ARTICLE IN PRESS

Available online at www.sciencedirect.com

ScienceDirect

Advances in Space Research xxx (xxxx) xxx

ADVANCES IN SPACE RESEARCH (a COSPAR publication)

www.elsevier.com/locate/asr

A stacked machine learning model for the vertical total electron content forecasting

Eric Nana Asamoah [a,b], Massimo Cafaro [a,b,*], Italo Epicoco [a], Giorgiana De Franceschi [b], Claudio Cesaroni [b]

[a] Department of Engineering for Innovation, University of Salento - Via Per Monteroni, Lecce 73100, Italy
[b] Istituto Nazionale di Geofisica e Vulcanologia, Via di Vigna Murata 605, Roma 00142, Italy

Received 5 March 2024; received in revised form 20 April 2024; accepted 27 April 2024

Abstract

We design four Artificial Neural Network (ANN) models, namely a Multilayer Perceptron (MLP), a Convolutional Neural Network (CNN), an Extreme Learning Machine network (ELM) and an ensemble model based on stacking (STACKED) to forecast the ionospheric vertical Total Electron Content (vTEC) from 1 to 24 hours in advance on a single station at mid latitude. The time series used, spanning from January 2006 to December 2018, includes vTEC, provided by the Global Navigation Satellite System (GNSS) receiver at Tsukuba (TSKB), Japan (36.06^o N, 140.05^o E) and suitable external drivers selected among several helio-geophysical parameters. The selection of appropriate external drivers is made by rating their relevance on the vTEC at forecasting timescales (from 1 to 24 hours). The process, based on eleven machine learning models, highlights that the most important external drivers are: the 10.7 cm Solar Flux (F10.7), the magnitude B_T of the Interplanetary Magnetic Field (IMF), and the Auroral Electrojet (AE) index. The forecasting performance of the four models (MLP, CNN, ELM, STACKED) is then analysed. The analysis relies on three statistical metrics to compare actual and forecasted vTEC: the coefficient of determination (R^2), the Mean Absolute Error (MAE), and the Root Mean Squared Error (RMSE). Additionally, descriptive statistics are presented using box and whisker plots. The four ANN models show a quite satisfactory capability to forecast vTEC when applied to the test dataset which represents 10% of the available data from 2006 to 2018. Furthermore, by conducting a Wilcoxon signed rank test, it is shown that statistically significant improvements are achieved by the STACKED model with regard to MLP, CNN and ELM. On average, by analysing the forecasted (from 1 to 24 hours in advance) vs the actual vTEC, the STACKED model achieves $R^2 = 0.816, RMSE = 0.426$ TECu, and $MAE = 0.296$ TECu (1 TECunit $= 10^{16}$ electrons/m^2) whilst MLP, CNN and ELM show respectively $R^2 = 0.808, 0.812, 0.803, RMSE = 0.436$ TECu, 0.431 TECu, 0.441 TECu and $MAE = 0.304$ TECu, 0.299 TECu, 0.312 TECu.

Keywords: Total electron content; Machine learning; Regression; Stacking

1. Introduction

Around 50 km up to 1000 km and higher, a partially ionized gas known as the Earth's ionosphere envelops the globe. It forms as a result of photoionization caused by the Sun's ultraviolet radiation, and as a result, it varies throughout the day and with season, reaching its peak of ionization at local noon. In addition to the "normal" ionospheric diurnal variation, the ionosphere may experience disturbances, as a result of a variety of solar originating phenomena (e.g., the Coronal Mass Ejections (CME)), which propagate to the magnetosphere-ionosphere system

* Corresponding author at: Department of Engineering for Innovation, University of Salento - Via Per Monteroni, Lecce 73100, Italy.
E-mail address: massimo.cafaro@unisalento.it (M. Cafaro).

https://doi.org/10.1016/j.asr.2024.04.055
0273-1177/© 2024 COSPAR. Published by Elsevier B.V.

(Mendillo, 2006) and collectively make up what is known as "Space Weather".

To quantify the level of ionization of the upper atmosphere and its variability especially under Space Weather events, scientists frequently make use of the vertical Total Electron Content (vTEC), derived from the slant TEC (sTEC, i.e. total number of electrons integrated between satellite radio transmitter and the ground-based receiver) obtained by applying a calibration technique to double frequencies GNSS code and phase observables and a mapping function projecting the sTEC onto a perpendicular path by assuming the ionosphere as a thin layer (Cesaroni et al., 2015; Mannucci et al., 1993).

An example of how complex the ionosphere becomes under Space Weather events is given by Ngwira et al. (2019), that investigated vTEC under three interplanetary shocks strucking the Earth's magnetosphere in quick succession on June 22-23, 2015. The authors found that the ionosphere was being driven by a variety of factors, including high-latitude injection, quick penetration electric fields, disturbance dynamo effect, neural winds, and composition changes.

Especially under geospatial disturbed conditions, the ionosphere represents the biggest contributor to the error budget for GNSS positioning applications, with a significant impact on GNSS integrity, accuracy, and availability. An example is given by the investigation of the 6 September 2017 X-Class Solar Flares (SFs) conducted by Yasyukevich et al. (2018), on the effects of the two SFs on GPS, GLONASS, and Galileo-based navigation. It was revealed that the SF did not result in GNSS receivers losing their ability to track signals accurately, whereas the positioning error increased by roughly three times in GPS precise point positioning solution.

As a contribution to countermeasures to mitigate the errors introduced by the ionospheric threats, several vTEC forecasting models have been developed using different approaches. These approaches may be divided into two general types: (i) traditional methods; (ii) Machine Learning (ML) based methods. In turn, traditional methods may be divided into three categories: mathematical, physical, and empirical (Chapman, 1931a; Chapman, 1931b). Mathematical models focus on the description of the distributions/dynamics of ionospheric particles (Nijimbere, 2020), whereas physical model are based on physics principles, for the computation of the distribution of electrons, ions, and neutral gases in the ionosphere (Schunk et al., 1986). Additionally, empirical models are data-driven models that characterize the distribution and behavior of ionospheric parameters using experimental data. The most well-known empirical model of the ionospheric environment is the International Reference Ionosphere (IRI) (Bilitza, 2001). NeQuick, NeQuick-G, IRI-Plas, adaptive autoregressive models are some additional examples of the empirical models used by the scientific community (Nava et al., 2006; Bilitza et al., 2017).

ML models are widely used, and flourishing in several fields of study, including image processing and computer vision for classification, detection, recognition, language translation and regression problems. In recent years, many related works have been done using ML techniques for ionospheric forecasting.

Zewdie et al. (2021), presented a data-driven forecasting of vTEC using a Long-Short Time Memory (LSTM) deep recurrent neural network. In the process of selecting the input parameters to train the algorithm, they used the Random Forest algorithm to perform regression analysis and estimate the importance of input parameters. The relative importance of 34 different parameters - including the solar flux, the solar wind density, the speed of the three components of interplanetary magnetic field, Lyman-alpha, Kp, Dst and the Polar Cap (PC) index - has been analyzed. The LSTM method was applied to forecast the vTEC up to 5 hours ahead. A good forecast was achieved with low RMSE but the RMSE increases as they forecast further into the future.

Ionospheric space weather forecasting is recommended in the study of Mallika et al. (2018) to improve the GNSS's accuracy. They developed a suitable ionospheric forecasting technique to capture, in particular, the ionospheric disturbances. This approach uses a mix of Principal Component Analysis (PCA) and Artificial Neural Network (ANN) techniques to forecast the TEC values of the ionosphere. The authors show that the recommended approach performs more effectively than the earlier models. The downside of the proposed technique is that it requires a large training dataset, which increases computation time and implementation complexity.

Han et al. (2021), spent time figuring out how to predict ionospheric vTEC values at three IGS GNSS monitoring sites at the low-latitude area (16^o S to 10^o S) during times of strong solar activity and magnetic storms. Multilayer perceptron (MLP), LSTM, Adaptive Neuro-Fuzzy Inference System (ANFIS) based on subtractive clustering, and Gradient Boosting Decision Tree (GBDT) were four distinct ML models utilized in this study during high solar activity and magnetic storm times. The findings indicate that ML algorithms outperform the global ionospheric map prediction model, and that the GBDT model is the best performing algorithm among them in ionosphere prediction situations.

An innovative method was used in the study of Cesaroni et al. (2020) to mimic the characteristics of the ionosphere, particularly during disturbed times. The suggested method makes use of the Global Ionospheric Map (GIM), provided by the International GNSS Service (IGS), and then applies a nonlinear autoregressive neural network with external input (NARX) to chosen GIM grid points for the 24-hour single-point vTEC forecasting, taking into account the actual and predicted geomagnetic conditions.

A LSTM neural network was utilized in the work of Liu et al. (2020) to forecast the 256 spherical harmonic (SH) coefficients that are needed to create global ionospheric maps. Multiple input data sets, including historical time series of the SH coefficients, solar extreme ultraviolet (EUV) flux, disturbance storm time (Dst) index, and hour of the day, are utilized to train the LSTM neural network. The LSTM model has been built using a variety of combinations of the aforementioned parameters, and it has been found that the model which uses all four parameters performs the best. The SH coefficients are then predicted using the top-performing LSTM model, and the 256 projected SH coefficients are then utilized to recreate the global hourly vTEC maps. The results show that the first/second hour vTEC RMSE during storm time is 1.27/2.20 TECu and 0.86/1.51 TECu during quiet time, showing that the developed model performs well during both quiet and storm times.

Given the ability of nonlinear modelling, Huang and Yuan (2014) investigated the potential of this approach when applied to forecast vTEC. They suggested a Gaussian mixture model–improved Radial Basis Function (RBF) neural network. The developed RBF network model was trained, validated, and tested using data from stations located at various latitudes, including estimated TEC overhead of GPS ground stations BJFS (39.61^o N, 115.89^o E), WUHN (30.53^o N, 114.36^o E), and KUNM (25.03^o N, 102.80^o E) for six months in 2011. By contrasting the RBF network model's performance with that of a conventional multi-layer feedforward network trained using the back propagation (BP) technique in terms of predetermined error criteria, it was determined how well it performed. According to the findings of the prediction, the absolute error within 1.5 TECu is above 90% at stations BJFS and WUHN and drops to 75% at station KUNM, which is located at a low latitude.

With the research mentioned above and others, ML demonstrates its superior performance compared to traditional methods. However, certain elements need to be taken into account in order to further enhance the performance of ML models over longer predicting horizons. The fact that the lag between the solar cycle, geomagnetic forcing, and the ionosphere's reaction is not well understood with regard to predicting horizons is a definite factor to be taken into account. Again, a number of ML models have demonstrated positive outcomes, and each of them has distinct qualities that are completely tapped into. An example, the MLP (McCulloch and Pitts, 1943; Theodoridis, 2015), is the most basic neural network with the capability to solve complex nonlinear problems, can handle large amounts of input data and can achieve high accuracy (e.g., (Mallika et al., 2018)). The convolution step in a Convolutional Neural Network (CNN) extracts and reduces the number of parameters via weight sharing (Theodoridis, 2015). The Extreme Learning Machine (ELM) model provides good generalization performance and learns faster than models trained using the backpropagation algorithm (Huang et al., 2006; Huang et al., 2011; Huang, 2014; Huang, 2015).

In this work, we propose a novel method that could be used for ionospheric forecast. The goal is to forecast vTEC from 1 to 24 hours in advance using our proposed ML based algorithm which, by stacking several fully independent neural networks together, builds one robust ensemble model. This STACKED model is a sequence-to-one model, which uses different external drivers depending on the forecasting timescales. The base neural network models include MLP, CNN and ELM. In section II, we present the data used and show how we select the most relevant features (external drivers used as input parameters) for the prediction task. Moreover, we briefly recall how MLP, CNN, ELM and the STACKED model work. In Section III we provide and discuss the experimental results. Finally, in Section IV we draw our conclusions and highlight possible future investigation.

2. Data and methods

2.1. Data preprocessing

The National Aeronautics and Space Administration (NASA) Archive of Space Geodesy[1] provided the Global Navigation Satellite Systems (GNSS) data used in this study. These are daily 30-second Receiver Independent Exchange (RINEX) compressed GNSS data from the mid-latitude station at Tsukuba (TSKB; 36.06^o N, 140.05^o E) in Japan for the period January 2006 to December 2018. TSKB mid-latitude station has been selected in this first attempt to build a novel ML model (STACKED) to avoid the complexity of the ionosphere at high and low latitude. From RINEX files, we extracted the vTEC firstly by using the calibration technique developed Ciraolo et al. (2007) and reported in Cesaroni et al. (2015); Cesaroni et al. (2021). This algorithm is able to estimate sTEC at each Ionospheric Pierce Point (IPP) by computing a constant bias for each arc of observation inlcuding the Differential Code Bias (DCB), the phase ambiguity and all the non-zero mean errors affecting the Geometry Free Linear Combination of the GNSS phase and code observables. The assumption for this estimation is that the ionosphere can be considered as a thin layer at IPP of 350 km so that the sTEC can be projected to the vTEC by applying a geometric mapping function. The vTEC spatial beavhiour is modeled by a polynomial function of the Modified Dip Latitude (MODIP) and Local Time that is also used to compute the vTEC over the TSKB GNSS station.

