LSTM-Based Battlefield Electromagnetic Situation Prediction

Hengchang Zhang, Shengjie Zhao(✉), and Rongqing Zhang

School of Software Engineering, Tongji University, Shanghai 201804, China
{hengchang_zhang,shengjiezhao,rongqingz}@tongji.edu.cn

Abstract. In the modern battlefield, the complex electromagnetic environment has brought considerable challenges to assessing the electromagnetic situation. Traditional electromagnetic situation assessment methods are difficult to process massive high-dimensional data, making it challenging to predict the electromagnetic situation. In recent years, the in-depth development of deep learning has provided a breakthrough in the electromagnetic situation field. However, related research mainly focuses on threat assessment and electromagnetic situation complexity prediction. There are few papers on electromagnetic situation prediction using machine learning. In this paper, we propose an electromagnetic situation prediction method based on long short-term memory (LSTM). We first build an attack-defense model based on deep reinforcement learning to simulate the electromagnetic situation. Then we use LSTM to predict the development of the situation and improve the loss function to reduce the prediction error. Furthermore, we analyze the impact of different situation features on the final win rate and use a small amount of situation information to predict the win rate with high accuracy. The experimental results show that this method can effectively predict the electromagnetic situation, providing excellent decision support for commanders.

Keywords: LSTM · Electromagnetic situation · Situation prediction

1 Introduction

The battlefield electromagnetic situation refers to the state and the development trend of the confrontation between the two sides on the battlefield. With the wide application of spectrum equipment and electromagnetic technology, the battlefield electromagnetic environment has become increasingly complex, which poses a con-

This work is supported in part by the National Key Research and Development Project under Grant 2019YFB2102300 and 2019YFB2102301, in part by the National Natural Science Foundation of China under Grant 61936014 and 61901302, in part by the Fundamental Research Funds for the Central Universities (China), and is also funded by State Key Laboratory of Advanced Optical Communication Systems Networks, China.

J. Xiong et al. (Eds.): MobiMedia 2021, LNICST 394, pp. 768–781, 2021.
https://doi.org/10.1007/978-3-030-89814-4_57

siderable challenge to assessing and predicting the electromagnetic situation. Most existing situation assessment methods focus on providing commanders with the past and current electromagnetic conditions but cannot predict future development, so it is difficult to formulate strategies based on the development trends. Therefore, accurately predicting and evaluating the battlefield electromagnetic situation has become an urgent problem in the future battlefield.

Endsley [1] divided the situation assessment process into three stages: situation awareness, situation comprehension, and situation prediction. At present, the research on electromagnetic situation mainly focuses on the first two stages, namely electromagnetic situation awareness and electromagnetic situation comprehension. Cai *et al.* [2] analyzed the elements of the battlefield electromagnetic situation, and used a signal population distribution method to calculate the complexity of the battlefield electromagnetic environment. Shen *et al.* [3] designed a 3D electromagnetic situation visualization system based on geographic information system (GIS). Yuan *et al.* [4] proposed an index system to characterize the complexity of the electromagnetic environment, and predicted the complexity based on Bayesian networks.

Traditional electromagnetic situation assessment methods are complex and difficult to process massive high-dimensional data. Deep learning methods, as a comparison, have more powerful representation learning capabilities and can automatically extract various complex features from the original data. In recent years, various deep learning algorithms have been proposed for different types of problems, such as Convolutional Neural Networks [5], Recurrent Neural Networks [6], and Residual Networks [7], etc. These methods are widely used in image recognition, object detection, language modeling, and other fields [8]. In the employment of deep learning in electromagnetic situation investigation, related research focuses on electromagnetic threat prediction [9] and electromagnetic interference prediction [10], but there are few papers about the battlefield electromagnetic situation prediction.

In this paper, we propose an LSTM-based electromagnetic situation prediction method for the two-sided battlefield, which can achieve efficient prediction of various electromagnetic situation parameters with high accuracy. This method requires a large amount of data for model training. To solve the problem of insufficient data sources, we first establish an attack-defense model based on deep reinforcement learning. Then, we build an LSTM situation prediction model and improve the loss function to smooth the prediction results. We use the simulated data to train and test the model, and find that compared to MSE loss, the improved loss function can reduce the prediction error (RMSE) by about 10.35%. In addition, we analyze the impact of different situation features on the final win rate by analysis of variance and use a small amount of situation information to predict the win rate with high accuracy.

