A Short-Term Forecast Model of foF2 Based on Elman Neural Network

Jieqing Fan 1, Chao Liu 1, Yajing Lv 1, Jing Han 2 and Jian Wang 2,*

1 School of Electrical and Electronic Engineering, North China Electric Power University, Beijing 102206, China
2 China Research Institute of Radiowave Propagation, Qingdao 266107, China
* Correspondence: wangjian@criep.com

Received: 8 June 2019; Accepted: 9 July 2019; Published: 10 July 2019

Abstract: The critical frequency foF2 of the ionosphere F2 layer is one of the most important parameters of the ionosphere. Based on the Elman neural network (ENN), this paper constructs a single station forecasting model to predict foF2 one hour ahead. In order to avoid the network falling into local minimum, the model is optimized by the improved particle swarm optimization (IPSO). The input parameters used in the model include local time, seasonal information, solar cycle information and magnetic activity information. Data of the Wuhan Station from 2008 to 2016 were used to train and test the model. The prediction results of foF2 show that the root mean square error (RMSE) of the Elman neural network model is 4.30% lower than that of the back-propagation neural network (BPNN) model. The RMSE is further reduced by 8.92% after using the IPSO to optimize the model. This indicates that the Elman neural network model optimized by the improved particle swarm optimization is superior to the BP neural network and Elman neural network in the forecast of foF2 one hour ahead at Wuhan station.

Keywords: foF2; Elman neural network; improved particle swarm optimization; forecast

1. Introduction

The critical frequency foF2 of the ionosphere F2 layer is of great significance for the prediction of high frequency (HF) communication frequency and has been a hot spot in ionosphere research. Influenced by solar, geomagnetic and other factors, the change of foF2 is a time sequence nonlinear process. The back-propagation neural network has been widely used to predict foF2 over the past few decades because it can fit nonlinear processes well. For instances, predicting the noon value of foF2 at Grahamstown [1]; predicting the trend of foF2 at 19 Ionosphere stations in the Asia/Pacific sector [2]; comparing neural network-based predictions of foF2 with the IRI-2012 model in Southeast Asia [3]; using the data of 13 stations to establish a global foF2 prediction model [4].

The performance of the back-propagation neural network (BPNN) model is better than the traditional empirical model, such as the IRI model [5]. However, it also has the problems of slow convergence and falls easily into the local minimum. Thus, other algorithms are used to train neural network models to overcome these shortcomings, such as the genetic algorithm [6] and AdaBoost-BP algorithm [7], and have achieved good results.

However, none of the above models can accurately reflect the temporality of foF2, because they are all feed-forward neural networks. Meanwhile, the recursive neural network (RNN) with memory function has been widely used in the prediction of time sequence and achieved good results. For examples, visual tracking [8], load dispatch of microgrid [9] and total electron content [10]. Therefore, in this paper, we developed a model based on the Elman neural network, which is a kind of RNN, to predict foF2 one hour ahead. The improved particle swarm optimization was used to train the model.
2. Model Principle

2.1. Elman Neural Network (ENN)

The Elman neural network was first proposed by J. L. Elman in 1990 for speech processing. Based on the neural network, the ENN adds the context layer to connect the input and output of the hidden layer, which enable the network to store previous state and have dynamic properties. Compared with BP neural networks, ENN is better able to capture information about sequence data. The change of foF2 is stable in a short time, and the correlation of foF2 in adjacent time is great. Thus, we chose the Elman neural network to build the model. The Elman neural network structure is shown in Figure 1:

![Multi-hidden layer Elman neural network](image)

*Figure 1. Multi-hidden layer Elman neural network.*

The Elman neural network consists of four layers, which are input layer, hidden layer, context layer and output layer [11]. The input layer transmits external signals to the hidden layer. The context layer stores the output signal of the hidden layer and serves as the input for the next step of the hidden layer. Then, the hidden layer activates the signal it receives and passes it to the next layer. Finally, the signal is output by the output layer.