The vTEC dataset includes 2880 values per day and has missing values. These are fixed by using a forward linear interpolation within one hour gap. To fix a gap in a range from one to four hour, both forward and backward linear interpolation are used. For extended gap durations we utilise the median values computed on the basis of the preceding 27 days. Then, the vTEC dataset is downsampled by

[1] https://cddis.nasa.gov/archive/gnss/data/daily

averaging 10 points at a time to convert from a 30 seconds dataset to a 5 minutes dataset. The resulting 5 minutes downsampled vTEC is plotted in Fig. 1, splitted into three partitions for training (70%), validation (20%), and testing (10%).

The external drivers used in this study (Table 1) are downloaded from https://cdaweb.gsfc.nasa.gov for the period under consideration (January 2006 - December 2018). Some of the external drivers may have different timesteps. To obtain 5 minutes external drivers datasets, we fix their variable timestep by continual padding using the previous value until we reach the next one, e.g., as in the case of Kp (tri-hourly) and F10.7 (daily measure at noon). Finally, external drivers missing values are fixed using linear interpolation. Fig. 2 shows the percentage of missing values of external drivers and vTEC.

At the end of this preprocessing phase we obtain a consolidated vTEC and external drivers dataset (five minutes sampling) for the entire period (2006 to 2018). To understand which external drivers are suitable as input for our models in the training, validation and testing phases, we ranked them in order of importance with regard to their impact on vTEC forecasting. The algorithm is described in the following Section.

2.2. Feature ranking

We use a feature ranking algorithm known as *permutation of feature importance*, where a feature corresponds to an external driver. In this approach, to evaluate a feature's importance, we first train a model where external drivers are inputs and vTEC is the output: this allows ranking the external drivers on the basis of their effects on vTEC forecasting. Before using the permutation of features importance technique, we first checked the multicollinearity of the external drivers as shown in Fig. 3 using a Pearson's correlation coefficient (ρ) threshold equal to 0.5 to discard those parameters with $|\rho| \geqslant 0.5$. This allows reducing the number of external drivers used as input to the model. However, the multicollinearity based exclusion process shall guarantee that the three groups of parameters (Table 1) have at least one index in the remaining set of them. With this in mind, and by considering the results in Fig. 3, we excluded Kp, Dst, AL, AU, flow pressure and SSN before applying the permutation of feature importance technique (since after removing them the remaining indices are uncorrelated). Firstly, in this approach the dataset has been normalized using Z-scores:

$$z = (x - \mu)/\sigma, \tag{1}$$

where x is a dataset sample and μ and σ are, respectively, the mean and standard deviation of the dataset.

Since the external variables are characterized by different magnitude and variability, we do this to scale and center the dataset. The model is then evaluated. During the test, the baseline performance by the coefficient of determination (R^2) (Wright, 1921) is computed. A feature is shuffled, the performance is computed, and its performance is

Table 1
External drivers used for our study.

	Parameters
Auroral Electrojet	AU, AL, AE
Geomagnetic	Dst, SYM-H, Kp
Magnetic and Solar	F10.7, SSN, proton density, flow pressure, solar wind (Proton QI), the magnitude of the interplanetary magnetic field B_T (nT), vector components (Bx, By, Bz), flowspeed (V) and vector components (Vx, Vy , Vz) (km/s)

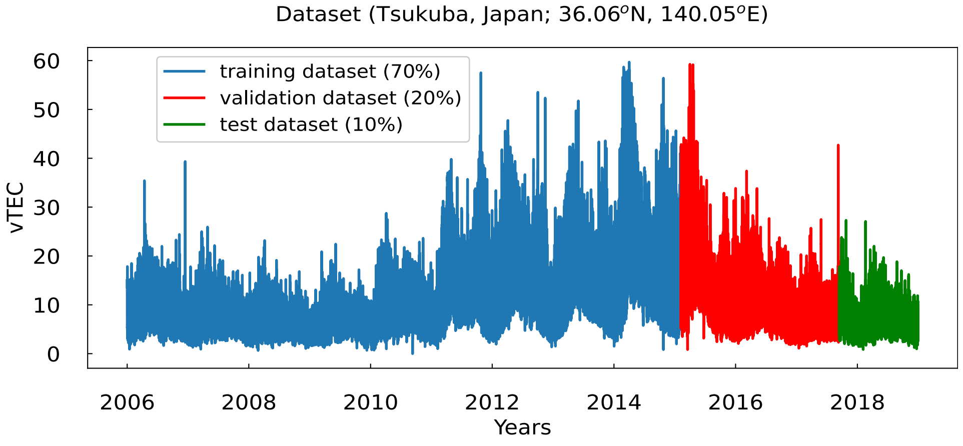


Fig. 1. The vTEC data for the Tsukuba station spanning from the year 2006 to 2018 splitted into training (blue), validation (red) and test (green) datasets. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

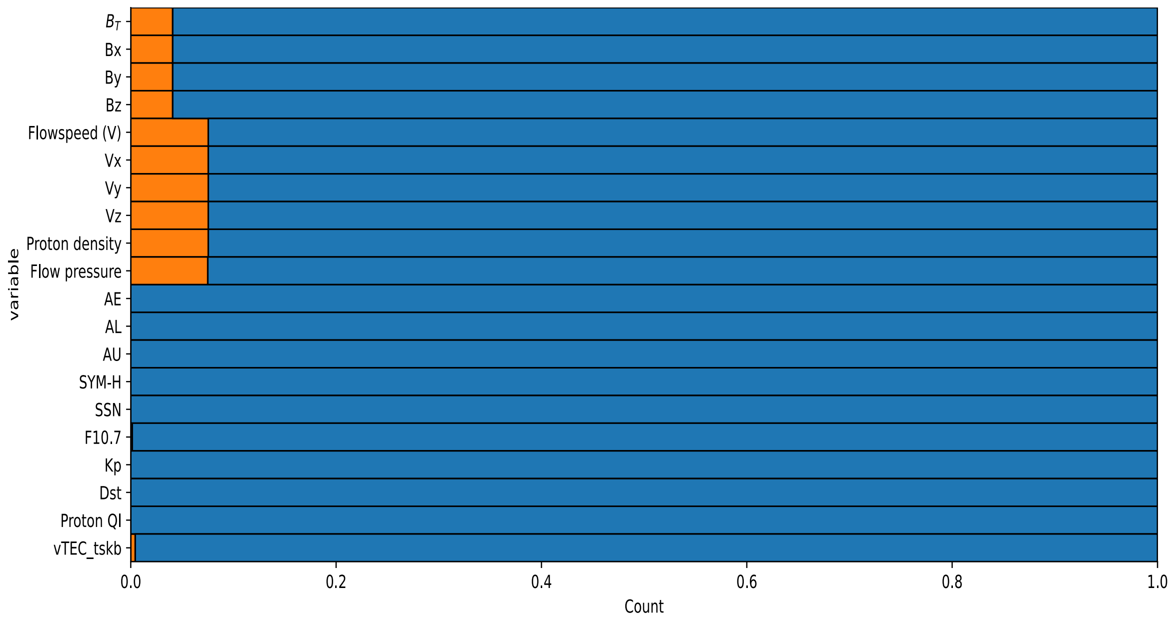


Fig. 2. vTEC and external drivers considered (see Table 1), including the fraction of missing values (in orange). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

compared with the baseline performance in each case to determine the significance of each feature. The significance of the characteristic is closely correlated to the magnitude of the change between the baseline performance and the performance obtained after each shuffling (Altmann et al., 2010). If a feature's performance differs noticeably from the baseline after the shuffle, it is of high relevance. If a feature's performance doesn't alter dramatically, it is of low relevance. Algorithm 1 (https://scikit-learn.org/stable/modules/permutation_importance.html.) provides the pseudo-code of the permutation of feature importance algorithm.

Algorithm 1. Permutation of feature importance

```
1: Inputs: fitted predictive model $m$, tabular dataset
   (training or validation) $D$
2: Compute the baseline performance score $s$ of the
   model $m$ on data $D$ (e.g., $R^2$ for a regressor)
3: for each feature $j$ (column of $D$): do
4:     for each repetition $k \in \{1, \dots, K\}$: do
5:         Randomly shuffle column $j$ of dataset $D$ to
           generate a corrupted version of the data named $\tilde{D_{k,j}}$
6:         Compute the score $s_{k,j}$ of the model $m$ on
           corrupted data $\tilde{D_{k,j}}$
7:     end for
8:     Compute importance $i_j = s - \frac{1}{K}\sum_{k=1}^{K} s_{k,j}$ for
       feature $f_j$
9: end for
```

Since the lag between the solar cycle, the geomagnetic forcing and the response of the ionosphere is not very well known, we performed the permutation of feature importance algorithm considering different forecasting timescales between 1 and 24 hours to take into account the possible different significance of the features for different forecasting horizons. To determine the importance of the external drivers, we used eleven different ML models including: Random Forest, Extreme Gradient Boosting, Gradient Boosting, linear regression, LASSO, Adaptive Boosting, Decision Tree, Extra Tree, KNN, Bagging (using SVR as estimator), Voting (using as estimators: Adaptive Boosting, Decision Tree, KNN, Random Forest, Extra Tree, Gradient Boosting, Extreme Gradient Boosting). Bagging (Breiman, 1996) and Voting (Dietterich, 2000) are used to combine the other ML models in order to get the best feature importance score. Owing to the fact that any two optimization algorithms (in this case, ML models based on solving an optimization problem) are equivalent when their performance is averaged across several different problems (in this case, various forecasting horizons) (Wolpert and Macready, 1997), we average the output of all the models after performing the feature ranking using all eleven models. Indeed, on the basis of the "no free lunch theorem" (Wolpert and Macready, 1997), averaging the features ranking score provided by different ML models used to perform the feature ranking itself, allows avoiding the bias possibly introduced by a specific algorithm. With this information, the average significance score over all fore-

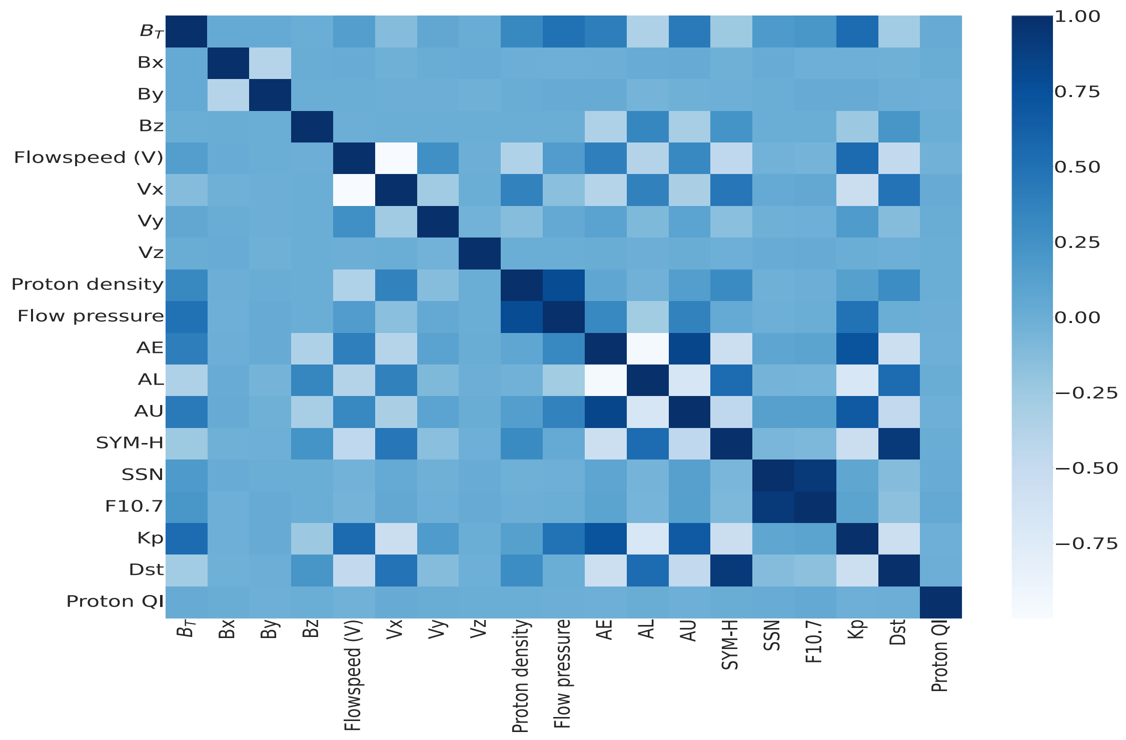


Fig. 3. Multicollinearity between external drivers computed using a threshold of 0.5. Accordingly, we removed Dst, SSN, flow pressure, AL, AU and Kp.

casting horizons is computed, and Fig. 4 depicts the behaviours of the external drivers with regard to various forecasting timescales. In Fig. 4, it is observed that for all the forecasting horizons, F10.7 (red dots) ranked first with the highest average score of importance. The magnitude B_T of the Interplanetary Magnetic Field (IMF) (blue dots) is the second important external driver for 1 hour and 24 hour forecasting horizons. For the interval ranging from 2 to 23 hours of forecasting horizons, the second best external driver is the Auroral Electrojet (AE) (green dots). F10.7, B_T and AE are briefly described below:

- F10.7: this index has the highest ranking across all predicted horizons. vTEC tends to exhibit a decrease during periods characterized by low solar activity, whereas it shows an increase during periods of high solar activity (Mukesh et al., 2020; Shenvi et al., 2023). Solar activity has a direct impact on the ionosphere, thereby influencing vTEC. The F10.7 solar flux is a frequently employed metric for quantifying solar activity. It has a strong correlation with sunspot counts and serves as an indicator for Ultraviolet (UV) and visible solar radiation levels. Consequently, the utilization of F10.7 is justified due to its established reliability as an indicator of solar activity magnitudes.
- B_T: this is shown to be the second most influential factor for forecasting horizons at 1 and 24 hours. B_T represents the overall intensity of the IMF. The metric encompasses the amalgamation of magnetic field intensity in the north-south, east-west, and inward-facing solar vs outward-facing solar orientations.
- AE: it exhibits a consistent ranking as the second most reliable indicator for predicting horizons ranging from 2 to 23 hours. The purpose of AE is to offer a comprehensive, numerical assessment of magnetic activity within the auroral zone, which is generated by intensified ionospheric currents occurring both below and within the auroral oval. Ideally, the concept refers to the comprehensive extent of variation at a given moment in time from the baseline values of the horizontal magnetic field (h) in the vicinity of the auroral oval.