The rest of this paper is organized as follows: Sect. 2 proposes the attack-defense model based on deep reinforcement learning to generate data for situation prediction. Section 3 introduces the LSTM model we use. Section 4 introduces our experiments and the results on situation prediction. Section 5 concludes the paper.

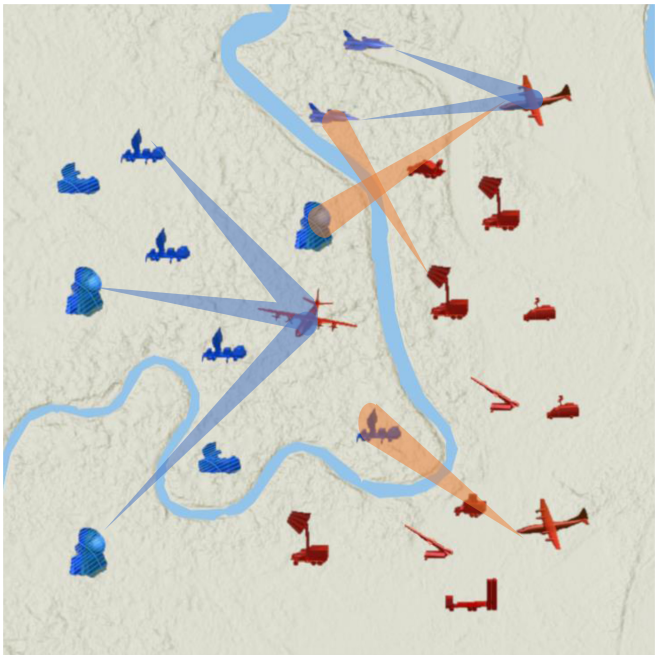


Fig. 1. The simulated battlefield.

2 Attack-Defense Model

The battlefield electromagnetic situation is obtained from the original battlefield data through data fusion methods. However, the battlefield data are confidential to all countries, and it is hard to get. Therefore, we build an attack-defense model, a type of wargaming [11], to generate simulated data. Wargaming has a long history. It is mainly used to simulate the battlefield environment, on which the commander can deploy their strategies and prove their ideas. The traditional wargaming model requires people to manually set the battlefield environment and manually manipulate the battlefield elements to advance the war. The cost of acquiring data in this way is unacceptable. In 2017, Deepmind proposed AlphaGo Zero [12], which is a Go program based on deep reinforcement learning. The program generates data through self-playing games and uses the data for training, thus becoming a Go expert from scratch. This inspires us to build a deep reinforcement learning model to compete with each other in the simulated battlefield environment and use the data to analyze and predict the battlefield situation.

2.1 Situation Description

Figure 1 is the scene diagram of the attack-defense model. In one game, the two sides participating in the game each has a certain number of missiles and other

equipment. Use $s_t = [s_t^r, s_t^b]$ to represent the state of two sides at time-step t. s consists of two matrices with the same shape.

The shape of s^r or s^b is 3 rows and $M+W$ columns. Each column represents one type of equipment. The first row represents the equipment's quantity, the second represents the code of the equipment, and the third represents the total remaining value of the equipment in this column. The first M columns represent M types of missiles. There can be multiple missiles of each type, so the first row of the first M columns can take a number greater than 1. The last W columns represent equipment other than missiles. There is at most one piece of equipment in each column, so the first row of the last W columns can only be 0 or 1.

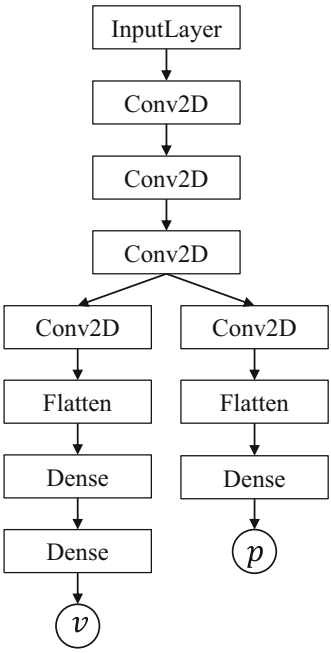


Fig. 2. Structure of the CNN used in the attack-defense model.

