2018-01-01

Forecasting Space Weather Using Deep Learning Techniques

Sumi None Dey
University of Texas at El Paso, sdey2@miners.utep.edu

FORECASTING SPACE WEATHER USING DEEP LEARNING TECHNIQUES

SUMI DEY

Master's Program in Computational Science

APPROVED:

Olac Fuentes, Chair, Ph.D.

Amy Wagler, Ph.D.

Laura Boucheron, Ph.D.

Charles Ambler, Ph.D.
Dean of the Graduate School

Dedication

to my

Family

with love

FORECASTING SPACE WEATHER USING DEEP LEARNING TECHNIQUES

by

SUMI DEY

THESIS

Presented to the Faculty of the Graduate School of

The University of Texas at El Paso

in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE

Master's Program in Computational Science

THE UNIVERSITY OF TEXAS AT EL PASO

December 2018

Acknowledgements

I am thankful to my Almighty God for his blessings shown me throughout my education.

I would like to express my sincere gratitude to my research advisor Dr. Olac Fuentes, faculty of the Department of Computer Science at The University of Texas at El Paso, for his advice, enduring patience and inspiration to complete my thesis successfully.

I would also like to thank the other committee members Dr. Amy Wagler, faculty of the Department of Mathematical Science at The University of Texas at El Paso and Dr. Laura Boucheron, faculty of the Klipsch School of Electrical and Computer Engineering at The New Mexico State University for their effort and time.

I would like to thank the faculty and the staffs of Computational Science Program of the University of Texas at El Paso for their hard work and dedication to complete my thesis.

My warmest gratitude goes to my parents, my sister for their love and constant support. They had always confidence in me even when I didn't have for myself.

My sincere thanks go to my all friends and loved ones for their contribution to my work.

Abstract

Solar activity gives rise to various kinds of space weather among which solar flares have serious detrimental effects on both near-Earth space and our upper atmosphere that will have consequent influence in our lives. For example, solar flares can damage satellite infrastructure, hinder power grids, disrupt Global Positioning Systems (GPS) and disrupt long-distance communication. Airplane pilots, cabin crew and astronauts can be affected by the harmful radiation released from the Sun. As a result, there is a need of a methodology to forecast space weather accurately. In this work, we have developed a deep learning architecture to do the short-range prediction of the X-ray flux and particle flux emitted by the Sun in continuous time by making use of time series data. We considered two baselines to evaluate our approach; in one baseline, the latest available flux is used as the prediction; in the other baseline, the average of the previous fluxes with a particular time window is used as the prediction. Our developed model outperforms both of the baselines. Our proposed work consists of extending the prediction methods by exploiting information extracted from sequences of solar images.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

The dynamic conditions and events on the Sun in near-Earth space and in our upper atmosphere have an important role in both human lives and technology. This phenomenon is described by space weather. Solar activities give rise to various kind of space weather. These activities are solar flares, coronal mass ejections, high-speed solar wind, and solar energetic particles. The solar magnetic field is the ultimate source of all of these activities.

1.1 Solar Flares

1.1.1 What Are Solar Flares

A sudden release of distorted magnetic fields that produces a huge amount of energy and drives that energy into space creates a sudden flash of light known as a solar flare. Solar flares are considered as the largest explosive events of our solar system. Flares can last from minutes to hours. This electromagnetic emission travels at the speed of light. Sometimes the released energy accelerates very high energy particles such as protons and electrons. It takes about tens of minutes for these high energy particles to reach to the Earth [1].

1.1.2 Solar Flares Events

Solar flares itself a strong event and become stronger when they are associated with some other solar activity, for example, coronal mass ejections or solar wind. This kind of events started to be recorded around 150 years ago. Some real world examples of solar flare events are following:

1. One of the most powerful solar flare event happened on September 1859. This event is known as the "Carrington event". The event was first reported by one of England's foremost solar astronomers named Richard Carrington. Because of that event, high energy particles entered into Earth's atmosphere and overpowered the Earth's protective magnetic field which resulted in a huge destruction on the ground. Telegraph system disrupted worldwide. Telegraph offices were ignited by the spark. Colorful aurora were seen at near tropical latitudes over Cuba, the Bahamas, Jamaica, El Salvador, and Hawaii. Even when batteries were disconnected, the messages were transmitted because of electric currents in the wires induced by Aurora [12].

2. One of the historical powerful series of solar storms happened in August 1972. The solar storms were associated with solar flare, geomagnetic storm and high energy particles. This storm set off sea mines in Vietnam. This event disrupted satellite, hindered communication-grid and electric service. This event occurred between the Apollo 16 and Apollo 17 lunar missions. If this would happen during one of the mission, the particles could have hit astronauts outside of Earth's protective magnetic field and the result could have been life-threatening.

3. From mid-October to early November 2003, one of the largest solar flare events occurred. This is actually the largest solar flare event that has been recorded by Geostationary Operational Environmental Satellite (GOES) system. The Sun's magnetic field lines were stretched because of the flare and then all of a sudden the magnetic field lines stretched beyond their limit. As a result, there was a gigantic explosion on the Sun's surface called coronal mass ejections(CME). CMEs are capable of exploding billions of tons of electrified gas and subatomic particles into space at a speed of five million miles per hour. This event disrupted communication, satellite-based system. A 90-minute blackout in Sweden was caused because of this event. Aircraft controllers changed their route to avoid high altitudes near the polar regions.

1.1.3 Goal

The goal of the project is to develop algorithms to accurately forecast solar X-ray flux and particle flux in continuous time using a combination of previous flux measurements and sequences of solar images. For this purpose, we will exploit Geostationary Operational Environmental Satellite (GOES) X-ray flux 5-minute time series data, GOES particle flux 5-minute time series data and Solar Dynamics Observatory (SDO) images.

1.1.4 Significance of The Goal

Space weather events, in particular solar flares, can damage satellite infrastructure, hinder power grids, disrupt Global Positioning Systems (GPS) and damage long-distance communication. A recent study [12] estimated that a single severe space weather event could cause an economic loss of between 0.5 trillion and 2.7 trillion USD. Airplane pilots, cabin crew and astronauts need to stay in space. They can be affected by the harmful radiation released from the particles. Forecasting X-ray flux and particle flux in continuous time is a necessary step towards fully automated prediction of space weather events.

1.2 Thesis Statement

Forecasting of X-ray and particle fluxes in continuous time can be improved using a deep learning architecture trained on a combination of time series data and sequences of image data.

1.3 Research Questions

In this research, we are looking for the answers of the following questions:

1. How to forecast the X-ray and particle fluxes in continuous time from time series data (Figure 1.1)?

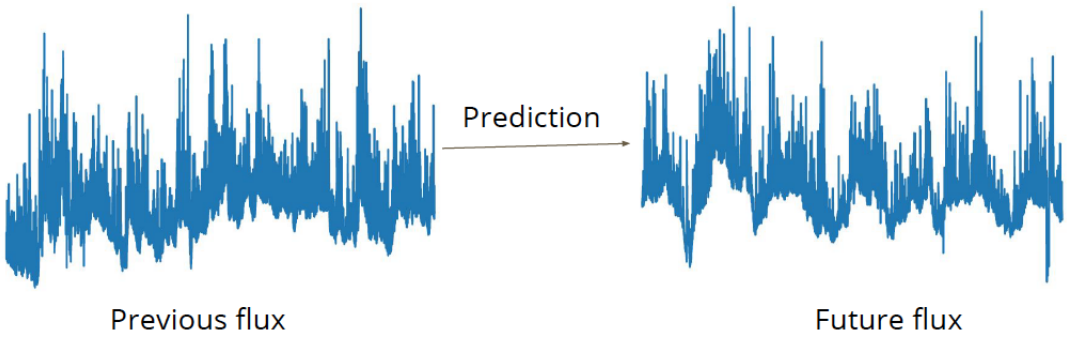


Figure 1.1: Research question 1

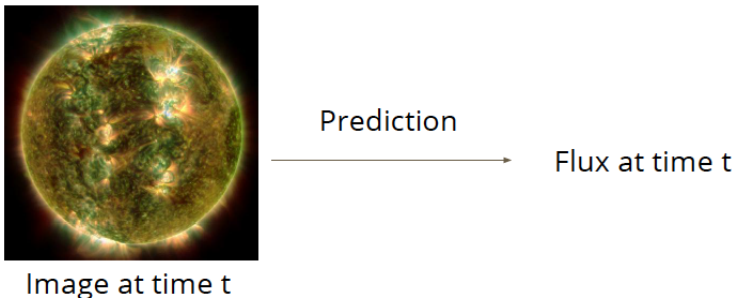


Figure 1.2: Research question 2

2. How can we predict the X-ray and particle fluxes in continuous time from the solar images (Figure 1.2)?

3. Can prediction be improved by extracting temporal information from sequences of solar images (Figure 1.3)?

4. How can we combine sequences of solar images and history of fluxes to improve predictions (Figure 1.4)?

1.4 Expected Contributions

We are expecting to achieve the following:

1. Development of a deep neural network architecture that reduces the error for the

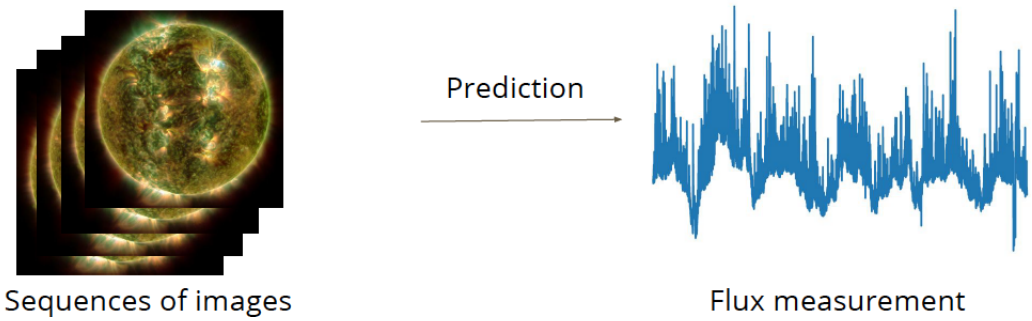


Figure 1.3: Research question 3

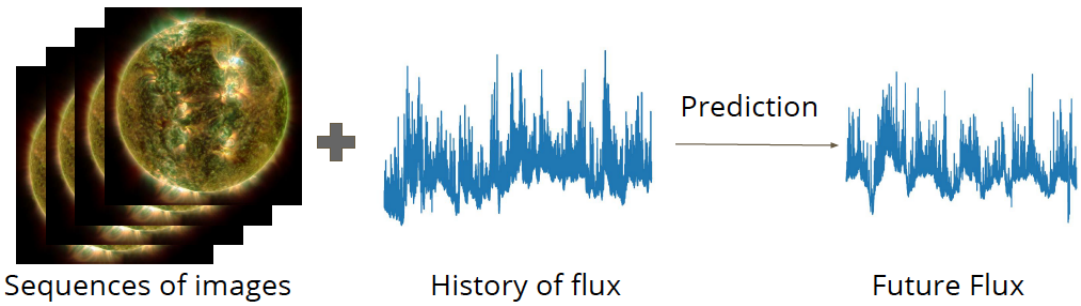


Figure 1.4: Research question 4

prediction of X-ray and particle fluxes in continuous time from time series data.