2.3. STACKED model

To improve the performance of a ML predictive model, different approaches have been introduced considering an ensemble of other models as base learners. Bagging, one possible method, is effective in reducing the variance exhibited by the base models; it uses majority voting among the produced outputs (Bühlmann, 2012; Altman and Krzywinski, 2017; Maclin and Opitz, 1997; Theodoridis, 2015). Another approach, called boosting (Bühlmann, 2012; Schapire, 1999; Sabzevari et al., 2018; Theodoridis, 2015; Maclin and Opitz, 1997), is usually helpful in reducing the bias exhibited by the base models; it uses weighted voting among the produced outputs. Finally, a different ensemble approach, called stacking and based on a hierarchy of models, was introduced by Wolpert (1992) with the goal to reduce the bias of the base models. Usually, two layers of models are used. The lower layer in the hierarchy includes the base models, whilst the higher layer includes

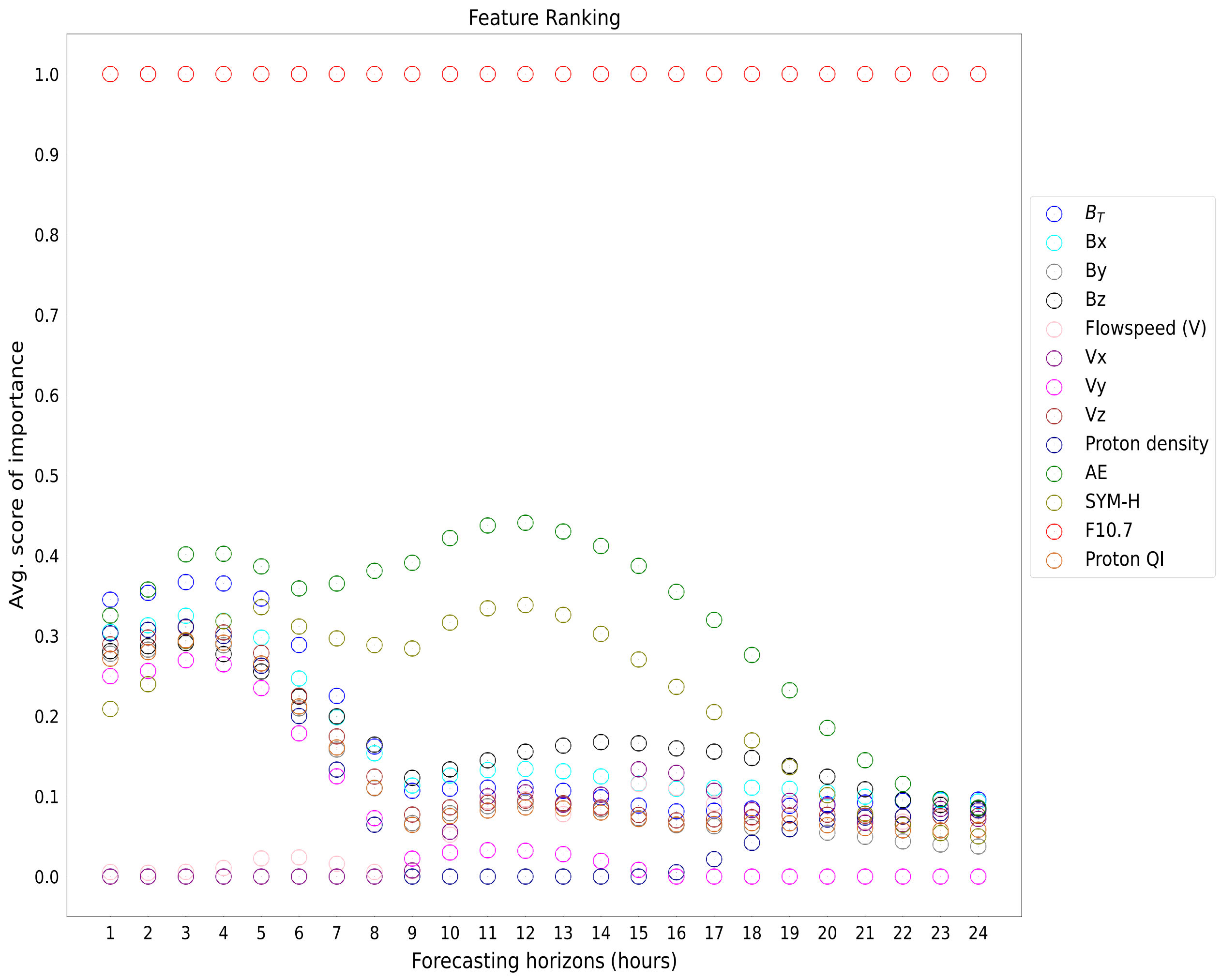


Fig. 4. Ranking of external drivers for vTEC forecasting for different forecasting horizons (1 to 24 hours in advance).

the combined model, which is trained on the outputs provided by the base models.

Our STACKED model uses ANN models: MLP, CNN and ELM as its base ANN models. The STACKED model is a sequence-to-one model exploiting the individual advantages of its base learners (MLP, CNN, ELM) (Taghizadeh-Mehrjardi et al., 2020; Pavlyshenko, 2018). A MLP is the most basic neural network, trained using the backpropagation algorithm during the learning phase. A CNN is also an ANN model trained using backpropagation. This network introduces a convolution step, a nonlinear step, and the pooling step allowing the extraction of the most important features and reducing the number of parameters via weight sharing (Theodoridis, 2015). ELM training, on the other hand, is not based on the backpropagation algorithm. As a consequence, the learning phase of this model is orders of magnitude faster than the corresponding ANN models trained using the backpropagation algorithm. ELM is known to provide good generalization performance (Huang et al., 2006; Huang et al., 2011; Huang, 2014; Huang, 2015).

Fig. 5 depicts a high level workflow that emphasizes the STACKED model training phase.

We begin with data preprocessing (vTEC and external drivers), and perform feature ranking to select the best two external drivers (from now on, we shall refer to them respectively as Ext 1 and Ext 2) for each forecasting horizon. We use a supervised learning approach and a sliding window algorithm (Xiong et al., 2021), which is applied to the time series (vTEC, Ext 1 and Ext 2) data from January 2006 to December 2018. This preprocessing step is required in order to prepare the input dataset (pairs $<x, y>$ in which x is one sample of the dataset and y the corresponding label) for the predictive models (i.e., the base learners: MLP, CNN, and ELM). The sliding window has a fixed width of 288 data points a day (5 minutes data) and the resulting outcome from the sliding window is partitioned into training, validation, and testing data sets which includes vTEC, Ext 1 and Ext 2. The training and validation data sets are used in the training phase by MLP, CNN, and ELM. The outputs of the base learners are the forecasted vTEC values for both the training and

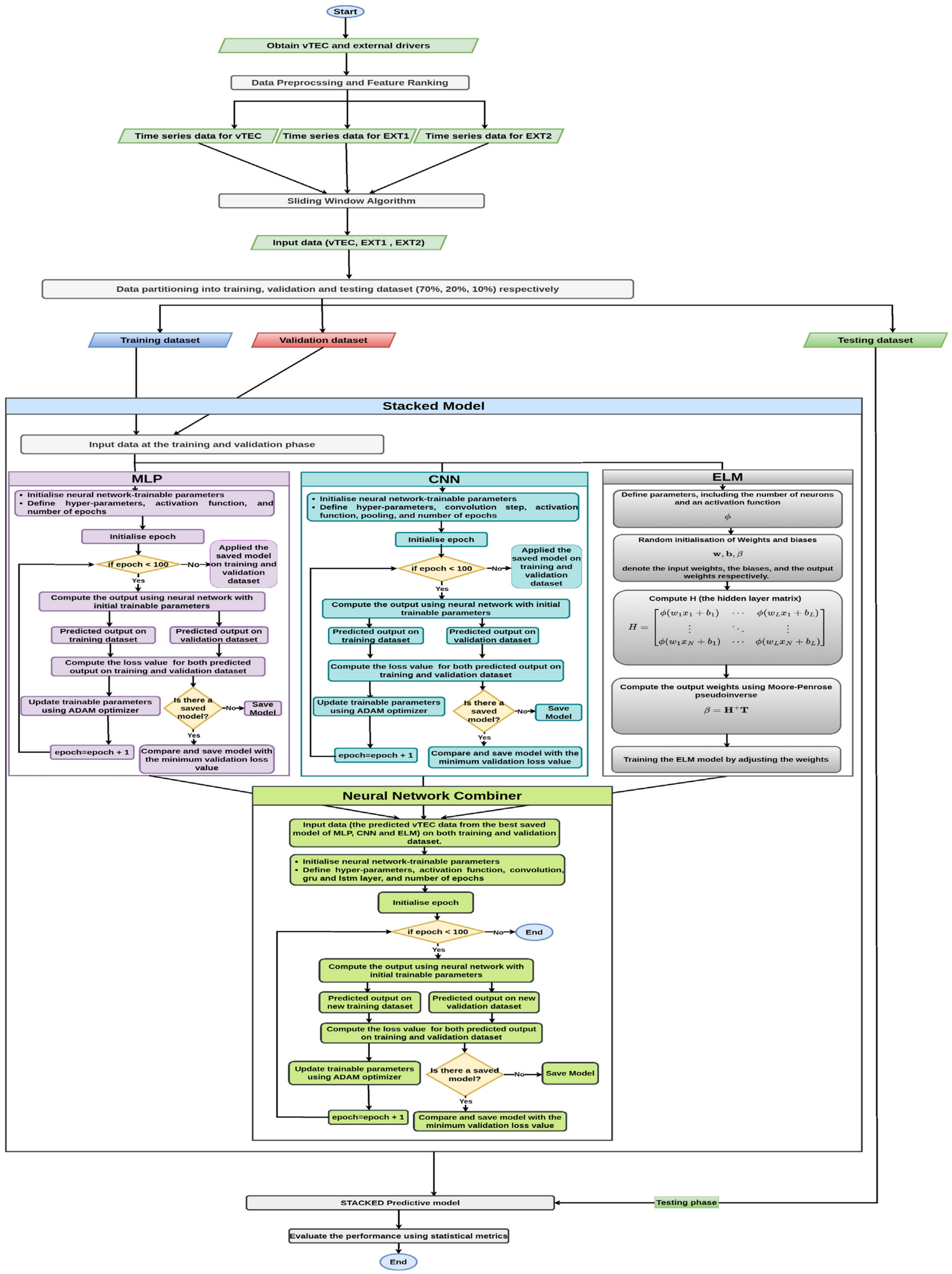

Start
Obtain vTEC and external drivers
Data Preprocssing and Feature Ranking
Time series data for vTEC
Time series data for EXT1
Time series data for EXT2
Sliding Window Algorithm
Input data (vTEC, EXT1 , EXT2)
Data partitioning into training, validation and testing dataset (70%, 20%, 10%) respectively
Training dataset
Validation dataset
Testing dataset
Stacked Model
Input data at the training and validation phase
MLP
• Initialise neural network-trainable parameters
• Define hyper-parameters, activation function, and number of epochs
Initialise epoch
if epoch < 100
No
Yes
Applied the saved model on training and validation dataset
Compute the output using neural network with initial trainable parameters
Predicted output on training dataset
Predicted output on validation dataset
Compute the loss value for both predicted output on training and validation dataset
Update trainable parameters using ADAM optimizer
Is there a saved model?
Save Model
epoch=epoch + 1
Compare and save model with the minimum validation loss value
CNN
• Initialise neural network-trainable parameters
• Define hyper-parameters, convolution step, activation function, pooling, and number of epochs
ELM
Define parameters, including the number of neurons and an activation function
ϕ
Random initialisation of Weights and biases
$\mathbf{w}, \mathbf{b}, \beta$
denote the input weights, the biases, and the output weights respectively.
Compute H (the hidden layer matrix)
$H = \begin{bmatrix} \phi(w_1x_1+b_1) & \cdots & \phi(w_Lx_1+b_L) \\ \vdots & \ddots & \vdots \\ \phi(w_1x_N+b_1) & \cdots & \phi(w_Lx_N+b_L) \end{bmatrix}$
Compute the output weights using Moore-Penrose pseudoinverse
$\beta = \mathbf{H}^{+}\mathbf{T}$
Training the ELM model by adjusting the weights
Neural Network Combiner
Input data (the predicted vTEC data from the best saved model of MLP, CNN and ELM) on both training and validation dataset.
• Initialise neural network-trainable parameters
• Define hyper-parameters, activation function, convolution, gru and lstm layer, and number of epochs
End
Predicted output on new training dataset
Predicted output on new validation dataset
STACKED Predictive model
Testing phase
Evaluate the performance using statistical metrics
End

Fig. 5. Workflow (related to the training phase) and architecture of our STACKED model, leveraging MLP, CNN and ELM base learners.

validation periods. These forecasted vTEC values become the training and validation datasets for the STACKED model. Similarly, by applying the base learners to the testing dataset (i.e., the remaining 10% of the dataset), we obtain as output the forecasted vTEC during the testing phase. This output is then used by the STACKED model in its testing phase. Finally the forecasted vTEC is provided by the STACKED model without the need of the external drivers (which, instead, are used by the base learners).