2.2 Decision Model

The game is turn-based. Both sides attack alternatively. In each time-step of the game, one side chooses one of its missiles to attack the other side's non-missile unit, so the decision can be denoted by an $M * W$ matrix. The probability of a successful attack is related to the accuracy of the missile acc_i and the opponent's overall defense value def, as in

$$prob = \frac{c \cdot acc_i}{c + def} \tag{1}$$

where c is a preset parameter. If a unit is successfully hit, the unit's value will be reduced by the missile's attack power. If the value of a unit is reduced to 0, then the unit will be removed.

The decision model's purpose is to make the appropriate decision π according to the current state s. We use a neural network guided MCTS [12] to search for the best move. Figure 2 describes the structure of the neural network $(p, v) = f_\theta(s)$.

2.3 Model Training Process

The neural network f_θ is trained based on the data generated from games of self-play.

At the initial state $i = 0$, the neural network's parameter θ_0 is randomly initialized. Then at each iteration $i > 0$, multiple games are played. At each time-step t in one game, an MCTS search $\pi_t = a_{\theta_{i-1}}(s_t)$ is executed under the guidance of the neural network. A game terminates at time-step T when one side's total value becomes 0 or when the game exceeds a maximum length. A sample (s_t, π_t, z_t) is produced at each time-step t, where $z_t = 1$ if the current player is the final winner of the game, and $z_t = -1$ if otherwise. The neural network is trained from the data (s, π, z) to minimize the difference between p and π, also v and z.

When the training of the neural network f_θ is completed, we then use the model to play more games, and use the data generated to analyze the situation.

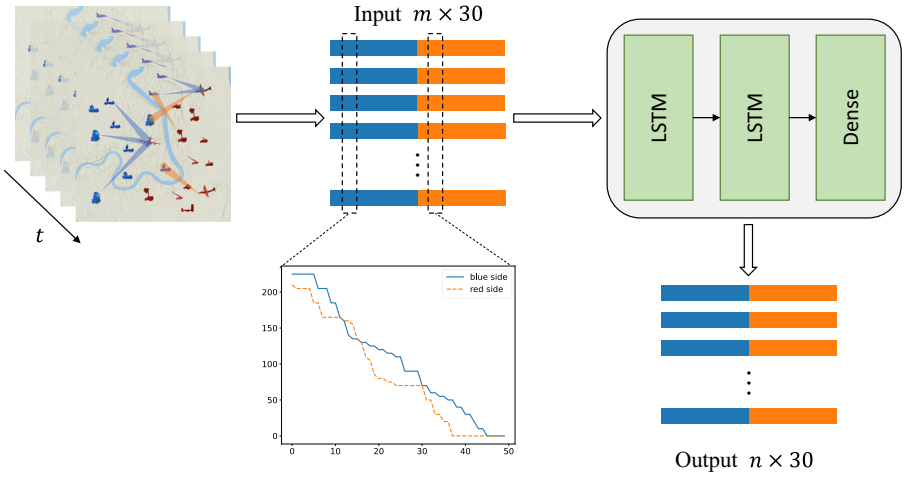


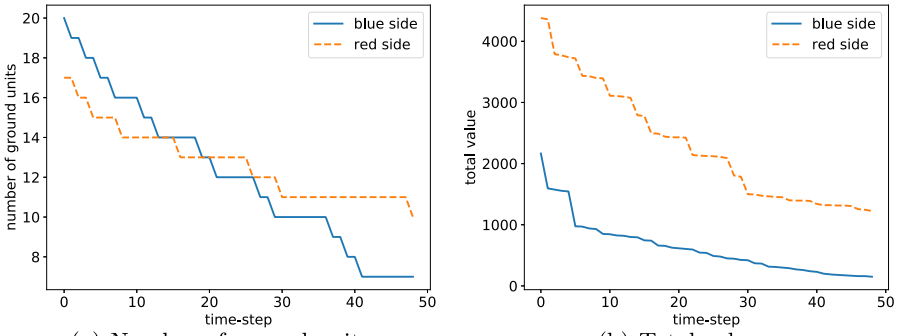
Fig. 3. Overall architecture of the LSTM.

