Probabilistic Short-Term Solar Driver Forecasting with Neural Network Ensembles

Joshua D. Daniell[1] **and Piyush M. Mehta**[1]

[1]Dept. of Mechanical and Aerospace Engineering
West Virginia University Morgantown, WV 26505

Key Points:

- A new striped sampling approach is developed to achieve improved performance on limited data.
- Ensemble methods show promise for short-term forecasting of JB2008 solar drivers and provide robust and reliable uncertainty estimates.
- Stacked neural network ensemble methods provide the best short-term solar driver forecasts.

Corresponding author: Joshua Daniell, jddaniell@mix.wvu.edu

Abstract

Space weather indices are used to drive forecasts of thermosphere density, which directly affects objects in low-Earth orbit (LEO) through atmospheric drag force. A set of proxies and indices (drivers), $F_{10.7}$, $S_{10.7}$, $M_{10.7}$, and $Y_{10.7}$ are used as inputs by the JB2008 thermosphere density model. The United States Air Force (USAF) operational High Accuracy Satellite Drag Model (HASDM), relies on JB2008, and forecasts of solar drivers from a linear algorithm. We introduce methods using long-short term memory (LSTM) model ensembles to improve over the current prediction method as well as a previous univariate approach. We investigate the usage of principal component analysis (PCA) to enhance multivariate forecasting. A novel method, referred to as striped sampling, is created to produce statistically consistent machine learning data sets. We also investigate forecasting performance and uncertainty estimation by varying the training loss function and by investigating novel weighting methods. Results show that stacked neural network model ensembles make multivariate driver forecasts which outperform the operational linear method. When using MV-MLE (multivariate multi-lookback ensemble), we see an improvement of RMSE for $F_{10.7}$, $S_{10.7}$, $M_{10.7}$, and $Y_{10.7}$ of 17.7%, 12.3%, 13.8%, 13.7% respectively, over the operational method. We provide the first probabilistic forecasting method for $S_{10.7}$, $M_{10.7}$, and $Y_{10.7}$. Ensemble approaches are leveraged to provide a distribution of predicted values, allowing an investigation into robustness and reliability (R&R) of uncertainty estimates. Uncertainty was also investigated through the use of calibration error score (CES), with the MV-MLE providing an average CES of 5.63%, across all drivers.

Plain Language Summary

Objects in low-Earth orbit, are affected by atmospheric drag, which depends on density. A currently used thermosphere density model, JB2008, relies on a collection of space weather solar drivers as inputs. Currently, these drivers rely on a linear forecasting algorithm, which supplies single point forecasts for 6 days. No current methods exist for providing probabilistic forecasts and uncertainty estimates for three of the drivers. In this work, we introduce a machine learning approach which provides a probabilistic forecast of all solar drivers used by JB2008. The new approach uses a combination of individual predictions, which can be combined to produce less error than the current operational method. This new approach provides improvement over single point forecasting made by the operational method and provides a measure of confidence in the forecasted values.

1 Introduction

Just a few decades ago, the number of objects in LEO was small; the detection, tracking, and identification of artificial objects, known as catalog maintenance, was relatively easy. However, in the last two decades there has been an exponential growth in total objects, especially due to large satellite constellations. LEO has quickly become the most populated orbital region, and these objects pose immediate danger to multi-billion dollar space assets and human spaceflight missions. As the total number of artificial objects in LEO grows, catalog maintenance has become non-trivial. A need for more real-time knowledge of the space environment has caused a shift to space domain awareness (SDA), which stresses the ability to accurately predict an object's orbital state.

For objects in LEO, atmospheric drag accounts for the largest source of uncertainty in predicted state. Atmospheric drag is directly tied to thermosphere heating and neutral atmosphere density of the thermosphere. The properties of Earth's upper atmosphere are heavily impacted by solar activity, specifically extreme ultra-violet (EUV) irradiance. Changes in thermosphere density occur with changing solar activity levels. High energy EUV solar radiation is absorbed by the Earth's upper atmosphere and causes large den-

sity variations due to heating. This change in thermosphere density directly impacts the dynamics of LEO objects in the thermosphere. More robust predictions for solar EUV will lead to more accurate modeling of density and orbit propagation.

The $F_{10.7}$ solar radio flux proxy is one of the most widely used proxies for solar activity. (Tapping, 2013) describes the proxy measurement as a "determination of the strength of solar radio emissions in a 100 MHz-wide band centered on 2800 MHz (a wavelength of 10.7 cm) averaged over an hour". $F_{10.7}$ has a high correlation with both sunspot number and solar EUV irradiance, seen in both (Svalgaard & Hudson, 2010) and (Vourlidas & Bruinsma, 2018); and is considered a good indicator for thermosphere heating (K. W. Tobiska et al., 2009). $F_{10.7}$ is the recognized historical EUV proxy and daily values have been recorded consistently since 1947.

Care should be taken to describe the difference between index and proxy;$F_{10.7}$ has been found to be correlated with solar EUV but is not a direct measure. Similar to (R. J. Licata et al., 2020), we encourage the distinction that a proxy is an indirect measure, while an index relies on direct measurements. The $F_{10.7}$ proxy is reported in solar flux units (SFU), where

$$1\ SFU = 10^{-22} \frac{W}{Hzm^2}$$

Three more indices and proxies were introduced for use as inputs to the Jacchia-Bowman 2008 (JB2008) empirical thermosphere density model. $S_{10.7}$, $M_{10.7}$, and $Y_{10.7}$, which we refer to as *drivers*, map energy from specific solar irradiances to major thermosphere layers. (K. W. Tobiska et al., 2009) Using the four solar drivers, JB2008 provides significant improvement in empirical thermosphere density modeling. (K. W. Tobiska et al., 2009) $S_{10.7}$, $M_{10.7}$, and $Y_{10.7}$ are scaled with linear regression to units of SFU, to be consistent with the historical $F_{10.7}$ and enable direct comparison. This scaling is done to more easily qualitatively compare various indices/proxies. (K. W. Tobiska et al., 2009)

The $S_{10.7}$ index (Bowman et al., 2008) is the integrated 26-34 nm irradiance measured by the Solar Extreme-ultraviolet Monitor (SEM) instrument on the NASA/ESA Solar and Heliospheric Observatory (SOHO), and is used to represent heating in regions near 180 or 200 km. SET provides an operational backup for SEM data processing as well as provides values of $S_{10.7}$. SEM has been making measurements since December 1995. A more specific description of the process for scaling the data and converting to solar flux units (SFU), units consistent with other drivers, are discussed by Tobiska et al. (K. W. Tobiska et al., 2009). Daily values for index are archived and available since January 1, 1997;

The $M_{10.7}$ proxy (Bowman et al., 2008) is created from the Magnesium II (Mg II) core to wing ratio, which originates from the NOAA (National Oceanic and Atmospheric Administration) satellites (NOAA -16,-17,-18). The satellites host the Solar Backscatter Ultraviolet (SBUV) spectrometer, which can make solar UV measurements. MG II is a proxy for solar FUV and EUV emissions that is mapped into the lower thermosphere and represents heating in the thermosphere regions between 95-110 km. MG II is translated into SFU as discussed in the work by Tobiska et al. (K. W. Tobiska et al., 2009). Daily values for $M_{10.7}$ are archived and available since January 1, 1997.

The $Y_{10.7}$ index (Bowman et al., 2008) is created from the combination of both GOES/XRS 0.1-0.8 nm x-ray observations and Lyman-α, which is measured by the SOSTICE instrument on the UARS and SORCE satellites as well as the SEE instrument on TIMED. The observations made are represented by an $L_{10.7}$ and $X_{10.7}$ index, which are combined by Tobiska et al. (K. W. Tobiska et al., 2009) to form the $Y_{10.7}$ index, representing heating in regions between 85-100 km. Data for $Y_{10.7}$ has been reported daily and archived since January 1, 1997.

JB2008 takes these solar drivers and two geomagnetic drivers to predict density at discrete global positions and at a variety of altitudes. The United States Air Force (USAF) uses the operational High Accuracy Satellite Drag Model (HASDM) for satellite drag modeling. HASDM relies on an adjustment of the density nowcast made by JB2008, using satellite observations. to produce corrected densities for use in orbit propagation. Space Environment Technologies (SET) is currently contracted by the USAF to provide forecasted driver values for use with HASDM. SET uses a linear auto-regressive algorithm for forecasting driver values, specifically the "TS_FCAST" subroutine in IDL (Independent Data Language) (R. J. Licata et al., 2020).

Historically, forecasting of model drivers have relied on deterministic methods. Work done by (Daniell & Mehta, 2023b), (R. Licata et al., 2021), and (R. J. Licata et al., 2022) have shown methods which provide probabilistic forecasting using neural network models. Recent work by (Daniell & Mehta, 2023b) showed that neural network ensemble methods outperformed the linear algorithm for univariate $F_{10.7}$ prediction and provided a well calibrated probabilistic forecast. The linear algorithm used by SET is the only current method for forecasting $S_{10.7}$, $M_{10.7}$, and $Y_{10.7}$, so it is important to explore a similar approach to enable probabilistic forecasting of these drivers.

There has been a dramatic increase in the past few decades in the number of objects in LEO, especially due to large satellite constellations, leading to the need for better understanding of predicted orbital state, to avoid collisions. This increase in objects has led to the shift from SDA to space traffic management (STM), where operators must perform risk-assessments and potential high-cost maneuvers to avoid collisions. By providing improved short-term forecasts of model drivers to JB2008, more accurate short-term forecasts of density can be made. Additionally by providing probabilistic forecasts of model drivers, models will be able to provide robust and reliable uncertainty estimates for density. An improvement in driver forecasts would lead to an improvement in density forecasts made by JB2008, which would lead to an improvement in drag modeling performed by HASDM and would enhance STM efforts.

The paper is organized as follows. In Section 2, we introduce the dataset for the drivers and necessary data preparation procedures for our machine learning approaches. In Section 3 we discuss LSTM models, hyperparameter tuning, the proposed neural network ensemble approach, error metrics, and the methods for uncertainty quantification. In Section 4; we investigate ensemble diversity and methods for combining forecasts. We also compare the results of the multivariate ensemble method against the univariate linear algorithm and the univariate ensemble method used in (Daniell & Mehta, 2023b). We also discuss the probabilistic forecast uncertainty estimates by performing uncertainty quantification (UQ).