The architectures of the ANN models used as base and meta learners are detailed in Section 2.4.

2.4. Artificial neural networks

An ANN (McCulloch and Pitts, 1943) interconnects many artificial neuron units organized into layers. By changing the synaptic weights to minimize a predetermined cost function, learning is accomplished. Although training neural networks is computationally demanding, the backpropagation algorithm was a breakthrough because it made it possible to train neural networks on a collection of input-output training samples (LeCun et al., 1989; Rosenblatt, 1958; Theodoridis, 2015). From this perspective, an ANN can be thought as a nonlinear parametric model ($\hat{y} = f_\theta(x)$), where θ represents the weights/biases present in the network and x and $\hat{y}$ represent respectively the input and output of the neural network. Therefore the process of training ANN appears not to be different from any other paramtric prediction model. Three essential components are required: (i) a set of training samples, (ii) a loss function, $\mathscr{L}(y, \hat{y})$, and (iii) an iterative algorithm, such as gradient descent, to optimize the empirical loss,

$$J(\theta) = \sum_{n=1}^{N} \mathscr{L}(y_n, f_\theta(x_n)) \tag{2}$$

2.4.1. Multilayer perceptron

An MLP is a type of feedforward ANN, consisting of fully connected layers. Fig. 6 shows our MLP architecture. Training a MLP using the backpropagation algorithm requires repeatedly computing many diffeent gradients, which are involved in the solution of the optimization problem related to the MLP loss function. In turn, this may lead to vanishing and exploding gradient problems. In handling such situations, several factors comes into play: (i) the specific activation function used (ii) the loss function (iii) the initialization of trainable parameters. Since the backpropagation algorithm flows first forward and then backwards, the derivatives of the nonlinear functions used and the weights associated to the neuron units are multiplied together so that the number of relevant products grows quickly. Obviously, this problem is exacerbated in deep networks (i.e., networks with hundreds or thousands of layers). It is well known that using as activation function the Rectified Linear Unit (ReLU) not only ensures the required nonlinearity of the network but also helps reducing the training time (Krizhevsky et al., 2012). ReLU is defined as

$$ReLU(z) = \max\{0, z\} \tag{3}$$

ReLU does not suffer from saturation and its derivative is equal to one when the neuron operates in its active region ($z > 0$). Since the loss function of a MLP is tightly coupled to the type of output nonlinearity that is used, a “wrong” combination can severely affect the speed at which the network learns during training. Moreover, correctly initializing the trainable parameters is also crucial with regard to causing vanishing and/or exploding gradients. Therefore, Glorot and Bengio (2010) proposed an initialization method (Xavier or Glorot initialization) which helps avoiding this problem.

Mizutani and Dreyfus (2001) provides a complexity analysis of the supervised MLP-learning algorithm using backpropagation. The MLP architecture is grouped into Forward Pass (FP) and Backward Pass (BP). These are further subdivided into three processes each. FP processes include i) node net-input computation, ii) node activation (or output) evaluation and iii) error (or objective function) evaluation, whilts BP processes include i) node sensitivity (or delta) evaluation, ii) gradient computation and iii) parameter updating (using gradient). The complexity for the total operations per epoch is given in Table 2.

In Table 2 N is the total number of layers (including the first input layer), d denotes the number of training data, s represents a particular layer, P_s is the number of nodes in layer s excluding bias node, P_N is the number of output nodes in the terminal (output) layer N. T_s denotes the operations for evaluating $f^s(\cdot)$ where $f^s(\cdot)$ is the activation function in layer s and finally V_s represents the operations for evaluating $f^{s\prime}(\cdot)$ where $f^{s\prime}(\cdot)$ is the activation function derivative.

2.4.2. Convolutional neural network

CNN is another kind of feedforward ANN. Different models have been created to match the characteristics of the data for various applications (Theodoridis, 2015) but CNN is one of the major breakthrough appeared in the context of image analysis. Indeed, in the original model the neurons were connected to resemble the organization of the visual cortex of a cat. In our architecture we have three small CNNs whose outputs are fused and passed to a fully connected network. For each CNN, the first step is to apply a convolution operation to the feature vector, followed by a nonlinear activation function and by a pooling step in order to extract the most important features and reduce the dimensionality. Let $\mathbf{H} \in R^{m\times m}$ and $\mathbf{I} \in R^{l\times l}$, then the convolution operator is defined as

$$\mathscr{O}(i,j) = \sum_{t=1}^{m}\sum_{r=1}^{m} h(t,r) I(i+t-1, j+r-1) \tag{4}$$

where $m < l$. The CNN architecture we developed is shown in Fig. 7. In general, the combination of large sized inputs (with regard to the number of neuron units in the network’s input layer) and deep neural networks (with regard to the large number of hidden layers) give rise to an explosion in the number of parameters, which in turn seriously challenges the network’s generalization performance. This would require a huge amount of training data in order to

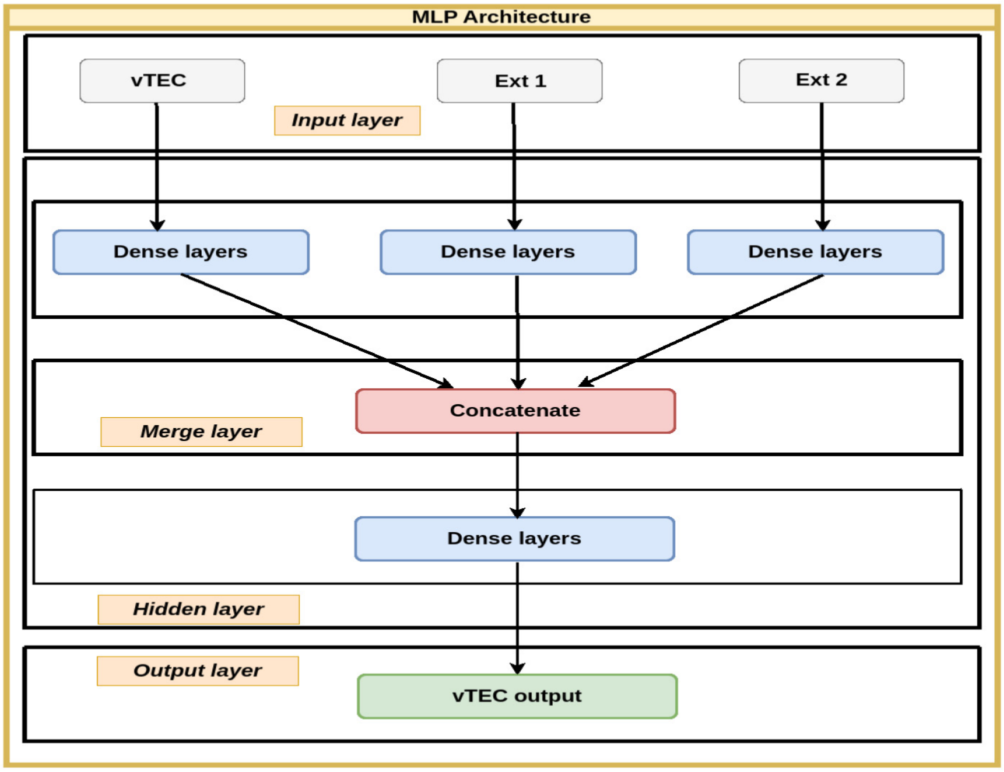


Fig. 6. Architecture of our MLP model. In this architecture we exploit multiple networks whose outputs are then fused together.

Table 2
Complexity analysis for MLP (for details see (Mizutani and Dreyfus, 2001)).

	Forward Pass (FP)
i) Node net-input computation	$2d\sum_{s=2}^{N}(P_{s-1}+1)P_s = 2dn$
ii) Node activation (or output) evaluation	$d\sum_{s=2}^{N}T_sP_s$
iii) Error (or objective function) evaluation	$dP_N + 2dP_N$
	Backward Pass (BP)
i) Node sensitivity (or delta) evaluation	$dP_N + d\sum_{s=1}^{N-1}P_{s+1}(V_{s+1}+1) + 2d\sum_{s=2}^{N-1}P_sP_{s+1}$
ii) Gradient computation	$d\sum_{s=1}^{N-1}(P_s+1)P_{s+1} = dn$ for incremental or $2dn$ for batch
iii) Parameter updating (using gradient)	$2d\sum_{s=1}^{N-1}(P_s+1)P_{s+1} = 2dn$ for incremental or $2n$ (independent of d) for batch

cope with the overfitting tendency of the network. But, in CNNs, the use of convolution, nonlinearity and pooling helps tackling the issue of parameter explosion via weight sharing; additionally, convolution, nonlinearity and pooling also allow the network extracting useful statistical information from the input data. The number of operations required to process a 1D-CNN layer is given as

$$n_f n_i n_k \cdot OutputSize \tag{5}$$

where n_f is the number of filters, n_i is the input vector size, n_k is the kernel size and the $OutputSize$ is defined as

$$OutputSize = \left\lceil \frac{n_s + 2padding - dilation(n_k - 1) - 1}{stride} + 1 \right\rceil \tag{6}$$

where n_s is the input sequence size (for details and other metrics of computational complexity for a CNN layer see Freire et al. (2024)).

2.4.3. Extreme learning machine

ELM (Huang et al., 2004; Huang et al., 2006; Huang et al., 2011; Cambria et al., 2013) is a feedforward ANN with one or more layers of hidden nodes, and it requires

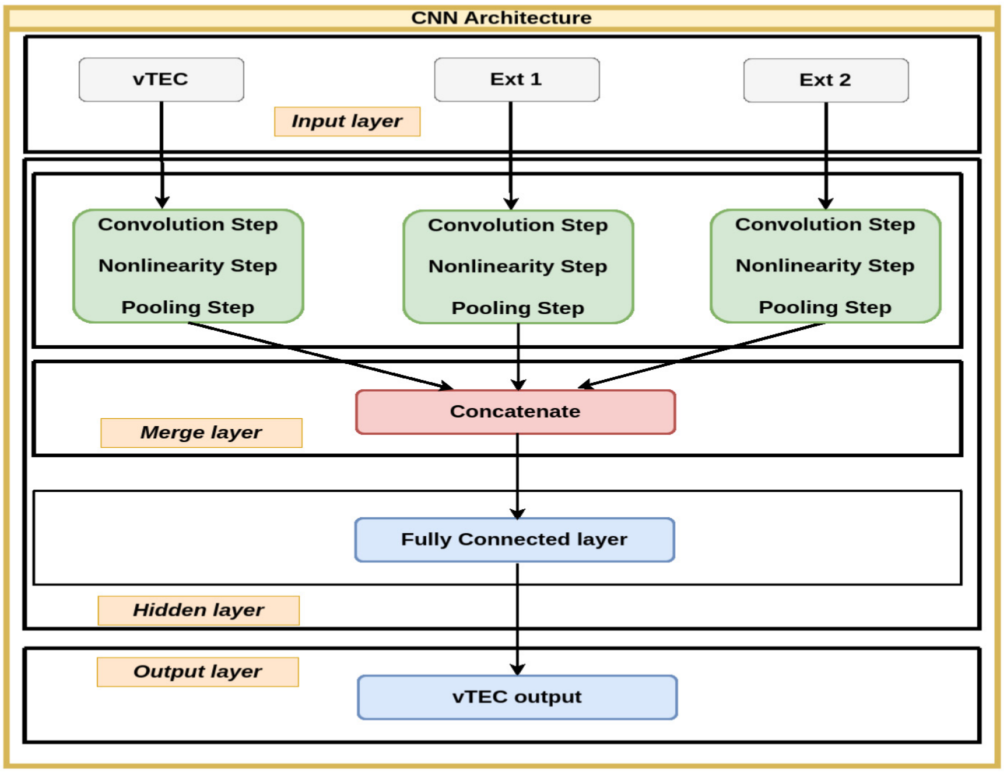


Fig. 7. Architecture of our CNN model. In this architecture we have three CNN whose outputs are flattened, fused and passed as an input to a fully connected neural network.

tuning for hidden node parameters as well as the weights that connect inputs to hidden nodes. The weights of the nodes of hidden layers may be randomly chosen and never updated, or they may be passed down unchanged from predecessors to successors neurons. The output weights of hidden nodes are typically learnt in a single step, which is equivalent to learning a linear model in most situations. While achieving greater generalization performance, the learning speed of the ELM can be thousands of times quicker than that of conventional feedforward network learning algorithms like the backpropagation algorithm. In Akusok et al. (2015), the authors presented a methodology and toolbox for highly scalable ELMs.

