Neural Decision Tree: A New Tool for Building Forecast Models for Plasmasphere Dynamics

Yu Lu[1*], Irina S. Zhelavskaya[2], Chunming Wang[1]

[1]Department of Mathematics, University of Southern California Los Angeles, CA 90089, USA.
[2]Helmholtz Centre Potsdam, GFZ German Research Centre for Geosciences, Potsdam, Germany

Key Points:

- Neural Decision Tree is an effective tool for building space weather forecast models.
- More elaborate neural network structure initialized using NDT can provide higher performance and training efficiency.
- Physics-based model constraints with statistical assumptions can have significant impact on models built through machine-learning techniques.

*Sponsorship of the Living With a Star Targeted Research and Technology NASA/NSF Partnership for Collaborative Space Weather Modeling is gratefully acknowledged. Portions of the research for this paper were performed at the Jet Propulsion Laboratory, California Institute of Technology under contract with NASA.

Corresponding author: Yu Lu, leoluyu@gmail.com

Abstract

The Neural Decision Tree (NDT) is a hybrid supervised machine-learning algorithm that combines the self-limiting property of a decision tree (CART) algorithm with the artificial neural network (ANN). We demonstrate the use of NDT for a regression problem of building a prediction model for the plasmasphere electron density with solar and geomagnetic measurements as inputs. Our work replicates the work by Zhelavskaya et al. reported in their 2017 article (I. S. Zhelavskaya, 2017) to show that NDT makes available sophisticated network layout for building a predictive model, thus taking advantage of *deep-learning potential* of the neural network. We also demonstrate that with the ability to automatically select an appropriate network layout, as well as, effective initialization, the NDT algorithm allows research scientists in space weather to focus more of their attention on physically and statistically relevant aspects of using machine-learning techniques. In fact, our example highlights the facts that the basic assumptions of standard supervise machine-learning problems are often unsatisfied in real-world space weather applications. Greater attention to these fundamental issues may create significantly different solutions to space weather forecast problems.

1 Introduction

The fascination for machine-learning technology has taken the space weather community, as well as, the geophysical scientific community in general by storm (Camporeale, 2019), (Chantry, 2021). A large number of astonishing and impressive performance of models supported by machine-learning technology have been reported in conferences and journal publications (I. S. Zhelavskaya, 2017), (Huntingford et al., 2019), (Reichstein et al., 2019), (Grönquist et al., 2021), and (Kashinath et al., 2021). One of the attractive aspects of machine learning techniques is the wide applicability of their framework. In particular, the basic concept of supervised learning in which a collection of paired input variables and desired outputs is used as training data to derive a predictor for the output variables from the new input values is widely applicable. However, behind the easy accessibility of these techniques are the complex construction of generic models and deep mathematical rationale to support the statistical validity of the model as a product of the training process. The widely used artificial neural network (ANN) is a perfect example for illustrating the challenges of adopting general machine learning techniques for geophysics and space weather applications.

As most people who have attempted to use ANN as a basic forecast model know, the usually already challenging task of deciding which of the available observable quantities a forecast should depend on becomes even more complex when the answer may also be linked to which structure of ANN one chooses to use. In fact, the more variables we include as inputs to a model, the more complex an ANN tends to be. Since training of an ANN is essentially a high dimensional non-convex optimization process, we often run into the "curse of dimensionality" in which the space of parameters defining a model is so vast that the search for an optimal solution becomes illusive. The increased complexity of a model also needs a proportionally increasing volume of training data for its calibration, thus compounding the difficulty for model development. In areas that have adopted machine learning techniques as dominant approaches for model development, such as image, handwriting, and voice recognition, considerable experiential knowledge often provides valuable guidelines for the structure and size of the ANN needed for a new application. This is not the case in most geophysics research areas in general and in the space weather community,specifically. Due to the vast diversity of applications, it is also unlikely that widely applicable guidelines can be developed in the near future.

Emerging techniques in the machine learning community have begun to offer solutions to model structural selection. One example of these techniques is the Neural Decision Tree (NDT) (Biau et al., 2018), (Lu & Wang, 2020). Unlike an ANN, a decision

tree is grown by partitioning training data into subsets according to the criterion that intends to minimize overall information uncertainty entropy or simply the non-homogeneity in the subsets. Although a commonly used decision tree algorithm selects splitting criteria according to a single component of the vector of input parameters, the technique has shown to usually offer good partitions of the space of parameters to substantially facilitate regression modeling. The decision tree's growth strategy provides a self-limiting characteristic that can provide a high-level assessment of the complexity of a problem. Once a decision tree establishes a preliminary partition of training data, an algorithm is developed to map a decision tree to a multi-layer neural network. The newly structured and initialized ANN is then iteratively optimized. This hybrid approach, referred to as Neural-Decision Tree, has been demonstrated in many benchmark AI classification applications to provide significantly superior performance than ad hoc selection of network structure with randomized initialization of weights (Lu & Wang, 2020).

Our research reported in this paper represents our first attempt to use NDT for a regression problem for space weather applications. Unlike classification problems in which the model outputs are integers representing the categories that a data point should belong to, the outputs of a regression problem tend to be real-valued variables continuously dependent on input parameters. Indeed, as shown in (I. S. Zhelavskaya, 2017), the purpose of a plasmasphere dynamic model is to predict electron density distribution in the Earth's plasmasphere at a given time based on available measurements of solar and geomagnetic activities. As explained in (I. S. Zhelavskaya, 2017), a 2-dimension density field in a sun-fixed plane can adequately represent a 3-dimensional density field. Computational experiments have led Zhelavskaya et al. to select an effective ANN model that can reproduce plasmasphere density for various historically known conditions. Indeed, the ultimately successful model was identified through a process of essentially trial-and-errors. Our collaboration stems from a desire to evaluate the capability of NDT in shortening the process of discovery of promising model structures. In particular, we are interested in investigating the following issues:

- Can NDT automatically discover an ANN with comparable or less complexity as those found in (I. S. Zhelavskaya, 2017) that delivers similar performance in prediction?
- Can NDT provide any computational advantage in terms of convergence rate in the training process?
- Since a NDT is inherently multi-layer, do multiple hidden layers offer a substantial improvement over a single hidden layer ANN?

Our research has shown positive answers to all the above questions. Moreover, by focusing our attention on more physically relevant issues and basic mathematical frameworks for regression problems, we are able to produce more physically coherent and statistically meaningful models. We believe that our results demonstrate that NDT is a beneficial machine-learning technique specifically for new space weather forecast applications.

In this manuscript, we shall present the basic construct of a NDT and the statistical consistency theorem for the resulting ANN in Section 2. We shall compare the performance of NDT in terms of model complexity, prediction error RMSE reduction, and convergence rate in model training in Section 3. As we have indicated previously, the streamlining of the process of structuring an effective NDT allowed us to focus on more high-level issues related to the prediction model. In Section 4 we present our efforts to incorporate additional physical and statitsical considerations in the generation of predictive plasmssphere models. In the concluding Section 5 we shall provide further discussions on NDT and potential benefits that it can offer to the space weather forecast community.

2 Construction and Theoretical Framework of Neural Decision Tree

Broadly speaking, machine learning (ML) is a set of methods that can systematically detect patterns in data and then use the uncovered patterns to make inference for future data or to support other decision-making in the presence of uncertainties (Murphy, 2012). The most widely formulated applications for ML are in the form of *supervised learning* problems. The goal is to establish a mapping from input x to output y. Two of the most commonly used supervised learning techniques are Decision Tree by Classification and Regression Tree (CART) and Artificial Neural Network (ANN).

2.1 CART and ANN

A Decision Tree models the output y by first partitioning the d-dimensional feature space for x into disjoint subsets and then fitting a simple function between x and y within each subset. For a regression problem, CART fits an average model within each subset. The evaluation criterion of a tree split is based on the mean square error (MSE) reduction in y as the following:

$$\Delta_{MSE} = \frac{N_p}{N} MSE(parent) - \frac{N_l}{N} MSE(left\_child) - \frac{N_r}{N} MSE(right\_child),$$

where N_p, N_l, N_r, and N are the number of data in parent, left child, right child, and the entire training set respectively. The CART is then constructed by iteratively selecting the most discriminating attribute x_j and value b to partition a parent set into left-child subset ($x_j < b$) and right-child subset ($x_j \geq b$). The selection of x_j and b in each partition is based on a greedy algorithm yielding the largest MSE reduction. consequently, the decision tree provides a sub-optimal partition of the feature space. The growth of a decision tree is self-limited by a threshold for the minimal MSE reduction for each partition. Additionally, setting maximum tree depth can also effectively avoid over-complex trees. Indeed, an excessively complex tree does not perform well when tested with data that is not part of training data.

On the other hand, an ANN models the output y by applying a non-linear activation function to a linear combination of the outputs of the previous layer, starting with the input x as the outputs of the zero-th layer or input-layer. Initial weight parameters in the linear combination are typically randomly selected. Optimization of the weights is carried out by iterative gradient-based optimization methods.

