Methodology.

Sean Klein, Kevin Russell, Ataish Nehra
Supervised by: Dr. Fatema Nafa
Northeastern Data Science Capstone

1 Approach.

Our approach for the project was to aggregate multiple datasets to make a single dataset in order to reflect climate change and its potential relationship with the number of natural disasters. Furthermore, the data had to be restructured in order to reflect the time sequential nature of the data. To predict Total Disasters with climate change, we chose to utilize LSTM and Attention LSTM cells to predict Total Disasters as a sequence to sequence problem. We also chose to use the Forward-Forward algorithm to predict Total Disasters as a continuous value problem. Each algorithm's approach required special treatment to the data along with architectural design. We then chose a few different evaluation metrics to compare and contrast the models to find the best model represented to predict the data.

1.1 Visual Architecture.

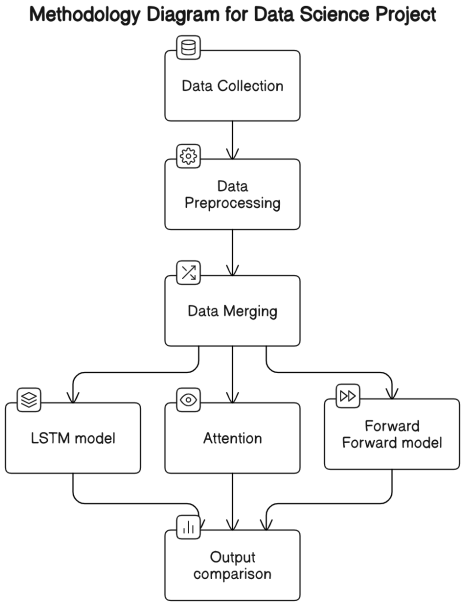

Figure 1: *A Flowchart showing a step-by-step architecture of our methodology to build the ML model for predicting natural disasters.*

Long Short-Term Memory :

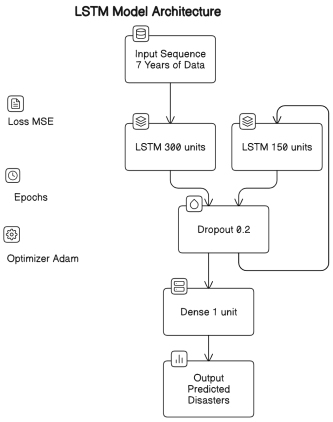

Figure 2: *Architectural flowchart of the final LSTM model, depicting two LSTM layers with dropout, processing a 7-year input sequence. The model employs the Adam optimizer and MSE loss over 150 epochs, culminating in a single-unit output for disaster predictions.*

Attention :

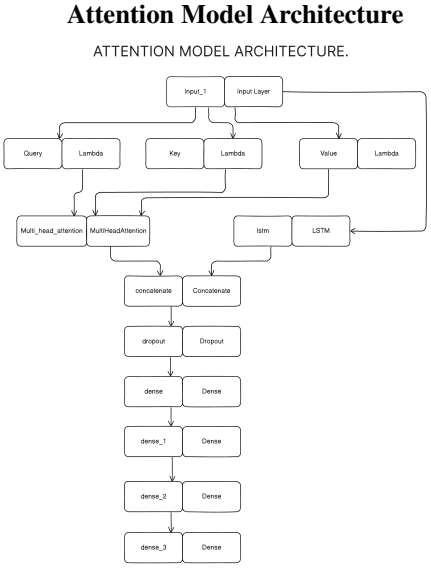

Figure 3: *The attention neural network uses the lstm model for the base, has a multihead attention layer, a dropout layer, 3 ReLu dense layers, and then a Softmax dense layer to return a single output.*

Forward-Forward :

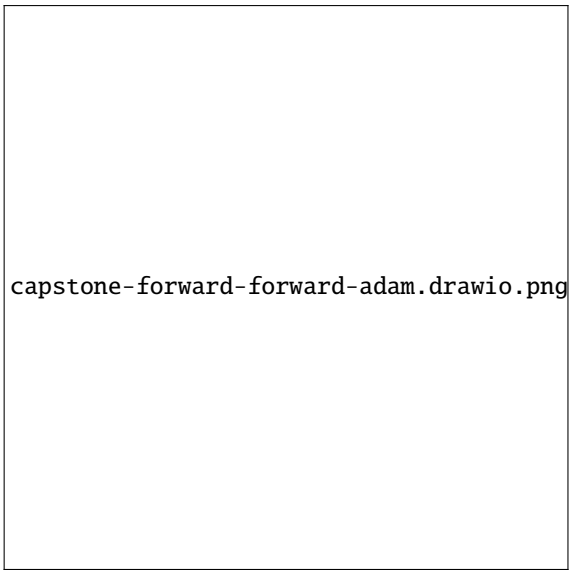

Figure 4: *This visual showcases the Training Architecture for the Forward-Forward Algorithm where (red) represents the positive goodness and (blue) represents negative goodness that is either trained inside the layer itself or passed to the next fully connected layer*