ELM can be formally described as follows. Given a collection of N unique training pairs $(\mathbf{x}_i, \mathbf{t}_i), i \in [1, N]$ with $\mathbf{x}_i \in R^d$ a dataset sample and $\mathbf{t}_i \in R^c$ its corresponding label. Then the output equation for a Single-Layer Feed-forward Network (SLFN) with L hidden neurons is as follows:

$$\sum_{j=1}^{L} \beta_j \phi\left(\mathbf{w}_j \mathbf{x}_i + b_j\right), i \in [1, N] \tag{7}$$

where ϕ represents the activation function and $\mathbf{w}_j, b_j$ and β_j are respectively the input weight, the bias and the output weight of the j-th hidden neuron. The relation between the inputs $\mathbf{x}_i$ of the network, the target outputs $\mathbf{t}_i$ and the estimated outputs $\mathbf{y}_i$ is given by

$$\mathbf{y}_i = \sum_{j=1}^{L} \beta_j \phi\left(\mathbf{w}_j \mathbf{x}_i + b_j\right) = \mathbf{t}_i + \epsilon_i, i \in [1, N] \tag{8}$$

where ϵ_i is a term related to the approximation error committed by the network. Representing an ELM network in matrix form, we have the following:

$$\mathbf{H} = \begin{bmatrix} \phi(\mathbf{w}_1 \mathbf{x}_1 + b_1) & \cdots & \phi(\mathbf{w}_L \mathbf{x}_1 + b_L) \\ \vdots & \ddots & \vdots \\ \phi(\mathbf{w}_1 \mathbf{x}_N + b_1) & \cdots & \phi(\mathbf{w}_L \mathbf{x}_N + b_L) \end{bmatrix} \tag{9}$$

$$\beta = \left(\beta_1^T, \cdots, \beta_L^T\right)^T, \mathbf{T} = \left(\mathbf{y}_1^T, \cdots, \mathbf{y}_N^T\right)^T \tag{10}$$

One ELM issue is that frequently this model is overdetermined, i.e., there are more training data samples N than hidden neurons L ($N > L$). The model is determined when the number of training data samples is equal to the number of hidden neurons ($N = L$), and it is under-determined when the hidden neurons are more than the training data samples ($N < L$)). To obtain a solution for an over-determined system, the Moore-Penrose generalized inverse matrix with regularization is used for numerical stability, and the solution is given by the minimum L_2 norm of the training error. The same approach can also be used in the determined and under-determined cases. The following equations define the Moore-Penrose generalized inverse of the ELM matrix $\mathbf{H}$.

$$\mathbf{H}\beta = \mathbf{T} \tag{11}$$

$$\beta = \mathbf{H}^{\dagger}\mathbf{T} \tag{12}$$

$$\mathbf{H}^{\dagger} = \left(\mathbf{H}^{T}\mathbf{H}\right)^{-1}\mathbf{H}^{T} \tag{13}$$

where $\mathbf{H}^{\dagger}$ represents the Moore-Penrose generalized inverse. However, $\left(\mathbf{H}^{T}\mathbf{H}\right)^{-1}$ shows numerical instability when $\mathbf{H}^{T}\mathbf{H}$ is close to singular. Hence, the Moore-Penrose generalized inverse with regularization is given as $\mathbf{H}^{\dagger} = \left(\mathbf{H}^{T}\mathbf{H} + \alpha\mathbf{I}\right)\mathbf{H}$, where $\alpha = 50\epsilon$ and ϵ is the machine precision for the used type of floating point numbers (Akusok et al., 2015; Rao et al., 1972). Our ELM architecture is shown in Fig. 8. The computational complexity for pseudo-inverse solution (i.e., Eqn (10)) of ELM is given as $\mathcal{O}\left(\widetilde{N}Lc\right)$ where $L \times c$ is the dimension of β and $\widetilde{N}$ is the dimension projected to the hidden layer (see reference (Akusok et al., 2015)).

2.4.4. Neural network combiner (meta learner)

The meta learner is the model which is situated at the second level in the hierarchy of the stacked model, above the base learners. At this level, an ANN is also used to combine the forecasted vTEC from the individual base learners. The architecture of our stacked meta-learner is shown in Fig. 9. The inputs are the vTEC forecasted values obtained by the MLP, CNN and ELM base learners. These outputs are combined together, obtaining the training and validation datasets required to train and validate the meta learner to forecast the vTEC values. The architecture is based on three neural network layers: CNN, Gated Recurrent Unit (GRU) and Long-Short Term Memory (LSTM) layers. The latter two networks are examples of Recurrent Neural Network (RNN). Weight sharing is applied to CNN, GRU and LSTM layers.

The LSTM unit was invented by Hochreiter and Schmidhuber (1997) due to the drawback of deep RNN i.e., the vanishing and exploding gradient problems which manifest themselves in the Backpropagation Through Time (BPTT) algorithm. An LSTM unit has the built-in ability to control the information into and out of the system's memory through nonlinear elements know as gates. A gate is equivalent to applying weight (ranging between 0 and 1) on the flow of information. Among the different gates, there are forget, input and output gates. A forget gate ($\mathbf{f}$) decides whether to retain the previous state or forget it; an input gate ($\mathbf{i}$), allows introducing new information and quantifying its importance; an output gate ($\mathbf{o}$), provides the current state on the basis of the previous hidden state ($\mathbf{h_{t-1}}$) and input vector ($\mathbf{x_t}$). In the LSTM unit, there are two types of memory-related variables that are propagated, namely the cell state vector $\mathbf{s}$ and the hidden variables vector $\mathbf{h}$. Fig. 10a shows the basic unit of LSTM and the corresponding updating equations are summarized below.

$$\mathbf{f_t} = \sigma\left(\mathbf{U^f x_t} + \mathbf{W^f h_{t-1}} + \mathbf{b^f}\right) \tag{14}$$

$$\mathbf{i_t} = \sigma\left(\mathbf{U^i x_t} + \mathbf{W^i h_{t-1}} + \mathbf{b^i}\right) \tag{15}$$

$$\bar{\mathbf{s}} = \tanh(\mathbf{U^s x_t} + \mathbf{W^s h_{t-1}} + \mathbf{b^s}) \tag{16}$$

$$\mathbf{o_t} = \sigma(\mathbf{U^o x_t} + \mathbf{W^o h_{t-1}} + \mathbf{b^o}) \tag{17}$$

$$\mathbf{s_t} = \mathbf{s_{t-1}} \circ \mathbf{f} + \mathbf{i} \circ \bar{\mathbf{s}} \tag{18}$$

$$\mathbf{h_t} = \mathbf{o} \circ \tanh(\mathbf{s_t}) \tag{19}$$

where $\circ$ denotes element-wise product between vectors or matrices. $\mathbf{f_t} \in R^h$ represents the forget gate, $\mathbf{U^k} \in R^{h\times d}$ is a matrix containing the weights of the inputs ($k = f, i, s, o$), $\mathbf{W^k} \in R^{h\times h}$ is a matrix containing the weights of the recurrent connections ($k = f, i, s, o$) and $\mathbf{b^k} \in R^h$ is the bias vector ($k = f, i, s, o$). $\mathbf{x_t} \in R^d$ is the input vector and $\mathbf{i_t} \in R^h$ is the input gate. σ represents the logistic sigmoid function, $\mathbf{o_t} \in R^h$ is the output gate, $\mathbf{s_t} \in R^h$ is the state vector at time $\mathbf{t}$ and $\mathbf{h_t} \in R^h$ is the hidden vector at time $\mathbf{t}$. The computational complexity of a LSTM layer is given as $T_s h(4d + 4h + 3)$, where T_s denotes the number of time steps in the layer that should be taken into account, as the process is repeated T_s times (see, Freire et al. (2024) for details and other computational complexity metrics).

GRU is a variant of LSTM proposed by Chung et al. (2014). This gated unit seeks to address the drawback of LSTM i.e., the training time. LSTM takes longer time in training due to its sophisticated internal unit as shown in Fig. 10a. Chung et al. (2014) simplified the internal unit of LSTM from three gates (forget, input and output gate) to two gates (reset and update gate) thereby improving the training time at the same time achieving better performance as well (e.g. see Iluore and Lu (2022)). The reset state ($\mathbf{r_t}$), helps determining to what extend the previous hidden state must be forgotten, whilst the update gate ($\mathbf{z_t}$) allows the model to precisely identify the amount of relevant historical information from prior timesteps that should be transmitted to future steps. The internal structure of a GRU is shown in Fig. 10b and the corresponding equations are summarized below.

$$\mathbf{z_t} = \sigma(\mathbf{U^z x_t} + \mathbf{W^z h_{t-1}} + \mathbf{b^z}) \tag{20}$$

$$\mathbf{r_t} = \sigma(\mathbf{U^r x_t} + \mathbf{W^r h_{t-1}} + \mathbf{b^r}) \tag{21}$$

$$\tilde{\mathbf{h}}_{\mathbf{t}} = \tanh\left(\mathbf{U^h x_t} + \mathbf{W^h}(\mathbf{r_t} \circ \mathbf{h_{t-1}}) + \mathbf{b^h}\right) \tag{22}$$

$$\mathbf{h_t} = (1 - \mathbf{z_t}) \circ \mathbf{h_{t-1}} + \mathbf{z_t} \circ \tilde{\mathbf{h}}_{\mathbf{t}} \tag{23}$$

where $\mathbf{z_t} \in R^h$ is the updated gate, $\mathbf{U^l} \in R^{h\times d}$ and $\mathbf{W^l} \in R^{h\times h}$ are the unknown parameter matrices to be learned ($l = z, r, h$). $\mathbf{r_t}$ denotes the reset gate and $\circ$ element-wise multiplication. $\mathbf{x_t} \in R^d$ is the input vector and $\mathbf{r_t} \in R^h, \sigma$

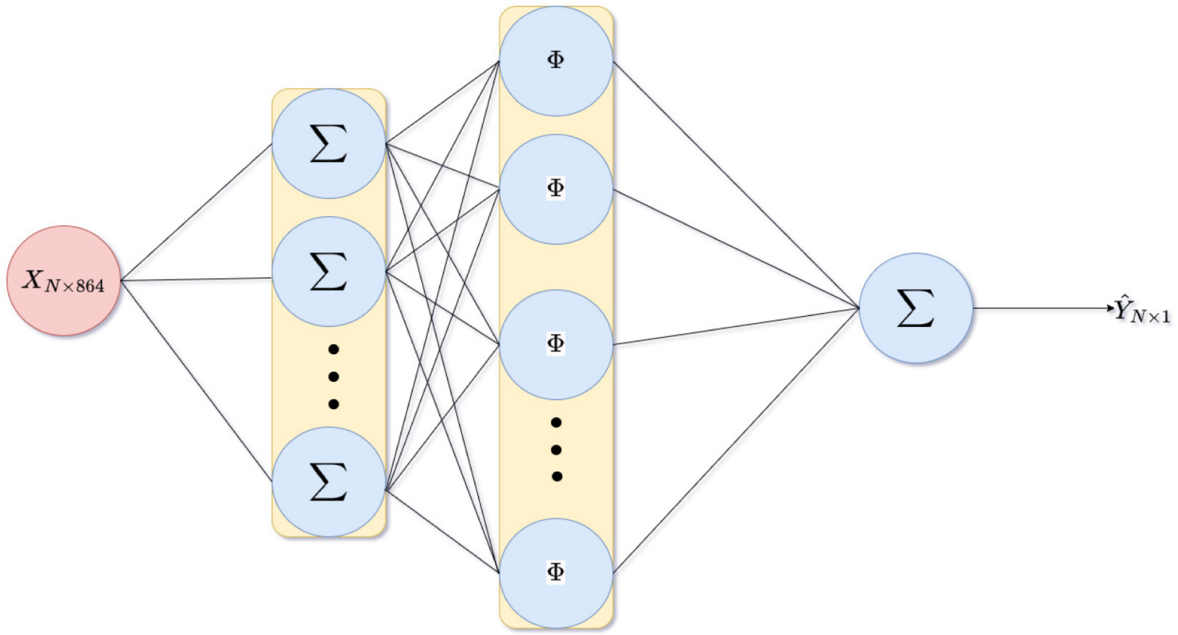


Fig. 8. Architecture of our ELM model. There are two hidden layers, $\sum$ represents the linear operator performed on the first hidden layer and Φ the use of the activation function (tanh) on the second hidden layer.