A single tree node can be treated as a single network neuron with an indicator activation function. To compare a neuron and a tree node, let s represent an elementary neuron with input $x \in \mathbb{R}^d$:

$$s(x) = a(w^T x - b), \qquad w \in \mathbb{R}^n, b \in \mathbb{R}, \tag{1}$$

where $a : \mathbb{R} \mapsto [0, 1]$ is referred to as an activation function. When $a = \mathbb{I}$ is the indicator function for non-negative real numbers, the function s can be rewritten as

$$s(x) = \begin{cases} 1 & w^T x - b \geq 0, \\ 0 & w^T x - b < 0. \end{cases}$$

As a result, the neuron s essentially creates a partition of $\mathbb{R}^d$ by the hyperplane $w^T x - b = 0$ into two subsets $S_1 = s^{-1}(1)$ and $S_0 = s^{-1}(0)$. The action of a decision node in a binary tree is indeed a such partition as well, except that a common decision tree partitions the feature space according to the value of a single component x_j of feature vector x. Thus, by taking $w = e_j$, the partitions created by s have the form

$$S_1 = \{x \in \mathbb{R}^d, x_j \geq b\}, \quad S_0 = \{x \in \mathbb{R}^d, x_j < b\}.$$

Consequently, by representing every decision node in a binary tree with an elementary neuron of the above form, it is possible to determine from the outputs of these neurons which leaf node an input vector x should be placed in. Since each leaf is assigned

with a node average in a regression tree, it is therefore possible to reproduce the outcome of a regression tree exactly using a neural network in which activation functions are all indicator function.

2.2 Construction of the NDT

Once a decision tree is obtained by applying the CART algorithm on training data, the transition to a NDT requires two steps:

1. We construct a neural network (NN) using the step function $\mathbb{I}(x) = 1$ for $x > 0$ and $\mathbb{I}(x) = 0$ for $x \leq 0$, as activation function to replicate the input/output relationship of a decision tree and provide initial weights for the NDT.
2. We relax activation functions at various layers with strategically selected "smoother" activation function to relax the decision boundary from trees.

As a result, a typical NDT has two hidden layers that represent the set of decision and terminal nodes of the decision tree, respectively. We will denote the input $\mathbf{x} \in \mathbb{R}^{1\times d}$ as a row vector for notation simplicity in this section. Consider a standard binary tree T with K decision nodes. At a decision node j, the decision for splitting has the form $\mathbf{x}_{q(j)} < d_j$ where $\mathbf{x}_{q(j)}$ denote the $q(j)$'s attribute of the input $\mathbf{x}$. As a binary tree, T has $K+1$ leaves, and each leaf is assigned one single regression output.

The first hidden layer, $\mathbf{h}$, is constructed to replicate the set of decision nodes in T. Hence, $\mathbf{h} \in \mathbb{R}^{1\times K}$ contains K number of neurons. Given an input $\mathbf{x} \in \mathbb{R}^{1\times d}$ as a row vector, let $h_j = \mathbb{I}(\mathbf{x}W_j^{(1)} + b_j^{(1)})$ be the j^{th} neuron of $\mathbf{h}$. The initial weight vector $W_j^{(1)} \in \mathbb{R}^d$ and a offset $b_j^{(1)} \in \mathbb{R}$ for $j = 1 \dots K_n$ will be selected such that the output of the neuron equals to one when the criterion for the split of decision node j is verified, and zero otherwise. Note that the real-valued indicator function is

$$\mathbb{I}(t) = \begin{cases} 1 & \text{if } t > 0, \\ 0 & \text{otherwise,} \end{cases}$$

and the vector-valued function $\mathbb{I} : \mathbb{R}^m \mapsto \mathbb{R}^m, \mathbb{I}(\mathbf{t})$ is generalized by element-wise operation, i.e., $[\mathbb{I}(\mathbf{t})]_i = \mathbb{I}(t_i)$. For the splitting criterion $x_{q(j)} < d_j$ of the decision node j, the weight vector $W_j^{(1)}$ and the offset $b_j^{(1)}$ are initialized as the following:

$$W_{i,j}^{(1)} = \begin{cases} -1 & \text{if } i = q(j), \\ 0 & \text{otherwise} \end{cases}$$
$$b_j^{(1)} = d_j$$

for $i = 1 \dots d$. Hence, given any input $\mathbf{x}$, the output

$$\mathbf{h} = \mathbb{I}(\mathbf{x}W^{(1)} + b^{(1)})$$

is a binary $0, 1$ vector that represents the splitting results of the tree T where

$$\mathbf{W}^{(1)} = [W_1^{(1)}, \cdots, W_K^{(1)}] \in \mathbb{R}^{d\times K}, \quad \mathbf{b}^{(1)} = [b_1^{(1)}, \cdots, b_K^{(1)}] \in \mathbb{R}^{1\times K}.$$

The output of the second hidden layer $\mathbf{r} \in \mathbb{R}^{K+1}$ is designed as a binary vector with $K+1$ entries representing the $K+1$ leaves in a binary tree with K decision nodes. The j-th entry of $\mathbf{r}$, $r_j(x) = 1$ if the input x should be in the partition represented by the j-th leaves. It is important to note that each value of the binary vector $\mathbf{h}$ uniquely identifies a leaf on the tree. Thus for each neuron $r_j = \mathbb{I}(\mathbf{h}W_j^{(2)} + b_j^{(2)})$, the initial weights $W_j^{(2)} \in \mathbb{R}^K$ and offsets $b_j^{(2)} \in \mathbb{R}$ for $j = 1, \dots, K+1$ are defined such that when the value of input binary vector is associated with leaf j, the neuron produces an output of

one, and zero otherwise. Let $P_j \subset \{0,1\}^K$ denote the set of all possible binary vectors from the first layer that is associated with leaf node j. If for all vectors $p \in P_j$ the i-th component $p_i = 1$, then the criterion for the i-th decision must be verified for leaf j. Similarly, if for all vectors $p \in P_j$, $p_i = 0$ the criterion for the i-th decision must be false. On the other hand if for some $p \in P_j$, $p_i = 0$ and for some other $p \in P_j$, $p_i = 1$ then the i-th decision does not determine the adherence of input x to leaf j. The weights $W_j^{(2)}$ and offsets $b_j^{(2)}$ are given by: for $i = 1 \dots K$

$$W_{i,j}^{(2)} = \begin{cases} 1 & \text{if } p_i = 1, \quad \forall p \in P_j, \\ -1 & \text{if } p_i = 0, \quad \forall p \in P_j, \\ 0 & \text{if } p_i \text{ can be either 0 or 1 } \forall p \in P_j. \end{cases}$$

$$b_j^{(2)} = -\Big[\sum_{\{i:W_{i,j}^{(2)}=1\}} 1 \Big] + \frac{1}{2}.$$

Hence, the output of the second layer

$$\mathbf{r} = \mathbb{I}(\mathbf{h}W^{(2)} + b^{(2)})$$

is also a binary vector with only a single component equals to 1 which, for a given input $\mathbf{x}$, indicates that it belongs to the designated partition of T.

The intuition of such initialization is the following: if an input $\mathbf{x}$ belongs leaf node j in T, then

$$\begin{aligned} \mathbf{h}W_j^{(2)} &= \sum_{\{i:W_{ij}^{(2)}=1\}} 1 \\ \mathbf{h}W_j^{(2)} + b_j^{(2)} &= \sum_{\{i:W_{ij}^{(2)}=1\}} 1 + b_j^{(2)} \\ &= \frac{1}{2}. \end{aligned}$$

Otherwise, $\mathbf{h}W_j^{(2)} + b_j^{(2)} < -\frac{1}{2}$. Consequently, an indicator activation yields

$$\mathbb{I}(\mathbf{h}W_j^{(2)} + b_j^{(2)}) = \begin{cases} \mathbb{I}(1/2) & = 1, \text{ if } x \text{ belongs leaf j} \\ \mathbb{I}(-1/2) & = 0, \text{ if } x \text{ does not belong leaf j.} \end{cases}$$

The output layer has a single neuron for the regression problem, and it represent the final output from the tree T. The neuron will select the regression output from the associate leaf node. Let $\{C_1, \dots C_{K+1}\}$ be the regression output for leaf node $\{1, \dots K+1\}$ and $W^{(3)} \in \mathbb{R}^{(K+1)\times 1}, b^{(3)} \in \mathbb{R}$ be the weight and offsets from the layer $\mathbf{r}$ to the output layer. The initialization of $W^{(3)}$ and $b^{(3)}$ are given by

$$\begin{aligned} W_j^{(3)} &= C_j \\ b^{(3)} &= 0 \end{aligned}$$

for $j = 1, \dots K+1$. At last, the neural network output is

$$y^{(3)} = \mathbf{r}W^{(3)} + b^{(3)}.$$

Essentially, the NDT here is the regression version of the NDT in (Lu & Wang, 2020). The main purpose of initializing an ANN with a decision tree is that the partition of the feature space created by CART offers a rough approximation of the level sets of the true

classifier. However, the restrictive use by CART of only hyperplanes perpendicular to axes of the feature space is unlikely to be optimal for an efficient approximation.