2 Methodology

The operational model for driver forecasting utilizes the "TS_FCAST" subroutine in the Interactive Data Language (IDL), which was discussed by (R. J. Licata et al., 2020), can be seen in the following equation,

$$x_t = \sum_{i=1}^{P} \theta_i \, x_{t-i} + \epsilon_t \tag{1}$$

where P is the order of the model, ϵ_t is an error term, θ are scalar coefficients, and x_{t-i} are the values of the driver i days prior. This algorithm is a P-th order linear auto regressive model which captures persistence and recurrence. (W. K. Tobiska et al., 2008) The linear algorithm was implemented by (Daniell & Mehta, 2023b) for comparisons with

neural network ensemble approaches. An extensive benchmarking of this method was presented by (R. J. Licata et al., 2020).

The SET algorithm is deterministic, providing a single output for a given input. The model also makes iterative forecasts, requiring forecasts to be "chained" together to reach a prediction horizon larger than one day. It should be noted that the SET algorithm is unable to provide any uncertainty estimates in its forecasts due to the deterministic nature of the algorithm. This algorithm is the method that we will be making comparisons to, considered the baseline.

This work introduces a novel approach for preparing machine learning data sets, a novel approach for preparing multivariate data, and a novel approach for probabilistic forecasting all solar drivers. We investigate the performance of neural network model ensembles for proving the first ever probabilistic forecasts of $S_{10.7}$, $M_{10.7}$, and $Y_{10.7}$. We also investigate methods similar to the univariate approaches used by (Daniell & Mehta, 2023b). We seek to answer the question, "will multivariate models provide better forecasts than four separate univariate models?"

2.1 Data

The dataset for $F_{10.7}$ is the largest, with observed values being recorded since 1947. The other 3 drivers are limited by data being produced by spacecraft, for example $S_{10.7}$ data prior to the launch of the SOHO spacecraft would not be possible as no measurements would have been made. Due to this limitation, archived daily values for drivers exists between January 1, 1997 and the present day, seen in Figure 1. Missing values are noticed in the $S_{10.7}$ driver between 6/25/1998 and 10/24/1998 and linear interpolation was used to fill in missing data, accounting for about 1% of the $S_{10.7}$ driver data.

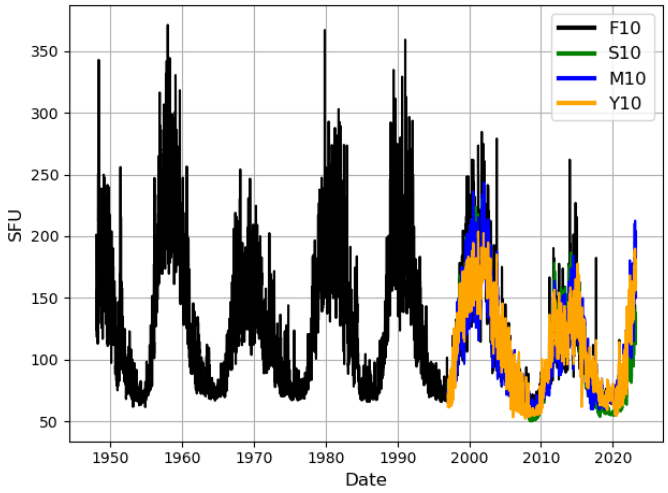


Figure 1: It is desired for a neural network model to see many repeated patterns during training. In the case of the newer drivers, several phases of the solar cycle have only been seen a few times.

2.2 Data Preparation

It is necessary to carefully preprocess data to efficiently apply and train neural network models. It was shown that normalization of data is a critical step to both improve

results and decrease computational time. (Sola & Sevilla, 1997) To normalize the data, we use the standard normalization equation,

$$\tilde{D} = \frac{D - mean(D)}{standard\,deviation(D)} = \frac{D - \mu_D}{\sigma_D} \tag{2}$$

where D represents any arbitrary driver.

Previous forecasting efforts, such as the work done by (Daniell & Mehta, 2023b), (Stevenson et al., 2022), (Luo et al., 2022), and (Huang et al., 2009) leveraged historical values of $F_{10.7}$ as inputs, referred to as auto-regressive (AR) modeling. Since the SET method is AR, we limit ourselves to using only previous driver values for making predictions to maintain consistency Additionally, we consider historical values of multiple drivers simultaneously in an effort to improve forecasting results.

To apply neural networks, we must provide the network with input/output pairs, which are used by the model during training. By providing pairs to the model, comparisons between model predictions and true outputs can occur. We consider the sliding window method to split the data; providing inputs as the previous observations L (Lookback) and the future H (Horizon) driver values. (Daniell & Mehta, 2023b)

(Daniell & Mehta, 2023b) identified the importance of the lookback range used in the models, stating that a combination of short-term and longer-term lookbacks can be beneficial to probabilistic model performance. To provide a direct comparison to the linear SET method, we choose a 6-day horizon, which is consistent with the thorough solar driver forecasting analysis performed by (R. J. Licata et al., 2020).

2.2.1 Creating Data Subsets for Machine Learning

Training of a neural network model is the step where changes are made to weights and biases within the model in order to improve the output, it can be thought of as teaching by example. (Wasserman & Schwartz, 1988) The model makes an educated guess based on the inputs, compares with the expected output, changes weights/biases, and makes another educated guess and compares once more. Training allows for the weights and biases to be adjusted to minimize a given optimization loss function. The weights and biases are internal to the model and can be adjusted to improve results.

For typical regression and model selection/training work flows, we split the data into 3 subsets; training data, validation data, and testing data. The training data is what is used to teach the model by example; the model is provided with training data as an input, it makes a prediction, and predictions are compared against the expected output. This training set is used to adjust model weights and biases, which will change future model outputs. Validation is used as a method to mitigate model over-fitting, or a "remembering" of the training data. The validation data is used after a training step and can provide insight into how well the model can generalize on data it has not seen yet. (Xu et al., 2016) claim that the model validation step is the most important part of building a supervised model. The test set, is data that has been entirely hidden from models during training steps. By keeping a dataset hidden, the model is "tested" on entirely new data and the model's ability to form generalizations can be evaluated more clearly.

2.2.2 Choosing a Validation Scheme

The holdout method is one of the most typical methods for splitting data into the three sets. A percentage for splitting, typically used in machine learning (ML), is a 70%/15%/15% split for training/validation/testing sets. When considering time series data, such as the data set used in this work, typical holdout methods would preserve the temporal order of data by partitioning a percentage of data at the end of the full set, referred to as the

test set. After the first partition is made, a second partition is made using a specified percentage of data at the end of the non-test set. This second partition will contain the data which can be used for training and validation steps.

In machine learning, the amount of data used to train a model greatly impacts the performance of final models. By providing larger number of samples to the model during training, better generalization can be expected. However, when training data is limited, worse generalization can be expected. It is important to ensure that the validation scheme chosen supplies the models with enough training data so that models can learn on statistically consistent data. In this case, ensuring that data sets capture similar levels of solar activity. We are limited by the amount of historical data that exists for the non-$F_{10.7}$ drivers.

If one were to use typical holdout validation methods, or even cross validation, there exist several issues:

- Training, validation, and test sets may be based on differing portions of the solar cycle; i.e. differing activity levels.
- If using traditional holdout, the test set may not be statistically consistent to the training or validation sets. This would lead to good test set prediction only when data is similar with that of the training or validation sets.
- If a model is trained and validated on only solar maximum data; one would expect over prediction or poor performance when predictions at solar minimum occur. A similar problem would occur if training was done on solar minimum data, and solar maximum predictions were desired.

To combat these issues, we introduce a novel method, *striped sampling*. Striped sampling involves a structured data splitting to ensure adequate data for effective model training while also statistically balancing the dataset. First, data is split into weekly input output pairs; such that for every 10 weeks, 6 weeks are used for training, the following 2 weeks are used for validation, and the following 2 weeks are used for testing. The striping method results in a nearly desired 60%/20%/20% split; samples of such splitting are shown in Figure 2.

Using the traditional holdout method on the full dataset, significant statistical differences were seen between training, validation, and test data, seen in Figure 3a. By using striped validation, we effectively capture similar statistics between our training, validation, and testing sets; seen in Figure 3b. This result indicates that, with limited data, a more intelligent method for splitting data, such as striping, can create more useful datasets for ML methods.

Striped sampling of data results in sets that are statistically consistent, but still have minor differences, especially as solar activity level increases. Small variations can be seen, most notably, $Y_{10.7}$ with values between 70 - 130 SFU in 3b. We consider the striped sampling approach to be beneficial for sampling data and this work will implement it.

2.2.3 PCA Rotation

Principal component analysis, or PCA, is considered the most popular multivariate statistical technique and likely to be the oldest multivariate technique. PCA is typically used on large dimension data; compressing the size of the set while keeping the more important information. PCA involves a transformation from the original data into linear combinations of the original variables known as principal components (PCs). The PCs are calculated in such a way as to maximize variance between the PCs and constrains the components to be orthogonal to one another (Abdi & Williams, 2010).