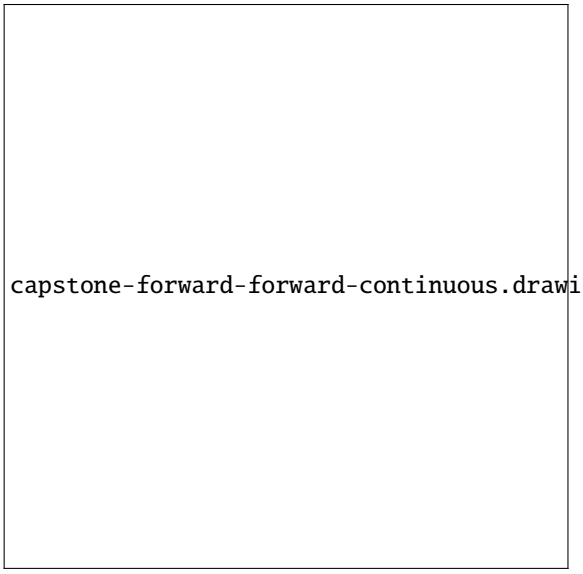

Figure 5: *The top architecture is the prediction of the Forward-Forward algorithm that yields positive goodness from each fully connected layer. Each positive goodness' are concatenated to make a matrix that is passed into a singular linear layer for training.*

1.2 Purpose:

The purpose of this project is to accurately predict natural disasters, and to show that climate change does have an impact on the environment. The project also compares and contrast top-of-the-line neural networks to see how they fair compared to an experimental Forward-Forward model. Additionally, we hoped to figure out how to improve and modify earlier iterations of the Forward-Forward model to extend beyond image classification. Our hope was to either build an architecture for continuous value prediction or time-series prediction.

2 Problem Statement.

2.1 Define the Problem.

The project aims to address the challenge of predicting natural disasters, with a focus on storms and other catastrophic events, as a time-series and regression problem using machine learning techniques, specifically neural networks. The primary problem being tackled is the unpredictability and often sudden occurrence of natural disasters, which can lead to significant human and environmental losses. By leveraging neural network architectures such as TCNN (Temporal Convolutional Neural Network), RNN (Recurrent Neural Network), LSTNN (Long Short-Term Neural Network), and Attention NN, the project aims to develop models capable of analyzing climate change data over time.

2.2 Significance.

In industry, there is a growing demand for practical applications of AI and ML in addressing real-world challenges. The project's goal of predicting natural disasters has direct implications for disaster management and emergency response industries. Governments, humanitarian organizations, and other stakeholders in disaster-prone areas are increasingly recognizing the value of data-driven approaches to improve early warning systems and preparedness strategies. The project, by combining AI, ML, and climate science, aligns with the industry's need for actionable insights to mitigate the impact of natural disasters.

3 Data Collection and Preparation.

3.1 Data Sources:

The data was sourced from the International Monetary Fund's six public datasets on climate change and natural disasters [1]. The IMF collaborated with multiple different reputable organizations in order to collect and aggregate the data into an easy to access web page. The Surface Temperature dataset was sourced from the Food and Agriculture Organization Corporate Statistical (FOASTAT)[2] Database and collected by the National Aeronautics and Space Administration's[3] Goddard Institute; Carbon Dioxide Concentrations were provided by the National Oceanic and Atmospheric Association(NOAA)[4] Global Monitoring Laboratory; Sea Level Changes were from the NOAA's Laboratory for Satellite Altimetry; Forest and Carbon data was collected by the IMF in collaboration with FOASTAT; Land Cover is from FOA land cover with additional work done by the IMF; and the Natural Disasters data is sourced by EM-DAT [5] which is an international database specifically for disasters and their epidemiology.

3.2 Data Description:

The dataset was constructed from the following 5 datasets: Surface Temperature, Forest and Carbon, Land Cover Accounts, Sea Level Change and Frequency of Climate Disasters. Originally there was a sixth dataset pertaining to Co Concentrations, however it only recorded the CO concentrations for the entire world, so it did not align with the structure of the dataset that we were constructing. The dataset was created where each row represents the climate records and number of disasters over a year for a specific country. This

means that when the dataset is grouped by country then ordered by year, it creates a dataset of multiple small time series of twenty eight years for each country.