In order to enable optimization techniques such as stochastic gradient descent (SGD) to train the ANN that initialized with a decision tree, we replace the indicator $\mathbb{I}(x)$ by a smooth (differentiable almost everywhere) activation function $\sigma(x)$ in the second step of constructing a NDT. The selection of activation functions can have a significant impact on the performance of the final NDT. Our experience indicates that the lacking of a strategic selection of activation functions, a NDT may gain significantly fewer advantages from the CART initialization compare to an arbitrarily constructed ANN.

From the input $\mathbf{x}$ to the first hidden layer $\mathbf{h}$, our experience suggests the use of bounded Rectified Linear (ReL) activation function

$$\sigma_1(x) = \min(\max(0, x), 1) \tag{2}$$

where $\sigma_1(x)$ is the activation for $h_j = \sigma_1(\mathbf{x}W_j^{(1)} + b_j^{(1)})$. This selection ensures that $\sigma_1(x)$ has a strict 0 as a lower bound. The upper bound of 1 also yield clear indication of whether the input x belongs to the left or right child. Therefore, $\mathbf{h}$ partially preserves the splitting criterion of the decision tree. For second layer $r_j = \sigma_2(\mathbf{h}W_j^{(2)} + b_j^{(2)})$, we suggest using the standard logistic function $\sigma_2(x) = \frac{1}{1+e^{-x}}$. Because the second layer corresponds the leaf node that represent the rigid decision boundary of CART, having a "smoother"(differentiable everywhere) logistic function can effectively optimize the decision boundary. At last, the output layer is given by $y_j^{(3)} = \mathbf{r}W_j^{(3)} + b_j^{(3)}$.

2.3 Statistical Consistency of the NDT

An essential characteristic of a desirable algorithm is the convergence of the optimally constructed regression map toward the 'true' regression map as the volume of training data, and the degree of freedom of the regression map tend toward infinity. Algorithms with these characteristics are referred to as statistically consistent. (Lu & Wang, 2020) provides proof for the consistency theorem for binary classification, which can be easily generalized to multi-classification. One significant difference between a regression problem and a classification problem is that there is not necessarily a lower and an upper bound for the output y of a regression problem. Preliminary data processing and transformation is often required to map the application-specific output y to an output vector $\hat{y}$ that only takes value in a bounded interval. In general, we assume the processed output will be bounded by the constant 1. We shall state our main consistency theorem below.

Theorem 2.1 (Main Result: Strongly Universal Consistency of m_n) *Let $(X, Y) \in \mathbb{R}^d \times [-1, 1]$ be a random vector with joint probability density function $\mu_{X,Y}$. We denote the minimum variance regression map by $m(x) = E(Y|X = x)$ which is considered the 'true' regression map. Let $(X_1, Y_1), \ldots, (X_n, Y_n)$ be i.i.d. samples of (X, Y) and $D_n = \{(X_i, Y_i)\}_{i=1}^n$. Let $\mathcal{F}_n$ be the class of neural networks defined above, and let m_n be the empirical L_2 loss minimizer in $\mathcal{F}_n$ that depends on D_n. If*

$$\frac{K_n^2 \log(K_n^4)}{n} \to 0$$

and $\inf_{f \in \mathcal{F}_n} \mathbb{E}_X(f(X) - m(X))^2 \to 0$ as $n \to \infty$, then for any distribution for (X, Y),

$$E_X(m_n(X) - m(X)) = \int |m_n(x) - m(x)|^2 \mu_X(dx) \to 0 \ \ a.s.$$

3 Developing a Regression Based Neural Decision Tree Model for Forecasting Plasmasphere Dynamics

The initial goal of our work is to evaluate the NDT's ability to produce an ANN model with comparable performance to the PINE model reported in (I. S. Zhelavskaya, 2017) with minimal manual adjustment. Unlike PINE which is a single hidden layer neural network, ANN models generated by the NDT algorithm always have at least two hidden layers which is structurally more complex.

As reported in (I. S. Zhelavskaya, 2017), the plasmasphere electron density used to train PINE is derived from the upper hybrid frequency, which is retrieved from measurements by the Electric and Magnetic Field Instrument Suite and Integrated Science Instrumentation Suite (EMFISIS) on the Van Allen Probes satellites using the Neural-network-based Upper hybrid Resonance Determination (NURD) algorithm (I. Zhelavskaya, 2016). The input variables for the models are selected through repeated experimentation by Zhelavskaya et al. to include recent time-history of solar and geomagnetic parameters originally obtained from NASA's OmniWeb data service. Table 1 below shows a complete list of attributes for the model inputs X.

Table 1: Attributes in the input for PINE and NDT models for plasmasphere dynamics

Row Index	**Name**	**Time Stamp**
1	AE	Current
2	kp	Current
3	SymH	Current
4	F107	Current
5	L	Altitude in a.u.
6	MLT	Magnetic local time
7-12	AE avg	Averages for AE over previous 3,6,12,24,36,48 hours
13-18	kp avg	Averages for kp over previous 3,6,12,24,36,48 hours
19-24	SymH avg	Averages for SymH over previous 3,6,12,24,36,48 hours
25-30	F10.7 avg	Averages for F10.7 over previous 3,6,12,24,36,48 hours

It is helpful to note that plasma density data are retrieved along the spacecrafts' orbit over time; therefore, the sampling in spatial variables L and MLT are entirely dependent on the trajectory of the Van Allen Probes. The sampling frequency for the rest of the input variables varies from 3 hours to one second. The moving averaged values over intervals of different lengths help to provide stability of the model. While the training data consists of an extensive collection of matched pairs X_i, y_i where y_i is the plasmasphere electron density at a specific location given by (L_i, MLT_i), the actual utility of the resulting model for predicting the plasmasphere dynamics is to produce the entire electron density field over the Earth equatorial plane for a given set of solar and geomagnetic data X. This constitutes an extension of the traditional supervised learning paradigm in the sense that for each input vector X, the actual intended output is a 2-dimensional scalar field. However, the training data available to us consists of point-wise values of the desired field at different times. An analogy in the context of image recognition would be trying to determine if an image is that of a dog when instead of given the entire image, we have only one single pixel of the image at a given time. This extension substantially increases the challenge for model training. Consequently, there are essential features for the desired output field that are not explicitly represented by the data. We shall discuss these additional properties in the next section. In this section, we focus our attention on constructing a regression model using NDT that can accurately

reproduce the plasmasphere electron density at discrete points. In particular, we would like to attempt to answer the following questions:

1. Can a NDT-initiated neural network with similar model complexity automatically produce the performance in terms of prediction least square error similar to PINE?
2. Does NDT provide substantially favorable initialization that the convergence rate for the training process is accelerated compared with a randomly initiated network as seen in (Lu & Wang, 2020)?
3. Does NDT initiate neural network deliver robustness in optimization similar to what we have seen for other problems (Lu & Wang, 2020)?
4. Can NDT-initiated neural networks with reduced model complexity produce comparable performance in terms of prediction error?

Before presenting the NDT's construction of plasmasphere dynamics models, it is helpful to provide a brief description of our use of the data set prepared by Zhelavskaya and her colleagues. As mentioned previously, the total data set available consists of matched pairs of solar and geomagnetic measurements to plasmasphere electron density at a specific altitude L and geomagnetic local time MLT covering the time period from October 1st, 2012 to May 12th, 2016. In the training and model selection work by Zhelavskaya and her colleagues, this data set is partitioned into training $\mathcal{T}$ and testing or validation subsets $\mathcal{V}$ with a ratio of 9 to 1 in data volume by randomized sampling without repetition. To simplify the direct comparison of model performance, we use the equivalent partitions as Zhelavskaya et al. in all comparisons of RMSE among the models.

In selecting a suitable network structure for PINE, Zhelavskaya et al. consider single-hidden layer neural networks with {23,30,38,45,53} neurons as candidates structures. To decide on an appropriate size for the network, they have used the approach of 5 fold cross-validation to select a structure with the lowest RMSE. That is, by partitioning the training subset $\mathcal{T}$ into 5 equal-sized subsets and using any 4 of them for model training and the remaining one for measuring RMSE performance. The average of the 5 RMSE values represents the performance for the specific size of the neural network. It is worth reminding us that since all training of neural network for PINE follow the typical approach of random initialization of the weights defining a network, a single model evaluation involves two sources of randomization: selection of data making the 4-subsets of the 5 folder cross-validation and the randomization of the initial weights. As a result, a meaningful assessment of the performance of a network structure also involves a repeated training process for each training-validation step in the 5 fold cross-validation to average out the effect of randomized initialization. The enormous computational efforts required to select suitable models among candidate designs render consideration of more elaborate network structures prohibitively expensive. Indeed, with just 5 candidate model structures and m randomized initialization for each step in the 5 fold cross-validation, a total of $25m$ model training and validation process is required. If the approach is to be extended to two hidden layers structures, the combinatorial explosion of candidates will make the selection nearly impossible computationally.