2. Development of a deep neural network to predict the X-ray and particle fluxes in continuous time from solar images.

3. Development of a deep learning model for long term prediction of X-ray and particle fluxes using sequences of solar images.

4. Development of a deep learning architecture to forecast X-ray and particle fluxes from a combination of sequences of solar images and time series data.

1.5 Outline

The writing is organized as follows: Chapter 2 describes the background of this proposal. Some related works to this research are presented in chapter 3. Chapter 4 describes the objectives and methodologies that are implemented in this work to achieve the expectations of this work. Preliminary results are discussed in chapter 5. Finally, my work plan, the timeline is shown in chapter 6.

Chapter 2

Background

A family of algorithms is called Neural Networks, which are the basic building blocks of the deep learning algorithm. Generally, a neural network consists of three layers - input layer, hidden layers and output layer. Input layer receives input of various formats, for example, number, text, audio, image pixels and so on. Hidden layer does data manipulation, mathematical calculation, feature extraction and so on based on the information it receives from input layer. Output layer receives information from hidden layer and based on that it generates the desired output. Each node in the network carries some weights to the next layer and there is an assigned bias to each layer. Result of the weighted sum of inputs with an addition of bias is passed to activation functions and based on that output layer generates the final output. Then a cost function calculates the error. To minimize the error, weights need to adjust. To do this, we backpropagate the error through the network.

There are various types of neural networks. Among them Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) are the most popular one.

2.1 Convolutional Neural Network (CNN)

2.1.1 Convolution Operation

The convolution operation is usually denoted by asterisk. The convolution of f and g is defined as follows

$$s(t) = (f * g)(t) = \int_{-\infty}^{\infty} f(a)g(t-a)da \tag{2.1}$$

and described as weighted average of $f(a)$ at the moment t where the weighting is given by $g(-a)$ with a shift of amount t. The weighting function emphasizes different parts of the input function based on the value of t.

The function f is known as the input, the function g is called the kernel and the output is referred to as the feature map in convolutional network terminology.

t will be discretized while working with data and then t can only take on integer values. In that case the discrete convolution can be defined as:

$$s(t) = (f * g)(t) = \sum_{a=-\infty}^{\infty} f(a)g(t-a) \tag{2.2}$$

When the input is a multidimensional array, then the kernel is also a multidimensional array. These multidimensional arrays are referred to tensors. Both input and kernel elements are stored separately. It is assumed that both input and kernel functions are zero everywhere but the finite set of points for which the values are stored. Consequently, the infinite summation can be expressed as a summation over a finite number of array elements. In this case convolution is defined as

$$S(i,j) = (F * G)(i,j) = \sum_{m} \sum_{n} F(m,n)G(i-m, j-n) \tag{2.3}$$

One of the nice property of convolution is commutative property. So the previous equation is equivalent to (2.4).

$$S(i,j) = (G * F)(i,j) = \sum_{m} \sum_{n} F(i-m, j-n)G(m,n) \tag{2.4}$$

The following figure shows how 2-D convolution works.

2.1.2 Pooling

After performing several convolutions in parallel to produce a set of linear activations, each linear activation is run through a nonlinear activation function, for example, the

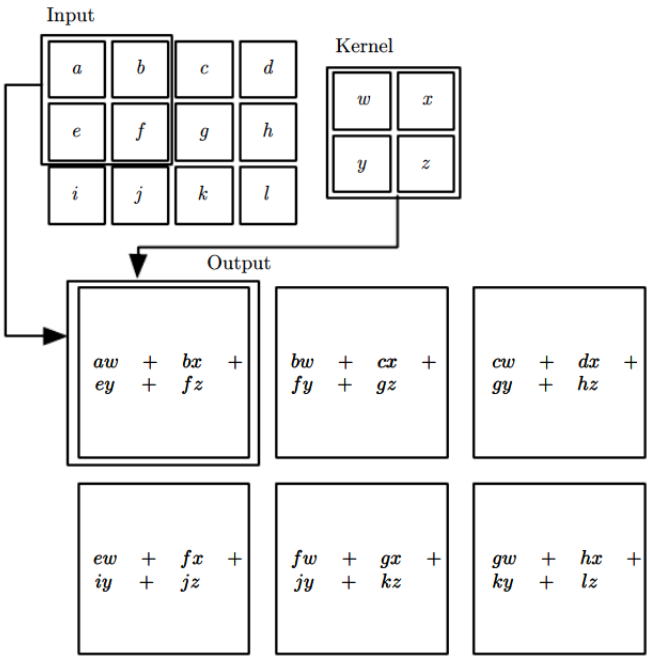


Figure 2.1: An example of 2-D convolution

rectified linear unit. This is called detector stage. Then pooling function is used to replace the output of the network at a certain location with a summary statistic of the nearby outputs. To implement pooling there are several non-linear functions - max pooling, average pooling, L2 norm, weighted average. Max pooling uses the maximum value of a rectangular neighborhoood, average pooling takes the average of a rectangular neighborhoood.

It is assumed that the presence of some feature is more important than its exact location relative to other features. In this case translation invariance property is very useful. And

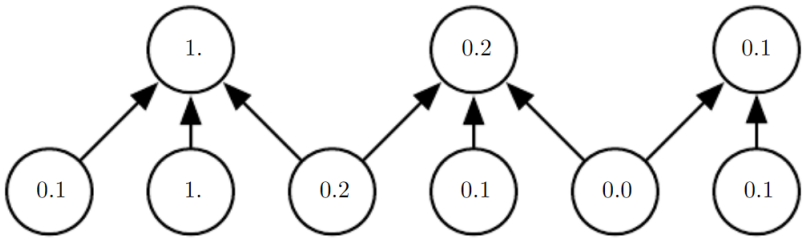


Figure 2.2: An example of max pooling

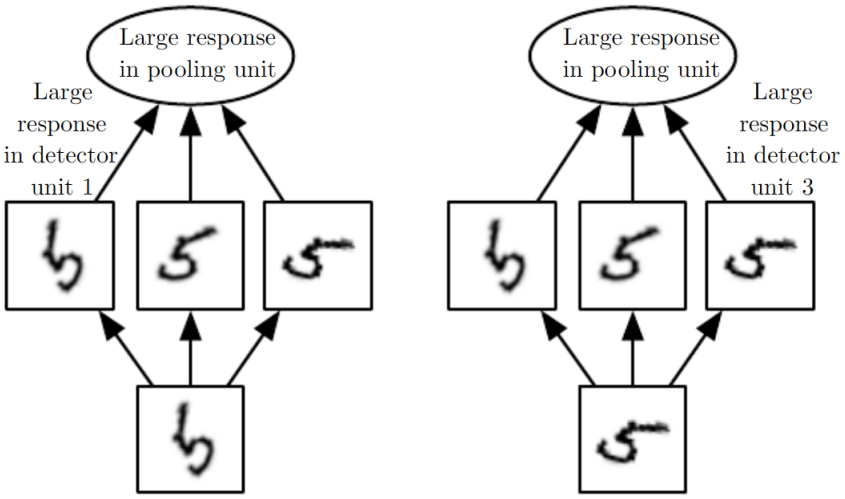


Figure 2.3: An example of learned invariance

pooling helps to make the representation approximately translation invariant.

Pooling over spatial regions produces invariance to translation, but pooling over the outputs helps the features to learn which transformations to become invariant to.

Usually pooling units are fewer than detector units because pooling uses the summary statistics for pooling regions spaced k pixels apart rather than 1 pixel apart. That means the next layer has approximately k times fewer inputs to process which is computationally efficient as well as statistically efficient. This also reduce the memory requirements to store the bottlenecks.

2.1.3 Activation

Sigmoid

The mathematical form of sigmoid function is

$$\sigma(x) = \frac{1}{1 + \exp(-x)} \tag{2.5}$$

This activation function is very popular among deep learning researchers and engineers because of its advantages -

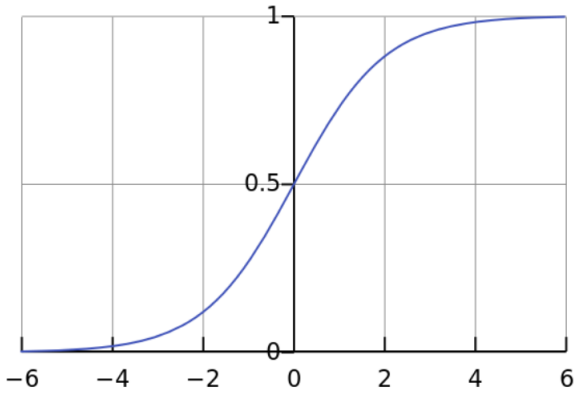


Figure 2.4: Sigmoid

- By nature sigmoid function is nonlinear, so the linear combination of this function is also nonlinear. This function has smooth gradient also.
- This function takes any real value as input and output is always in the range $(0, 1)$, while the range for linear function is $(-\infty, \infty)$.
- From the figure 2.4 of the sigmoid function, it is clear that the function is very steep in the range $(-2, 2)$. That means in this range the function will change significantly for a small change in x. Consequently, there will be gradient which is greater than zero. It is also clear that the output of this function is between 0 and 1, which is good for a classifiers as this function is making a clear distinctions on prediction.