Suppose $m$ is the length of input, the input set is:

$$X = [x_1, x_2, \cdots x_m]$$

(1)

$n$ is the total number of hidden layers and $t$ is the input set number. The output of the hidden layer at $t$ step is:

$$H_i(t) = \begin{cases} 
  f(W_iX(t) + W_iZ_i(t)) & i = 1 \\
  f(W_iH_{i-1}(t) + W_iZ_i(t)) & 1 < i \leq n 
\end{cases}$$

(2)

where $H_i(t)$ represents the output of the hidden layer $i$, $W_i$ represents the weight matrix from the hidden layer $(i-1)$ to the hidden layer $i$, $Z_i(t)$ represents the output of the context layer $i$, which is given by:

$$Z_i(t) = \begin{cases} 
  H_1(t-1) & t > 1 \\
  0 & t = 1 
\end{cases}$$

(3)

$f$ represents the activation function of the hidden layer, which can be calculated from the following:

$$f(x) = \frac{1}{1 + e^{-x}}$$

(4)
and \( y(t) \), which represents the output of the network, is given by:
\[
y(t) = W_{n+1}H_n(t)
\]
where \( W_{n+1} \) represents the weight matrix from the hidden layer \( n \) to the output layer.

2.2. Improved Particle Optimization Algorithm

The most classic training algorithm for neural networks is the back-propagation algorithm. The algorithm iteratively updates the model parameters by using the derivative of the error function to the model parameters. However, when the model function is a non-convex function, the algorithm tends to converge to a local minimum. Therefore, this paper chooses the improved particle swarm optimization algorithm as the optimization algorithm. The improved particle swarm optimization (IPSO) algorithm is a group stochastic optimization technique inspired by the foraging behavior of birds. By pursuing the optimal particle instead of the derivative, the movement of the whole group evolves from disorder to order in the solution space, and ultimately the optimal solution is obtained.

2.2.1. Particle Swarm Optimization for ENN

The particle swarm optimization algorithm randomly initializes a group of particles, each of which has three characteristics of position, velocity and fitness function value, representing a potential solution to the problem. The position of the particle with the optimal function fitness is marked as the optimal particle position. The position of each particle is updated according to its historical optimal positions and the global optimal position [12]. Finally, particles converge to the global optimal solution. The process of particle swarm optimization for the ENN is as follows:

1. In order to calculate the value of the weight matrix in the ENN, the element of the weight matrix is transformed into a vector, which is the position of the particle. Suppose \( k \) is the length of the particle, then \( m \) represents the number of particle iterations. The position of the particle can be expressed as \( P(m) = (p_1(m), p_2(m) \cdots p_k(m)) \). The velocity of the particle determines the change of its position in the next iteration, which can be expressed as \( V(m) = (v_1(m), v_2(m) \cdots v_k(m)) \). Initialize \( P(1) \) and \( V(1) \) between \([0,1]\).

2. Evaluating the fitness of each particle by using the root mean square error (RMSE); the calculation formula is as follows:
\[
\text{RMSE} = \sqrt{\frac{1}{N} \sum_{t=1}^{N} (y(t) - f_o(t))^2}
\]
where \( y(t) \) represents the output of the network, \( N \) represents the number of input sets, \( f_o(t) \) represents the observed value. \( P_{\text{best}} = (p_1, p_2 \cdots p_l) \) represents the optimal position that the particle has searched thus far, and \( G_{\text{best}} = (g_1, g_2 \cdots g_l) \) represents the optimal position that all particles has searched thus far. The initial \( P_{\text{best}} \) and \( G_{\text{best}} \) take the position of the first particle.

3. Replace \( P_{\text{best}} \) or \( G_{\text{best}} \) with the position of the particle, if the current fitness of the particle is better than \( P_{\text{best}} \) or \( G_{\text{best}} \). Update the speed and position of the particle as follows:
\[
\begin{align*}
V_i(m + 1) &= \omega V_i(m) + c_1 r (P_{\text{best}} - P_i(m)) + c_2 r (G_{\text{best}} - P_i(m)) \\
P_i(m + 1) &= P_i(m) + V_i(m + 1)
\end{align*}
\]
where \( \omega \) represents the weight coefficient, \( c_1 \) and \( c_2 \) represent the non-negative acceleration constants and \( r \) is a random number between \([0,1]\). In this paper, we make \( \omega = 0.8, c_1 = c_2 = 2.1 \).

