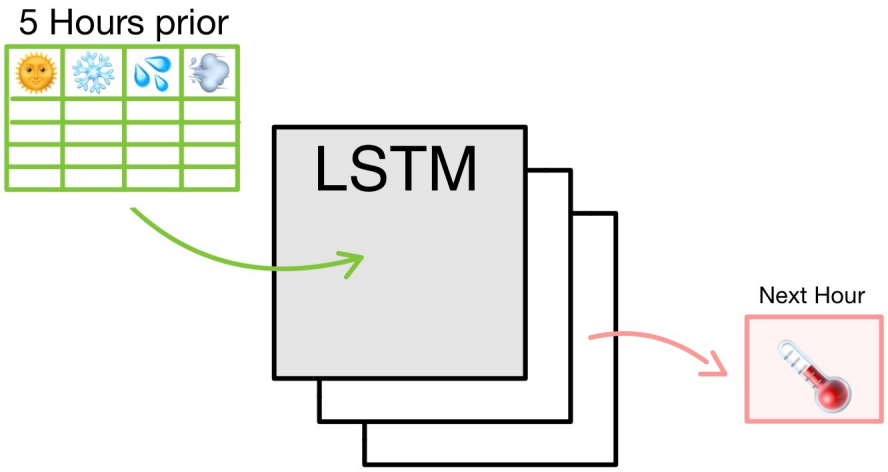

Figure 9: Diagram of Model Flow

Quantitatively, we determined at our best, this network was able to achieve an RMSE 0.45. This leaves room to improve our model by continuing to tune our hyper-parameters.

This was especially rewarding due to the challenges faced while researching how to create a network that was able to ingest these series of multivariate data and then produce one temperature output. One of these challenges was managing all of the data that was available to use. Many times, not having enough data to use is a limitation for a model but in this case, there were multiple sources with different data across all of them including source [7]. We managed this very well by using the data-prepossessing strategies outlined in the corresponding section, one major strategy being combining datasets.

4 Link to Google Colab

`https://colab.research.google.com/drive/1IQ-zKpyy-Yk1flKSa-byiks4nEUF6r5r?usp=sharing`

References

[1] E. and C. C. Canada, "Government of Canada / gouvernement du Canada," Climate, https://climate.weather.gc.ca/historical$_data/search_historic_data_e.html$.

[2] Keith, M. (2022, January 13). Exploring the LSTM neural network model for Time Series. Medium. https://towardsdatascience.com/exploring-the-lstm-neural-network-model-for-time-series-8b7685aa8cf: :text=One%20of%20the%20most%20advanced,more%20parameters%20to%20be%20learned.

[3] Basnet, B. (2016, November 18). LSTM optimizer choice?. Data Science & Deep Learning. https://deepdatascience.wordpress.com/2016/11/18/which-lstm-optimizer-to-use/

[4] Endalie, Demeke, Getamesay Haile, and Wondmagegn Taye. (2022, March 1). Deep learning model for daily rainfall prediction: case study of Jimma, Ethiopia. https://iwaponline.com/ws/article/22/3/3448/85304/Deep-learning-model-for-daily-rainfall-prediction

[5] Brownlee, J. (2020, October 21). Multivariate time series forecasting with lstms in Keras. MachineLearningMastery.com. https://machinelearningmastery.com/multivariate-time-series-forecasting-lstms-keras/

[6] University of Toronto (2023, July 13). Week 8 Recurrent Neural Networks - Part II- LSTM model overview from lecture

[7] Government of Canada Weather Data. https://climate.weather.gc.ca/historical_data/search_historic_data_e.html