The main drawback of this activation function is outside of the range $(-2, 2)$, the function changes very little for any amount of change in x. That means the gradient will be close to zero. This problem is called "vanishing gradients" problem. This is an undesirable property. In this case the gradient becomes zero and there will be no flow of signal to its weights and recursively to its data.

Hyperbolic tangent

Mathematically hyperbolic tangent is expressed as

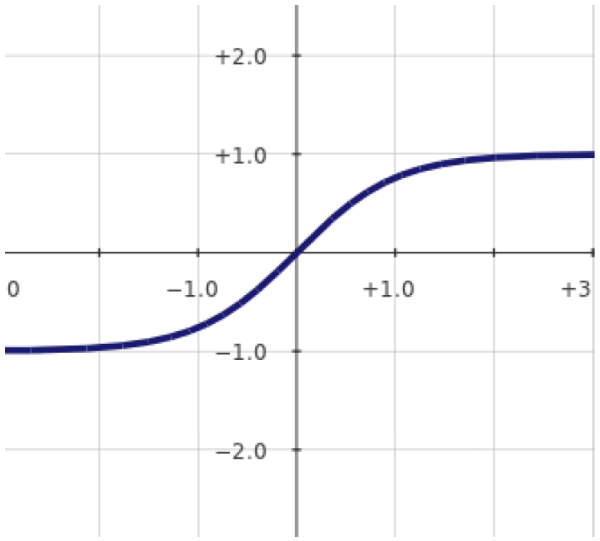


Figure 2.5: Hyperbolic tangent

$$tanh(x) = \frac{2}{1 + \exp(-2x)} - 1 \tag{2.6}$$

which can be written as

$$tanh(x) = 2\sigma(2x) - 1 \tag{2.7}$$

That means *tanh* is scaled sigmoid function. The characteristics of this activation function is similar to sigmoid function. This function also takes any real value and the output range is $(-1, 1)$. The gradient of this function is stronger than the sigmoid function. In addition to this, *tanh* is zero centered unlike sigmoid function.

Rectified Linear Units (ReLU)

ReLU is the most popular and widely used activation function. This function is defined as the positive part of its argument, $R(x) = max(0, x)$. That means this function takes real-valued number and the output is x if x is greater than or equal to zero and zero if x is less than zero.

Sigmoid or *tanh* activation is dense meaning that to describe the output of a network almost all the activations are processed, which is costly. But ReLU makes the network

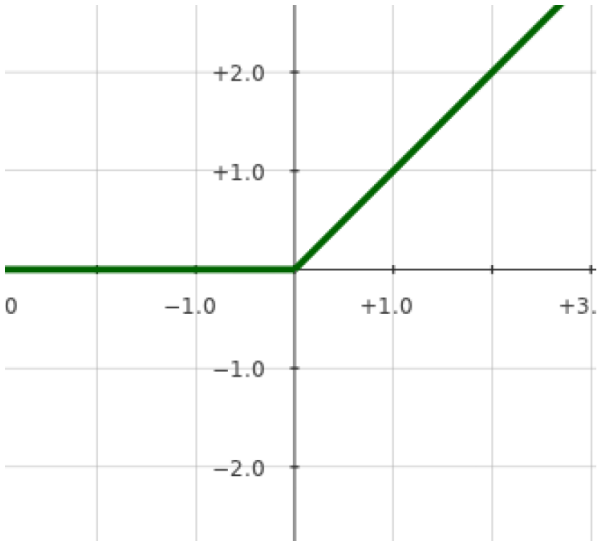


Figure 2.6: Rectified Linear Units

lighter and efficient by not activating some neurons in the network.

The characteristic of the ReLU is that for negative values of x, this function is zero, which means the gradient zero. This will result in not adjusting the weights of the neurons during backpropagation. So these neurons will never activate again. This gives rise to "dying ReLU" problem.

2.1.4 Loss Function

Loss function measures the performance of the predicted result. Our objective is to minimize the loss function. Mean Squared Error (MSE) is used as the loss function to measure the distance between the predicted result and true result. MSE is given by,

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2, \tag{2.8}$$

where $y_i's$ are the true values and $\hat{y}_i's$ are the predicted values.

Another loss function is Mean Absolute Error (MAE), which is given by,

$$MAE = \mid y_i - \hat{y}_i \mid, \tag{2.9}$$

where $y_i's$ and $\hat{y}_i's$ are the true and predicted values respectively.

2.2 Recurrent Neural Network

Recurrent Neural Network is a special kind of neural network to process a sequence of values $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, ..., \mathbf{x}^{(n)}$. This has the ability to process sequences of variable length.
The key idea of recurrent neural networks is to share the parameters across different parts of the model. This sharing plays an important role when a specific information appears at multiple positions of a sequence.
The output of a recurrent network is a sequence, where each member of the output is a function of previous members of the output and the output is obtained from using the same rule applied to the previous outputs.
There is a wide variety of a recurrent neural network. For examples:

Example - 1 : Recurrent networks with recurrent connections between hidden units produce an output at each time step (Figure 2.7). This maps an input sequence of x values to a corresponding sequence of output o values. Loss L is the measurement of the variation of the output o values with the target values y.
In the figure $\boldsymbol{x}^{(t)}$ represents the input, $\boldsymbol{h}^{(t)}$ represents the hidden layer activations, $\boldsymbol{o}^{(t)}$ represents the output, $\boldsymbol{y}^{(t)}$ represents the target and $\boldsymbol{L}^{(t)}$ represents the loss at each time step t.

To develop the forward propagation equations for this recurrent network, we assume the hyperbolic tangent activation function as the activation function for the hidden units. We also assume the output is discrete. The output is the unnormalized log probabilities of each

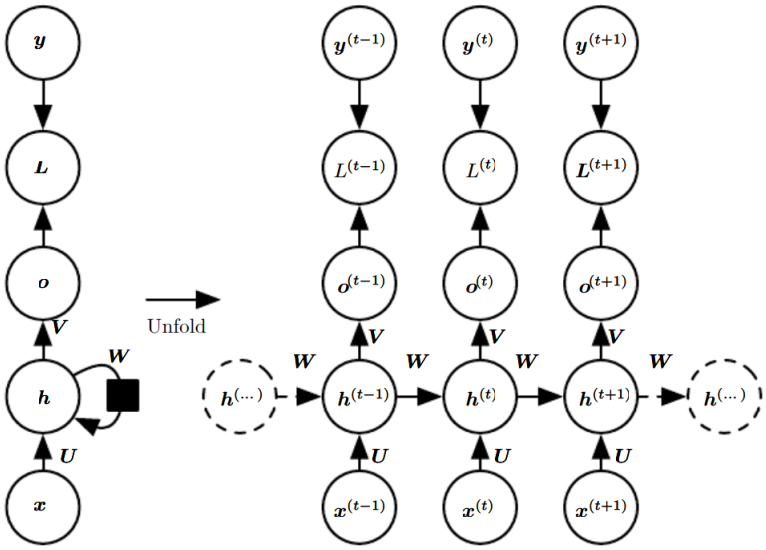


Figure 2.7: Recurrent Neural Network - Type 1

possible targets. Then the softmax operation is applied to obtain a vector $\hat{\boldsymbol{y}}$ of normalized probabilities over the output. $\boldsymbol{h}^{(0)}$ is the initial state. And the updated equations are the following:

$$\boldsymbol{a}^{(t)} = \boldsymbol{b} + \boldsymbol{W}\boldsymbol{h}^{(t-1)} + \boldsymbol{U}\boldsymbol{x}^{(t)} \tag{2.10}$$

$$\boldsymbol{h}^{(t)} = tanh(\boldsymbol{a}^{(t)}) \tag{2.11}$$

$$\boldsymbol{o}^{(t)} = \boldsymbol{c} + \boldsymbol{V}\boldsymbol{h}^{(t)} \tag{2.12}$$

$$\hat{\boldsymbol{y}}^{(t)} = softmax(\boldsymbol{o}^{(t)}) \tag{2.13}$$

Here, $\boldsymbol{U}$, $\boldsymbol{V}$ and $\boldsymbol{W}$ represent the input-to-hidden, hidden-to-output and hidden-to-hidden connections respectively. And $\boldsymbol{b}$ and $\boldsymbol{c}$ are the bias vector along with the weight matrices.

Example - 2 : Recurrent networks having recurrent connections only from the output at one time step to the hidden units at the next time step produce an output at each time step (Figure 2.8). This recurrent network is trained to put a specific output into $\boldsymbol{o}$ and only the information in $\boldsymbol{o}$ is allowed to send to the future. The previous $\boldsymbol{h}$ is connected to the current $\boldsymbol{h}$ through the output produced by the previous $\boldsymbol{h}$. There is a lack of important information from the past unless $\boldsymbol{o}$ is very high-dimentional. It may easier to train but this

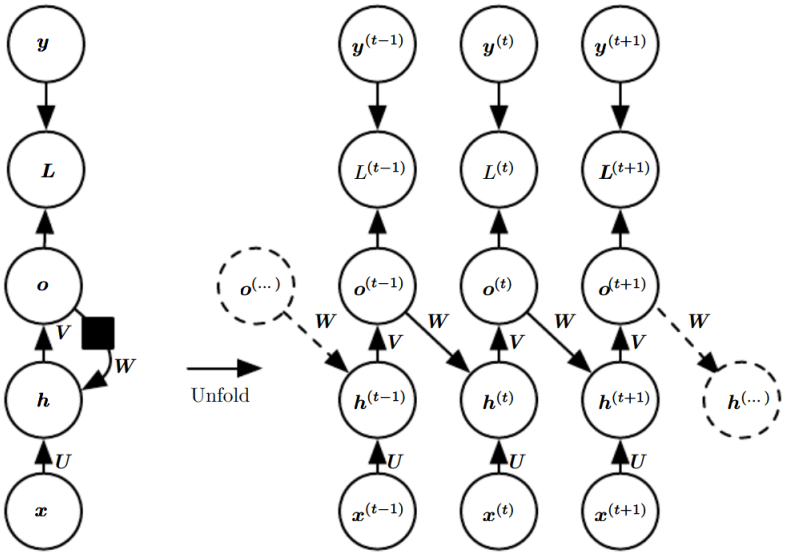


Figure 2.8: Recurrent Neural Network - Type 2

makes the recurrent network less powerful [23].