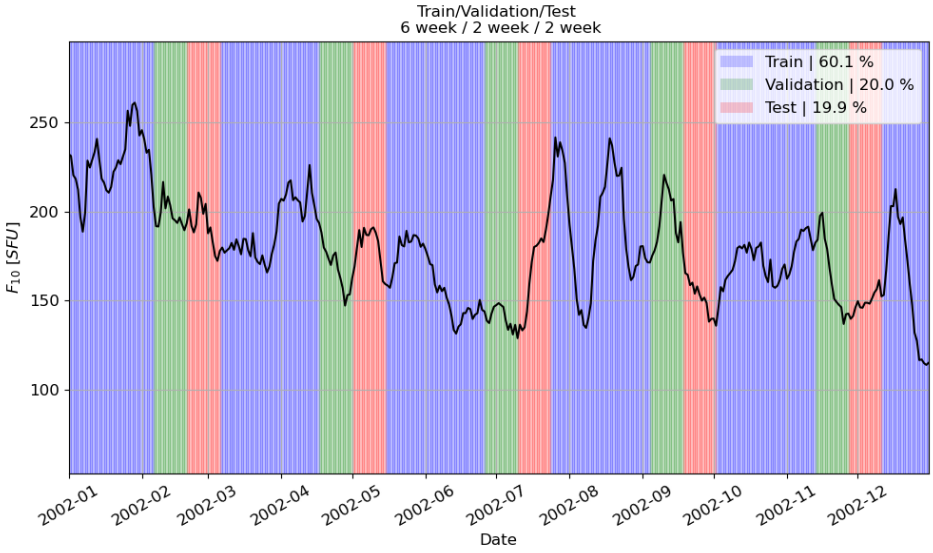
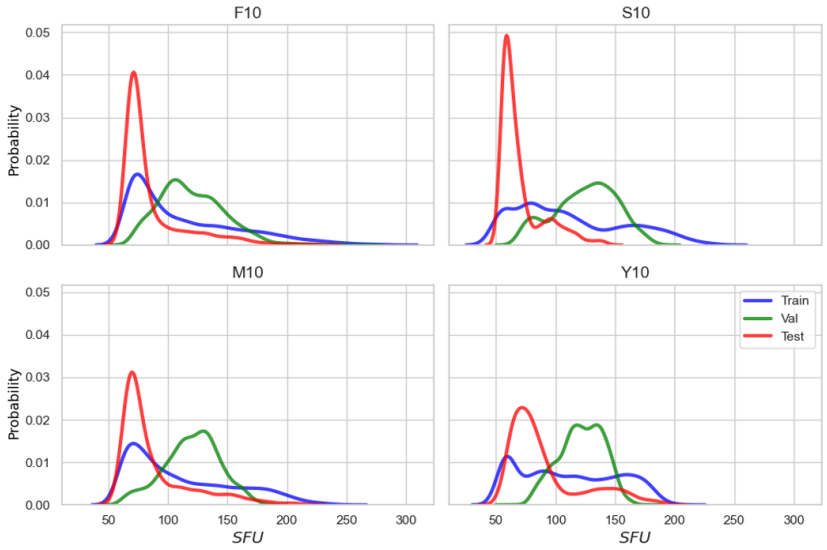


Figure 2: By creating "chunks" of input/output pairs, we allow the ML models to see sequential data while also providing statistically similar data for training, validation, and testing.

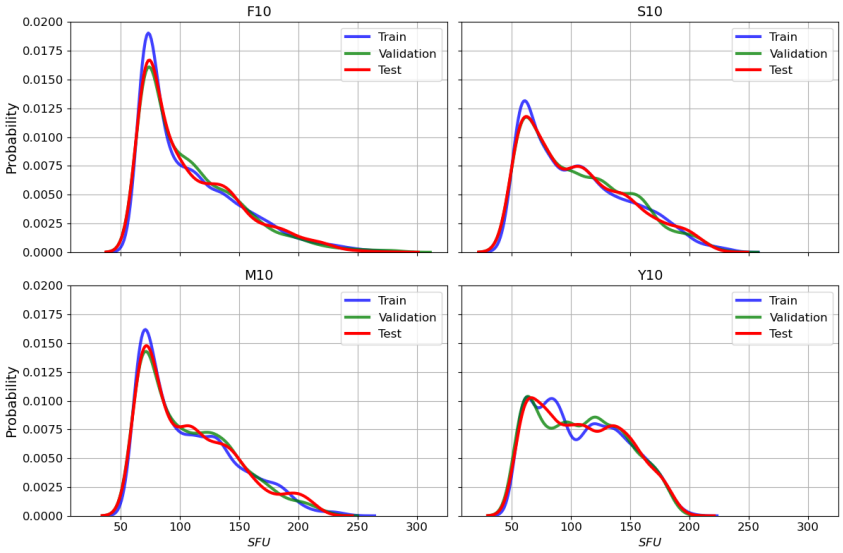
Typical methods using PCA would truncate PCs, to reduce the dimension of the dataset, allowing for easier applications of machine learning techniques; especially those involving very large or high dimension data. Since the PCs are a linear combination of the initial space, the dimensions may not be interpretable; these new components no longer represent the original driver values. The PCA algorithm is as follows:

1. Starting with a time series of four drivers (which are separated into the training, validation, and test sets), standardize the data based on statistics of the training set.
2. Calculate the covariance matrix based on the training set; a 4x4 symmetric matrix that contains the covariances associated with all pairs of variables, which is performed via *Numpy.cov()* in Python.
3. Compute eigenvectors and eigenvalues of covariance matrix $\mathbf{C}$ to identify PCs.
4. Sort eigenvalues and associated eigenvectors based on scale of eigenvalue (maximizing variance) and construct a feature vector matrix.
5. Perform ML methods (training, validation, and prediction) in the PCA rotated space using the feature vector matrix.
6. Transform predictions back into the original space and reverse standardize the outputs.

Redundant information is contained within the solar drivers and it may be considered less important to forecast them all at once. Applying ML techniques to make multivariate forecasts on highly correlated variables could be less useful. By applying a technique like PCA, we "untangle" our data, and force our dimensions to be orthogonal and have a maximized variance (less correlated). We have not seen a PCA rotation method used in the forecasting of density model drivers and introduce a rotation similar to the one discussed by (Abdi & Williams, 2010). Most ML applications for machine learning truncate PCs due to the small amount of variance that they capture (Hu et al., 2016). We consider the typical PCA algorithm with no truncation, simply a "rotation" to max-



(a) Approximate PDFs indicate that the average solar activity level of the sets are drastically different; leading to difficulty with traditional machine learning sampling methods.

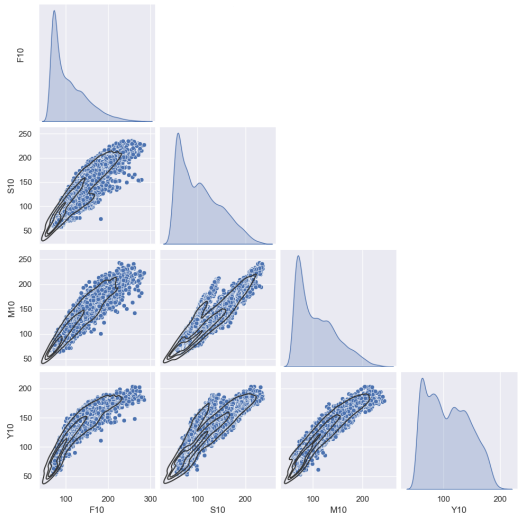


(b) By capturing similarly approximated PDFs between datasets using striped validation, we give machine learning models the best chance to effectively generalize and reduce potential bias.

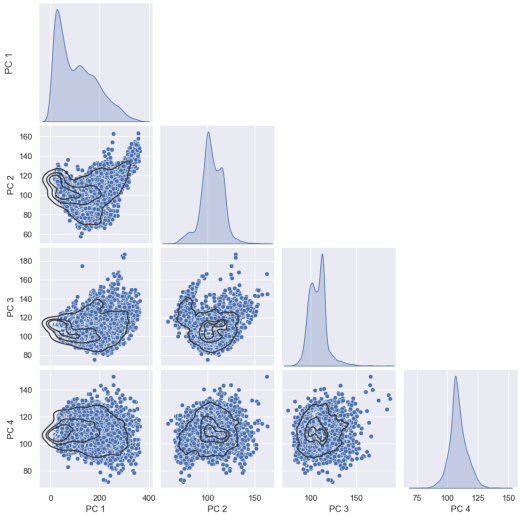
Figure 3: **Top**: Holdout methods are used to split the data into the three ML subsets, which produces inconsistent statistics. **Bottom**: Striped sampling allows for consistent statistics between subsets.

imize variance and create orthogonal PCs. We aim to investigate whether de-correlation of the drivers can improve forecasting.

By applying PCA rotation, we create a set principal components that are drastically different than the original drivers. The original driver distributions, seen in Fig-



(a) **Main Diagonal:** Approximate distributions for individual drivers. **Lower Entries:** Correlation between pairs of drivers.



(b) **Main Diagonal:** Approximate distributions for individual PCs. **Lower Entries:** Correlation between pairs of PCs.

Figure 4: **Top**: Raw driver data is highly correlated and may hold promise for an ML approach known as transfer learning. **Bottom**: PCA rotation yields PCs which are significantly less correlated and should be investigated as ML model inputs.

ure 4a, were very similar; PCA rotation provides PCs with more unique distributions, seen in Figure 4b. The rotated data can be used by neural networks in nearly the same way as unrotated data. The only difference in the process of evaluating models with PC inputs, is that PCA based models will require data to be rotated back to the original driver space after predictions are made. To our knowledge, this is the first application of PCA rotation in the field of solar driver forecasting.

2.3 Long-Short Term Memory (LSTM) and Training

An important model type for time-series forecasting is Long-Short Term Memory (LSTM), which was introduced by (Hochreiter & Schmidhuber, 1997). LSTM models have been used extensively in time-series forecasting problems such as stock market prediction (Bhandari et al., 2022), terrestrial weather forecasting (Karevan & Suykens, 2020), and the domain of space weather and forecasting; (Luo et al., 2022),(R. Licata et al., 2021) and (Benson et al., 2021).

LSTM models leverage a feature known as the hidden cell state to "remember" information that had been provided to the model earlier (Hochreiter & Schmidhuber, 1997). LSTM models are commonly used in leveraging prior information without being directly used as an input. For example, text prediction algorithms such as those seen in email and smart phones, leverage LSTM models to use prior text to suggest an expected output based on what has been typed. In this work our goal is to utilize LSTM models for forecasting drivers by considering both directly input data at the current time step and the "remembered" prior inputs/outputs of the model.

It is important to note that during LSTM training and prediction steps, data cannot be temporally disjoint. Data preparation is non-trivial and the LSTM cell state must be cleared between each sample that is input into the model. For example, training samples contain 7 days of input/output pairs for 6 weeks and then there are 4 weeks of validation and test data. If the model were to see the next training sample directly after the first, it would attempt to use "short-term memory" of data from 4 weeks ago, which would drastically decrease model performance. By using a "striped" validation approach, seen in Figure 2, the temporal ordering of the data between the training, validation, and test samples is preserved.

To our knowledge, approaches for forecasting of the three drivers: $S_{10.7}$, $M_{10.7}$, and $Y_{10.7}$; with methods other than the linear method used by SET have not yet been introduced. We aim to provide multivariate forecasting for all four drivers using LSTM neural network model ensembles. We provide the first probabilistic method for forecasting $S_{10.7}$, $M_{10.7}$, and $Y_{10.7}$ and aim to improve on errors seen in the SET algorithm. Additionally, we aim to provide robust and reliable uncertainty estimates for each of the drivers.