As presented in Section 2, NDT selects the network architecture and the initial weights for neurons based on the decision tree, which is created through preliminary processing of training data. This removes the need for repeated training to average out the effect of random initialization as was the case in a common neural network evaluation. Moreover, a single criterion on either the minimum threshold for RMSE reduction when creating a new decision node in the tree or the maximum number of nodes required automatically allows the construction process of the NDT to select a promising network layout involving two hidden layers with appropriate initial weights for the neurons. In fact, since the construction of CART is relatively insensitive to the volume of data used as shown in (Lu & Wang, 2020), it allows us to bypass the cross-validation step in establishing a reliable and representative performance measure for a given network structure.

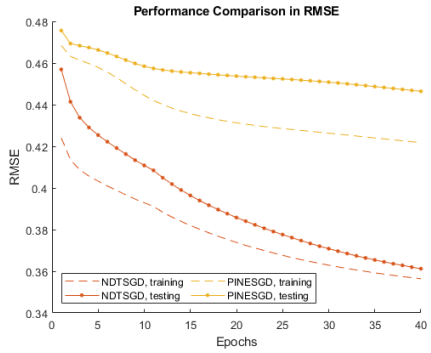
As a result, in this section, all performance comparisons between the final selection for the PINE model with 45 neurons and models created by the NDT algorithm are derived using the entire subset $\mathcal{T}$ for training and evaluated on the subset $\mathcal{V}$. Table 2 compares model selection approaches for NDT and PINE.

Table 2: Approaches for model selection for PINE and for NDT based approach

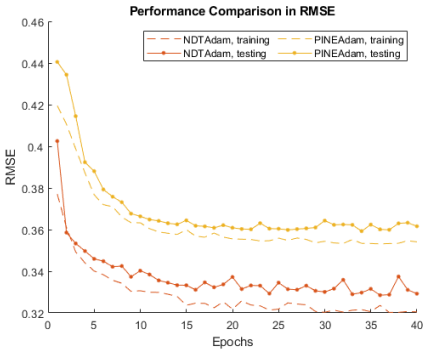
	NDT	PINE
Candidate architectures	2 hidden-layers	1 layer
Network initialization	Replicating CART	Random weights
Scoring RMSE	Training $\mathcal{T}$ and validation $\mathcal{V}$	5-fold cross-validation using $\mathcal{T}$ Training $\mathcal{T}$ and validation $\mathcal{V}$

The most popular optimization algorithm for training a neural network is the Stochastic Gradient Descent (SGD) method, for which the gradient with respect to the weight vector of the performance of a single data point or a small patch of data points is evaluated using the highly efficient backward propagation algorithm. The weights are then updated by a small fixed fraction, often referred to as a step, in the negative direction of the gradient vector. SGD is particularly attractive for applications involving continuous learning when incremental data availability allows continuous improvement of a model. As a first-order optimization technique, SGD does have a tendency, in some cases, to be slow in final convergence to a local minimum. In these situations, quadratic quasi-Newton methods such as Levenberg-Marquardt (LM) often provide improved convergence. However, the price for 'faster' convergence in terms of the number of iterations is often much more computationally intensive iterations. As a result, LM algorithm-based training typically uses gradient evaluation on the model performance over the entire training data set. Our experiments indicate that NDT-created neural networks can achieve significantly faster convergence than the network structure used by PINE with randomized weights when the SGD algorithm is used. As shown in Figures 1a and 1b, the decrease in RMSE is much faster during the training for NDT than for PINE. In these experiments, the entirety of the nearly 3 million training data points is group into 293 patches for 10,000 data points each for a SGD update of weights defining a network. When all 293 have been used once, the optimization process is said to reaches one epoch. The patches are then being reused in a new epoch of the training process. At the end of each epoch, the RMSE is evaluated on the entire training data set $\mathcal{T}$ and validation data set $\mathcal{V}$. From Figures 1a and 1b we observe that not only a much faster reduction of RMSE for NDT as the training progress than that for PINE, the rate of decrease also shows a smoother approach in Figure 1a to a local minimum without the intermediate slowing down as seen for PINE.

In the previous efforts by Zhelavskaya et al., it was found that the LM optimization technique was necessary to deliver slightly lower RMSE for both training and validation. Our experiments also confirm their observation. However, a common variant of the SGD method, Adaptive Moment Estimation (Kingma & Ba, 2014), often referred to as Adam algorithm with similar efficiency as SGD method, can produce near-identical performance in terms of final RMSE level as LM algorithm as shown in Table 3 below. As we can see in Table 3, using the Adam algorithm, the RMSE level for NDT is nearly identical to that of PINE when trained with the LM algorithm, although LM seems able to reduce RMSE of NDT to an even lower level for both the training and validation data. A relevant question is whether or not these minuscule differences have any significance



(a) Changes in the sum of RMSE as functions of iteration number during the training of PINE and NDT in SGD.



(b) Changes in the sum of RMSE as functions of iteration number during the training of PINE and NDT in Adam

Figure 1: Comparison of Rate of reduction of RMSE for NDT and PINE using first order gradient descent type of optimization methods.

statistically or in terms of model prediction accuracy. We shall attempt to address this issue later.

Table 3: Robust Optimizer

	NDT			PINE	
Optimization Algorithm	SGD	Adam	LM	SGD	LM
Training RMSE	0.3226	0.3158	0.3043	0.3649	0.3145
Testing RMSE	0.3316	0.3282	0.3204	0.3811	0.3282

The NDT model used in the comparison shown in Table 3 above is a model for which we limited the total number of decision nodes in CART to 25 so that the overall dimension of the weight vector for the resulting NDT is nearly identical to the PINE model with 45 neurons. We have also experimented in NDT models with a much lower degree of freedom involving a much smaller number of neurons in the network. Indeed, as shown in Table 4, compared with the default NDT initiated by a CART with 25 decision nodes, CARTs with 15 or 10 decision nodes initialize the NDT to produce comparable or even lower RMSE levels when optimized with the LM algorithm.

Since the ultimate goal of our work is to produce a predictive model for plasmasphere dynamics, or more concretely, generate electron density field on the equatorial plane for a given solar and magnetic condition specified by the input vector X, we plotted in Figure 2 the predicted electron density field for all four models listed in Table 4 for a time period of known plasmasphere storm from June 26 to June 27, 2001. As we can see in Figure 2 the difference in the model predictions are pretty minuscule consistent with their RMSE performance despite their substantial difference in model complexity. However, the computation intensity in training these models can be vastly different as illustrate in Table 5 below. As we can see, the time required for training a model with

Table 4: Comparison of final RMSE for different NDT constructed models and PINE.

	NDT			PINE
# of Decision nodes for NDT	25	15	10	
Dimension of weight params	1478	738	443	1441
Fraction to dimension of PINE	100%	50%	30 %	100%
Training RMSE	0.3043	0.3081	0.3198	0.3145
Testing RMSE	0.3204	0.3162	0.3256	0.3282

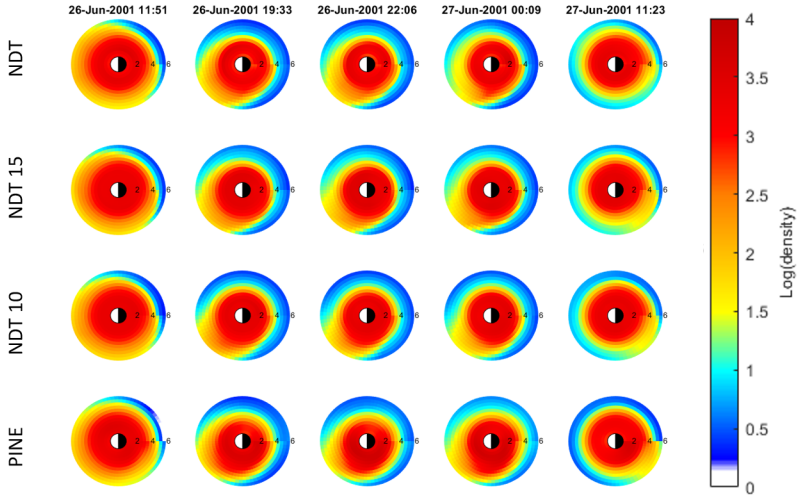


Figure 2: Model predictions of electron density field for June 26-27, 2001 storm.

a higher number of weights can be an order of magnitude longer than one that requires a fraction number of weights. In addition to being more robust and stable, models with fewer parameters tend to have much higher information content measured by AIC or BIC indices. The fast training process also allows us to explore other critical issues relevant for developing a regression-based model as we shall discuss in the next section. Our experimental results demonstrate that the NDT algorithm can deliver high-performance regression neural network models through inherently sophisticated multiple hidden layer structures.