2.2.1 LSTM

Generally, recurrent neural networks have a chain form of repeating modules of neural network [2]. In standard RNNs, the structure of this repeating module is very simple, such as a single hyperbolic tangent layer (Figure 2.9).

But in LSTM, this repeating module has four layers and they are interacting in a very special way (Figure 2.10).

The first step of LSTM is to decide what information it needs to throw away from the cell state. This cell state is the key to LSTM (Figure 2.11). The cell state runs straight down the entire chain, with very little linear interactions. So the information flows along it without any change. One of the structure of LSTM, called gates, controls the process of removing or adding information in the cell states. Gates are consisting of a sigmoid layer and and a pointwise multiplication operation (Figure 2.12). The sigmoid layer decides how

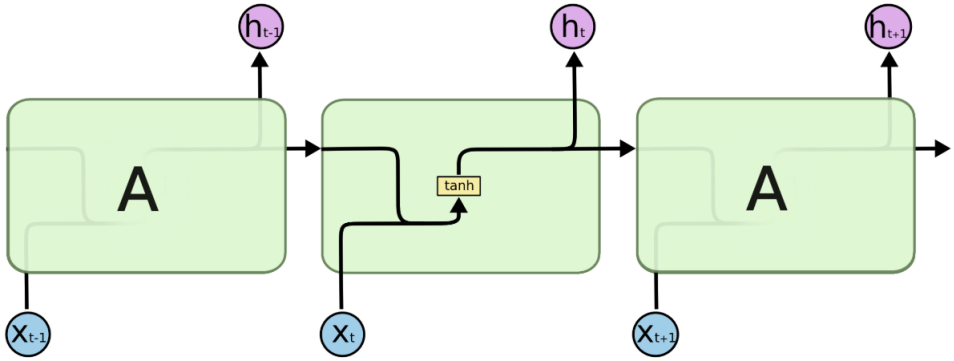


Figure 2.9: Standard RNN

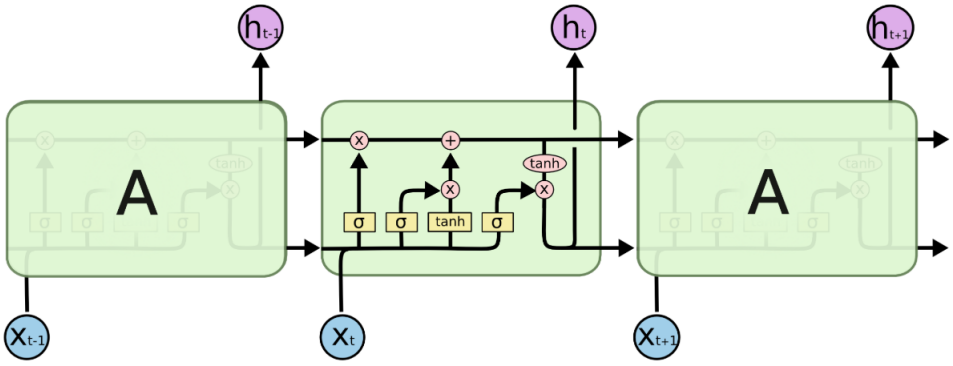


Figure 2.10: LSTM

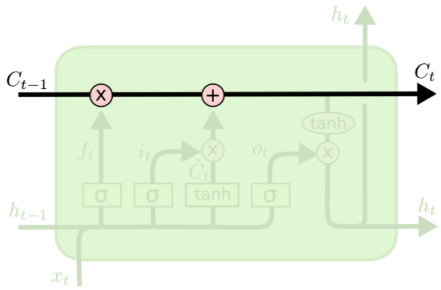


Figure 2.11: Cell of LSTM

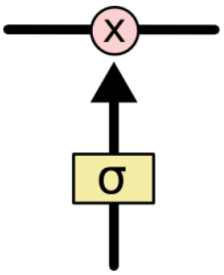


Figure 2.12: Structure of gate of LSTM

much of each component should go through. The output of sigmoid layer is between 0 and 1, where 0 means no information is going through and 1 means all information are going through. In the first step forgate gate layer (Figure 2.13) decides what information needs to remove from the cell state.

$$f_t^j = \sigma\left(b_f^j + \sum_k U_f^{j,k} x_t^k + \sum_k W_f^{j,k} h_{t-1}^k\right), \tag{2.14}$$

where b_f, U_f and W_f represents the biases, input weights and recurrent weight for the forget gates respectively, x_t is the current input vector and h_t is the current hidden layer vector that contains the outputs of all the LSTM cells.

In the next step, LSTM decides what new information needs to store in the cell state. This is done in two parts - input gate layer decides what values need to update (Figure 2.14) and tanh layer creates a vector of new candidates value, $\widetilde{C}_t$, to update.

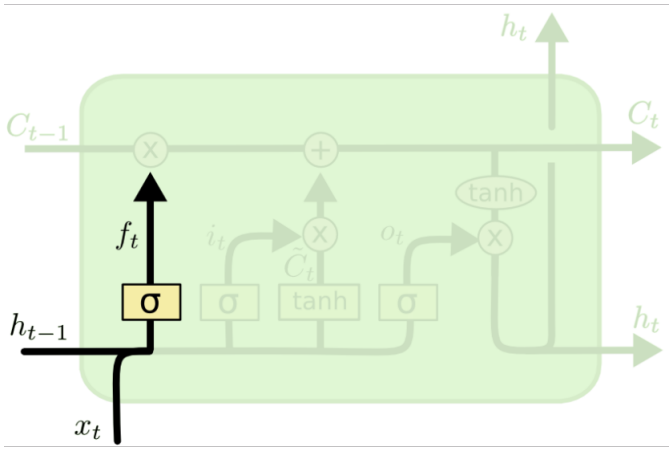


Figure 2.13: Forgate gate of LSTM

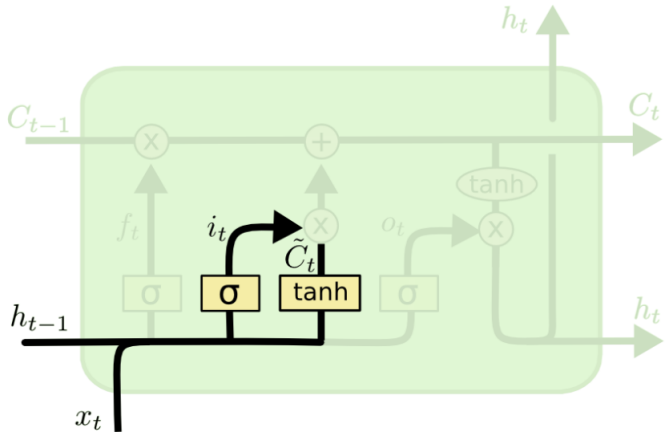


Figure 2.14: Input gate of LSTM

$$i_t^j = \sigma\left(b_i^j + \sum_k U_i^{j,k} x_t^k + \sum_k W_i^{j,k} h_{t-1}^k\right), \tag{2.15}$$

$$\widetilde{C}_t^j = tanh\left(b_C^j + \sum_k U_C^{j,k} x_t^k + \sum_k W_C^{j,k} h_{t-1}^k\right), \tag{2.16}$$

where b_i, U_i and W_i represents the biases, input weights and recurrent weight for the input gates respectively. b_C, U_C and W_C are the biases, input weights and recurrent weights for the hyperbolic tangent layer to create the candidate values.

With all theses information cell state will be updated (Figure 2.15).

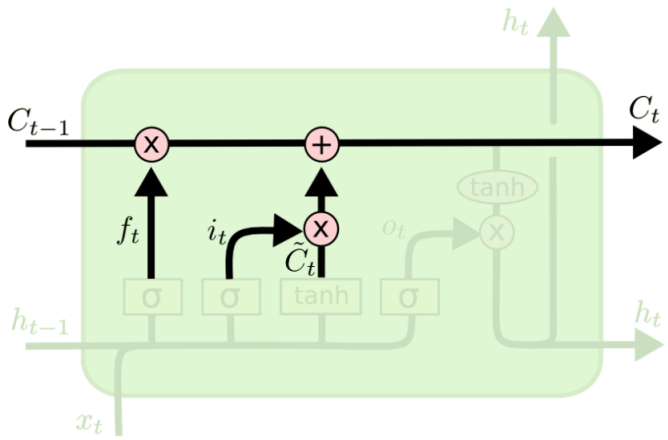


Figure 2.15: Updated state of LSTM

$$C_t^j = \sigma\left(f_t^j * C_{t-1}^j + i_t^j * \widetilde{C}_t^j\right), \tag{2.17}$$

The output gate o_t^j that uses a sigmoid unit for gating stops the output h_t^j of the LSTM cell:

$$h_t^j = tanh\left(C_t^j\right) o_t^j \tag{2.18}$$

$$o_t^j = \sigma\left(b_o^j + \sum_k U_o^{j,k} x_t^j + \sum_k W_o^{j,k} h_{t-1}^k\right), \tag{2.19}$$

where b_o, U_o and W_o represents the biases, input weights and recurrent weight for the output gates respectively.

Chapter 3

Related Work

Space weather, specifically solar flare, is largely studied in recent years [35, 5, 16, 36, 45, 12]. Still the mechanism of solar flares is an unresolved mysterious topic in solar physics. Various kinds of works have been done to predict solar flares.

3.1 Space Weather Prediction Taxonomy

Forecasting space weather accurately is still in it's infancy stage despite of the availability of huge amount of related data. Our proposed taxonomy divides space weather prediction techniques into two groups - *Data Driven* and *Model Driven*. Next we divide *Data Driven* into two groups, one is *SVM, K-nearest neighbors, Logistic Regression* and the other is *Deep Learning*. Then Deep Learning is divided into two groups: *Convolutional Neural Network* and *Recurrent Neural Network*.