4. The network training stops and the \( G_{\text{best}} \) is output, when \( G_{\text{best}} \) satisfies the convergence condition, or the number of iterations reaches the maximum. Otherwise, the training returns to the second step and continues to iterate until the termination condition is satisfied.
2.2.2. Improvement of Particle Swarm Optimization

Particle swarm optimization has a fast convergence speed. However, the similarity between the particles is high at the later stage of training, which makes the training of the model easy to fall into local optimum. Therefore, the following improvements have been made to the particle swarm optimization algorithm in this paper:

1. Using local averages instead of global averages can make training more difficult to fall into local minimum but take more time [13]. Therefore, this paper combines two methods. Suppose each particle has $s$ neighbors, the $s+1$ particles are represented as $\{P_1, P_2, \cdots P_{s+1}\}$. The average position of these $s+1$ particles is:

$$P_a = \frac{\sum_{i=1}^{s+1} P_i}{s+1} \quad (8)$$

Then, the speed update formula of the particle swarm algorithm becomes:

$$V_i(m+1) = \omega V_i(m) + c_1 r(P_{best} - P_i(m)) + c_2 r(G_{best} - P_i(m)) + c_3 r\omega_1(P_{all} - P_i(m)) \quad (9)$$

where $c_3$ is the non-negative acceleration constants, $\omega_1$ is the weight coefficient of $P_{all}$. Here, we make $c_3=0.2$, and:

$$\omega_1(m) = 1 - 0.8m/m_{\text{max}} \quad (10)$$

where $m_{\text{max}}$ is the maximum number of iterations. In this paper, we set this value to 300.

2. At the later stage of training, all particles are sorted according to their fitness function values and re-initialized the lower-ranking particles, if the global optimum particle has not changed or changed little after many iterations. In this paper, all particles are sorted if the function fitness of $G_{best}$ decreases by less than 0.1 MHz after 40 iterations. Then, the last 25% of the particles are initialized and the initialization range is set to $[-3,3]$ to ensure the diversity of particles.

3. Data Settings and Model Construction

3.1. Setting of Training Data

For an effective prediction of $\text{fo}_F2$, the ionosphere data of Wuhan Station from 2008 to 2013 is used to train the model. The ionosphere data inevitably contain gaps due to ionosphere disturbances. If the data for the previous hour and the next hour are both present, linear interpolation is used to complete the missing data. Otherwise, the data is discarded. Details of the input parameters are described below.

3.1.1. Diurnal and Seasonal Variation

Due to the influence of the earth’s rotation and revolution, the ionosphere has obvious diurnal and seasonal variations, which makes $\text{fo}_F2$ have the same variation. Hour number (HR) from 0 to 23 and day number (DN) from 1 to 365 are used to describe diurnal and seasonal variations, respectively. In order to avoid unrealistic numerical discontinuity, HR and DN are converted to quadrature components:

$$HRS = \sin\left[\frac{2\pi \times HR}{24}\right] \quad (11)$$

$$HRC = \cos\left[\frac{2\pi \times HR}{24}\right] \quad (12)$$

$$DNS = \sin\left[\frac{2\pi \times DN}{365}\right] \quad (13)$$

$$DNC = \cos\left[\frac{2\pi \times DN}{365}\right] \quad (14)$$
3.1.2. Solar and Magnetic Activities

Solar variation causes periodic changes of the ionosphere. As the measurement of solar variation, we chose sunspots number (SSN) and 10.7 cm solar flux F10.7 as input parameters. Sporadic changes in the ionosphere are mostly related to geomagnetic disturbances. Moreover, we chose the Kp index and Dst index to describe magnetic activities [14].

It is more difficult to forecast foF2 during the geomagnetic disturbances period, and some specialized studies have been carried out [15]. However, in this paper, we did not filter the data from the geomagnetic disturbances, but used all the data to train the model.

3.1.3. Present Value of foF2

A large amount of data analysis indicates that the value to be predicted has a good correlation with the value of the present time. Thus, the value of the present time is added to the input set.

Finally, we got nine inputs to predict foF2, as shown in Table 1.