Table 5: The models and training algorithms are select for similar final RMSE performance. The times are measured on a personal computer wit a Intel® Core™ i7-4790 Processor CPU, a NVIDIA GeForce GTX 970 GPU and a total of 32 GB ROM.

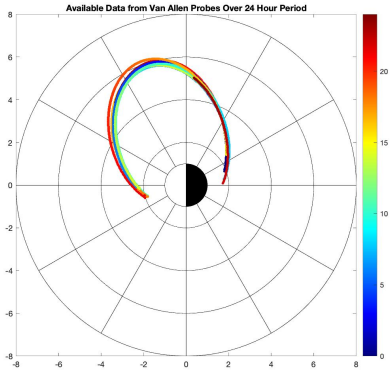
	NDT				PINE
# of nodes for NDT	25	25	15	10	
Optimization Algorithm	Adam	LM	LM	LM	LM
Train/Test RMSE	0.32/0.33	0.30/0.32	0.31/0.32	0.32/0.33	0.31/0.33
Time (minutes)	13.13	162.16	28.76	**11.91**	158.06

4 A Broader View of the Task of Modeling Plasmasphere Dynamics

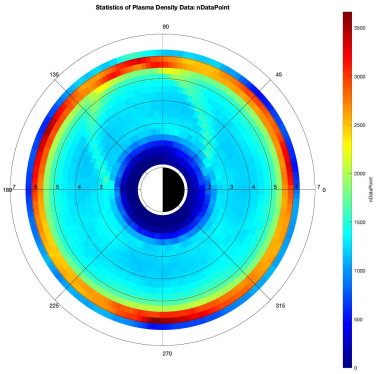
As we have indicated, at the beginning of Section 3, that the construction of a plasmasphere dynamics model based on the type of data available to us is particularly challenging. Unlike most supervised learning applications, for each solar and magnetic condition, our training data is not the ultimate model response which should be the electron density field in the Earth's equatorial plane. Instead, each data point merely provides the density at a specific point in the plasmasphere. Since data are collected along the orbit of Van Allen Probes, the amount of data available over a 24 hour time period covers only a small fraction of the space in the plasmasphere as shown in Figure 3a. It would take several months worth of data to cover a significant portion of the plasmasphere. The underlying values for the solar and magnetic conditions can undergo substantial changes over this period of time. Consequently, the problem of obtaining a predictive model of electron density distribution for plasmasphere using solar and magnetic field observation is extremely challenging and even seemly unrealistic. We will give more discussions on this aspect of the model in the next section. We also note that the spatial distribution of data is highly non-uniform as shown in Figure 3b. This is, of course, a result of the orbit for the Van Allen Probes where the orbit reaches its highest point and tangential to the circle at $L = 6$ on the equatorial plane. Consequently, a much larger number of training data is available at altitude $L = 6$. A closer examination of the preliminary descriptive statistical analysis of the available data shows both the average and empirical standard deviation of electron density are systematically spatially dependent (Figure 4a and 4b). (Figure 4a and 4b).

We recall the fundamental assumptions that leads to statistical consistency of the regression analyses include the following:

1. Residual errors in data points are independent and identically distributed. Thus, the least square regression leads to the optical estimation of the mean electron density.

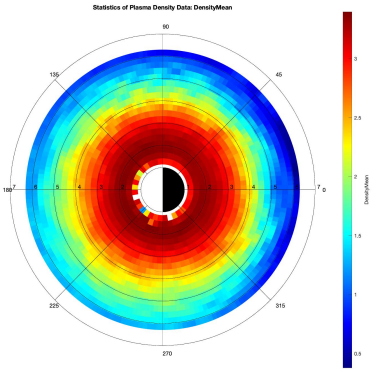


(a) Data available for a period of 24 hours from Van Allen Probes

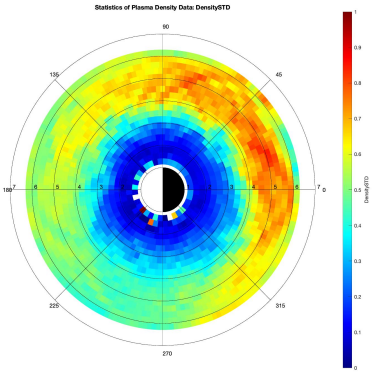


(b) Spatial Distribution of Data

Figure 3: Since Van Allen Probes collect data along their orbits, instantaneous global imaging of the plasmasphere density field is obviously unavailable, and spatially non-uniform distribution of the data is inherent to the measurement approach.



(a) Average electron density in the plasmasphere



(b) Empirical standard deviation of electron density in the plasmasphere

Figure 4: Local statistics of training data shows distinct spatial variability in both average and standard deviation of electron density.

2. The distribution of training data should reflect the distribution of conditions that require prediction. Since the true goal of our prediction is the electron density on the entire equatorial plane at a given time, ideally, the data points should be uniformly distributed. Moreover, the electron density of all points on the equatorial plane is clearly not identically distributed. Indeed, the density at lower altitude is substantially higher than high altitude region as shown in Figure 4a.

Another property inherent in our understanding of physics is that electron density should be spatially continuous. However, when spatial coordinates L and MLT are used, the spatial input data are defined over a rectangular area of $[0, 6]\times[0, 24]$. As far as the training algorithm is concerned, no information is indicating at the boundary at $MLT = 0$ and $MLT = 24$ are actually the same spatial point. On the other hand, a transformation to Cartesian coordinate $x_m = L\cos 2\pi \cdot (MLT/24)$, and $y_m = \sin 2\pi \cdot (MLT/24)$ would explicitly guarantee the continuity across the boundary at $MLT = 0$. Naturally, when training data volume is large and densely covers all areas of the space for input variables, the optimal regression predictor would generally produce a spatial continuous electron density field. However, data from the Van Allen Probes are not sufficiently dense near the region where $MLT = 0$. As a result, we can clearly see spatial discontinuity at $MLT = 0$ in the PINE prediction for a storm period of June 26-27, 2001, when the model is trained with geolocation of data is registered in polar coordinates, see Figure 5. Figure 5 also shows that spatial discontinuity is removed for NDT prediction when training data is geolocated in Cartesian coordinates.

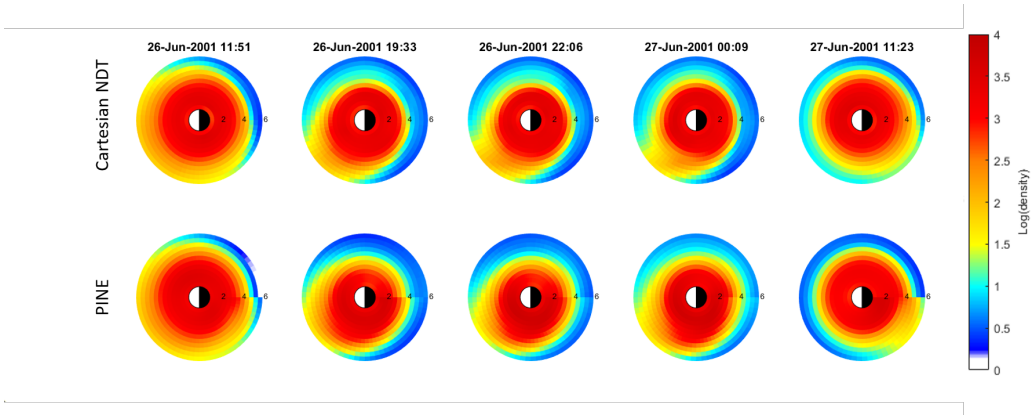


Figure 5: When Cartesian coordinates are used for the geolocation of training data in the NDT training process, the spatial continuity in the prediction of electron density field is achieved. Since polar geolocation is used in PINE's training, the electron density field produced by PINE can have visible spatial discontinuities.