3.1.1 Model Driven Techniques

Techniques in this category do not use any machine learning algorithm. This category involves statistical approach to classify the sunspots [29], analyzing the mechanism of solar activity [45], comparing flare forecasting method [6], investigating economic impact because of space weather [12], exploring the risks for lives related to solar flare event [27]. This also involves understanding the physical mechanisms of the origin of solar flares [38, 36, 19, 10, 18].

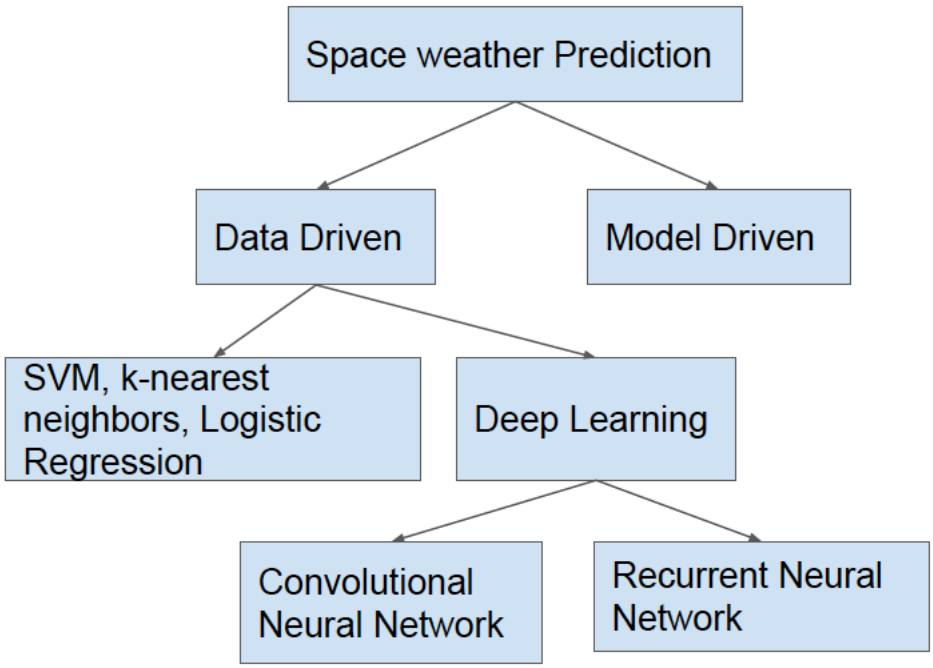


Figure 3.1: Solar flares prediction taxonomy

3.1.2 Data Driven Techniques

This techniques use machine learning algorithm to train the data and then do the prediction on the test data.

3.1.2.1 Deep Learning Techniques

This subgroup of techniques use deep neural network for the prediction. In recent years, deep neural networks have lead to breakthrough results on various types of problems. There are two types of deep neural networks - 1) Convolutional Neural Network and 2) Recurrent Neural Network.

Convolutional Neural Network

Convolutional Neural Networks have achieved great success in face recognition [22] and handwritten character recognition [24]. After that, this has been very popular among scientists and engineers. After gaining excellent performance in ImageNet challenge [11],

deep learning scientists and engineers have applied Convolutional Neural Network in other image-related tasks, for example, image captioning [20], visual game playing [30].

A deep network called "Dynamic Convolutional Layer", in which filters will vary from input to input during testing, is applied to do a short range weather prediction in [21]. [17] suggested a a convolutional sequence-to-sequence autoencoder model to predict undiscovered weather situations from previous satellite images. [31] developed a deep Convolutional Neural Network to predict the solar flares in a time window from 20 minutes to 120 minutes using Geostationary Operational Environmental Satellite (GOES) X-ray flux data. [33] presented a deep neural network named Deep Flare Net (DeFN) to calculate the probability of flares occurring in the next 24 hours using SDO images.

Recurrent Neural Network

Recurrent Neural Network is a great success for recognizing patterns that are defined by temporal distance. Recurrent Neural Network has been applied in music generation [13, 8], text generation [40, 14], image generation [15], image caption generation [43, 20].

To study solar wind-magnetosphere from the prediction of geomagnetic storm recurrent neural network is used in [42]. Prediction of two sunspot-related time series is done in [28] using a recurrent neural network. Weather conditions and isolation have influence on photovoltaic (PV) system, accepted as an alternative source of energy. In [44] a recurrent neural network is used to forecast the power output of PV system. To forecast hourly and daily solar irradiation a diagonal recurrent wavelet neural network is used in [9]. Solar wind prediction is made in [32] using hybrid computation methods so as wavelet decomposition and recurrent neural networks.

3.1.2.2 Other Machine Learning Techniques

This techniques involves machine learning approaches, for example, k-nearest neighbors, logistic regression, random forest, support vector machine, neural network to exploit the huge amount of data available in real time. In [26, 25] solar flares and proton events are predicted applying a classification algorithm called SVM-KNN which is a combination of

support vector machine and k-nearest neighbors. To predict the probability for each solar active region to produce X-, M-, or C- class flares during the next 1-day time period a logistic regression model is proposed in [39]. [7] proposed support vector machine model to forecast M- and X-class solar flares exploiting the Solar Dynamics Observatorys Helioseismic and Magnetic Imager data. In [46] support vector machine is applied to do short-term solar power prediction. [37] established a correlation between solar flares and sunspot groups exploiting the National Geophysical Data Center. A combination of K-nearest neighbors and nearest centroid algorithms is implemented in [41] to predict the solar flares based on the relation between the maximum ratio of the flare flux and nonflare background. In this work, the GOES X-ray flux data and the Space Weather Prediction Center flares catalog is used. Three machine learning techniques - k-NN, SVM and ERT - are compared in [34] to forecast solar flare and the result showed that k-NN performs the best.

Chapter 4

Objectives and Methodologies

Through this chapter, we have discussed about the datasets we have used. We have also described our purposes using these datasets and finally the approaches we have followed to achieve our goals.

4.1 Data Description

4.1.1 X-ray dataset

Solar activities describe the dynamic conditions and events on the Sun in near-Earth space and in our upper atmosphere, gives rise to various kind of space weather. These activities include solar flares, coronal mass ejections, high-speed solar wind, and solar energetic particles. A solar flare is an emission of immense magnetic energy from the sun. This is associated with the active regions and some of these active regions have sunspots. Flares are monitored in X-rays. There are are two X- ray wavebands those are the measurement of the real-time energy flux, measured by the X-Ray detectors, that reaches to earth. The satellites - GOES-10, GOES-11, GOES-13, GOES-14 and GOES-15 - are used to collect the data. The data is available for every 5 minutes. The shorter wavelength is from 0.05 to 0.4 nanometers, which is called harder X-ray channel and the longer wavelength is from 0.1 to 0.8 nanometers, which is called softer X-ray channel. All the data are available on online [3]. To investigate the forecasting the softer X-ray channel is used in this work.

4.1.2 Particle Flux dataset

GOES-10, GOES-11, GOES-13 and GOES-15 satellites are used to collect the data. The data is available for every 5 minutes. Different types of energetic particles are here. Particles with energy greater than 1 megaelectronvolt (MeV), greater than 5 megaelectronvolt (MeV), greater than 10 megaelectronvolt (MeV), greater than 30 megaelectronvolt (MeV), greater than 50 megaelectronvolt (MeV) and greater than 100 megaelectronvolt (MeV). Energy paticle data can be found on online [4]. We have predicted all types of energetic particles in this work.

4.1.3 Data Preprocessing

The values in the data, we are working with, are really very small. So we used the logarithmic transformation on the original data. Then again we transformed the resulting data in a way that the mean of the finally transformed data will be zero. In the X-ray flux dataset, the softer X-ray flux for year 2011 and 2012 is used for the experiment. 70% of the data is used as the training set and 30% data is used as test set. In the particle dataset, data is used from June 2015 till now. 80% of the data is used as training set and the rest is used as the test set.

Now, we will discuss thoroughly about our objectives and our approaches. We used two baseline results to compare the performance of our developed network.

4.2 Objectives

The followings are the objectives of this research :

1. Developing a deep neural network architecture to forecast the solar X-ray and particle fluxes in continuous time from time series data.

2. Developing a deep learning model to predict the X-ray and particle fluxes from the solar images.

3. Developing a deep network architecture to predict the flux from sequences of solar images.

4. Developing a deep network model to predict the flux from a combination of history of fluxes and sequences of images.

4.3 Baselines

We considered non-machine learning approaches as our baselines. They are as follows:

4.3.1 Baseline 1 : Baseline prev

In this baseline, the latest available flux is used as prediction.

4.3.2 Baseline 2 : Baseline avg

The average of the previous fluxes with particular choice of the number of previous time step is used as prediction here.

4.4 Methodology 1 : 1-D Convolutional Neural Network (CNN)

1-D CNN performs really well in sequence processing. This considers time as the independent dimension and extracts interesting features from local input patches and allows for modularized representation. 1-D CNN is successful in many fields, for example - audio generation, machine translation etc.

4.4.1 Data Preparation

1-D CNN requires the shape of input as a 3D tensor - (number of samples, lookback, number of features), where lookback is the number of previous samples we are looking at for each sample. So we need to convert our input into a 3D tensor to feed into the 1-D CNN. We can customize the lookback to analyze the behavior of the network.

4.4.2 Model Architechture

We have used 16 different windows (filters) in the first layer and each of the them has sliding window (kernel) size 5. Rectified Linear Unit is the activation function here. The output of the first layer of the network is fed into the second layer as input. We have used 16 filters in the second layer where each of them has kernel 5 like the first layer. After using stack of convolutional layers we have used a maxpooling layer to reduce the complexity of the output and to prevent overfitting the data.The output of this maxpooling layer will be half of the input of the layer as we have used maxpooling size 2 here. On the top of the maxpooling layer, we have used a flatten layer to convert the output of the maxpooling layer, which is a 2 dimensional arrays, into a single linear vector. On top of that we have used a fully connected layer. Mean squared error is the loss function here.

4.4.3 Prediction

Two approaches are followed here to do the forecasting.

4.4.3.1 Single Timestep Prediction

While training the data, the network learns all the parameters from the training data. Then these learned parameters are used to do prediction for a single timestep in future.