Table 1. Inputs of the model.

| FACTOR                        | Diurnal and Seasonal Variation | Solar and Magnetic Activities | Present Value |
|-------------------------------|-------------------------------|-------------------------------|---------------|
| VARIABLE                      | HRS HRC DNS DNC SSN F10.7 Kp Dst foF2 |

3.2. Architecture of the ENN

In order to build the ENN model, the number of hidden layers and the number of neurons in each hidden layer need to be determined. First, we assumed that the number of hidden layer neurons is a fixed value and the number of hidden layers is a variable. After determining the optimal number of hidden layers, the optimal number of neurons in each hidden layer is calculated.

However, there is still no method to accurately calculate the number of hidden layer neurons in the neural network, and only the approximate range can be calculated. The formula is as follows:

\[ N_h = \sqrt{N_i + N_o + \alpha}, \]

where \( N_h \) is the number of hidden layer neurons, \( N_i \) is the number of input layer neurons, \( N_o \) is the number of output layer neurons and \( \alpha \) is a random number between [1,10]. Too few neurons in the hidden layer will increase the model error and too many neurons will make the model training more complicated. According to formula (15), we determine that the number of the hidden layer neurons range from 6 to 12, and set the number of hidden layers to vary from 1 to 5. These models are trained with the BP algorithm and evaluated by RMSE. The result can be seen in Figure 2.

![Figure 2. Root mean square error (RMSE) of Elman neural network (ENN) model under different hidden layers.](image-url)
It can be seen from Figure 2 that the model takes the minimum value when the number of hidden layers is 3, regardless of the number of neurons in hidden layers. Thus, the number of hidden layers is set to 3.

Then, the number of each hidden layer neurons was set to vary from 6 to 12. The number of neurons from the first hidden layer to the third hidden layer is expressed as a combination \( \{h_1, h_2, h_3\} \), \( h \in [6, 12] \). Finally, 343 combinations are obtained. According to the order of \( \{6, 6, 6\} \), \( \{6, 6, 7\} \) \( \ldots \) \( \{12, 12, 12\} \), these combinations are numbered. The result is shown in Figure 3.

**Figure 3.** RMSE of ENN model under different combination of hidden layers.

As shown in Figure 3, the learning ability of the model increases and RMSE decreases with the increase of the number of hidden layer neurons. The numbers of five combinations which have the smallest RMSE are 242, 268, 294, 327 and 333. Since the increase of the number of hidden layer neurons will increase the training complexity of the IPSO, this paper uses the combination of \( \{11, 9, 7\} \), which is numbered 268, as the number of hidden layer neurons.

4. Discussion

The IPSO-ENN model constructed in this paper has five layers, which are one input layer, three hidden layers and one output layer. The input layer has nine neurons, corresponding to nine inputs. The three hidden layers contain eleven, nine and seven neurons, respectively. The output layer has one neuron that represents the predicted value of foF2. The ENN model, which is trained by BP algorithm, and the BPNN model are chosen as the comparison of the IPSO-RNN model. Both the ENN model and the BPNN model are generated by Matlab neural network toolbox and have the same configuration as the IPSO-ENN model. The data of Wuhan Station in 2014 and 2016 are used to test the forecasting ability of all models. 2014 is the high year of solar activity, while 2016 is the low year of solar activity. The RMSE and relative error (RE) are used to evaluate the prediction results of foF2. The RE formula is as follows:

\[
RE = \frac{1}{N} \sum_{n=1}^{N} \frac{|y - f_o|}{f_o} \times 100
\]

(16)

where \( N \) is the number of input samples, \( f_o \) is the observed value and \( y \) is the predicted value. The prediction results are shown in Table 2. As can be seen from the table, the ENN model has the smaller RMSE than the BPNN model, regardless of whether the solar activity is high or low. After using the IPSO algorithm to optimize the ENN model, the RMSE of the ENN model is further reduced. Specifically, the RMSE of the ENN model is reduced by about 4.30% compared with the BPNN model, and the RMSE of the IPSO-ENN model is reduced by about 8.92% compared with the ENN model. The RMSE in 2014 is larger than that in 2016, suggesting that it is more difficult to predict foF2 in high solar years. However, the RE in 2014 is smaller than that in 2016. This is because foF2 is larger in high solar years.
Table 2. Forecast error. IPSO: improved particle swarm optimization; ENN: Elman neural network; BPNN: back-propagation neural network.