The deviation from the basic statistical assumptions for regression underlying model training may mean in practice that the same relative residual errors in electron density region weigh significantly more in the model training process than low-density regions or regions where a higher abundance of data have oversized importance. The ability of NDT to easily select a suitable network configuration enables us to quickly explore the approaches that can address these high-level data analysis issues that stem from our understanding of the physical properties of the plasmasphere. A usual remedy for the disparity in spatial and statistical data distribution is by re-scaling of raw data. In particular, we can partition the plasmaspheric region into areas where data density and statistics are similar. In our case, the partitions are according to altitudes. Let $\mathcal{A}_k, k = 1, \cdots, K$

be defined by

$$\mathcal{A}_k = \{(l, mlt), l_{k-1} \le l < l_k\}.$$

Consider localized sample mean and standard deviation defined by

$$\bar{y}_k = \frac{1}{N_k} \sum_{(l_i, mlt_i) \in \mathcal{A}_k} y_i, \quad \sigma_k = \frac{1}{N_k - 1} \sum_{(l_i, mlt_i) \in \mathcal{A}_k} (y_i - \bar{y}_k)^2,$$

where $N_k = |\{(l_i, mlt_i) \in \mathcal{A}_k\}|$. Then a normalized version of electron density is defined by

$$\hat{y}_i = \frac{y_i - \bar{y}_k}{\sigma_k}, \quad \forall (l_i, mlt_i) \in \mathcal{A}_k. \tag{3}$$

When a new regression neural network is trained to predict $\hat{y}$ instead of y, the training data are more consistent with the statistical assumptions for regression analysis. In the subsequent discussion, we refer to a model trained with data scaled by local statistics as statistically scaled models. Naturally, the output $\hat{y}(x)$ of a statistically scaled model must be restored to the original scale by

$$y(x) = \hat{y}(x)\sigma_k + \bar{y}_k, \quad \forall (l, mlt) \in \mathcal{A}_k.$$

Similarly, we could remedy the non-uniform spatial distribution of data by scaling. Let ρ_k be the number density of data points in the region $\mathcal{A}_k$. We can replace the standard deviation in (3) by $\hat{\sigma}_k = \sigma_k / \sqrt{\rho_k}$. We refer to a model trained with variable weights for data points as a weighted model. The scaling and weighing of data are equivalent to the change of the regression performance metric. It is therefore expected that the new models would produce larger RMSE in their predictions when tested against validation data set than previous training when lowering RMSE is the optimization criterion. However, these new variants of models may provide a better representation of plasmasphere dynamical features when compared to actual imagery of the plasmasphere electron density field. To illustrate the effects of our data transformation, we simulated plasmasphere electron density field during the storm of June 26-27, 2001 as in (I. S. Zhelavskaya, 2017), see Figure 6.

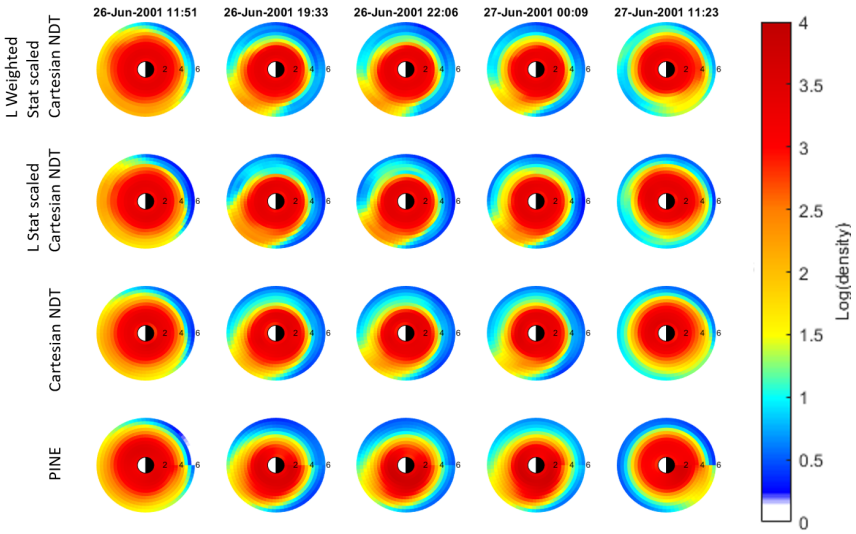


Figure 6: Effect of weighted stat scaled and stat scaled only

As a reference, we show the prediction of plasmasphere density under normal conditions defined by the mean values of the solar and magnetic input parameters in Figure 7. Not surprisingly, all four variant models show essentially the same density field.

However, comparing with Figure 7, we observe in Figure 6 that all four models show the enhancement of electron density in the mid-afternoon (low-left) region of the equatorial plane as a clockwise rotation during the on-set of the storm at around 12 UTC on June 26, 2001. As the storm progressed, we observe a significant depletion of electron density at high altitudes. At the same time, a remnant of the enhancement at around 15 MLT persisted for at least 6 hours until 0 UT on June 27, 2001, before the density field returned to a near-normal state. The four variant models give somewhat different predictions of this temporary period. In fact, all DNT models with Cartesian spatial registration of data show a much slower process with enhancement persists strongly in the afternoon (lower-left) region. Also, the progression of the decline of the enhanced region seems more detailed in NDT predictions with a much more localized enhanced region toward the end of the storm at around 0 UT on June 27. Although a determination of which of these variant models are consistently capable of producing more realistic predictions of plasmasphere dynamics during storm conditions cannot be resolved by anecdote comparison shown here, the NDT variants presented show that careful data representation can alter the final construction of the trained model. The different scaling and weighing of training data provide effective ways to construct a plurality of models that may deliver more reliable predictions for plasmasphere conditions in an ensemble.

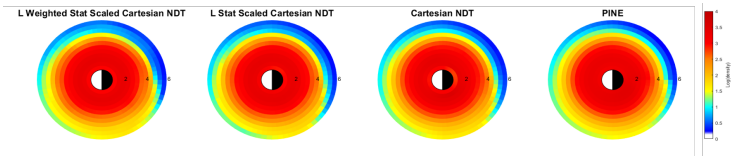


Figure 7: Effect of weighted stat scaled and stat scaled only on the average of the entire data set.

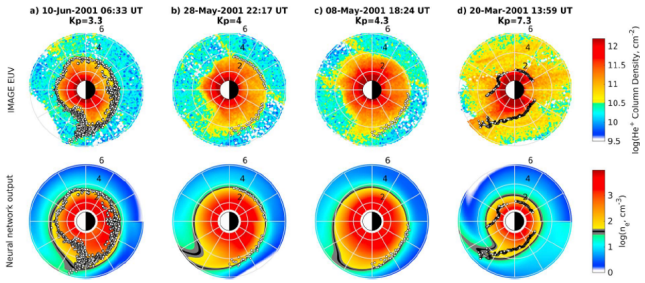
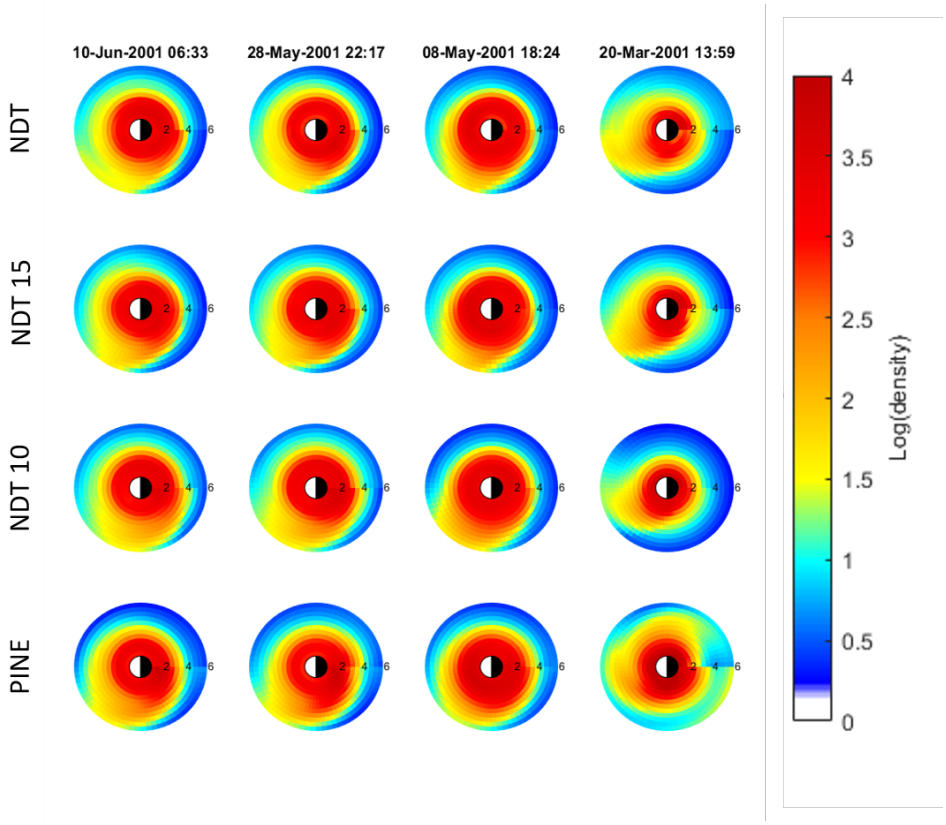
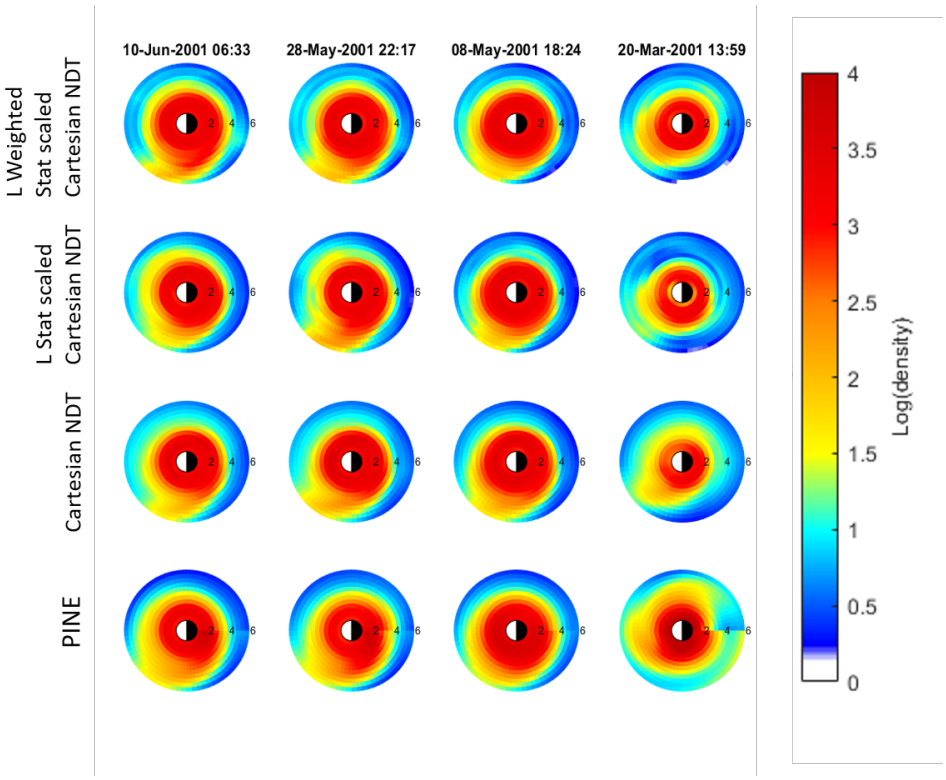