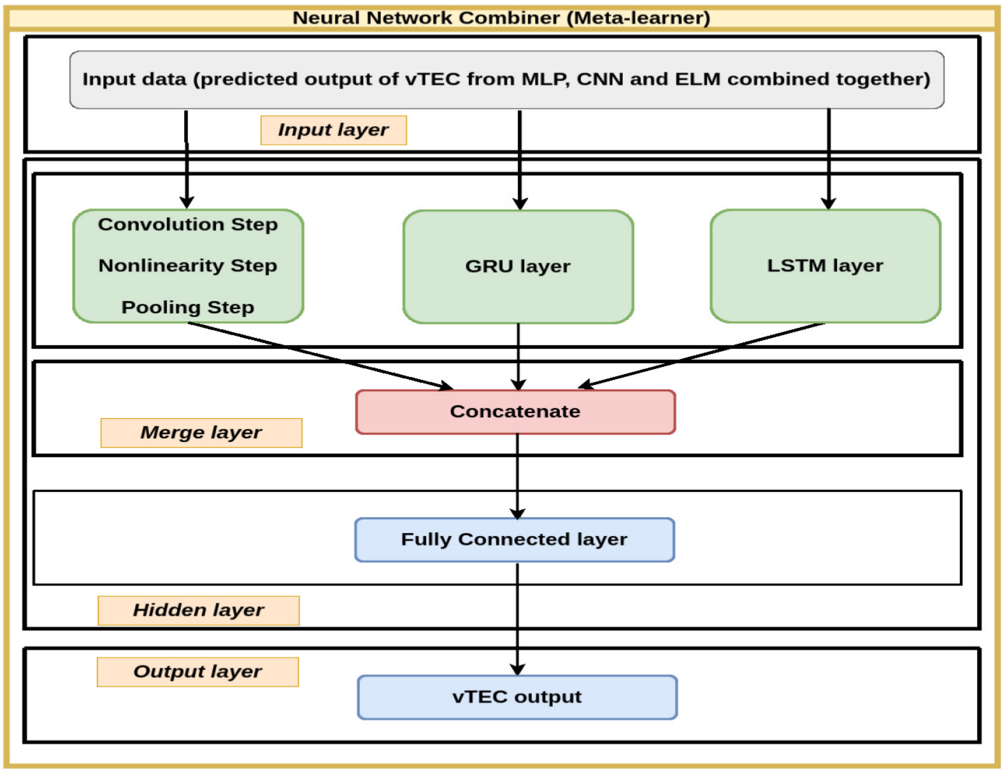
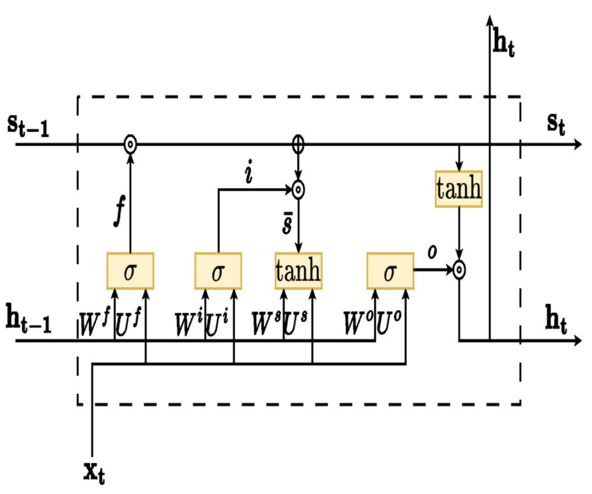
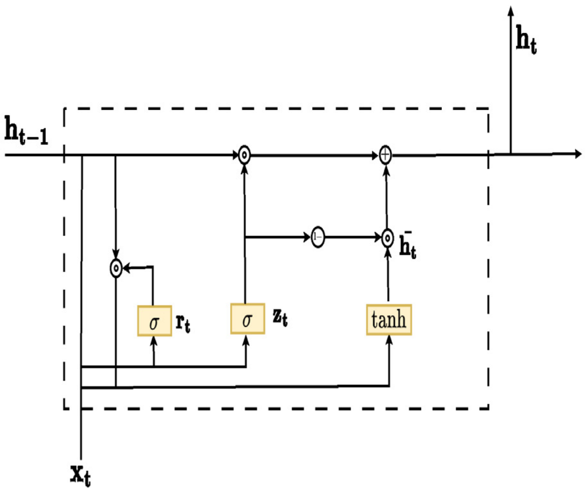


Fig. 9. Architecture of our meta-learner model. The CNN, GRU and LSTM outputs are fused and passed as an input to a fully connected neural network.



(a) Basic unit of LSTM.



(b) Basic Unit of GRU.

Fig. 10. RNN units.

represents the logistic sigmoid function, $\mathbf{h_t} \in R^h$ is the hidden vector at time $\mathbf{t}$ and $\tilde{\mathbf{h}}_t \in R^h$. The computational complexity of this LSTM layer is given as $T_s h(3d + 3h + 3)$, where T_s denotes the number of time steps in the layer that should be taken into account (since the process is repeated T_s times).

3. Results

In order to evaluate the forecasting performance of the four models (MLP, CNN, ELM and the STACKED model), the following metrics are used: R^2 (Nagelkerke et al., 1991; Renaud and Victoria-Feser, 2010), Mean Absolute Error (MAE) (Willmott and Matsuura, 2005; Chai and Draxler, 2014) and Root Mean Square Error (RMSE) (Willmott and Matsuura, 2005; Chai and Draxler, 2014). R^2, MAE and RMSE are defined in Eq. (24), (27) and (28) respectively:

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}}, \tag{24}$$

where SS_{res} and SS_{tot} are respectively the residual sum of squares and the total sum of squares. These are defined as

$$SS_{res} = \sum_i (y_i - \hat{y}_i)^2 \tag{25}$$

$$SS_{tot} = \sum_i (y_i - \bar{y})^2, \tag{26}$$

where y_i and $\hat{y}_i$ are the actual and predicted values for the i-th sample respectively, and $\bar{y}$ is denoted as the mean of y.

$$\mathrm{MAE} = \frac{1}{N}\sum_i^N |y_i - \hat{y}_i| \tag{27}$$

where y_i and $\hat{y}_i$ are the actual and predicted values for the i-th sample respectively, and N is the total number of samples.

$$RMSE = \sqrt{\frac{\sum_i^N (y_i - \hat{y}_i)^2}{N}} \tag{28}$$

where y_i and $\hat{y}_i$ are actual and predicted values for the i-th sample respectively, and N is the total number of samples.

R^2, MAE, and RMSE are evaluated for the four models by comparing the forecasted (from 1 to 24 hours in advance) and actual vTEC during the testing period (from 13 September 2017 to 31 December, 2018) at the TSKB (36.06^o N, 140.05^o E) GNSS receiver station. The minimum, maximum and mean values of the three statistical metrics as function of the four models are presented in Table 3. On average, the models' performance for vTEC forecasting is quite satisfactory, and the STACKED model shows a slight improvement with regard to MLP, CNN, ELM.

In Fig. 11, R^2, MAE, and RMSE are plotted for all the models under investigation as a function of the forecasting horizon. For all the models, R^2 decreases from 1 to 5 hour forecasting horizon, then remains constant through out the rest of the forecasting time. MAE and RMSE confirm this behaviour by increasing from 1 to 5 hour forecasting horizons, then remain constant for the rest of the forecasting periods. The STACKED model (black dots) provides the best performance at 1 hour forecasting horizon with $R^2 = 0.956$, MAE $= 0.1477$ TECu and RMSE $= 0.210$ TECu. The lowest R^2 value for the STACKED model is equal to 0.796, and it is related to the 18th hour forecasting horizon, with corresponding MAE $= 0.314$ TECu and RMSE $= 0.451$ TECu. The blue dot in Fig. 11 depicts

Table 3
Descriptive statistics of the evaluation metrics (R^2, MAE, RMSE).

Models	R^2			MAE (TECu)			RMSE (TECu)		
	min	max	mean	min	max	mean	min	max	mean
STACKED	0.796	0.956	0.816	0.147	0.314	0.296	0.210	0.451	0.426
CNN	0.790	0.951	0.812	0.154	0.320	0.299	0.222	0.458	0.431
MLP	0.786	0.949	0.808	0.158	0.322	0.304	0.227	0.462	0.436
ELM	0.781	0.953	0.803	0.155	0.331	0.312	0.217	0.468	0.441

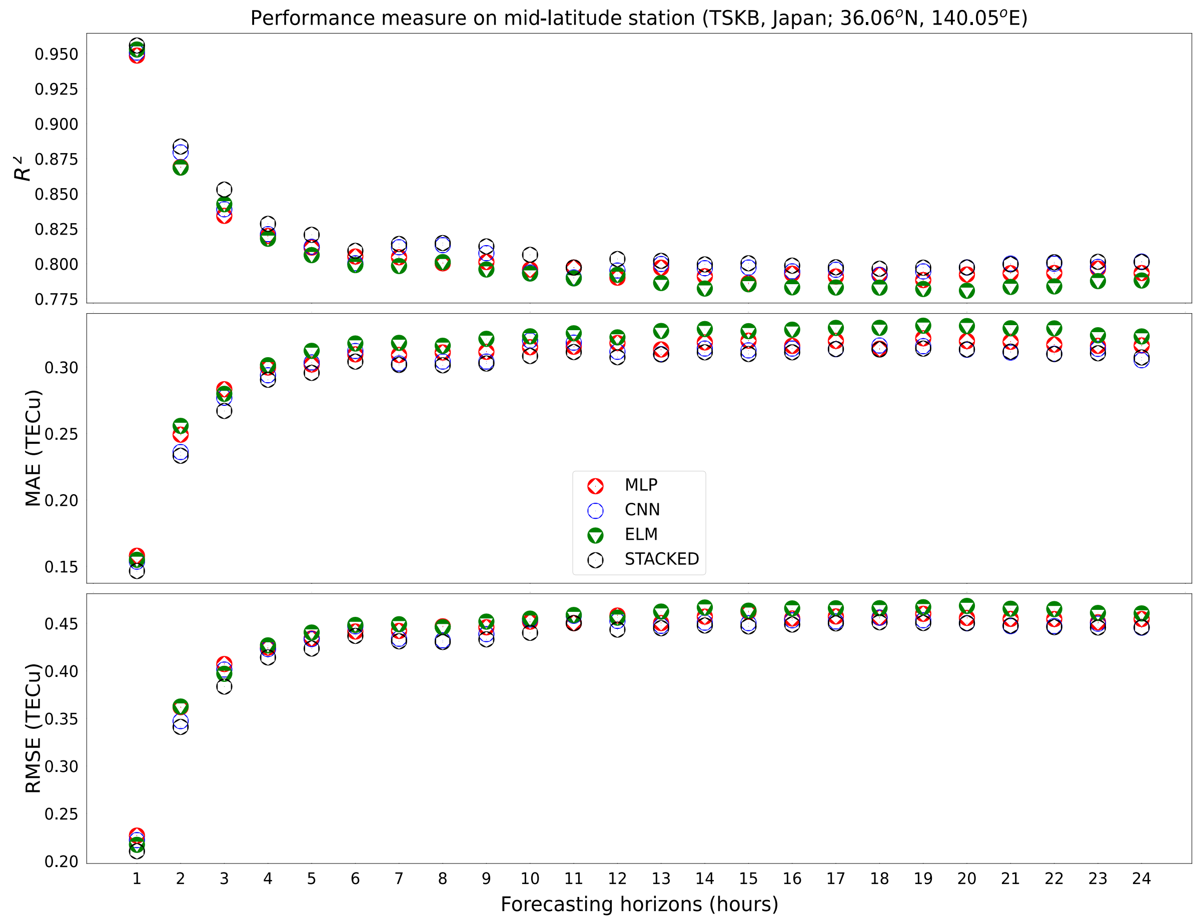


Fig. 11. R^2, MAE and RMSE of MLP, CNN, ELM and STACKED models from 1 to 24 hours of forecasting horizons.

the CNN model, which ranks second in the performance evaluation. At the 1 hour forecasting horizon, it achieves a maximum R^2 value equal to 0.951 and the minimum MAE (0.154 TECu) and RMSE (0.222 TECu). The CNN worst performance occurs at the 11th hour forecasting horizon ($R^2 = 0.790$, MAE $= 0.320$ TECu and RMSE $= 0.458$ TECu). MLP (red dot) and ELM (green dot) perform worse than the STACKED and CNN models. At the 1 hour forecasting horizon MLP and ELM obtain their maximum R^2 value (respectively equal to 0.949 and 0.953), with minimum MAE (respectively equal to 0.158 TECu and 0.155 TECu) and RMSE (respectively equal to 0.227 TECu and 0.217 TECu). MLP and ELM reach their worst forecasting performance respectively at the 15th and 20th hour forecasting horizons.

Fig. 12 depicts the box and whisker plot representing the statistical metrics of R^2, MAE, and RMSE. All models exhibit positive skewness for the R^2 metric and negative skewness for the MAE and RMSE metrics. Additionally, there are three outliers for each of these metrics. Comparing the STACKED model to the other ones under test, the metrics reveal for the STACKED model a significant skew in both the positive (with regard to R^2) and negative (with regard to MAE and RMSE) directions. Additionally, the dispersion of the models is roughly equivalent. The median value of the STACKED model provides a marginal improvement in comparison to the MLP, CNN, and ELM models.