2.3.1 Transfer Learning (Univariate)

A common practice in the field of machine learning involves using a previously trained model (or set of models) as a starting point, known as *transfer learning.* Transfer learning is a powerful tool that allows the use of an already made model without the need for excessive training or hyperparameter selection. Due to the lack of a large available data set for $S_{10.7}$, $M_{10.7}$, and $Y_{10.7}$, a transfer learning approach should be investigated. The work done by (Daniell & Mehta, 2023b) showed a univariate approach that enabled a probabilistic forecast of $F_{10.7}$ using NN ensembles. It can be seen in Figure 4a, the drivers with limited data correlate well with the $F_{10.7}$ proxy. Transfer learning may provide reasonable forecasts for tasks that are related; such as highly correlated variables (Qureshi et al., 2017). Transfer learning can also significantly improve the efficiency in learning by exploiting the relatedness between a data-scarce target task and a data-abundant source task (Gerace et al., 2022).

To accomplish transfer learning, NN models which have been created for forecasting $F_{10.7}$ can be used for the other drivers. Models already trained for solar proxy prediction should be considered, such as models created by (Daniell & Mehta, 2023b). To prepare for transfer learning, data for the drivers must be formatted identically to the data used for the original model. Models loaded for transfer learning may be starting "ahead" of newly created models and may be able to provide good performance without the need for excessive training, only needing a small amount of training to "fine-tune"

model weights and biases (Weiss et al., 2016). The models can then be evaluated on the training, validation, and test data.

Based on the work done by (Daniell & Mehta, 2023b), we select the univariate *MLE LSTM*. The MLE LSTM is an ensemble approach, which is selected due to its good performance metrics and well behaving uncertainty estimates. The MLE LSTM showed reasonable performance improvements over the SET algorithm when forecasts of $F_{10.7}$ were made and provided robust uncertainty estimates. For the remainder of the work, we refer to the application of a MLE for univariate data as UV-MLE (UniVariate Multi-Lookback Ensemble) and multivariate case as MV-MLE (MultiVariate Multi-Lookback Ensemble).

2.3.2 Univariate Approach (UV-MLE)

A logical next step, is to use the method described by (Daniell & Mehta, 2023b) to create MLE that are tuned and trained specifically for $S_{10.7}$, $M_{10.7}$, and $Y_{10.7}$. The univariate approach showed an improvement over the SET linear method, as well as the NOAA Space Weather Prediction Center (SWPC) forecasts. The same steps are used to construct a set of models to forecast drivers, using the new training, validation, and test data splitting methods discussed in Section 2.2.1. A hyperparameter tuner is constructed for each driver and a set of lookbacks and backwards averaged values are considered for the generating of neural network ensemble members. Using the same methods as the authors, approximately 180 models are created for each driver and are trained and used to predict separately.

2.3.3 Multivariate Approach (MV-MLE)

Due to the high correlation between drivers, seen in Figure 4a, trends seen in one driver are most likely to be seen in the other drivers. Rather than limit a model to a single stream of data, we can consider a model which is input four sets of previous driver values and provides a forecast for all four variables simultaneously. Such a model would increase the dimensions of considered data by a factor of four; all inputs and outputs would involve all four drivers as opposed to just one. By considering all drivers simultaneously, patterns seen in one driver may be useful for forecasting of another driver. For example, a short-term decrease in $S_{10.7}$ may indicate a similar drop would occur in $M_{10.7}$, even if a pattern is not seen in previous $M_{10.7}$ data. In this work, simultaneous multivariate forecasting is performed and compared with univariate methods. This work is done to determine if such multivariate methods are more beneficial than univariate methods. Two approaches will be explored for multivariate forecasting; we will prepare data by standardization (Equation 2) and consider both standard and PCA rotated inputs.

2.3.4 Model Training

During neural network model training, an optimization loss is necessary to "teach" the model if it is doing well. Most often, it is desired to minimize the loss values. A minimized loss indicates that the model has found an optimal combination of weights and biases. Two of the most popular loss functions used in regression tasks are mean squared error (MSE) and mean absolute error (MAE),

$$MSE = \frac{1}{N}\sum_{i}^{N}(y_i - \hat{y}_i)^2 \tag{3}$$

$$MAE = \frac{1}{N}\sum_{i}^{N}|y_i - \hat{y}_i| \tag{4}$$

where N is the number of predictions made for a given set, y_i is the expected value of a given sample, and $\hat{y}_i$ is the output of the model for a given sample. It is desired to adjust network weights and biases such that the loss function is minimized; which is a procedure called training. These loss functions have been used successfully in the field of space weather by (Liu et al., 2020), (Stevenson et al., 2022), and (R. J. Licata et al., 2022).

Due to the exponential term in the MSE loss, large errors are penalized more than small errors. The nature of this loss function forces the model to focus on fitting samples with larger errors. As a loss function, MAE does not penalize large errors as much and may provide better performance for areas with lower errors. (Xu et al., 2016) concluded that by including multiple loss functions, the risk of overfitting may be mitigated. Based on the results from (Xu et al., 2016), the work done by (Stevenson et al., 2022), and the suggestions by (Daniell & Mehta, 2023b); we investigate the impact of different loss function on the performance of models.

2.4 Neural Network Ensembles

A neural network ensemble can be used to provide an improved forecast when compared to even individual best performing models (Hansen & Salamon, 1990). Neural network ensembles are created using multiple models to provide outputs for a given set of inputs. Typical regression models provide a deterministic forecast, which only provides a single output for a given input. A neural network ensemble uses the concept of diversity (Brown et al., 2005), to allow for predictions to be spread across models with different strengths. Neural network model diversity can be encouraged by considering varying architectures, model types, weight initialization, varying loss functions, and varied inputs. Variation of inputs have been used successfully in proxy forecasting by (Daniell & Mehta, 2023b) and (Stevenson et al., 2022).

(Daniell & Mehta, 2023b) explored NN ensembles constructed using a singular loss function, while (Stevenson et al., 2022) used multiple. An investigation into loss functions is performed to show the impact of changing the loss function on both error metrics and uncertainty estimates. To determine which models to use for our neural network ensemble, sets of model hyperparameters must be determined. Diversity is a key element into providing probabilistic forecasts with robust and reliable uncertainty estimates; it is necessary to investigate many potential sources of diversity. We investigate the effects of PCA rotation, varied loss function, model architecture, and weight initialization on prediction accuracy as well as uncertainty quantification. To provide models with a diverse set of architectures, we consider changing model hyperparameters; thus an optimal way of searching for good hyperparameters is needed.

Hyperparameters include model parameters such as; number of layers, number of neurons per layer, neuron activation function and dropout rate; which can be seen in Table 1. KerasTuner (a hyperparameter tuner) is used to generate models which make up the neural network ensemble. KerasTuner can be used to identify architectures and model hyperparameters which give a minimal loss based on a validation dataset. The results of KerasTuner can be used to generate models with varied architectures, creating diversity. The top three architectures for each lookback and loss function are used; the considered search space can be seen in Table 1.

2.4.1 Stacking Ensemble

Once the ensemble members are selected, one must carefully consider combination methods. In previous ensemble approaches for forecasting $F_{10.7}$; (Daniell & Mehta, 2023b) and (Stevenson et al., 2022) used an equal weighted output. Although an improvement was seen, one must consider that certain models are more skilled than others, well per-

Table 1: Tuning configurations to generate ensemble members at each lookback for multi-lookback ensemble methods.

UV-MLE and MV-MLE

Tuner Option	Choice	Parameter	Value/Range
Scheme	Bayesian Optimization	Number of LSTM Layers	[1-2]
Total Trials	50	LSTM Neurons	[32-256]
Initial Points	25	LSTM Activation	[tanh, sigmoid, softsign]
Repeats per Trial	2	Number of Dense Layers	[1-3]
Minimization Parameter	MSE or MAE	Dense Neurons	[64-256]
Epochs	50	Dense Activations	[relu, tanh, sigmoid, elu]
Optimizer	adam	Learning Rate	[.01, .001, .0001]
Batch Size	1	Dropout Rate	1% - 25%

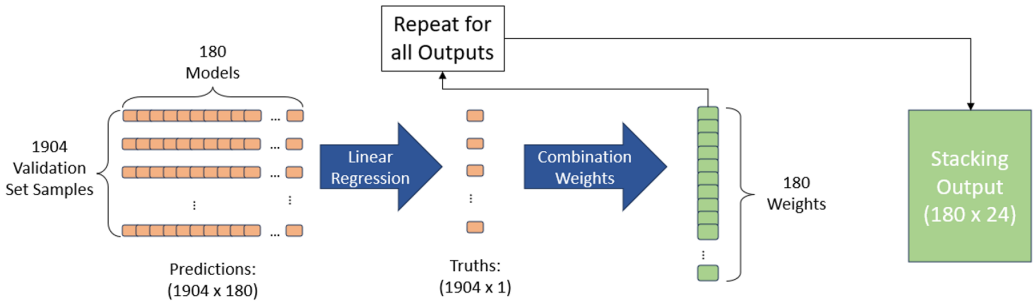


Figure 5: Twenty four linear regression models are fit using the validation data set, providing a 2-D array of weights. The weight array has 2 dimensions, 180: The number of models to combine and 24: The number of outputs per model (6 day prediction x 4 drivers).

forming models should be weighted more heavily than models with worse performance. Weighted ensembles performed better than unweighted for upper atmosphere models (Elvidge et al., 2023). We consider a *stacked* ensemble approach (Sridhar et al., 1996), which uses linear regression to optimally combine predictions. The stacking algorithm can be used to provides a set of weights, indicating which models have "more say" in the ensemble output.

Stacking, in practice, is the process of fitting a linear regression (Equation 1) of all model outputs and expected value over a set of samples. A representation of the stacking process can be seen in Figure 5. In this case, an ensemble made of 180 models will result in 180 coefficients (or weights), θ, associated with each output. In order to implement stacking, a set of predictions and ground truths must be used. We choose to use the validation set to determine stacking weights. We choose the validation set to avoid any potential leakage into the test set, keeping the set completely isolated from the set of models.