3 LSTM-Based Prediction Method

In the previous section, we build an attack-defense model to simulate battlefield electromagnetic data. This section proposes the situation prediction method based on LSTM. First, extract the situation features from the original data, then preprocess the data, in addition, build an LSTM model to predict the situation. Figure 3 is the overall architecture of the proposed method, where m and n are the number of input time-steps and output time-steps, respectively.

3.1 Feature Extraction

The data generated above represents the number and type of battlefield elements. For a game with T time-steps, the dimension of the data collected is $[T, 2, 3, 100]$. The original data are sparse, and it is difficult to obtain the overall situation from it. We extract 15 features from the original data, as is listed in Table 3. Figure 4 shows 2 of the 15 features of both sides in a 48-time-step game.



(a) Number of ground units (b) Total value

Fig. 4. 2 of 15 features in one game. (Color figure online)

As a result, the situation prediction problem is transformed into a time series prediction problem, which predicts the situation of both red and blue in the future m time-steps through the known situation of the red and blue sides of n time-steps.

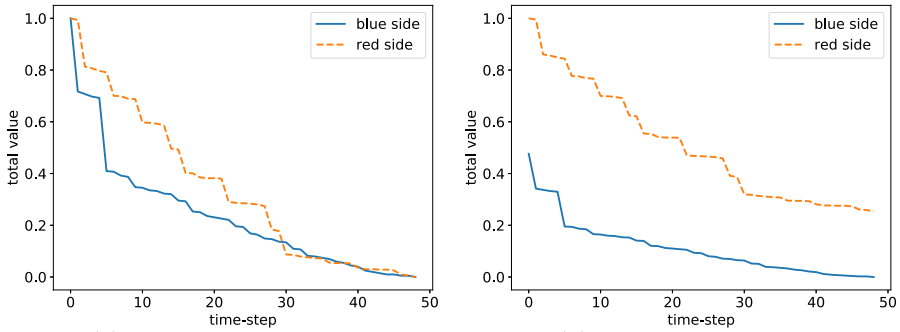
3.2 Data Preprocessing

As shown in Fig. 4, the difference between the absolute values of different features is very large. For example, the number of ground units is generally dozens, but the value of total units can reach several thousand. The vast difference between different features will affect the training of the model. The feature with a larger value will occupy a larger weight in the initial stage of training, which makes it

difficult for the feature with a relatively small value to play a role. Therefore, the raw data need to be normalized. We use the min-max normalization, namely:

$$x'_i = \frac{x_i - min(x)}{max(x) - min(x)}. \tag{2}$$

In this question, if the 30 features are normalized separately, the contrast between the two sides will be lost. Figure 4(b) represents the total value of the red and blue sides of 48 time-steps in one game. It can be seen that in the beginning, the total value of the red side is more than twice that of the blue side. If the two sides' features are normalized separately, after normalization, the total value is shown in Fig. 5(a). It can be seen that the total value of the two sides has both become 1 at the initial state.



(a) Normalize separately (b) Normalize together

Fig. 5. Comparison of two normalization methods. (Color figure online)

To avoid this problem, we normalize the same feature of both sides simultaneously. The result is shown in Fig. 5(b). This method maps the total value to the $[0, 1]$ interval while preserving the comparison between the two sides, and the normalized data are suitable for training in a neural network.

After feature extraction, the $[T, 2, 3, 100]$ raw data become the $[T, 30]$ features. To fully utilize the data for training the neural network, we need to convert the time series data to pairs of input and output sequences using the sliding window method. When predicting n time-steps with m time-steps, use data of the former m time-steps as input and the data of the following n time-steps as output. In this case, for a game with T time-steps, the number of samples that can be generated is $T - m - n + 1$.

3.3 LSTM Model

In time series forecasting problems, RNN (Recurrent Neural Network) [13] is a commonly used method. In recent years, RNN has achieved great success in many

problems such as speech recognition, machine translation, and image description. However, RNN is hard to handle the long-term dependencies [14]. To address this problem, LSTM [15] introduces the cell state to maintain long time memory and replaces the hidden layer neuron with a special LSTM structure with three gates, namely input gate, forget gate, and output gate. The input gate controls how much new information is added to the cell state. The forget gate decides how much cell state to forget. The output gate controls how much of the updated cell state to output.