To verify that the results highlighted by R^2, MAE and RMSE are not due to chance alone, we have performed the Wilcoxon Signed Rank Sum Test, which is a nonparametric statistical test (Wilcoxon, 1945). The aim is to compare paired groups, computing the differences between pairs and analyzing the resulting differences to infer if there is a statistically significant difference or not. This allows comparing different algorithms for the same task, and to understand if the results obtained - which show that one of the algorithms being tested provides better results than the others - are due to chance (in which case the results are misleading) or not (which means that the results are statistically significant).

To implement the Wilcoxon Signed Rank Sum Test, the null hypothesis is defined as follows: there is no statistically significant difference between the performance obtained by the STACKED model and the performance obtained by the base learners. The research hypothesis states that there is - indeed - a statistically significant difference. The level of statistical significance selected for the null hypothesis is $\alpha = 0.05$. The test statistics T is then computed as

$$T = \text{smallerof} \sum R_{+} \text{and} \sum R_{-} \tag{29}$$

where $\sum R_{+}$ is the sum of ranks with positive differences and $\sum R_{-}$ is the sum of ranks with negative differences. If the critical value (in our case, this value is 81; it is derived from the corresponding Wilcoxon signed rank test table) is greater than the test statistic T or the p-value falls in the rejection region (i.e. p-value $\leqslant \alpha$) we must reject the null hypothesis and conclude that there is a statistically significant improvement of the performance of the STACKED model with regard to the others. Table 4 reports the values obtained for all metrics, and allows rejecting the null hypothesis since the p-values at significance level $\alpha = 0.05$ are below 0.05 and the test statistics are less than the critical value (i.e., 81).

Finally, Fig. 13 shows an example of the forecasting capability of the STACKED model for vTEC over the test dataset of the TSKB station. The example pertains to the time frame from September 13 to October 17, 2017, and involves predicting horizons of 1 hour, 8 hours, 16 hours, and 24 hours. There were numerous solar flares in September and October 2017 that caused geomagnetic storms from G1 to G3 magnitudes (https://www.spaceweatherlive.com/en/archive/.html), for which the impacts are seen on the STACKED model performance in Fig. 13 on September 27-28, October 6-7 and October 11-15, 2017.

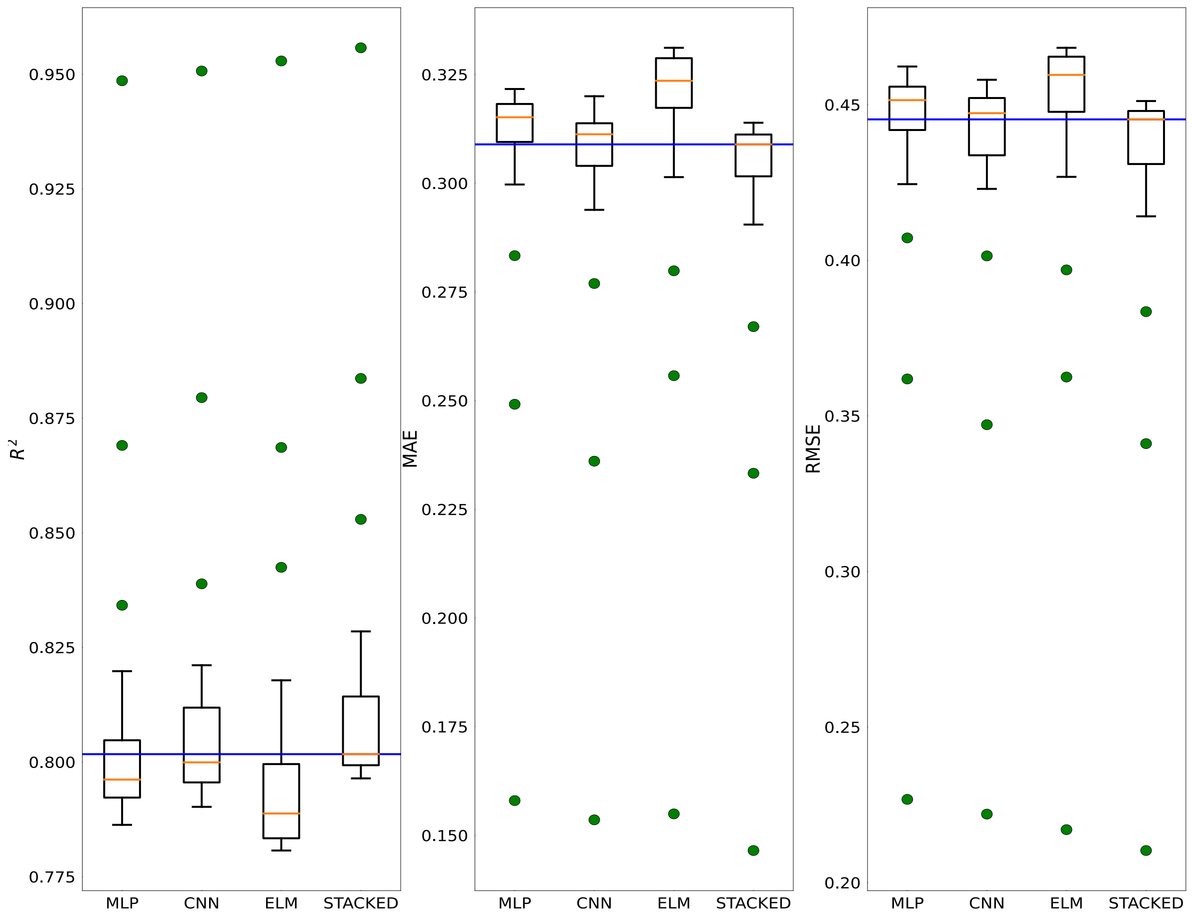


Fig. 12. Descriptive statistics for the performance of MLP, CNN, ELM and STACKED model on the full test dataset.

Table 4
Results of the Wilcoxon nonparametric statistical test. The *p*-values and the test statistic for each comparison are shown (STACKED vs MLP, STACKED vs CNN and STACKED vs ELM).

Using the statistical metric R^2			
$\alpha = 0.05$	STACKED vs MLP	STACKED vs CNN	STACKED vs ELM
p-value	2.384×10^{-7}	1.192×10^{-6}	1.192×10^{-7}
Test statistic	1.0	5.0	0.0
Using the statistical metric MAE			
p-value	1.192×10^{-7}	3.660×10^{-5}	1.192×10^{-7}
Test statistic	0.0	19.0	0.0
Using the statistical metric RMSE			
p-value	2.384×10^{-7}	1.192×10^{-6}	1.192×10^{-7}
Test statistic	1.0	5.0	0.0

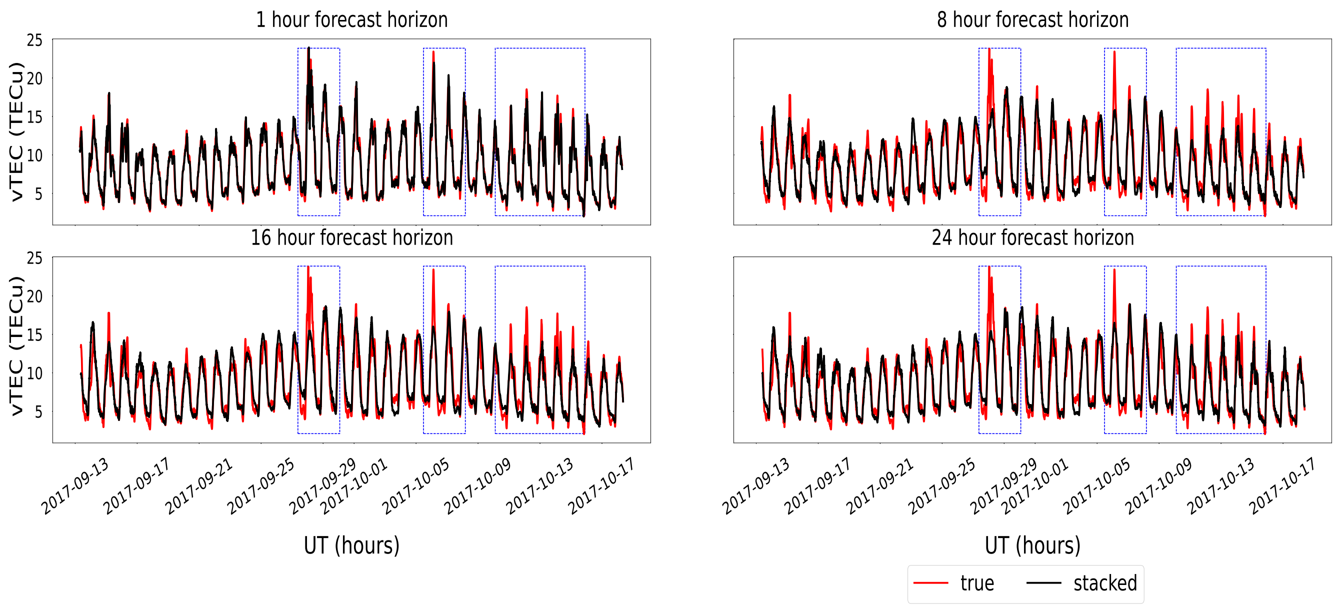


Fig. 13. Actual and predicted values using the STACKED model on the test dataset from September 13 to October 17, 2017 for forecasting at 1, 8, 16 and 24 hours in advance. The blue rectangular boxes are events on September 27-28, October 6-7 and 11-15, 2017.

4. Conclusions and future work

In this paper we have assessed the use of ANN models for vTEC forecasting, with regard to the MLP, CNN and ELM models. Moreover, we have designed a deep learning model based on the stacking technique. The primary advantage of the stacking ensemble technique lies in its capability of combining independent base learners thereby reducing their bias leading to a more accurate and robust predictive model. On the first level of the hierarchy there are MLP, CNN and ELM models as base learners, whilst at the second level of the hierarchy there is a meta-learner model. We have shown that our STACKED model provides better vTEC forecasting from 1 to 24 hours in advance. This performance's improvement is due to a reduction of the bias associated to the individual base learners, which is the effect of the stacking technique. The input to the models includes vTEC and appropriate external drivers selected among the helio-geophysical parameters available. These data came with missing values, a problem that has been fixed by using interpolation. The different timesteps of the external drivers were instead fixed by constant padding. The best pair of external drivers (depending on the forecasting timescale) have been determined by using the permutation of feature importance algorithm; this technique is based on several ML models, since there is not an optimal one for all the different forecasting horizons. The best external drivers were F10.7 (from 1 to 24 hour of forecasting horizons), B_T (at 1 and 24 hours of forecasting horizons) and AE (from 2 to 23 hours of forecasting horizons). The four models being assessed (MLP, CNN, ELM, STACKED) have been trained, validated and tested by using vTEC data from the TSKB receiver (36.06^o N, 140.05^o E) along 2006 to 2018.

The development of a single station model allows tuning the model parameters in order to improve the forecasting of the peculiar characteristics of the ionospheric variability over a specific location. Moreover, this simplifies the selection of the more impacting external drivers that are, in principle, different for different regions (e.g. latitudinal sectors) especially under disturbed helio-geophysical conditions. Several single station models, evaluated on properly distributed GNSS station over the globe, can be combined to obtain an accurate global forecasting including local ionospheric features.

The main findings are summarized as follows:

- On average, over the 24 hour forecasting horizons, R^2, MAE and RMSE obtained by the models comparing forecasted and actual vTEC range from 0.803 to 0.816 TECu for R^2; from 0.296 to 0.312 TECu for MAE and from 0.426 to 0.441 TECu for RMSE;
- Our STACKED model provides better performance than the others under test. This is confirmed by the R^2, MAE, and RMSE metrics;
- This improvement is statistically significant as shown by the Wilcoxon Signed-Rank Test. In this test, a significance level $\alpha = 0.05$ has been used and the corresponding p-values of the MLP, CNN and ELM models, compared to our STACKED model, clearly show rejection of the null hypothesis (it is worth recalling here that the null hypothesis has been defined as follows: there is no statistically significant difference between the performance obtained by the STACKED model and the performance obtained by the base learners); therefore, our STACKED model results are not due to chance, and are statistically significant.

The stacking ensemble approach seeks to reduce the bias of MLP, CNN, and ELM, hence improving the performance of the STACKED model. Future work will include a better selection and pre-processing of the external drivers that are characterized by different sampling (from minutes to 24 hours as in the case of F10.7) with regard to the vTEC data, as well as their derivative averaged or lagged values. Deep knowledge of the model performance will allow the use of receivers at different latitudes, that, all together, support investigating the model reliability as a function of the season, latitude, and space weather events that may occur.

Declaration of Competing Interest

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

References

Akusok, A., Björk, K.-M., Miche, Y., et al., 2015. High-performance extreme learning machines: a complete toolbox for big data applications. IEEE Access 3, 1011–1025. https://doi.org/10.1109/ACCESS.2015.2450498.

Altman, N., Krzywinski, M., 2017. Ensemble methods: bagging and random forests. Nat. Methods 14 (10), 933–935. https://doi.org/10.1038/nmeth.4438.

Altmann, A., Toloşi, L., Sander, O., et al., 2010. Permutation importance: a corrected feature importance measure. Bioinformatics 26 (10), 1340–1347. https://doi.org/10.1093/bioinformatics/btq134.