| YEAR | Annual Average | IPSO-ENN | ENN | BPNN |
|------|----------------|----------|-----|------|
|      | foF2/MHz       | RMSE/MHz | RE/%| RMSE/MHz | RE/%| RMSE/MHz | RE/%|
| 2014 | 8.19           | 0.75     | 7.58| 0.81     | 8.53| 0.85     | 8.66|
| 2016 | 6.24           | 0.68     | 8.58| 0.76     | 10.03| 0.79    | 10.20|

Figure 4 shows statistical histogram of the absolute error (AE) between measured foF2 and foF2 obtained from the IPSO-ENN model, the ENN model and the BPNN model. The AE formula is as follows:

\[ AE = f_p - f_o \]  

(17)

The top panel contains histograms for 2014 and the bottom panel contains histograms for 2016. A careful check of these histograms shows that the IPSO-ENN model has the largest AE peak and the most concentrated distribution, both in 2014 and 2016. For example, in 2014, 57.99% of AE are distributed between \([-0.5,0.5]\) and 84.82% of AE are distributed between \([-1,1]\) for the IPSO-ENN model. For ENN model and BPNN, the two ratios are 51.73% and 81.4% and 50.2% and 79.65%, respectively. Combining the above contents, we can find that the annual prediction of foF2 by IPSO-ENN model is more accurate, compared with the ENN model and the BPNN model.

Figure 4. The comparison of absolute error (AE) histograms. (a) AE histograms of the three models in 2014; (b) AE histograms of the three models in 2016.

Figure 5 shows the RMSE of each month in 2014 and 2016. It can be seen that the trend of monthly RMSE of the three models is basically the same and the IPSO-ENN model has the smallest RMSE in every month, which indicates that the IPSO-ENN model is more accurate for the prediction of foF2 in a single month.
Figure 5. The monthly performance comparison. (a) RMSE of three models at each month in 2014; (b) RMSE of three models at each month in 2016.

A further analysis was performed by investigating the predictability of foF2 diurnal variability. Two-day data from 2014 and 2016 were selected and they were plotted together with the predicted values from the IPSO-ENN model, the ENN model and the BPNN model, as shown in Figure 6. We found that all three models fit the rising and falling segments of foF2 curve well and reflect the diurnal trend of foF2. However, the IPSO-ENN model fit significantly better in the peaks and troughs of the curve. We also found that the diurnal variation of foF2 in 2016 was less than that of foF2 in 2014, which may have caused the RMSE in 2016 to be less than the RMSE in 2014.

In order to evaluate the forecasting ability of the model during the ionosphere disturbance period, the data of an ionosphere disturbance occurred on December 20, 2015 was selected for the test. This disturbance was caused by a geomagnetic storm, and the change of Kp index during the period is shown in Figure 7. Comparing Figures 7a and 7b, it can be found that ionosphere disturbances occur after geomagnetic storms. The change of foF2 during the ionosphere disturbance period is very intense, which leads to a large error in the prediction. The RMSE of the IPSO-ENN model during this period is 1.23 MHz, which is less than 1.36 MHz of the ENN model and 1.25 MHz of the BPNN model. Details of the ionosphere disturbance should be incorporated into the IPSO-ENN model to improve the prediction ability of the model, which requires further investigation.
Figure 7. Data during ionosphere disturbances. (a) Change of Kp index during ionosphere disturbances; (b) observed and predicted values during ionosphere disturbance.

5. Conclusions

In this paper, a new IPSO-ENN model for foF2 forecast is developed. Data of Wuhan Station from 2008 to 2016 are used to train the model and test its forecasting ability. Experiments show that the optimum structure of the IPSO-ENN model is three hidden layers containing 11, nine and seven neurons, respectively. By comparing with the ENN model and the BPNN model, we found that the prediction ability of the IPSO-ENN model is optimal at all time scales. During the ionosphere disturbance, the prediction RMSE of the IPSO-ENN model is also the smallest among the three models. Further research is needed to improve the predictive abilities of the IPSO-ENN model in the ionosphere disturbance period.