2.5 Metrics

As used in (Daniell & Mehta, 2023b) and (Stevenson et al., 2022), a relative metric can be used as a single value measure of model performance when considering multiple forecasst days. Relative metrics, introduced by (Yaya et al., 2017), are metrics which have been scaled with respect to another set of metrics and prevent larger horizon errors from dominating. We elect to maintain consistency with past work by using persistence as a baseline for comparison, due to its availability and role as a standard benchmark for time series forecasting. Relative metrics are defined by (Stevenson et al., 2022) as "the average, over all horizons, of the ratio of model performance to that of persistence",

$$Relative\, X = \frac{1}{H_{max}+1} \sum_{h=1}^{H_{max}} \frac{X_{model,h}}{X_{persistence,h}} \tag{5}$$

where X is a metric, and H_{max} is the largest horizon forecasted. We consider a maximum horizon of 6 days, which is the same horizon predicted by SET.

When performing univariate forecasting, metrics apply to a single variable and performance is easily captured by error metrics. Although it may seem easiest to evaluate metrics for multivariate methods across the entire output, it is important to investigate performance on each individual driver. When we evaluate combining models with unequal weighting, it is inevitable to find models that may perform better on certain drivers. We consider using common metrics like root mean squared error (RMSE), mean absolute percentage error (MAPE), and the Pearson Correlation Coefficient (R); Equations 6, 7, and 8 respectively. We select these to show general model performance. Metrics should be generated for all drivers to quantify the full skill of various models.

$$RMSE = \sqrt{\frac{1}{N}\sum_{i}^{N}(y_i - \hat{y_i})^2} \tag{6}$$

$$MAPE = \frac{100\%}{N}\sum_{i}^{N}\left|\frac{y_i - \hat{y_i}}{y_i}\right| \tag{7}$$

$$R = \frac{cov(\hat{y}, y)}{\sigma_{\hat{y}}, \sigma_y} \tag{8}$$

2.5.1 Uncertainty Quantification (UQ)

Each model provides forecasts for all drivers over a 6-day period, generating a set of forecasted values. We combine the predictions using the various methods discussed above to form an ensemble forecast, which provides a single point daily values for each variable for a 6-day period. The set of model predictions allow for a distribution of forecasted values to be generated, saved, and sampled for operations. Work done by (Paul et al., 2023) quantified the uncertainty in orbital state, illustrating the importance of analyzing uncertainty. (R. Licata et al., 2021) investigated the effects of driver uncertainty on orbital state and found that in-track position error was found to be larger when considering driver uncertainty.

A set of driver forecasts can be used to form probabilistic driver forecasts. Evaluating statistics across the forecast distribution allows the uncertainty of forecast to be quantified. The neural network ensemble approach in this work generates a combined (single point) forecast as well as a distribution for each driver. It is necessary to evaluate the robustness and reliability of uncertainty estimates similar to work by (Daniell

& Mehta, 2023b). By calculating the calibration error score (CES), a quantitative measure of a model's ability to provide reliable uncertainty estimates can be provided. The CES metric, originally by (Anderson et al., 2020), is modified by (R. J. Licata et al., 2022) for use in uncertainty quantification. CES quantifies the average deviation from perfect calibration in percentage, averaged across each output and is shown as,

$$CES = \frac{100\%}{r * m} \sum_{i=1}^{r} \sum_{j=1}^{m} \mid p(\alpha_{i,j}) - p(\hat{a}_{i.j}) \mid \tag{9}$$

where r is the number of model outputs, m is the number of prediction intervals investigated, p is the expected cumulative probability, and $\hat{p}$ is the observed cumulative probability. A broader explanation of the modified CES metric is given by (R. J. Licata et al., 2022).

We additionally generate a qualitative measure of uncertainty, which we refer to as a calibration curve. A calibration curve is a plot that shows the expected and observed cumulative probability, plotting of p vs $\hat{p}$ from Equation 9. Calibration curves show how well calibrated the uncertainty estimates are at capturing the expected percentage of true samples in the distribution. A model that is perfectly calibrated has a calibration curve with a slope of one. Models which are under or over confident have calibration curves with slopes of less than one or greater than one respectively.

A scaling factor can be applied to adjust the uncertainty estimates, which is referred to as σ-scaling. σ-scaling, introduced by (Laves et al., 2021), uses the validation set to "check" the validity of uncertainty estimates. This check provides a scaling factor based on the results and adjusts uncertainty estimates based on over or under prediction. For example, if a calibration curve shows a tendency to over predict on validation data, then a scaling factor can be generated to "correct" the uncertainty estimates. The scaling factor is generated as follows,

$$\sigma_S = \sqrt{S} * \sigma = \sqrt{\frac{1}{N} \sum_{i=1}^{N} \sigma_i^{-2} * (y - \hat{y})^2 * \sigma} \tag{10}$$

where S is the scaling factor , N is the number of samples in the validation set, σ_i is the sample standard deviation at step i, σ_S is the scaled standard deviation and $(y - \hat{y})^2$ is the squared error of prediction.

Another method for combining models, ensemble model output statistics (EMOS) was introduced by (Gneiting et al., 2005). EMOS is a post-processing technique which addresses forecast bias, underdispersion, and spread-skill relationship. EMOS relies on linear regression, to yield a probabilistic forecast; formed by a Gaussian predictive probability density function (PDF). The general form of the Gaussian predictive distribution,

$$\mathcal{N}(a + b_1 X_1 + ... + b_m X_m, c + dS^2) \tag{11}$$

where a, b_i, c, and d are regression coefficients, X_i are individual model forecasts, and S^2 is the ensemble variance. EMOS uses either the continuous ranked probability score (CRPS) or ignorance (IGN) scoring, to determine the linear regression coefficients. This distribution provides a probabilistic forecast which may outperform both the raw output and σ-scaling methods. EMOS techniques are investigated for their potential well calibrated probabilistic forecasts. With a method of providing the skill of a neural network ensemble, we can now provide a comparison of the operational method to the method discussed in this work.

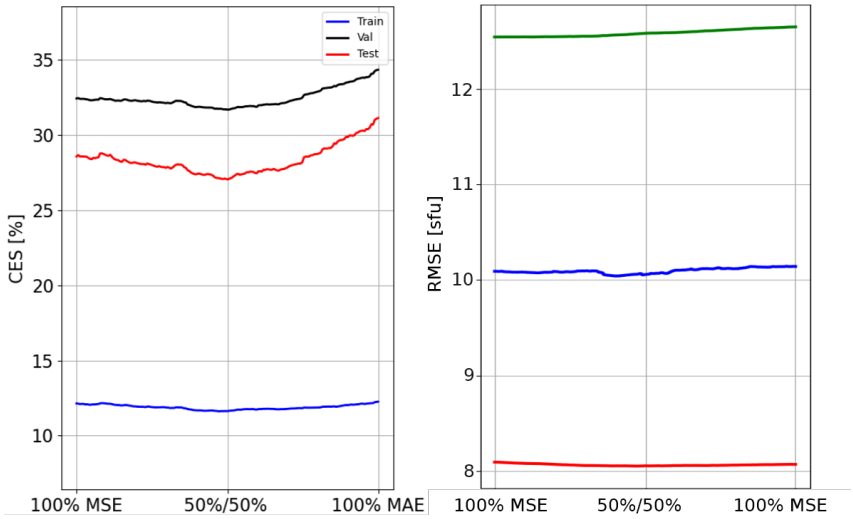


Figure 6: **Left:** A mixture of loss functions provide better calibrated uncertainty estimates. **Right:** The RMSE metric is not nearly as sensitive to loss function but may benefit from other sources of diversity.

3 Results

3.1 Diversity Through Loss Function

As a potential source of ensemble diversity, the optimization loss is investigated. By evaluating models with different loss functions, we can identify if the introduced diversity provides enhanced forecasts. We perform a sweep analysis by creating an ensemble composed of models with different loss functions. For this investigation, a model ensemble of size 100 was constructed using the 10 best architectures from KerasTuner, which were trained 10 times, with random weight initialization.

It is clear from Figure 6, introducing diversity to an ensemble by augmenting the learning process indeed changes performance and UQ metrics. We see a clear minimum appear in the CES metric. Using an equal contribution from MSE and MAE models, we can improve the calibration of the ensemble model. To produce reliable probabilistic forecasts, we seek to minimize the CES metric. By using a neural network model ensemble with varied loss functions, we decrease the CES metric by about 1% in the training and validation sets, as well as nearly 2% in the testing set. Performance metrics seen in Figure 6 indicates that diversity, by way of loss function variation, does not contribute much to the combined forecast error. Due to the improvement seen in CES and insignificant changes in performance error metrics; we opt for a neural network model ensemble which is constructed using an equal split of models trained with MSE and MAE loss functions.

3.2 Ensemble Member Combination Methods

Although a probabilistic forecast provides a distribution, it is critical to provide single point, or combined forecast values. The individual models are combined to improve overall prediction. To provide an improved single point forecast, considerations for how best to combine models were needed. In previous work, (Daniell & Mehta, 2023b) showed mathematical average could be used to combine ensemble members effectively. However, we believe it is necessary to investigate methods for a combined forecast other than averaging. Using both a stacked ensemble and by combining predictions using the

mathematical median, we determine more sophisticated methods, which outperform the previously used average prediction.

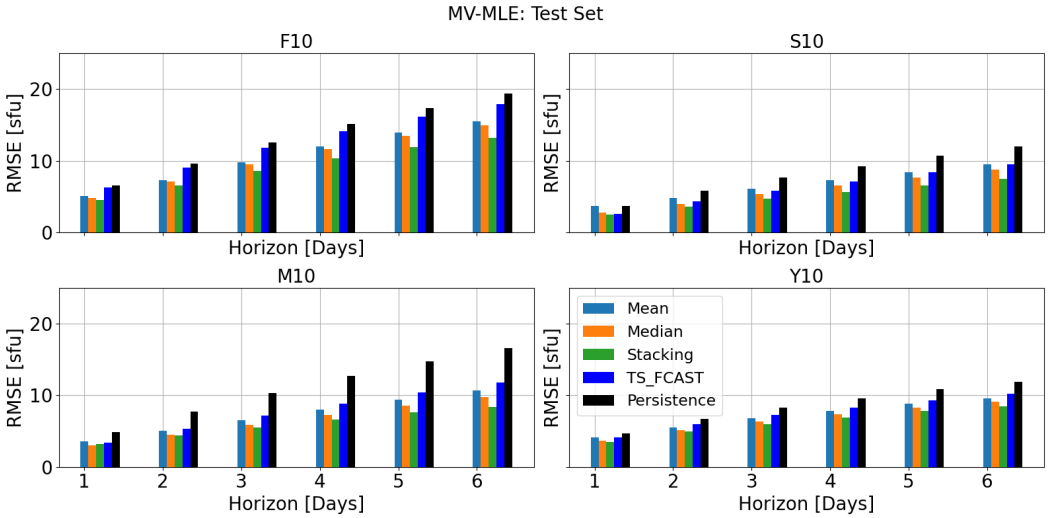


Figure 7: Non-mean combination of ensemble methods (orange & green) show improved errors over all horizons when compared to the persistence baseline and linear TS_FCAST methods.