4.4.3.2 Multiple Timetep Prediction

This is actually a chain prediction of single timestep. The current prediction is added in the input, at the same time the first observation is removed from the input, to fed into the network to predict the next timestep. In this prediction, a single model is used to predict even further in the future.

4.5 Methodology 2 : Long Short term Memory (LSTM)

LSTM is very popular in time series prediction, This has a lot of flexibility in modeling a problem as we have control over the parameters of the time series. For instance - when we are required to predict the current timestep using the previous information, we can use many to one model. Sometimes we also need to predict for multiple timesteps in future at once using the same previous information, in that case we can use many to many model.

4.5.1 Data Preparation

1-D CNN also requires a 3D tensor as input- (number of samples, lookback, number of features). So to be fed into the network the data is converted into three dimensional data. The behavior of the network is analyzed for different values of lookback.

4.5.2 Model Architechture

We have used a single LSTM layer with 16 units here. Hyperbolic tangent is the activation function here. On the top of this LSTM layer, we have used a fully connected layer. The loss function is same as 1-D CNN.

4.5.3 Prediction

Like 1-D CNN, two types of predictions are made here - (i) Single Timestep Prediction, (ii) Multiple Timestep Prediction.

4.6 Methodology 3 : 1-D CNN and LSTM

Here 1-D CNN is used to extract higher level features and that is fed into the LSTM as input.

4.6.1 Data Preparation

4D tensor is required as input of this network - (number of samples, subsequence, lookback, number of features). So the data is converted into 4D tensor, as required, and then fed into the network as input of 1-D CNN.

4.6.2 Model Architechture

We have used two convolutional layers with the Timedistributed wrapper and each of them has 16 units and each of the unit has 5 kernels. Output of one layer is fed into the next layer as input of the next layer. On top of that we have used a flatten layer and a fully connected layer. In this case mean squared error is the loss function as well.

4.6.3 Prediction

The parameters learned from the training data is used to predict the test data for a single timestep and multiple timesteps in the future.

Chapter 5

Preliminary Results

Through this chapter, we discuss our preliminary results for both the graphical and numerical presentation of the X-ray flux dataset and the particle dateset.

5.1 Preliminary Results For X-ray Dataset

We applied five techniques to predict the X-ray flux - Long Short Term Memory (LSTM), Chain LSTM, 1-D Convolutional Neural Network (Conv1D), Chain Conv1D and combination of Conv1D and LSTM (Conv1D+LSTM). We did experiment for different lookback - 24, 48, 72, 96 for the prediction in different future time - 5 minutes, 15 minutes, 30 minutes, 45 minutes, 60 minutes, 75 minutes, 90 minutes, 105 minutes and 120 minutes.

5.1.1 Graphical Presentation

The following graph represents the results for baseline prev, baseline avg, conv1d, chain conv1d, LSTM, chain LSTM, conv1d and LSTM with 24 lookback.

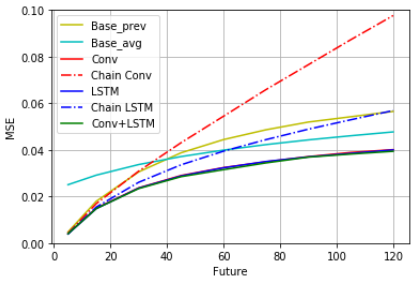


Figure 5.1: Forecasting in future for 24 lookback

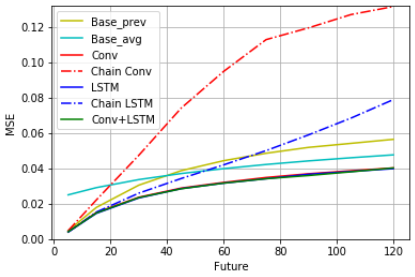


Figure 5.2: Forecasting in future for 48 lookback

The following graph 5.1 represents the result for 24 lookback.

The following graph 5.2 shows the results for 48 lookback.

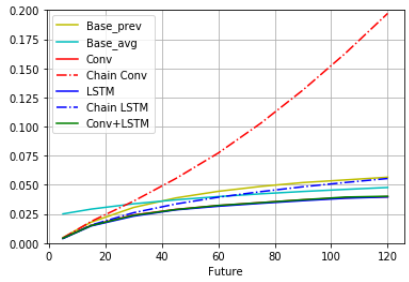


Figure 5.3: Forecasting in future for 72 lookback

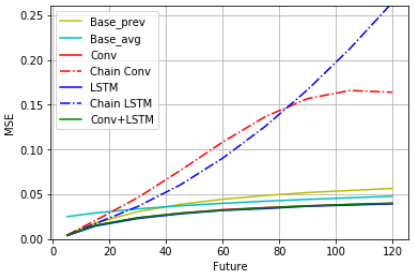


Figure 5.4: Forecasting in future for 96 lookback

72 lookback is used in the following result 5.3.

The following result 5.4 is obtained using 96 lookback.

5.1.2 Numerical Presentation

Following table 5.1 is the result in next 5 minutes for different previous flux.

Following table 5.2 represents the result in next 30 minutes for different previous flux.

Table 5.1: Results in next 5 minutes with different lookback

Techniques	MSE for 24 lookback	MSE for 48 lookback	MSE for 72 lookback	MSE for 96 lookback
Baseline prev	0.0048	0.0048	0.0048	0.0048
Baseline avg	0.0251	0.0308	0.0344	0.0368
LSTM	0.0039	0.0039	0.00395	0.004
Conv1D	0.0042	0.0047	0.0044	0.0044
Conv1D+LSTM	0.0042	0.0043	0.0042	0.00475

Table 5.2: Results in next 30 minutes with different lookback

Techniques	MSE for 24 lookback	MSE for 48 lookback	MSE for 72 lookback	MSE for 96 lookback
Baseline prev	0.0306	0.0306	0.0306	0.0306
Baseline avg	0.0337	0.0364	0.0386	0.0402
LSTM	0.0238	0.0237	0.0236	0.0232
Chain LSTM	0.0261	0.026	0.0261	0.0364
Conv1D	0.0241	0.0237	0.0237	0.0248
Chain Conv1D	0.0309	0.0475	0.03597	0.0463
Conv1D+LSTM	0.0238	0.0248	0.0248	0.0266

Following table 5.3 is the result in next 1 hour for different previous flux.

Table 5.3: Results in next 1 hour with different lookback

Techniques	MSE for 24 lookback	MSE for 48 lookback	MSE for 72 lookback	MSE for 96 lookback
Baseline prev	0.0444	0.0444	0.0444	0.0444
Baseline avg	0.0399	0.0408	0.0421	0.043
LSTM	0.032	0.032	0.0351	0.0315
Chain LSTM	0.0395	0.0421	0.0394	0.0903
Conv1D	0.0328	0.0338	0.0336	0.0332
Chain Conv1D	0.0544	0.095	0.0773	0.1082
Conv1D+LSTM	0.0321	0.0351	0.0351	0.0359

Following table 5.4 is the result in next 2 hours for different previous flux.

5.2 Preliminary Results For Particle Dataset

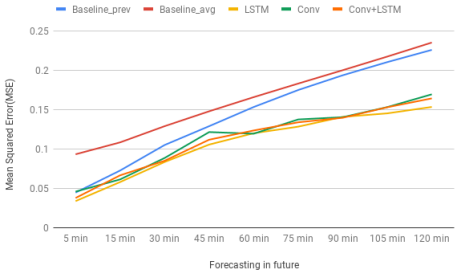
In the particle dataset, there are particles with 6 different energy - particles at greater than 1 megaelectronvolt (MeV), 5 megaelectronvolt (MeV), 10 megaelectronvolt (MeV), 10 megaelectronvolt (MeV), 30 megaelectronvolt (MeV), 50 megaelectronvolt (MeV), 100 megaelectronvolt (MeV). These are the feature of this dataset and we have predicted all of them. We have applied three techniques here - LSTM, Conv1D and combination of them to predict the particle emissions.

Table 5.4: Results in next 2 hours with different lookback

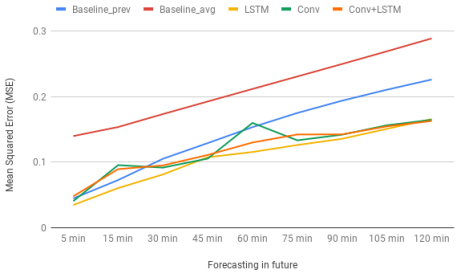
Techniques	MSE for 24 lookback	MSE for 48 lookback	MSE for 72 lookback	MSE for 96 lookback
Baseline prev	0.0565	0.0565	0.0565	0.0565
Baseline avg	0.0477	0.0469	0.04697	0.0472
LSTM	0.0402	0.0404	0.0431	0.0414
Chain LSTM	0.0569	0.079	0.0554	0.264
Conv1D	0.0402	0.0413	0.0427	0.0415
Chain Conv1D	0.0977	0.1317	0.1972	0.1638
Conv1D+LSTM	0.0399	0.0417	0.0488	0.0488

5.2.1 Graphical Presentation

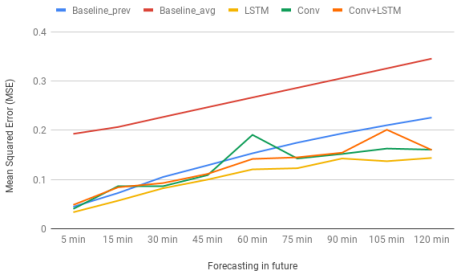
The following graph 5.5 represents the results for the particles at greater than 1 megaelectronvolt (MeV) with different lookback.