Figure 8: The top row is EUV images for the times indicated in the titles, and the bottom row is the final model output for those times. Events are ordered from left to right according to Kp (from low to high). The Kp index is shown in the titles as well.

As shown in (I. S. Zhelavskaya, 2017), comparison with EUV images can provide useful validation of model predictions. Reproduction of Figure 8 in (I. S. Zhelavskaya, 2017) shows examples of global density reconstruction by the resulting neural network model for four different events during the main phase plume formation. Compare to Figure 8, predictions provided by different versions NDT in comparison to the PINE model in Figures 9a and 9b shows similar characteristics in these model predictions. With lim-



(a) Predictions given by NDT models with different degree of freedom.



(b) Predictions given by NDT models trained with different scaled data.

Figure 9: Conditions characterized with different time and Kp index as those in Figure 8.

ited independent observation, quantitative comparison of performance among these models remains extremely challenging for the foreseeable future.

5 Discussion and Conclusion

Our numerical experimental results presented in Sections 3 and 4 show that NDT provides appropriate selection for the structure of neural network based on the available training data, and the method also leads to good initialization for the neural network. These features not only yield excellent performance in reducing residual regression errors as shown in Sections 3, but the fast convergence of NDT also enables us to focus on the physics and theoretical statistical aspect of the modeling problem.

Even though the comparison between models with different degrees of adherence to standard statistical theoretical assumptions and physical constraints seem to produce qualitatively similar predictions for the storm event of June 26-27, 2001, a deeper examination of these models can reveal substantial differences among them. For this purpose, we first perform a principal component analysis of the input parameters, i.e., AE, Kp, F107, SymH, and their near-time histories. More precisely, we first normalize each component of vector X as follows:

$$V_{i,j} = \frac{X_{i,j} - \bar{X}_i}{\sigma_i}, \quad \text{where} \quad \bar{X}_i = \frac{1}{N}\sum_{j=1}^{N} X_{i,j}, \quad \sigma_i^2 = \frac{1}{N-1}\sum_{j=1}^{N}(X_{i,j} - \bar{X}_i)^2, \tag{4}$$

for each of the components $i = 1, \cdots, 28$ of input vector X_j with $j = 1, \cdots, N$ by removing the components for L and MLT. Consider the eigenvalues λ_i^2 and eigenvectors u_i of the matrix VV^T. The values λ_i and vectors u_i are therefore principal values and principal components of the normalized data set $V_j, j = 1, \cdots, N$. Figure 10 shows that there are 5 to 6 dominant principal components for our training data set. Examining the

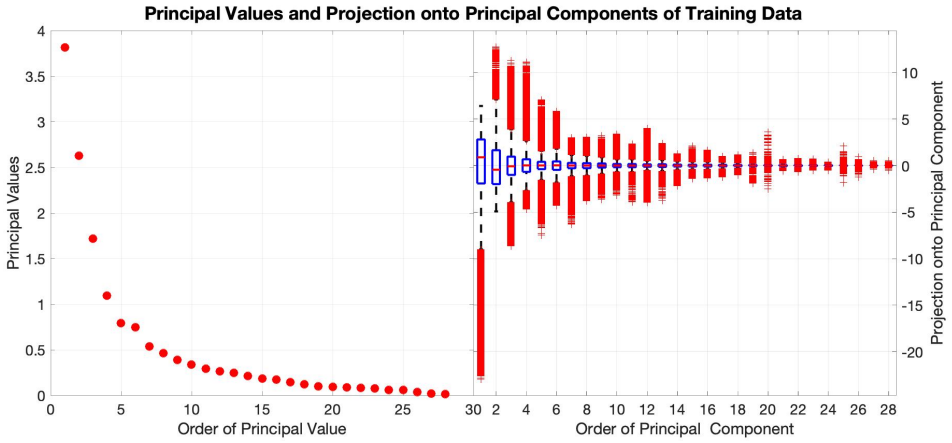


Figure 10: Distribution of Singular Values and projection of data onto principal directions

projections of data onto the principal components also reveals that outliers for the first and second principal components are clearly either all negative or all positive. Given the small number of these outliers and the fact that electron density data over the period of time when these outliers occur are very limited, we therefore do not expect the train-

ing neural network model for plasmasphere dynamics to be capable of modeling the extreme conditions represented by these outliers. Indeed, the prediction of plasmasphere density under conditions $X = \bar{X} \pm \lambda_i u_i \text{diag}(\sigma_1, \cdots, \sigma_{28})$ for the first 5 principal components in Figure 11 show signs of model saturation indicated by near-zero density at high altitude.

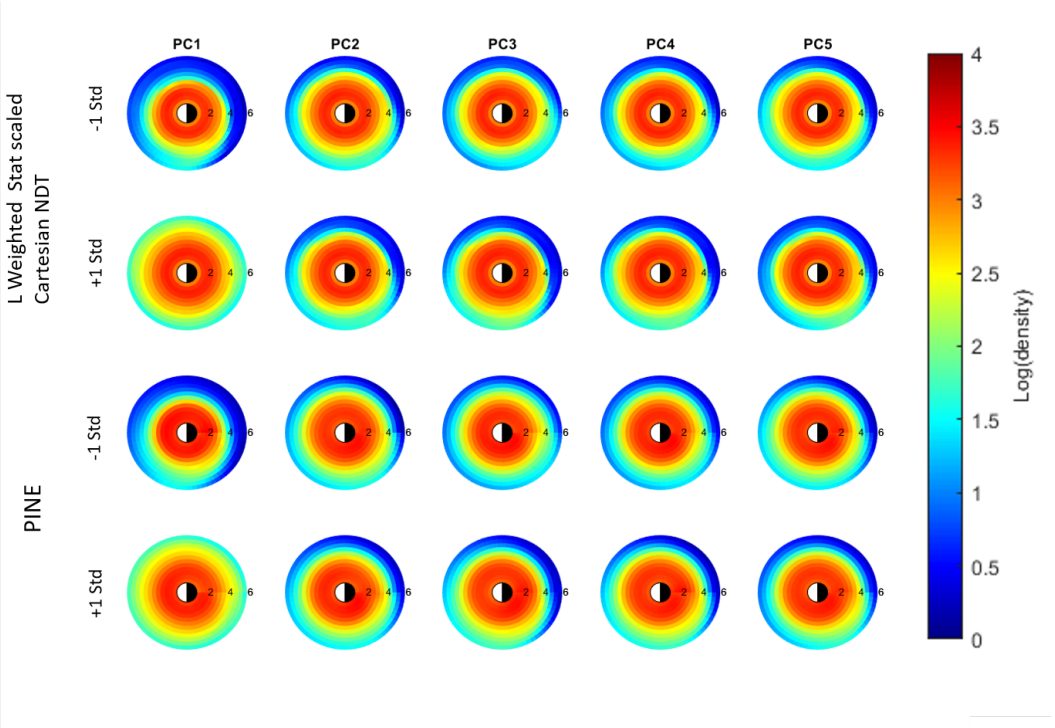


Figure 11: Electron density fields predicted by NDT and PINE for input parameters perturbed by one standard deviation in direction of the first 5 principle components respectively.