We explore the use of mean, median, and stacking for the predictions made by the individual members of the MV-MLE model on the test set data. Seen in Figure 7, it is clear that, when averaged, the MV-MLE model is able to provide better predictions across all horizons when compared to both the persistence baseline and the linear SET algorithm. We find that the averaged prediction works well, but using methods such as median prediction or stacking, further improvements are seen the performance metrics. We believe that by using the median predicted values, we eliminate the outliers that may occur when considering the mean prediction.

We perform a stacking approach using the validation set, providing a weigh associated with each model. Once the models have been combined using these weights, we see a dramatic increase in performance in nearly all drivers and horizons. We see that stacking increases performance at larger horizons more so than smaller horizons. This result may indicate that some models are better learners at larger horizons and have an associated larger weight. We see that predictions of $S_{10.7}$ using mean or TS_FCAST provide similar errors, it is not until stacking is performed do we see noticeable improvements.

When using median and stacking, we see a considerable improvement over the mean approach, see Table 2. When RMSE is averaged over the 6 day horizon, we see improvement in all drivers. Stacking outperforms median combination on the training, validation, and test sets. It should be noted that the significant improvement seen in the validation set is expected. This is due to the procedure used to create model weights, stacking weights were created to fit the validation data and therefore would perform well. Both the training and test can be considered better measures of true stacking performance. When compared to the mean combination, the lowest improvement over seen by stacking is 0.57 SFU; while the best improvements reach 2.19 SFU.

Table 2: The RMSE metric was averaged over the 6 day horizon for each combination method. The difference between the mean prediction and both median and stacking approaches are reported. Negative values indicated an improvement over the mean combination method, with bold indicating favorable values.

Combination Method	**Difference vs. Mean (RMSE)**			
	F_{10}	S_{10}	M_{10}	Y_{10}
Training Set				
Median	-0.28	-0.80	-0.64	-0.40
Stacking	**-0.65**	**-1.33**	**-1.12**	**-0.57**
Validation Set				
Median	-0.29	-0.79	-0.63	-0.37
Stacking	**-2.19**	**-1.86**	**-1.87**	**-1.24**
Test Set				
Median	-0.36	-0.79	-0.70	-0.47
Stacking	**-1.40**	**-1.57**	**-1.23**	**-0.86**

Due to the decrease in error; we consider the stacking approach to be the most useful method for combining our ensemble members. Since a validation or training set has already been "seen" by the models, we believe that the stacking approach can utilize such sets for another step in the machine learning process. A linear regression model is used to "learn" the best combination method for the neural network models; referred to as a *meta learner*. We show that with stacking, we can effectively use another machine learning step to enhance predictions. Based on the improvements in performance based error metrics, we select stacking as the preferred method for combining models.

3.3 Comparison of Forecasting Methods

We consider neural network models which have been trained using striped validation data, with PCA and non-PCA input data, MSE and MAE training loss functions, and have been combined via stacking. Due to the statistical similarity between the split datasets, we consider the test set as a primary indicator for model performance. The testing set has been completely hidden during training and stacking, and can be used to effectively measure performance error metrics. We aim to establish a preferred method for forecasting; one driver at a time, or simultaneously. Relative metrics for test set predictions are compared in Table 3.

Table 3 clearly shows that ensemble approaches, specifically MV-MLE outperform linear methods. The MV-MLE approach, with standard or PCA inputs, provide significant improvement over the SET method. When using MV-MLE with non-PCA inputs, we see an improvement of RMSE for $F_{10.7}$, $S_{10.7}$, $M_{10.7}$, and $Y_{10.7}$ of 17.7%, 12.3%, 13.8%, 13.7% respectively, over the SET method. It is clear that the SET method outperforms persistence and ensemble methods further improve on the SET method.

Transfer learning methods also perform well, an improvement is seen over the SET method in all cases. This improvement indicates that architectures and models developed for the $F_{10.7}$ driver can be applied to the other drivers, with adequate training and fine tuning of weights. Interestingly, we see a dramatic difference between the transfer learning and UV-MLE methods. We believe that the performance difference between these methods can be attributed to the amount of data available for training. The models developed originally for univariate forecasting of $F_{10.7}$ had substantially more data to de-

Table 3: Relative metric comparison of the SET linear method and stacked ensemble approaches on the test set. Metrics are scaled against the persistence baseline, and averaged over forecast horizons. Lower error metrics and higher correlation metrics are preferred, with a value of one exhibiting the same performance as persistence. The best performing values in each metric are highlighted in bold.

Driver	Relative Metric	SET	Transfer Learning	UV-MLE	MV-MLE	MV-MLE (PCA)
$F_{10.7}$	RMSE	0.927	0.799	0.911	**0.75**	0.773
	MAPE	0.939	0.823	0.904	**0.771**	0.805
	R	1.005	1.024	1.013	**1.029**	1.028
$S_{10.7}$	RMSE	0.854	0.735	0.738	0.731	**0.703**
	MAPE	0.835	0.758	0.755	0.803	**0.736**
	R	1.005	1.008	1.008	**1.01**	1.009
$M_{10.7}$	RMSE	0.761	0.646	0.751	0.623	**0.596**
	MAPE	0.771	0.687	0.764	0.658	**0.651**
	R	1.019	1.026	1.021	**1.029**	**1.029**
$Y_{10.7}$	RMSE	0.971	0.836	0.999	0.834	**0.832**
	MAPE	0.996	0.865	1.136	**0.863**	0.87
	R	1.003	1.009	1.002	**1.01**	1.009

velop models before being applied to the new drivers, while the UV-MLE models were limited to a much smaller historic dataset. It should be noted that comparison with univariate methods provides an indirect comparison to SWPC methods for $F_{10.7}$. UV-MLE methods could outperform transfer learning but may require a greater amount of historic data for $S_{10.7}$, $M_{10.7}$, and $Y_{10.7}$, to encourage ML efforts.

The ensemble requiring PCA rotated inputs seems to outperform the standard MV-MLE ensemble on the test set, for $S_{10.7}$ and $M_{10.7}$, while the standard MV-MLE performs better on $F_{10.7}$. The methods perform similarly on $Y_{10.7}$, with a difference of only 0.2% RMSE and 0.7% MAPE.

We show an additional comparison between the best performing, MV-MLE stacked ensemble approaches, seen in Table 4. Neither method stands out, when considering relative error metrics alone. We cannot definitively say whether PCA or non-PCA ensemble is preferred for probabilistic forecasting. We must instead look to uncertainty quantification to determine if one method is the better.

3.4 Quantified Uncertainties

The probabilistic forecasts are evaluated on the three datasets, seen in Table 5. We see that the CES varies across drivers; the calibration error scores associated with $F_{10.7}$ and $Y_{10.7}$ are smaller, while $S_{10.7}$ and $M_{10.7}$ have slightly larger CES values. In general, the raw outputs from both non-PCA and PCA inputs produce reasonable CES values. Regarding the effectiveness of σ-scaling, we notice that cases where CES is relatively large to begin with, are improved by scaling efforts. This can be seen most prominently in the $S_{10.7}$ driver, indicating that $S_{10.7}$ may benefit more from σ-scaling than other drivers. When calibrations are good to begin with, σ-scaling seems detrimental and leads to worse CES values.

We found that the EMOS method for probabilistic forecasting yielded unfavorable CES metrics for the multivariate ensemble methods. Due to the already reasonable cal-

Table 4: Relative metric comparison of the SET linear method and MV-MLE approaches on the training and validation sets. Metrics are scaled against the persistence baseline, and averaged over forecast horizons. Lower error metrics and higher correlation metrics are preferred, with a value of one exhibiting the same performance as persistence. The best performing values in each metric are highlighted in bold.

Driver	Relative Metric	SET	MV-MLE	MV-MLE (PCA)
Training Set				
$F_{10.7}$	RMSE	0.964	**0.632**	0.642
	MAPE	0.975	0.685	**0.677**
	R	1.008	1.043	**1.044**
$S_{10.7}$	RMSE	0.79	0.621	**0.585**
	MAPE	0.806	0.705	**0.633**
	R	1.007	1.012	1.011
$M_{10.7}$	RMSE	0.701	0.53	**0.504**
	MAPE	0.725	0.569	**0.555**
	R	1.024	**1.035**	1.034
$Y_{10.7}$	RMSE	0.983	**0.723**	0.73
	MAPE	0.985	**0.762**	0.771
	R	1.003	**1.013**	**1.1013**
Validation Set				
$F_{10.7}$	RMSE	0.96	**0.730**	0.759
	MAPE	0.982	**0.767**	0.794
	R	1.01	1.03	**1.028**
$S_{10.7}$	RMSE	0.811	0.745	**0.715**
	MAPE	0.783	0.814	**0.779**
	R	1.006	**1.009**	1.008
$M_{10.7}$	RMSE	0.704	0.64	**0.609**
	MAPE	0.715	0.681	**0.652**
	R	1.023	**1.029**	**1.029**
$Y_{10.7}$	RMSE	0.964	**0.835**	0.851
	MAPE	0.966	**0.863**	0.887
	R	1.003	**1.009**	1.008

ibration of the MV-MLE and MV-MLE (PCA) ensembles, the application of EMOS did not help. EMOS generally worsened CES metrics, ranging from 6-18%. We believe that potentially, the number of coefficients necessary for EMOS may have caused the poor performance. To apply EMOS, regression for every model and output is needed (180 models and 24 outputs). This high number of terms may have caused typical linear regression to fail, as EMOS has been typically used on much fewer outputs and models.

We choose to not apply σ-scaling or EMOS methods to the probabilistic forecasts. The application of σ-scaling helped marginally for some drivers but significantly worsened CES values for other drivers, and EMOS provided poor CES values overall. Direct use of the ensemble member outputs is a more intuitive method, and provides reasonable CES values. Additionally, no uncertainty scaling or regression must be performed after the prediction step, and no scaling terms need to be calculated. With no need for

Table 5: Calibration error score (CES) for ensemble methods when evaluated on all datasets. Lower values are better and bold terms indicate the best method for a given driver.