3.3 Preprocessing Steps:

The combined dataset has 899 samples and 30 features where 6 of these features are components of the target variable, Total Disasters, and a 7th feature is year. Mean was used to replace missing values except for Frequency of Natural Disasters where we assumed missing values are zeros. Duplicate and blank rows were dropped as well as same value columns. Luckily no blank rows were found nor was same value columns. IQR was used to detect outliers. All features are in numeric formats so no preprocessing of text or non-numeric values was needed. Data was normalized with MinMax Normalization for time series and Mean Normalization for continuous value prediction. Time series data was grouped into sequences by country, year to train the LSTM and Attention models.

4 Selection of Machine Learning Models

4.1 Model Consideration

Our team considered LSTM, RNN, Attention networks[6], and Linear Regression models because each network specializes in time series or continuous value predictions. The Forward-Forward algorithm [7] became interesting because it is an unorthodox method that has never been tested with continuous value prediction or time-series prediction. Since our dataset is climate change and our target is Total Disasters, we found each of these architectures appropriate to predict Total Disasters.

4.2 Final Model Selection

We decided to select the LSTM architecture [8] because we wanted to see the performance of short-term long-term memory in a sequence-to-sequence prediction. The attention network with LSTM cell was chosen because multi-head attention networks has proven to be very successful towards sequence to sequence prediction. Additionally, neural networks has always been a topic discussed at the end of several courses that Sean has taken, and Sean has been fascinated with building large architectures. The Forward-Forward algorithm [7] has never been tested against continuous value predictions, but the concept has shown promise towards supervised vision classification. The unique concept of the algorithm is what drove towards developing a method to predict continuous values. Unfortunately, there has been no evidence of continuous value prediction performed with the Forward-Forward algorithm.

5 Model Development and Training.

5.1 Architecture and Configuration.

Long Short-Term Memory: The model was built using TensorFlow's Keras API, featuring a stacked LSTM configuration [9] where the initial layer consisted of 300 units followed by a second LSTM layer with 150 units, both utilizing the 'tanh' activation function. To combat overfitting, a dropout of 0.2 was introduced after each LSTM layer. The model's configuration was such that it accepted input sequences shaped to include seven years of historical data, allowing the model to analyze and learn from an extended temporal context. For the compilation, the Adam optimizer was chosen for its efficiency and adaptive learning rate capabilities, and the model was trained using the Mean Squared Error (MSE) as the loss function, a standard choice for regression problems. The model was trained for 150 epochs, with a batch size of 32, to ensure that it had adequately learned from the training data without overfitting.

Attention: The Attention algorithm[6] was constructed using Tensorflow Keras, using an LSTM model for the base, the inputs are copied into three different variables (query, key, value) before then being pushed through a multihead-attention layer, those two layers are then concatenated before going through a dropout layer and then 3 different dense layers with an ReLu activation and then a dense layer with a softmax activation. The optimizer is Adam, the loss was MSE, and the number of epochs was set as 125.

Forward-Forward: The Forward-Forward Algorithm [7] uses three fully connected Forward-Forward layers, and a linear layer to predict continuous values in it's architecture. Each Forward-Forward layer was trained with Adam optimizer using a 0.06 learning rate, 150 epoch iterations, and 0.07, 4.0, 0.25 Forward-Forward thresholds. There were 256, 128, 128 neurons for each Forward-Forward layer respectively with the linear layer yielding one neuron. The linear layer was trained with 0.06 learning rate and 288 epoch iterations. Each Forward-Forward layer had a ReLU activation function applied after each layer to garnish nonlinearity to its activations.

5.2 Training Process.

Long-Short Term Memory: For the training process of my final LSTM model [10], I divided the dataset into training and testing sets using sklearn's train_test_split function, allocating 80% for training and 20% for testing to ensure a substantial amount of data was available for the model to learn from. To structure the data appropriately for time series prediction, I employed a sequence creation technique that formed input sequences from seven years of data, reflecting the model's architecture that was designed to process longer temporal sequences for improved forecast accuracy.

Normalization played a crucial role in preparing the data, and to achieve this, I standardized the features, not using MinMaxScaler as commonly done, but instead ensuring that the data fed into the LSTM was formatted correctly to preserve temporal dependencies without explicit scaling. I found this to be sufficient for the model to learn effectively from the data's natural distribution.

The batch size was set to 32 during training, a number that provided a good balance between model update frequency and convergence speed. By training over 150 epochs, the model had ample opportunity to learn from the data while monitoring the validation loss to prevent overfitting. The model was compiled with the Adam optimizer, known for its efficiency in handling sparse gradients on noisy problems, and Mean Squared Error (MSE) as the loss function to optimize, directly aiming to reduce the average magnitude

of errors in the predictions.