Bilitza, D., 2001. International reference ionosphere 2000. Radio Sci. 36 (2), 261–275. https://doi.org/10.1029/2000RS002432.

Bilitza, D., Altadill, D., Truhlik, V., et al., 2017. International reference ionosphere 2016: From ionospheric climate to real-time weather predictions. Space weather 15 (2), 418–429. https://doi.org/10.1002/2016SW001593.

Breiman, L., 1996. Bagging predictors. Mach. Learn., 24, 123–140. doi:10.1007/BF00058655.

Bühlmann, P., 2012. Bagging, boosting and ensemble methods. Handbook of computational statistics: Concepts and methods, pp. 985–1022. https://doi.org/10.1007/978-3-642-21551-3_33.

Cambria, E., Huang, G.-B., Kasun, L.L.C., et al., 2013. Extreme learning machines [trends & controversies]. IEEE Intell. Syst. 28 (6), 30–59. https://doi.org/10.1109/MIS.2013.140.

Cesaroni, C., Spogli, L., Alfonsi, L., et al., 2015. L-band scintillations and calibrated total electron content gradients over brazil during the last solar maximum. J. Space Weather Space Clim. 5, A36. https://doi.org/10.1051/swsc/2015038.

Cesaroni, C., Spogli, L., Aragon-Angel, A., et al., 2020. Neural network based model for global total electron content forecasting. J. Space Weather Space Clim. 10, 11. https://doi.org/10.1051/swsc/2020013.

Cesaroni, C., Spogli, L., De Franceschi, G., 2021. Ionoring: Real-time monitoring of the total electron content over italy. Remote Sensing, 13 (16). URL: https://www.mdpi.com/2072-4292/13/16/3290. doi:10.3390/rs13163290.

Chai, T., Draxler, R.R., 2014. Root mean square error (rmse) or mean absolute error (mae)?–arguments against avoiding rmse in the literature. Geoscient. Model Develop. 7 (3), 1247–1250. https://doi.org/10.5194/gmd-7-1247-2014.

Chapman, S., 1931a. The absorption and dissociative or ionizing effect of monochromatic radiation in an atmosphere on a rotating earth. Proceedings of the Physical Society (1926–1948), 43(1), 26. doi: 10.1088/0959-5309/43/1/305.

Chapman, S., 1931b. The absorption and dissociative or ionizing effect of monochromatic radiation in an atmosphere on a rotating earth part ii. grazing incidence. Proc. Phys. Soc. 43 (5), 483. https://doi.org/10.1088/0959-5309/43/5/302.

Chung, J., Gulcehre, C., Cho, K. et al. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555, doi:https://doi.org/10.48550/arXiv.1412.3555.

Ciraolo, L., Azpilicueta, F., Brunini, C., et al., 2007. Calibration errors on experimental slant total electron content (tec) determined with gps. J. Geodesy 81 (2), 111–120. https://doi.org/10.1007/s00190-006-0093-1.

Dietterich, T.G., 2000. Ensemble methods in machine learning. In International workshop on multiple classifier systems. Springer, pp. 1–15. https://doi.org/10.1007/3-540-45014-9_1.

Freire, P., Srivallapanondh, S., Spinnler, B., et al., 2024. Computational complexity optimization of neural network-based equalizers in digital signal processing: A comprehensive approach. J. Lightwave Technol., 1–25. https://doi.org/10.1109/JLT.2024.3386886.

Glorot, X., Bengio, Y., 2010. Understanding the difficulty of training deep feedforward neural networks. In Proceedings of the thirteenth international conference on artificial intelligence and statistics (pp. 249–256). JMLR Workshop and Conference Proceedings. URL: https://proceedings.mlr.press/v9/glorot10a.html.

Han, Y., Wang, L., Fu, W., et al., 2021. Machine learning-based short-term gps tec forecasting during high solar activity and magnetic storm periods. IEEE J. Select. Top. Appl. Earth Observ. Remote Sens. 15, 115–126. https://doi.org/10.1109/JSTARS.2021.3132049.

Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural Comput. 9 (8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735.

Huang, G.-B., 2014. An insight into extreme learning machines: random neurons, random features and kernels. Cognit. Comput. 6, 376–390. https://doi.org/10.1007/s12559-014-9255-2.

Huang, G.-B., 2015. What are extreme learning machines? filling the gap between frank rosenblatt's dream and john von neumann's puzzle. Cognit. Comput. 7, 263–278. https://doi.org/10.1007/s12559-015-9333-0.

Huang, G.-B., Zhou, H., Ding, X., et al., 2011. Extreme learning machine for regression and multiclass classification. IEEE Trans. Syst., Man, Cybernet., Part B (Cybernet.) 42 (2), 513–529. https://doi.org/10.1109/TSMCB.2011.2168604.

Huang, G.-B., Zhu, Q.-Y., & Siew, C.-K. (2004). Extreme learning machine: a new learning scheme of feedforward neural networks. In: 2004 IEEE international joint conference on neural networks (IEEE Cat. No. 04CH37541) (pp. 985–990). Ieee volume 2. doi: 10.1109/IJCNN.2004.1380068.

Huang, G.-B., Zhu, Q.-Y., Siew, C.-K., 2006. Extreme learning machine: theory and applications. Neurocomputing 70 (1–3), 489–501. https://doi.org/10.1016/j.neucom.2005.12.126.

Huang, Z., Yuan, H., 2014. Ionospheric single-station tec short-term forecast using rbf neural network. Radio Sci. 49 (4), 283–292. https://doi.org/10.1002/2013RS005247.

Iluore, K., Lu, J., 2022. Long short-term memory and gated recurrent neural networks to predict the ionospheric vertical total electron content. Adv. Space Res.. https://doi.org/10.1016/j.asr.2022.04.066.

Krizhevsky, A., Sutskever, I., Hinton, G.E., 2012. Imagenet classification with deep convolutional neural networks. Adv. Neural Inform. Process. Syst. 25. https://doi.org/10.1145/3065386.

LeCun, Y., Boser, B., Denker, J.S., et al., 1989. Backpropagation applied to handwritten zip code recognition. Neural Comput. 1 (4), 541–551. https://doi.org/10.1162/neco.1989.1.4.541.

Liu, L., Zou, S., Yao, Y. et al. (2020). Forecasting global ionospheric total electron content (tec) using deep learning. In AGU Fall Meeting Abstracts (pp. NG004–0017). volume 2020. doi: 10.1029/2020SW002501.

Maclin, R., Opitz, D., 1997. An empirical evaluation of bagging and boosting. AAAI/IAAI 1997, 546–551. https://doi.org/10.1109/ICNN.1997.613999.

Mallika, I.L., Ratnam, D.V., Ostuka, Y., et al., 2018. Implementation of hybrid ionospheric tec forecasting algorithm using pca-nn method. IEEE J. Select. Top. Appl. Earth Observ. Remote Sens. 12 (1), 371–381. https://doi.org/10.1109/JSTARS.2018.2877445.

Mannucci, A., Wilson, B., Edwards, C., 1993. A new method for monitoring the earth ionosphere total electron content using the gps global network paper presented at ion gps 93, inst. Of Navig., Salt Lake City, Utah, URL: https://hdl.handle.net/2014/36277.

McCulloch, W.S., Pitts, W., 1943. A logical calculus of the ideas immanent in nervous activity. Bull. Math. Biophys. 5 (4), 115–133. https://doi.org/10.1007/BF02478259.

Mendillo, M., 2006. Storms in the ionosphere: Patterns and processes for total electron content. Rev. Geophys. 44 (4). https://doi.org/10.1029/2005RG000193.

Mizutani, E., Dreyfus, S., 2001. On complexity analysis of supervised mlp-learning for algorithmic comparisons. In IJCNN'01. International Joint Conference on Neural Networks. Proceedings (Cat. No.01CH37222) (pp. 347–352 vol 1). volume 1. doi:10.1109/IJCNN.2001.939044.

Mukesh, R., Karthikeyan, V., Soma, P., et al., 2020. Forecasting of ionospheric tec for different latitudes, seasons and solar activity conditions based on oksm. Astrophys. Space Sci. 365 (1), 13. https://doi.org/10.1007/s10509-020-3730-x.

Nagelkerke, N.J. et al., 1991. A note on a general definition of the coefficient of determination. biometrika 78 (3), 691–692. https://doi.org/10.1093/biomet/78.3.691.

Nava, B., Radicella, S., Leitinger, R., et al., 2006. A near-real-time model-assisted ionosphere electron density retrieval method. Radio Sci. 41 (06), 1–8. https://doi.org/10.1029/2005RS003386.

Ngwira, C.M., Habarulema, J.-B., Astafyeva, E., et al., 2019. Dynamic response of ionospheric plasma density to the geomagnetic storm of 22–23 June 2015. J. Geophys. Res.: Space Phys. 124 (8), 7123–7139. https://doi.org/10.1029/2018JA026172.

Nijimbere, V., 2020. A mathematical model for the numerical simulations of traveling ionospheric disturbances/atmospheric gravity waves generated by the joule heating. Phys. Fluids 32 (7). https://doi.org/10.1063/5.0008649.

Pavlyshenko, B., 2018. Using stacking approaches for machine learning models. In: In 2018 IEEE second international conference on data stream mining & processing (DSMP). IEEE, pp. 255–258. https://doi.org/10.1109/DSMP.2018.8478522.

Rao, C.R., Mitra, S.K., et al., 1972. Generalized inverse of a matrix and its applications. In Proceedings of the sixth Berkeley symposium on mathematical statistics and probability, volume 1. University of California Press, Oakland, CA, USA, pp. 601–620. https://doi.org/10.2307/2344631.

Renaud, O., Victoria-Feser, M.-P., 2010. A robust coefficient of determination for regression. J. Stat. Plan. Inference 140 (7), 1852–1862. https://doi.org/10.1016/j.jspi.2010.01.008.

Rosenblatt, F., 1958. The perceptron: a probabilistic model for information storage and organization in the brain. Psychol. Rev. 65 (6), 386. https://doi.org/10.1037/h0042519.

Sabzevari, M., Martínez-Muñoz, G., Suárez, A., 2018. Vote-boosting ensembles. Pattern Recogn. 83, 119–133. https://doi.org/10.1016/j.patcog.2018.05.022.

Schapire, R., 1999. A brief introduction to boosting. IJCAI International Joint Conference on Artificial Intelligence, 2, 1401–1406. URL: http://rob.schapire.net/papers/Schapire99c.pdf. 16th International Joint Conference on Artificial Intelligence, IJCAI 1999; Conference date: 31-07-1999 Through 06-08-1999.

Schunk, R.W., Sojka, J.J., Bowline, M., 1986. Theoretical study of the electron temperature in the high-latitude ionosphere for solar maximum and winter conditions. J. Geophys. Res.: Space Phys. 91 (A11), 12041–12054. https://doi.org/10.1029/JA091iA11p12041.

Shenvi, N., Virani, H., et al., 2023. Forecasting of ionospheric total electron content data using multivariate deep lstm model for different latitudes and solar activity. J. Electr. Comput. Eng. 2023. https://doi.org/10.1155/2023/2855762.

Taghizadeh-Mehrjardi, R., Schmidt, K., Amirian-Chakan, A. et al. (2020). Improving the spatial prediction of soil organic carbon content in two contrasting climatic regions by stacking machine learning models and rescanning covariate space. Remote Sensing, 12 (7). URL: https://www.mdpi.com/2072-4292/12/7/1095. doi:10.3390/rs12071095.

Theodoridis, S., 2015. Machine learning: a Bayesian and optimization perspective. Academic press. https://doi.org/10.1016/C2013-0-19102-7.

Wilcoxon, F., 1945. Individual comparisons by ranking methods. Biomet. Bull. 1 (6), 80–83, URL: http://www.jstor.org/stable/3001968.

Willmott, C.J., Matsuura, K., 2005. Advantages of the mean absolute error (mae) over the root mean square error (rmse) in assessing average model performance. Clim. Res. 30 (1), 79–82. https://doi.org/10.3354/cr030079.

Wolpert, D.H., 1992. Stacked generalization. Neural networks 5 (2), 241–259. https://doi.org/10.1016/S0893-6080(05)80023-1.

Wolpert, D.H., Macready, W.G., 1997. No free lunch theorems for optimization. IEEE Trans. Evol. Comput. 1 (1), 67–82. https://doi.org/10.1109/4235.585893.

Xiong, P., Zhai, D., Long, C., et al., 2021. Long short-term memory neural network for ionospheric total electron content forecasting over china. Space Weather 19 (4). https://doi.org/10.1029/2020SW002706, e2020SW002706.

Yasyukevich, Y., Astafyeva, E., Padokhin, A., et al., 2018. The 6 september 2017 x-class solar flares and their impacts on the ionosphere, gnss, and hf radio wave propagation. Space Weather 16 (8), 1013–1027. https://doi.org/10.1029/2018SW001932.

Zewdie, G.K., Valladares, C., Cohen, M.B., et al., 2021. Data-driven forecasting of low-latitude ionospheric total electron content using the random forest and lstm machine learning methods. Space Weather 19 (6). https://doi.org/10.1029/2020SW002639, e2020SW002639.

Wright, S., 1921. Correlation and Causation. Journal of agricultural research 20, 557–585.