Author Contributions: Conceptualization, J.F. and J.W.; Methodology, J.F.; Software, C.L.; Investigation, J.F. and C.L.; Data Curation, J.H.; Visualization, Y.L.; Writing—original draft preparation, C.L.; writing—review and editing, Y.L. and J.F.

Funding: This research received no external funding.

Acknowledgments: The ionospheric data used in this paper are from Data Center for Geophysics, Data Sharing Infrastructure of Earth System Science, National Science & Technology Infrastructure of China. (http://geospace.geodata.cn). Other data are from Space Science Data Center. (www.geodata.cn)

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Williscroft, L.A.; Poole, A. Neural Networks, foF2, sunspot number and magnetic activity. Geophys. Res. Lett. 1996, 23, 3659–3662. [CrossRef]
2. Yue, X.N.; Wan, W.X.; Liu, L.B.; Ning, B.Q.; Zhao, B.Q. Applying artificial neural network to derive long-term foF2 trends in the Asia/Pacific sector from ionosonde observations. J. Geophys. Res. 2006, 111, 1–11. [CrossRef]
3. Noraset, W.; Kornyanat, H.; Pornchai, S.; Takuya, T. A comparison of neural network-based predictions of foF2 with the IRI-2012 model at conjugate points in Southeast Asia. Adv. Space Res. 2017, 59, 2934–2950.
4. Oyeyemi, E.O.; Poole, A.W.V.; McKinnell, L.A. On the global model for foF2 using neural networks. Radio Sci. 2005, 40, 1–15. [CrossRef]
5. McKinnell, L.A.; Poole, A.W.V. Neural network-based ionospheric modelling over the South African region. S. Afr. J. Sci. 2004, 100, 519–523.
6. Wang, R.P.; Zhou, C.; Deng, Z.X.; Ni, B.B.; Zhao, Z.Y. Predicting foF2 in the China region using neural networks improved by the genetic algorithm. J. Atmos. Sol. Terr. Phys. 2013, 92, 7–17. [CrossRef]
7. Zhao, X.K.; Ning, B.Q.; Liu, L.B.; Song, G.B. A prediction model of short-term ionospheric foF2 based on AdaBoost. Adv. Space Res. 2014, 53, 387–394. [CrossRef]
8. Li, P.X.; Wang, D.; Wang, L.J.; Lu, H.C. Deep visual tracking: Review and experimental comparison. Pattern Recognit. 2018, 76, 323–338. [CrossRef]
9. Wen, L.L.; Zhou, K.L.; Yang, S.L.; Lu, X.H. Optimal load dispatch of community microgrid with deep learning based solar power and load forecasting. *Energy* 2019, 171, 1053–1065. [CrossRef]

10. Acharya, R.; Roy, B.; Sivaraman, M.R.; Dasgupta, A. Prediction of ionospheric total electron content using adaptive neural network with in-situ learning algorithm. *Adv. Space Res.* 2011, 47, 115–123. [CrossRef]

11. Jia, W.K.; Zhao, D.A.; Zheng, Y.J.; Hou, S.J. A novel optimized GA-Elman neural network algorithm. *Orig. Artic.* 2019, 31, 449–459. [CrossRef]

12. Yun, P.P.; Ren, Y.F.; Xun, Y. Energy-Storage optimization strategy for reducing wind power fluctuation via Markov prediction and PSO method. *Energies* 2018, 11, 3393. [CrossRef]

13. Fang, H.Q.; Shen, Z.Y. Optimal hydraulic turbogenerators pid governor tuning with an improved particle swarm optimization algorithm. *Proc. CSEE* 2005, 25, 120–124.

14. Nakamura, M.I.; Maruyama, T.; Shidama, Y. Using a neural network to make operational forecasts of ionospheric variations and storms at Kokubunji. *J. Natl. Inst. Inf. Commun. Technol.* 2009, 56, 391–406. [CrossRef]

15. Mandrikova, O.V.; Polozov, Y.A.; Soloviev, I.S.; Fetisova (Glushkova), N.V.; Zalyaev, T.L.; Kupriyanov, M.S.; Dmitriev, A.V. Method of analysis of geophysical data during increased solar activity. *Pattern Recognit. Image Anal.* 2016, 26, 406–418. [CrossRef]

© 2019 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).