Attention: For the attention model[6], the train/test split was 70% and 30% respectively using sklearn library's train test split. The data was time structured using the method that Ataish had implemented but setting the number of steps to 1 so that the model could correctly interpret the data, and the features were normalized using sklearn library's MinMaxScaler by setting the min value to 0 and the max value to 1. The batch size for the attention neural network was set to 30.

Forward-Forward: Before training the Forward-Forward algorithm [7], the dataset was split into 80% train and 20% test datasets using torch's random split, which the dataset was loaded into a torch DataLoader using shuffle for the train dataloader. The data itself was transformed with z-score normalization and converted into a pytorch tensor. L2 layer normalization was applied between layers to stabilize training of the network.

5.3 Hyperparameter Tuning.

Long-Short Term Memory: Throughout the project, hyperparameter tuning was essential [11]. Initially, I set a high epoch count and optimized batch size for efficiency. As complexity increased in the second model, I introduced dropout and L2 regularization. Despite this, I needed to fine-tune further. For the final model, I extended input sequences to seven years and adjusted epochs to 150, which, along with a two-layer LSTM structure and an optimized learning rate via the Adam optimizer, significantly improved performance, yielding lower error metrics and a more accurate prediction model.

Attention: For the attention neural network[6], there were several different hyperparameters that needed to be tuned. Throughout each iteration of the model, I experimented with various different activation functions for the dense layers; the best results were given when using ReLu for most of them and then having the final dense layer have a softmax activation. There was a similar process with determining the optimizer and the loss functions for the architecture; where the finished model has Adam as the optimizer and MSE as the loss function. The number of epochs was the last hyperparameter that was tested with. I set the number of epochs to 750 and then plotted a graph where the X axis was number of epochs and the Y axis being the MSE values, by finding the elbow of the graph, I determined that the ideal number of epochs was around 125.

Forward-Forward: Tuning the Forward-Forward algorithm [7] became challenging since there are many combinations $O(\infty)$ to tune the Forward-Forward threshold. Additionally, each layer has its own learning rate, weight decay, epoch iterations, and various other hyperparameters that makes tuning the network unorthodox to the traditional neural network. Therefore, the learning rate was tuned first by evaluating loss, MSE, and MAE to find the best learning rate of 0.06 for each layer. The Epoch hyperparameter was tuned for each Forward-Forward layer, thereafter, where 150 epochs yielded the best evaluation for loss, MSE, and MAE. The Forward-Forward thresholds was tuned last for the Forward-Forward layers by evaluating loss, MSE, and MAE for each layer. Lastly, learning rate and epoch hyperparameters were tuned for the linear layer to find the best tuned fit.

References

[1] I. M. Fund. "Climate change data." (), [Online]. Available: https://climatedata.imf.org/pages/climatechange-data. accessed: 03.07.2024.

[2] U. Nations. "Foa: Food and agriculture organization of the united nations." (), [Online]. Available: https://www.fao.org/faostat/en/#home. accessed on: 03/14/2023.

[3] U. S. F. Government. "Nasa: National aeronautics and space administration." (), [Online]. Available: https://www.nasa.gov/. accessed on: 03/14/2024.

[4] U. D. of Commerse. "Noaa: National oceanic and atmospheric administration." (), [Online]. Available: https://www.noaa.gov/. accessed on: 03/14/2024.

[5] CRED and T. W. H. Organization. "Em-dat: The international disaster database." (), [Online]. Available: https://www.emdat.be/. accessed on: 03/12/2024.

[6] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, *Attention is all you need*, 2017. arXiv: 1706.03762 [cs.CL].

[7] G. Hinton, *The forward-forward algorithm: Some preliminary investigations*, 2022. arXiv: 2212.13345 [cs.LG].

[8] R. C. Staudemeyer and E. R. Morris, "Understanding lstm–a tutorial into long short-term memory recurrent neural networks," *arXiv preprint arXiv:1909.09586*, 2019.

[9] Y. Yu, X. Si, C. Hu, and J. Zhang, "A review of recurrent neural networks: Lstm cells and network architectures," *Neural computation*, vol. 31, no. 7, pp. 1235–1270, 2019.

[10] Y. Liu, D. Li, S. Wan, *et al.*, "A long short-term memory-based model for greenhouse climate prediction," *International Journal of Intelligent Systems*, vol. 37, no. 1, pp. 135–151, 2022.

[11] C. Natel de Moura, J. Seibert, and D. H. M. Detzel, "Evaluating the long short-term memory (lstm) network for discharge prediction under changing climate conditions," *Hydrology Research*, vol. 53, no. 5, pp. 657–667, 2022.