Train Set Driver	**MV-MLE** (Raw Output)	**MV-MLE (PCA)** (Raw Output)	**MV-MLE** (σ-scaled)	**MV-MLE (PCA)** (σ-scaled)
$F_{10.7}$	**1.92**	3.00	8.76	8.39
$S_{10.7}$	9.94	11.26	9.22	**8.15**
$M_{10.7}$	8.55	**5.59**	7.02	8.17
$Y_{10.7}$	3.93	**2.34**	3.89	5.13
Validation Set Driver	**MV-MLE** (Raw Output)	**MV-MLE (PCA)** (Raw Output)	**MV-MLE** (σ-scaled)	**MV-MLE (PCA)** (σ-scaled)
$F_{10.7}$	1.9	**1.87**	8.58	8.61
$S_{10.7}$	9.21	10.67	8.68	**8.41**
$M_{10.7}$	7.53	**5.50**	6.73	7.78
$Y_{10.7}$	3.70	**2.07**	3.64	4.78
Test Set Driver	**MV-MLE** (Raw Output)	**MV-MLE (PCA)** (Raw Output)	**MV-MLE** (σ-scaled)	**MV-MLE (PCA)** (σ-scaled)
$F_{10.7}$	3.08	**2.62**	8.48	8.57
$S_{10.7}$	8.64	10.63	8.36	**8.62**
$M_{10.7}$	7.74	**5.63**	6.66	8.33
$Y_{10.7}$	**3.18**	6.27	4.04	6.54

extra processing, using the direct predictions directly is less computationally expensive and can be considered quicker.

We choose the test set for evaluation since it has been unseen during training and is statistically similar to both the training and validation sets. The direct model outputs lead to the calibration curves seen in Figure 8. The MV-MLE (green) and UV-MLE (orange) models follow the same trends; methods are well calibrated for smaller confidence intervals, with a tendency to over predict when the confidence interval grows for $S_{10.7}$, $M_{10.7}$, and $Y_{10.7}$. The $F_{10.7}$ driver is very well calibrated, with only a minor tendency to under predict at very large confidence intervals. Neither MV-MLE or MV-MLE (PCA) stand out when examining the calibration curves. When averaged across all four drivers; the MV-MLE (PCA) ensemble yields an improvement in CES of only 0.16%. Based on the marginal differences, both methods can be considered useful, with a slight edge to MV-MLE (PCA) for uncertainty estimates.

4 Summary and Conclusions

In this work, the ability for neural network ensembles to provide simultaneous probabilistic forecasts for all four solar drivers used by the operational HASDM was investigated. A comparison between neural network ensemble methods and the currently used forecasting method was performed. Single point forecasting was significantly improved when compared to the linear algorithm used by SET. Due to the high correlation between solar drivers, transfer learning was investigated. Transfer learning leveraged models previously used in forecasting the $F_{10.7}$ driver and provided improvements over the SET method; reinforcing the ability for models to learn across solar drivers. The MV-MLE approach

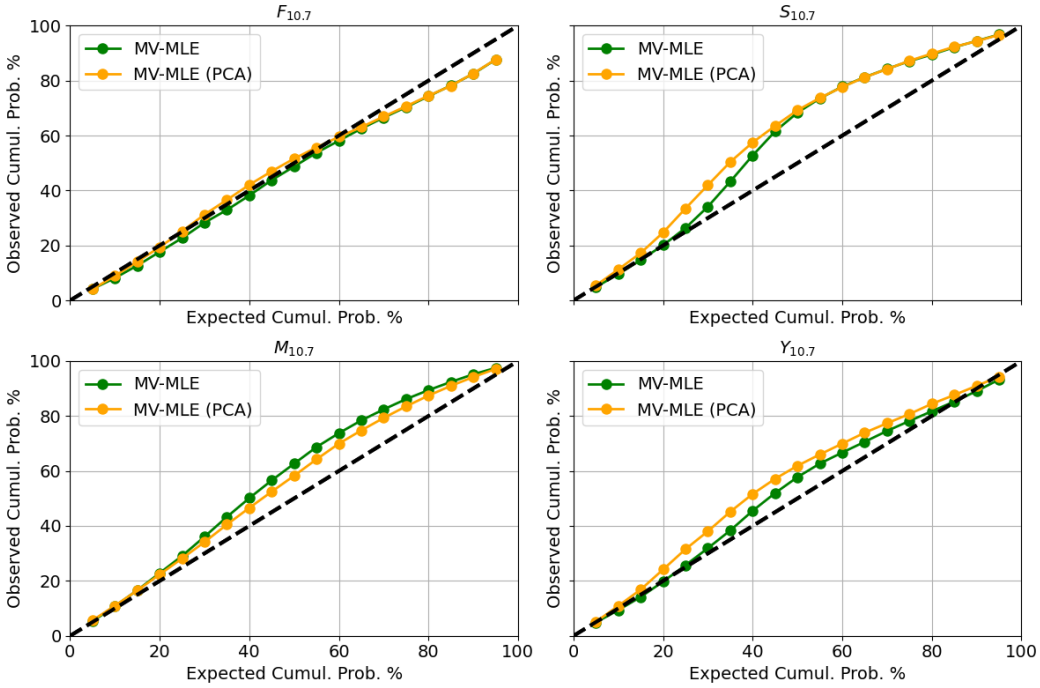


Figure 8: Evaluation of the multivariate ensemble methods on the test set. Curves above or below the 45^o line indicate over predicted uncertainty and under predicted uncertainty respectively. Curves closer to the 45^o dashed line are desired.

provided the most significant improvement of RMSE. RMSE of $F_{10.7}$, $S_{10.7}$, $M_{10.7}$, and $Y_{10.7}$ improved by 17.7%, 12.3%, 13.8%, 13.7% respectively when compared to the currently used method.

We provide a novel method for simultaneous probabilistic forecasting of $F_{10.7}$, $S_{10.7}$, $M_{10.7}$, and $Y_{10.7}$. Previously, no probabilistic forecasting methods existed for $S_{10.7}$, $M_{10.7}$, and $Y_{10.7}$. The probabilistic methods are well calibrated when using directly output predictions; providing an average CES of 5.63%, across all drivers. It is clear that simultaneous forecasting of drivers offers an improvement over univariate methods. We believe that multivariate neural network methods are useful for forecasting space weather indices.

To effectively apply ensemble methods, we introduce a new method called striped validation to create statistically similar training, validation, and test data sets. We believe that creating statistically similar data sets is critical for machine learning approaches in space weather, due to inadequate historic driver data. Without such splitting, models performed poorly and biases to certain solar activity levels were encountered. We additionally find that multivariate forecasting using PCA inputs offers little improvement over non-PCA; an average CES improvement of 0.16% over all drivers and sets is seen.

To investigate the diversity of our ensemble members, we varied the loss function used during training. We found that by using a mixture of MAE and MSE losses, we achieved similar performance metrics, with an improved calibration error score. We select an equal mixture of MAE and MSE models due to the minimal CES seen during a sweep analysis. Our analysis of ensemble diversity, by variation of loss function, is supported by the conclusions made by (Daniell & Mehta, 2023b).

Methods for combining individual predictions were investigated in this work. Median combination provided an improvement over traditional averaging. A stacked ensemble approach provided significant improvement over the traditionally used equal weighting, or average method. We used multiple linear regression on validation data to create weights associated with each model and output. A weighted combination of stacked models reduced the RMSE by an average of 1.22 SFU, when compared to the traditional averaging method.

The ensemble approaches used in this work allow a user to sample from a probabilistic range of values, rather than use a single deterministic value (like TS_FCAST). By forecasting a range of solar driver values, a prediction could contain both a combined forecast and associated uncertainty bounds, which creates a more robust and operationally useful forecast. Our novel ensemble method provides an improved forecast over the SET method and provides, for the first time, an approach capable of providing robust and reliable uncertainty estimates for $S_{10.7}$, $M_{10.7}$, and $Y_{10.7}$.

5 Future Work

A key requirement for application of machine learning techniques is access to adequate data sets, which are used to learn from. Due to the relatively small datasets for $S_{10.7}$, $M_{10.7}$, and $Y_{10.7}$, a difficulty is seen in training models effectively while using traditional methods. A potential solution for data depends on the time resolution of captured data. Daily values for all solar drivers are issued, but an enhanced resolution (less than 1 day) would provide ML techniques with more data to determine hidden patterns. Geomagnetic indices such as *DST* and *ap*, have a time resolution of 3 hours, a similar time cadence of solar driver measurement would yield an 8 fold increase in collected data. Such an increase in data would be expected to lead to dramatic and quick improvements in neural network modeling efforts.

With the constant advancements taking place in the field of machine learning, new time series techniques are constantly developed and should be investigated. Developed by (Vaswani et al., 2017), transformers are a current state of the art method for sequential data prediction. These models rely on self attention, multi-head attention, and positional encoding to enhance predictions. Transformers were briefly investigated in forecasting of solar drivers but did not provide noticeable improvement over the SET method. As a state of the art method, and constantly advancing topic, we believe further investigation into transformers may be beneficial for forecasting not just solar drivers, but all space weather indices.

While the methods presented in this work have demonstrated their superiority over the operational linear approach for short-term forecasting, it remains crucial to assess the efficacy of multivariate methods for longer-term forecasts. As of now, forecasts for $S_{10.7}$, $M_{10.7}$, and $Y_{10.7}$ use a six-day horizon, but future work extending the probabilistic forecast horizon may enhance our ability to plan for events such as re-entry and other critical activities.

Providing robust and reliable uncertainty estimates and probabilistic input drivers to JB2008 may provide improved density modeling and density uncertainty estimates. We believe that in the future, it is necessary to investigate the coupling between solar driver uncertainty and density modeling uncertainty. We believe that a framework which couples the major sources of uncertainty with orbit propagation is necessary, and would allow for more robust and reliable uncertainty estimates for the position and trajectories of tracked objects in LEO. An overall framework which links the solar driver uncertainty methods in this work with model uncertainty and predicted orbital state uncertainty should be a major focus for providing improved drag modeling with the operational HASDM.

6 Open Research

Software and data related to model development, data processing, figures, and method comparison are available as a Zenodo repository (Daniell & Mehta, 2023a). Additionally, this repository contains an example of forecasting using pre-trained models(Daniell & Mehta, 2023a). The repository does not requires registration for access and is licensed under Creative Commons Attribution 4.0 International; version information can be found in the repository.