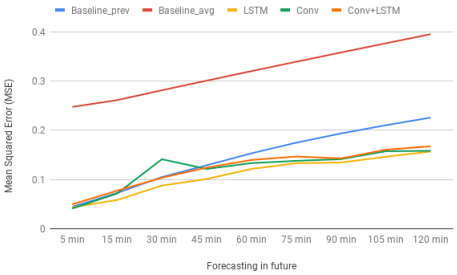
(a) Lookback 24



(b) Lookback 48



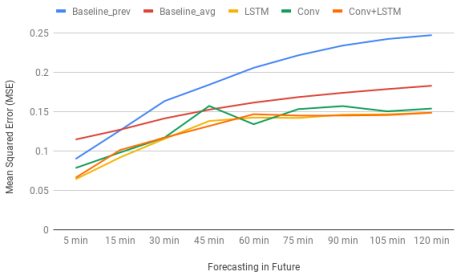
(c) Lookback 72



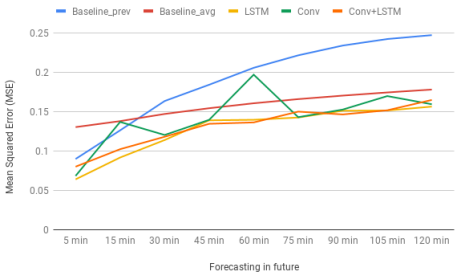
(d) Lookback 96

Figure 5.5: Results for the particles greater than 1 MeV

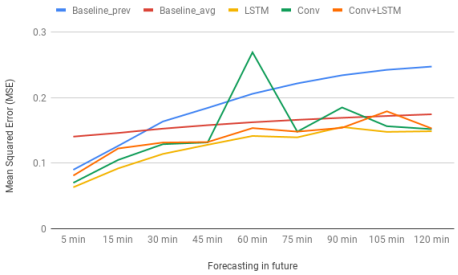
The following graph 5.6 shows the results for the particles at greater than 5 megaelectronvolt (MeV) with different lookback.



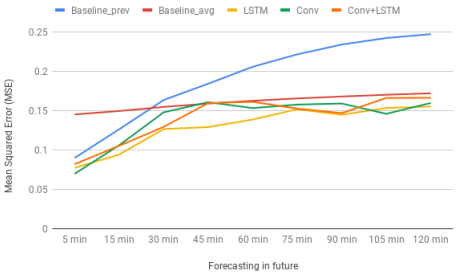
(a) Lookback 24



(b) Lookback 48



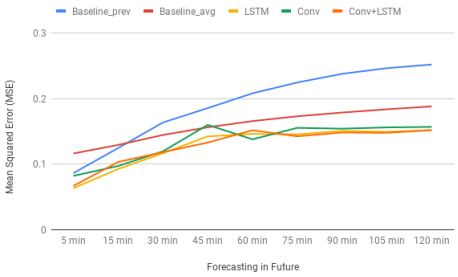
(c) Lookback 72



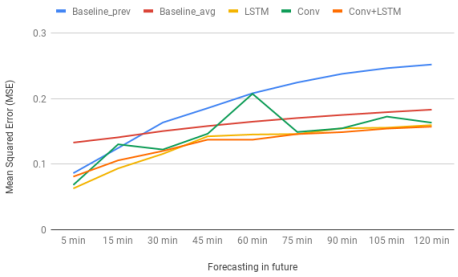
(d) Lookback 96

Figure 5.6: Results for the particles greater than 5 MeV

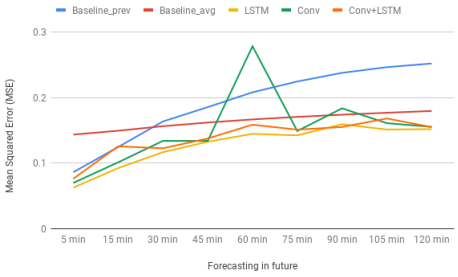
The results for the particles at greater than 10 megaelectronvolt (MeV) with different lookback is the following graph 5.7



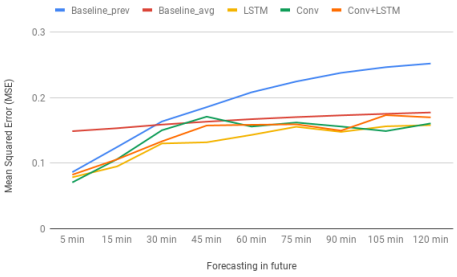
(a) Lookback 24



(b) Lookback 48



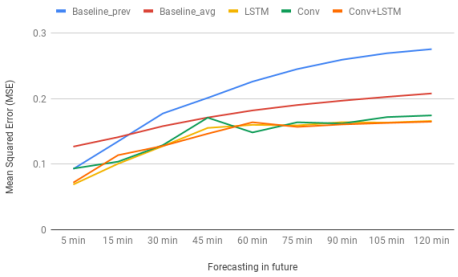
(c) Lookback 72



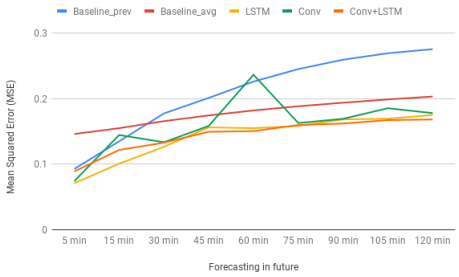
(d) Lookback 96

Figure 5.7: Results for the particles greater than 10 MeV

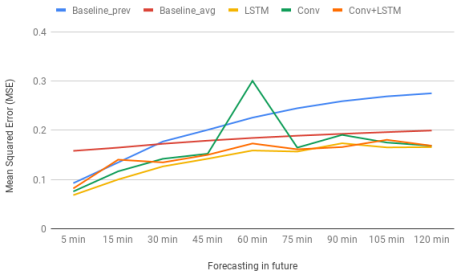
The following graph 5.8 shows the results for the particles at greater than 30 megaelectronvolt (MeV) with different lookback.



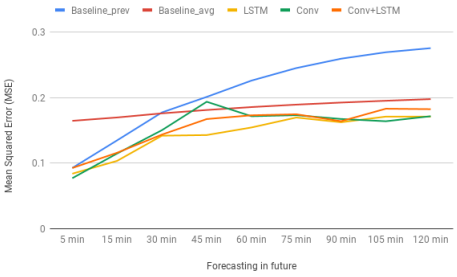
(a) Lookback 24



(b) Lookback 48



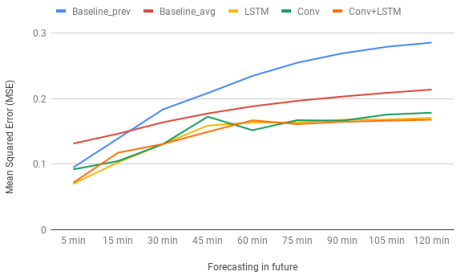
(c) Lookback 72



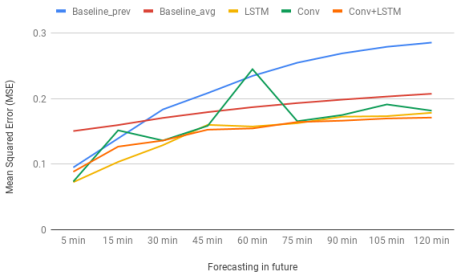
(d) Lookback 96

Figure 5.8: Results for the particles greater than 30 MeV

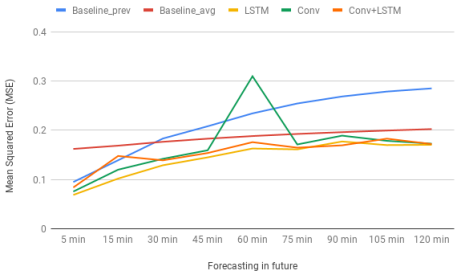
The following graph 5.9 represents the results for the particles at greater than 50 mega-electronvolt (MeV) with different lookback.



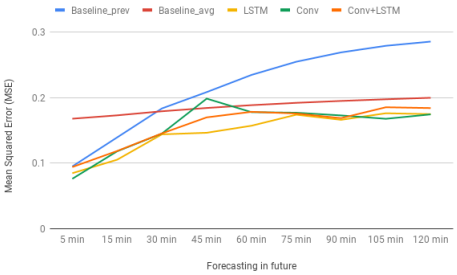
(a) Lookback 24



(b) Lookback 48



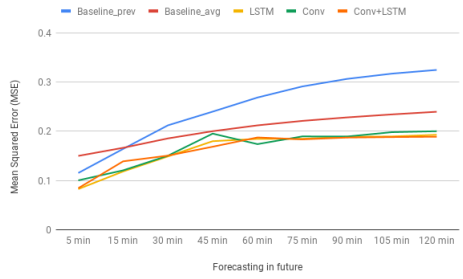
(c) Lookback 72



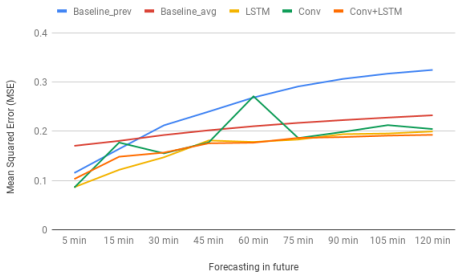
(d) Lookback 96

Figure 5.9: Results for the particles greater than 50 MeV

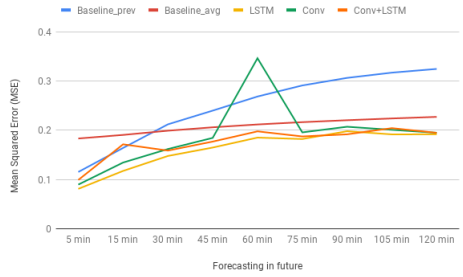
The results for the particles at greater than 100 megaelectronvolt (MeV) with different lookback is the following graph 5.10.



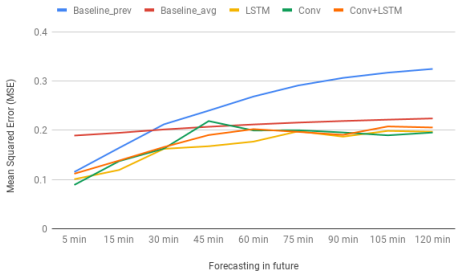
(a) Lookback 24



(b) Lookback 48



(c) Lookback 72



(d) Lookback 96

Figure 5.10: Results for the particles greater than 100 MeV

5.3 Discussion of Results

5.3.1 X-ray flux dataset

When the forecasting is made for far future, the average baseline is performing better than the LSTM, Conv1D, Conv1D+LSTM perform better than both of the baselines and their performance is almost similar. Error is reduced by around 18% by implementing LSTM. The performance of LSTM, Conv1D and Conv1D+LSTM improves with the increase of the lookback. LSTM. Conv1D and Conv1D+LSTM also do better than both of the baselines for far future forecasting.