Note from the right panel in Figure 10 that the outliers in the first five principal components often are far beyond one standard deviation away from the mean value. However, perturbation of input parameters by more than one standard deviation can sometimes lead to non-physical input. Therefore, results in Figure 11 actually understate the issues of model saturation. These results are entirely expected because of the limited availability of data during extreme conditions. The model saturation also reveals the limitation of data-driven models trained with our data regarding their ability to predict plasmasphere density under extreme conditions.

We are also interested in the systematic difference among the model variants in moderate conditions. In particular, we would like to understand whether or not the principal components identified in the solar and magnetic inputs of the models lead to physically meaningful characteristics in the predicted electron density field. To do this, we evaluate the difference in the predicted electron density field with input parameters perturbed by $\pm$ 1 standard deviation from the mean values, or the *difference of difference* for the predicted fields. In Figure 12, these differences are shown for the first five principal components for the weighted NDT and PINE. In addition to the spatial discontinuity at $mlt = 0$ that is visible in the PINE predicted electron density field in perturbation of principal components, there are also noticeable differences in perturbation of

input parameters along with other principal components. In particular, for both the 2nd and 4th principal components, the enhancement of electron density near midnight at high altitudes has much more finely resolved structures for the NDT model. Further compar-

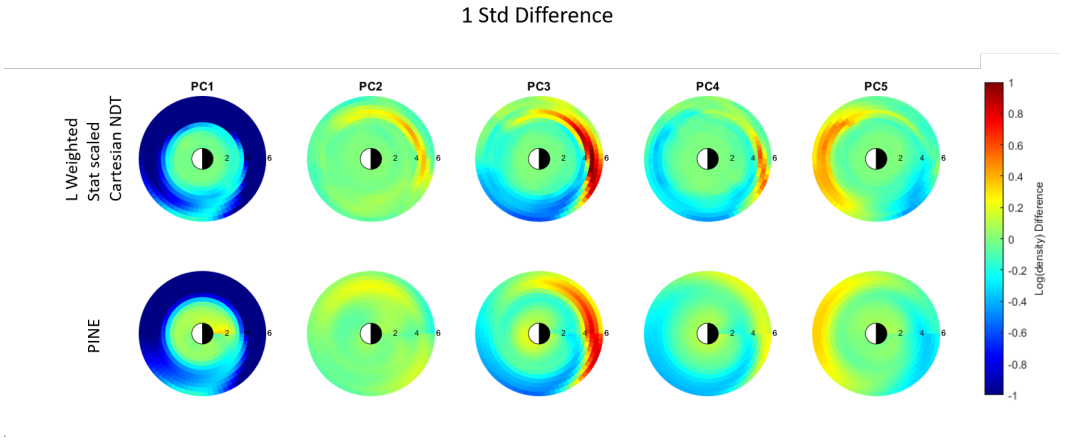


Figure 12: Difference in Electron density fields predicted by NDT and PINE for input parameters perturbed by ±1 standard deviation in direction of the first 5 principle components respectively.

ison among the variant NDT model shown in Figure 13 shows a progression of changes in the perturbation patterns. Indeed, when only the geolocation registration is changed from polar to Cartesian coordinates, the pattern produced by NDT are similar to those predicted by PINE without the spatial discontinuities at $mlt = 0$. However, other spatial features in the perturbed electron density fields emerge as additional scaling of data is introduced.

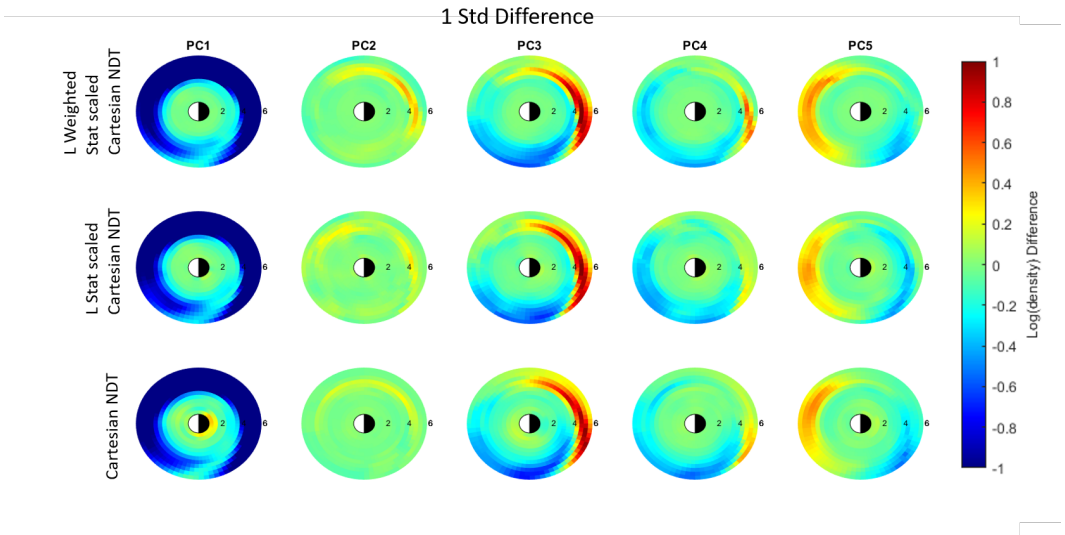


Figure 13: Difference in Electron density fields predicted by different models gentrained by NDT for input parameters perturbed by ±1 standard deviation in direction of the first 5 principle components respectively.

Without extensive independent validation data, it is difficult or impossible to conclude which model variants are more appropriate at representing the changes in the plasmasphere electron density field under characteristic changes in the input parameters. However, the models generated by NDT based on different physical and statistical considerations provide a wide range of alternative models for the prediction of the plasmasphere dynamics. When taken as an ensemble, we are more likely to capture the diversity of dynamical behavior of the plasmasphere.

In this paper, we have presented a new approach for constructing a regression neural network for plasmasphere dynamic model construction. The NDT approach naturally leads to a more sophisticated neural network structure than the traditional single hidden layer network. It is known in the machine-learning community that deep learning, which typically involves more hidden layers in neural networks, has the potential to capture a more complex relationship between input and output of a system. Our experience also reveals that even with a substantially smaller degree of freedom, a 2-hidden layer NDT trained model can outperform a single-layer model. However, the most attractive aspect of the NDT approach is its ability to identify appropriate network structures based on the decision tree initialization without prior experience. This feature is particularly relevant for the space weather community when only limited experience in machine-learning methods exists for many areas of applications.

6 Data Availability Statement

Data is available through (I. S. Zhelavskaya, 2017).

References

Biau, G., Scornet, E., & Welbl, J. (2018). Neural random forests. *Sankhya A*, *81*(2), 347–386.

Camporeale, E. (2019). The challenge of machine learning in space weather: Nowcasting and forecasting. (doi.org/10.1029/2018SW002061)

Chantry, C. H. D. P. P. T., M. (2021). Opportunities and challenges for machine learning in weather and climate modelling: hard, medium and soft ai. (doi.org/10.1098/rsta.2020.0083)

Grönquist, P., Yao, C., Ben-Nun, T., Dryden, N., Dueben, P., Li, S., & Hoefler, T. (2021). Deep learning for post-processing ensemble weather forecasts. *Philosophical Transactions of the Royal Society A*, *379*(2194), 20200092.

Huntingford, C., Jeffers, E. S., Bonsall, M. B., Christensen, H. M., Lees, T., & Yang, H. (2019). Machine learning and artificial intelligence to aid climate change research and preparedness. *Environmental Research Letters*, *14*(12), 124007.

I. S. Zhelavskaya, M. S., Y. Y. Shprits. (2017). Empirical modeling of the plasmasphere dynamics using neural networks. (doi.org/10.1002/2017JA024406)

I. Zhelavskaya, Y. S. W. K., M. Spasojevic. (2016). Automated determination of electron density from electric field measurements on the van allen probes spacecraft. (https://doi.org/10.1002/2015JA022132)

Kashinath, K., Mustafa, M., Albert, A., Wu, J., Jiang, C., Esmaeilzadeh, S., ... others (2021). Physics-informed machine learning: case studies for weather and climate modelling. *Philosophical Transactions of the Royal Society A*, *379*(2194), 20200093.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Lu, Y. L., & Wang, C. (2020). Validation of an alternative neural decision tree. In *2020 ieee international conference on big data (big data)* (pp. 3682–3691).

Murphy, K. P. (2012). *Machine learning: a probabilistic perspective*. MIT press.

Reichstein, M., Camps-Valls, G., Stevens, B., Jung, M., Denzler, J., Carvalhais, N., et al. (2019). Deep learning and process understanding for data-driven earth system science. *Nature*, *566*(7743), 195–204.