SET proprietary data used for verification of the SET algorithm are not made publicly available since they reside on operational servers run for the sole benefit of the USAF. Data are provided courtesy of Space Environment Technologies, 2019. These data have been provided to West Virginia University with license to use for scientific research. However, the verified TS_FCAST algorithm used by this work and accompanying forecast files used for this work are available within the repository, located within the Zenodo repository at URL `https://doi.org/10.5281/zenodo.10063536`

The JB2008 solar and geomagnetic indices are provided for scientific use courtesy of Space Environment Technologies and are available at `https://spacewx.com/jb2008/` (Space Environment Technologies, 2023). Figures were made with Matplotlib version 3.5.2 (Caswell et al., 2022) available under the Matplotlib license at `https://matplotlib.org/`.

Acknowledgments

This research is based upon work supported in part by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via 2023-23060200005. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported.

References

Abdi, H., & Williams, L. J. (2010, jul). Principal component analysis. *WIREs Computational Statistics*, *2*(4), 433–459. doi: 10.1002/wics.101

Anderson, G. J., Gaffney, J. A., Spears, B. K., Bremer, P.-T., Anirudh, R., & Thiagarajan, J. J. (2020). Meaningful uncertainties from deep neural network surrogates of large-scale numerical simulations. *arXiv preprint arXiv:2010.13749*.

Benson, B., Brown, E., Bonasera, S., Acciarini, G., Pérez-Hernández, J. A., Sutton, E., ... Baydin, A. G. (2021, December). Simultaneous multivariate forecast of space weather indices using deep neural network ensembles. doi: 10.48550/ARXIV.2112.09051

Bhandari, H. N., Rimal, B., Pokhrel, N. R., Rimal, R., Dahal, K. R., & Khatri, R. K. (2022, sep). Predicting stock market index using LSTM. *Machine Learning with Applications*, *9*, 100320. doi: 10.1016/j.mlwa.2022.100320

Bowman, B., Tobiska, W. K., Marcos, F., Huang, C., Lin, C., & Burke, W. (2008, jun). A new empirical thermospheric density model JB2008 using new solar and geomagnetic indices. In *AIAA/AAS astrodynamics specialist conference and exhibit*. American Institute of Aeronautics and Astronautics. doi: 10.2514/6.2008-6438

Brown, G., Wyatt, J. L., Tino, P., & Bengio, Y. (2005). Managing diversity in regression ensembles. *Journal of machine learning research*, *6*(9).

Caswell, T. A., Droettboom, M., Lee, A., de Andrade, E. S., Hoffmann, T., Klymak, J., ... Ivanov, P. (2022, May). *matplotlib/matplotlib: Rel: v3.5.2 [software].* Zenodo. Retrieved from `https://doi.org/10.5281/zenodo.6513224` doi: 10.5281/zenodo.6513224

Daniell, J. D., & Mehta, P. M. (2023a, November). *Probabalistic Short-Term Solar Driver Forecasting with Neural Network Ensembles Software and Data.* [Software]. Zenodo. Retrieved from `https://doi.org/10.5281/zenodo.10063536` doi: 10.5281/zenodo.10063536

Daniell, J. D., & Mehta, P. M. (2023b, sep). Probabilistic solar proxy forecasting with neural network ensembles. *Space Weather*, *21*(9). doi: 10.1029/2023sw003675

Elvidge, S., Granados, S. R., Angling, M. J., Brown, M. K., Themens, D. R., & Wood, A. G. (2023, mar). Multi-model ensembles for upper atmosphere models. *Space Weather*, *21*(3). doi: 10.1029/2022sw003356

Gerace, F., Saglietti, L., Mannelli, S. S., Saxe, A., & Zdeborová, L. (2022, feb). Probing transfer learning with a model of synthetic correlated datasets. *Machine Learning: Science and Technology*, *3*(1), 015030. doi: 10.1088/2632-2153/ac4f3f

Gneiting, T., Raftery, A. E., Westveld, A. H., & Goldman, T. (2005, may). Calibrated probabilistic forecasting using ensemble model output statistics and minimum CRPS estimation. *Monthly Weather Review*, *133*(5), 1098–1118. doi: 10.1175/mwr2904.1

Hansen, L., & Salamon, P. (1990). Neural network ensembles. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *12*(10), 993–1001. doi: 10.1109/34.58871

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735-1780. doi: 10.1162/neco.1997.9.8.1735

Hu, Z., Pan, G., Wang, Y., & Wu, Z. (2016). Sparse principal component analysis via rotation and truncation. *IEEE Transactions on Neural Networks and Learning Systems*, *27*(4), 875-890. doi: 10.1109/TNNLS.2015.2427451

Huang, C., Liu, D.-D., & Wang, J.-S. (2009, jun). Forecast daily indices of solar activity, f10.7, using support vector regression method. *Research in Astronomy and Astrophysics*, *9*(6), 694. Retrieved from `https://dx.doi.org/10.1088/1674-4527/9/6/008` doi: 10.1088/1674-4527/9/6/008

Karevan, Z., & Suykens, J. A. (2020, may). Transductive LSTM for time-series prediction: An application to weather forecasting. *Neural Networks*, *125*, 1–9. doi: 10.1016/j.neunet.2019.12.030

Laves, M.-H., Ihler, S., Fast, J. F., Kahrs, L. A., & Ortmaier, T. (2021, April). *Recalibration of aleatoric and epistemic regression uncertainty in medical imaging.* arXiv. doi: 10.48550/ARXIV.2104.12376

Licata, R., Mehta, P., & Tobiska, W. K. (2021, 02). Impact of driver and model uncertainty on drag and orbit prediction..

Licata, R. J., Mehta, P. M., Tobiska, W. K., & Huzurbazar, S. (2022, apr). Machine-learned HASDM thermospheric mass density model with uncertainty quantification. *Space Weather*, *20*(4). doi: 10.1029/2021sw002915

Licata, R. J., Tobiska, W. K., & Mehta, P. M. (2020). Benchmarking forecasting models for space weather drivers. *Space Weather*, *18*(10), e2020SW002496. Retrieved from `https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2020SW002496` (e2020SW002496 10.1029/2020SW002496) doi: https://doi.org/10.1029/2020SW002496

Liu, L., Zou, S., Yao, Y., & Wang, Z. (2020, nov). Forecasting global ionospheric TEC using deep learning approach. *Space Weather*, *18*(11). doi: 10.1029/2020sw002501

Luo, J., Zhu, L., Zhang, K., Zhao, C., & Liu, Z. (2022). Forecasting the 10.7-cm solar radio flux using deep cnn-lstm neural networks. *Processes*, *10*(2).

Retrieved from https://www.mdpi.com/2227-9717/10/2/262 doi: 10.3390/pr10020262

Paul, S. N., Licata, R. J., & Mehta, P. M. (2023, mar). Advanced ensemble modeling method for space object state prediction accounting for uncertainty in atmospheric density. *Advances in Space Research*, *71*(6), 2535–2549. doi: 10.1016/j.asr.2022.12.056

Qureshi, A. S., Khan, A., Zameer, A., & Usman, A. (2017, sep). Wind power prediction using deep neural network based meta regression and transfer learning. *Applied Soft Computing*, *58*, 742–755. doi: 10.1016/j.asoc.2017.05.031

Sola, J., & Sevilla, J. (1997, jun). Importance of input data normalization for the application of neural networks to complex industrial problems. *IEEE Transactions on Nuclear Science*, *44*(3), 1464–1468. doi: 10.1109/23.589532

Space Environment Technologies. (2023). *Solfsmy.* [Dataset]. (https://sol.spacenvironment.net/JB2008/indices/SOLFSMY.TXT)

Sridhar, D. V., Seagrave, R. C., & Bartlett, E. B. (1996, sep). Process modeling using stacked neural networks. *AIChE Journal*, *42*(9), 2529–2539. doi: 10.1002/aic.690420913

Stevenson, E., Rodriguez-Fernandez, V., Minisci, E., & Camacho, D. (2022). A deep learning approach to solar radio flux forecasting. *Acta Astronautica*, *193*, 595-606. Retrieved from https://www.sciencedirect.com/science/article/pii/S009457652100415X doi: https://doi.org/10.1016/j.actaastro.2021.08.004

Svalgaard, L., & Hudson, H. S. (2010). *The solar microwave flux and the sunspot number.* arXiv. doi: 10.48550/ARXIV.1003.4281

Tapping, K. F. (2013). The 10.7 cm solar radio flux (f10.7). *Space Weather*, *11*(7), 394-406. Retrieved from https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1002/swe.20064 doi: https://doi.org/10.1002/swe.20064

Tobiska, K. W., Bowman, B., & Bouwer, D. S. (2009). Solar and geomagnetic indices for the jb2008 thermosphere density model. *Space Environment Technologies Publication*.

Tobiska, W. K., Bouwer, S. D., & Bowman, B. R. (2008). The development of new solar indices for use in thermospheric density modeling. *Journal of Atmospheric and Solar-Terrestrial Physics*, *70*(5), 803-819. Retrieved from https://www.sciencedirect.com/science/article/pii/S1364682607003720 doi: https://doi.org/10.1016/j.jastp.2007.11.001

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, *30*.

Vourlidas, A., & Bruinsma, S. (2018, jan). EUV irradiance inputs to thermospheric density models: Open issues and path forward. *Space Weather*, *16*(1), 5–15. doi: 10.1002/2017sw001725

Wasserman, P., & Schwartz, T. (1988). Neural networks. II. what are they and why is everybody so interested in them now? *IEEE Expert*, *3*(1), 10–15. doi: 10.1109/64.2091

Weiss, K., Khoshgoftaar, T. M., & Wang, D. (2016, may). A survey of transfer learning. *Journal of Big Data*, *3*(1). doi: 10.1186/s40537-016-0043-6

Xu, C., Lu, C., Liang, X., Gao, J., Zheng, W., Wang, T., & Yan, S. (2016, dec). Multi-loss regularized deep neural network. *IEEE Transactions on Circuits and Systems for Video Technology*, *26*(12), 2273–2283. doi: 10.1109/tcsvt.2015.2477937

Yaya, P., Hecker, L., de Wit, T. D., Fèvre, C. L., & Bruinsma, S. (2017). Solar radio proxies for improved satellite orbit prediction. *Journal of Space Weather and Space Climate*, *7*, A35. doi: 10.1051/swsc/2017032