5.3.2 Particle flux dataset

In this case, the behavior of the result is almost similar to the X-ray flux dataset. The average baseline performs better in the far future than in the near future. LSTM, conv1D, Conv1D+LSTM outperform both of the baselines. LSTM. Conv1D and Conv1D+LSTM also perform better than both of the baselines for far future forecasting. For the particle dataset, LSTM performs best among all of them. The performance of Conv1D degrades for the prediction in around 1 hour. In the dataset, energetic particles with lower energy is a subset of the energetic particles with higher energy. That's is why the results for all the energetic particles are very close to each other.

For both datasets, LSTM for 24 lookback works best among all of the approached we applied here.

Chapter 6

Research Plan

Table 6.1: Activities - Part 1

Activity	Progress/Dates	Description
Initial Literature Review	80%	A comprehensive review of the literature regarding flux prediction.
Identify Limitations/Problems	75%	The identification of research opportunities.
Developing Research Question	100%	Formalization of research opportunities in the form of research questions.
Small Convolutional Network Implementation	100%	Implementation of a convolutional neural network to predict solar X-ray and particle fluxes from time series data.
Small Recurrent Network Implementation	100%	Implementation of a recurrent neural network to predict X-ray and particle fluxes from time series data.

Table 6.2: Activities - Part 2

Activity	Progress/Dates	Description
Initial Literature Review	80%	A comprehensive review of the literature regarding flux prediction with solar images.
Identify Limitations/Problems	70%	The identification of research opportunities.
Developing Research Question	100%	Formalization of research opportunities in the form of research questions.
Small Convolutional Network Implementation	30%	Implementation of a convolutional neural network to forecast flux from solar images.

Table 6.3: Activities - Part 3

Activity	Progress/Dates	Description
Publication 1	10%	Write a paper based on flux prediction from time series data.
Deep Neural Network Architecture	December 2019 - March 2019	Implementation of a deep learning model to forecast flux from sequences of images.
Deep Network Architecture	March 2019 - May 2019	Implementation of a deep learning architecture to forecast flux from a combination of sequences of images and history of fluxes.
Dissertation Writing	May 15, 2019 - December 18, 2019	Write dissertation as the other tasks are completed.
Dissertation Defense	December 1, 2019 - December 14, 2019	Defend dissertation in December.

References

[1] https://www.nasa.gov/content/goddard/the-difference-between-flares-and-cmes.

[2] http://colah.github.io/.

[3] http://darts.isas.ac.jp/pub/solar/sswdb/goes/xray/.

[4] ftp://ftp.swpc.noaa.gov/pub/lists/particle/.

[5] H. Alfvén and P. Carlqvist. Currents in the solar atmosphere and a theory of solar flares. *Solar Physics*, 1(2):220–228, 1967.

[6] G. Barnes, K. Leka, C. Schrijver, T. Colak, R. Qahwaji, O. Ashamari, Y. Yuan, J. Zhang, R. McAteer, D. Bloomfield, et al. A comparison of flare forecasting methods. i. results from the all-clear workshop. *The Astrophysical Journal*, 829(2):89, 2016.

[7] M. G. Bobra and S. Couvidat. Solar flare prediction using sdo/hmi vector magnetic field data with a machine-learning algorithm. *The Astrophysical Journal*, 798(2):135, 2015.

[8] N. Boulanger-Lewandowski, Y. Bengio, and P. Vincent. Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription. *arXiv preprint arXiv:1206.6392*, 2012.

[9] J. Cao and X. Lin. Study of hourly and daily solar irradiation forecast using diagonal recurrent wavelet neural networks. *Energy Conversion and Management*, 49(6):1396–1406, 2008.

[10] L. Comisso, M. Lingam, Y.-M. Huang, and A. Bhattacharjee. General theory of the plasmoid instability. *Physics of Plasmas*, 23(10):100702, 2016.

[11] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. Ieee, 2009.

[12] J. Eastwood, E. Biffis, M. Hapgood, L. Green, M. Bisi, R. Bentley, R. Wicks, L.-A. McKinnell, M. Gibbs, and C. Burnett. The economic impact of space weather: Where do we stand? *Risk Analysis*, 37(2):206–218, 2017.

[13] D. Eck and J. Schmidhuber. A first look at music composition using lstm recurrent neural networks. *Istituto Dalle Molle Di Studi Sull Intelligenza Artificiale*, 103, 2002.

[14] A. Graves. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013.

[15] K. Gregor, I. Danihelka, A. Graves, D. J. Rezende, and D. D. Wierstra. A recurrent neural network for image generation. arxiv preprint. *arXiv preprint arXiv:1502.04623*, 2015.

[16] J. Heyvaerts, E. R. Priest, and D. M. Rust. An emerging flux model for the solar flare phenomenon. *The Astrophysical Journal*, 216:123–137, 1977.

[17] S. Hong, S. Kim, M. Joh, and S.-k. Song. Psique: Next sequence prediction of satellite images using a convolutional sequence-to-sequence network. *arXiv preprint arXiv:1711.10644*, 2017.

[18] M. Janvier. Three-dimensional magnetic reconnection and its application to solar flares. *Journal of Plasma Physics*, 83(1), 2017.

[19] M. Janvier, G. Aulanier, and P. Démoulin. From coronal observations to mhd simulations, the building blocks for 3d models of solar flares (invited review). *Solar Physics*, 290(12):3425–3456, 2015.

[20] A. Karpathy and L. Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3128–3137, 2015.

[21] B. Klein, L. Wolf, and Y. Afek. A dynamic convolutional layer for short range weather prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4840–4848, 2015.

[22] S. Lawrence, C. L. Giles, A. C. Tsoi, and A. D. Back. Face recognition: A convolutional neural-network approach. *IEEE transactions on neural networks*, 8(1):98–113, 1997.

[23] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *nature*, 521(7553):436, 2015.

[24] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[25] R. Li, Y. Cui, H. He, and H. Wang. Application of support vector machine combined with k-nearest neighbors in solar flare and solar proton events forecasting. *Advances in Space Research*, 42(9):1469–1474, 2008.

[26] R. Li, H.-N. Wang, H. He, Y.-M. Cui, and Z.-L. Du. Support vector machine combined with k-nearest neighbors for solar flare forecasting. *Chinese Journal of Astronomy and Astrophysics*, 7(3):441, 2007.

[27] M. Lingam and A. Loeb. Risks for life on habitable planets from superflares of their host stars. *The Astrophysical Journal*, 848(1):41, 2017.

[28] S. Marra and F. C. Morabito. A new technique for solar activity forecasting using recurrent elman networks. *International Journal of Computational Intelligence*, 3(1):8–13, 2006.

[29] P. S. McIntosh. The classification of sunspot groups. *Solar Physics*, 125(2):251–267, 1990.

[30] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.

[31] T. A. Nagem, R. S. Qahwaji, S. S. Ipson, Z. Wang, and A. S. Al-Waisy. Deep learning technology for predicting solar flares from (geostationary operational environmental satellite) data. 2018.

[32] C. Napoli, F. Bonanno, and G. Capizzi. An hybrid neuro-wavelet approach for long-term prediction of solar wind. *Proceedings of the International Astronomical Union*, 6(S274):153–155, 2010.

[33] N. Nishizuka, K. Sugiura, Y. Kubo, M. Den, and M. Ishii. Deep flare net (defn) model for solar flare prediction. *The Astrophysical Journal*, 858(2):113, 2018.

[34] N. Nishizuka, K. Sugiura, Y. Kubo, M. Den, S. Watari, and M. Ishii. Solar flare prediction model with three machine-learning algorithms using ultraviolet brightening and vector magnetograms. *The Astrophysical Journal*, 835(2):156, 2017.

[35] E. N. Parker. The solar-flare phenomenon and the theory of reconnection and annihiliation of magnetic fields. *The Astrophysical Journal Supplement Series*, 8:177, 1963.

[36] E. Priest. *Magnetohydrodynamics of the Sun*. Cambridge University Press, 2014.

[37] R. Qahwaji and T. Colak. Automatic short-term solar flare prediction using machine learning and sunspot associations. *Solar Physics*, 241(1):195–211, 2007.

[38] K. Shibata and T. Magara. Solar flares: magnetohydrodynamic processes. *Living Reviews in Solar Physics*, 8(1):6, 2011.

[39] H. Song, C. Tan, J. Jing, H. Wang, V. Yurchyshyn, and V. Abramenko. Statistical assessment of photospheric magnetic features in imminent solar flare predictions. *Solar Physics*, 254(1):101–125, 2009.

[40] I. Sutskever, J. Martens, and G. E. Hinton. Generating text with recurrent neural networks. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 1017–1024, 2011.

[41] L. M. Winter and K. Balasubramaniam. Using the maximum x-ray flux ratio and x-ray background to predict solar flare class. *Space Weather*, 13(5):286–297, 2015.

[42] J.-G. Wu and H. Lundstedt. Geomagnetic storm predictions from solar wind data with the use of dynamic neural networks. *Journal of Geophysical Research: Space Physics*, 102(A7):14255–14268, 1997.

[43] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057, 2015.

[44] A. Yona, T. Senjyu, T. FunabaShi, et al. Application of neural network to one-dayahead 24 hours generating power forecasting for photovoltaic system [j]. *Intelligent Systems Applications to Power Systems*, 2007.

[45] M. E. Yousif. The solar flare mechanism.

[46] J. Zeng and W. Qiao. Short-term solar power prediction using a support vector machine. *Renewable Energy*, 52:118–127, 2013.

Curriculum Vitae

Sumi Dey was born on February 20, 1988. She is one of the daughters of Nepal Chandra Dey and Mina Rani Dey. She graduated from The University of Dhaka, Bangladesh, in 2009 with a Bachelor of Science in Mathematics. In the Spring of 2014, she got herself admitted into the Graduate School of The University of Texas at El Paso. While pursuing a master's degree in Mathematics she worked as a Teaching Assistant at the Mathematical Sciences department. She graduated in Fall 2016 with a Master's of Science in Mathematics. Then she entered into the Computational Science Program to pursue her PhD.

Email address : sdey2@miners.utep.edu