We build a neural network with two layers of LSTM with 128 units and one fully connected layer. We use Relu as the activation function and Adam as the optimizer. In addition, we add a penalty term to the MSE loss function to smooth the prediction results and improve the prediction accuracy, as in

$$loss = \mathrm{MSE} + \sum_i \sum_t \mathrm{Relu}(d_{t+1}^i - d_t^i) \tag{3}$$

where d_t^i is the value of the i-th feature at time-step t. In Sect. 4, we will introduce the prediction results and the comparison of different loss functions.

4 Experimental Results

In the previous section, we extract 30 features from the self-play data and propose an LSTM model to predict the electromagnetic situation. This section uses the simulated data generated by the attack-defense model to evaluate the prediction method. We further analyze the features by ANOVA (Analysis of Variance) and predict the win rate by a small part of these features.

4.1 Situation Prediction

We use the data generated by self-play to train and test the LSTM model proposed in Sect. 3. A total of 800 games are generated. Each game contains up to 50 time-steps. We randomly select 80% of the data as the training set and use 20% of the data as the test set. The batch size is set to 16.

Table 1. Results of different loss function

Loss function	RMSE (10^{-2})	MAPE (%)
MSE	1.2986	4.72
MSE+penalty	1.1642	3.44

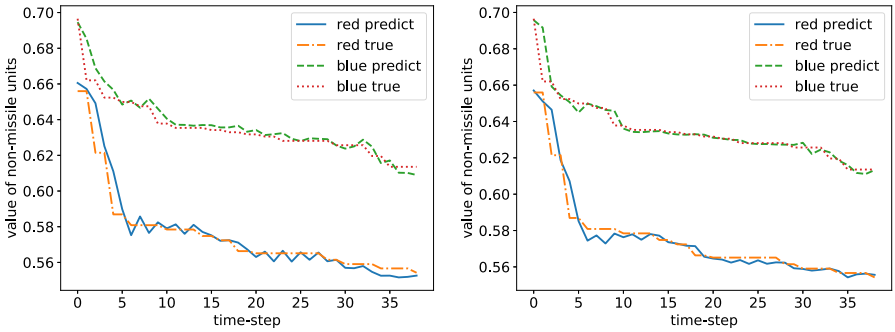
The result on the test set is shown in Table 1, the input time-step is 12, and the output time-step is 1. MAPE and RMSE are defined by

$$\mathrm{MAPE} = \frac{100\%}{n} \sum_{i=1}^{n} \frac{|\widetilde{y_i} - y_i|}{y_i} \tag{4}$$

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\widetilde{y_i} - y_i)^2} \tag{5}$$

where $\widetilde{y_i}$ and y_i indicate the predicted value and true value, respectively.

Figure 6 compares the result of the model with a pure MSE loss function and the model with an MSE combines penalty loss function. It can be seen that when the penalty is added, the prediction result is smoother and more close to the real situation.



(a) MSE loss function (b) Improved loss function

Fig. 6. Prediction results of different loss function.

In the attack-defense model, a move is decided only depends on the current state of both sides and has nothing to do with the path to the current state. In the situation prediction problem, however, more input time-steps can provide the neural network with more information about player preferences and equipment attributes, thereby improving prediction accuracy. We test a variety of combinations of input time-steps and output time-steps, and evaluate the results by RMSE. The results are listed in Table 2.

Table 2. RMSE of different input and output time-step combinations

RMSE (10^{-2})		Input Time-Step				
		1	2	3	4	5
Output Time-Step	1	1.5396	1.2524	1.2470	1.1832	1.1773
	2	1.5731	1.5277	1.4655	1.4327	1.4001
	3	1.8373	1.7904	1.7224	1.7101	1.6616
	4	2.0359	2.0386	1.9866	1.9206	1.8697
	5	2.3092	2.2801	2.2724	2.2617	2.1951

4.2 Win Rate Evaluation Based on Situation

Apart from the situation prediction, the evaluation of the current winning probability is also a critical subject. A proper assessment of the current situation is conducive to the formulation of the following action strategies. In the attack-defense model, the output v of the neural network $f_\theta(s)$ gives the win rate estimation, but the input is the entire current state s. We hope to evaluate the winning probability through the situation. On the one hand, it can reduce the dimension of input data and reduce data acquisition difficulty; on the other hand, it can simplify the model and improve prediction efficiency.

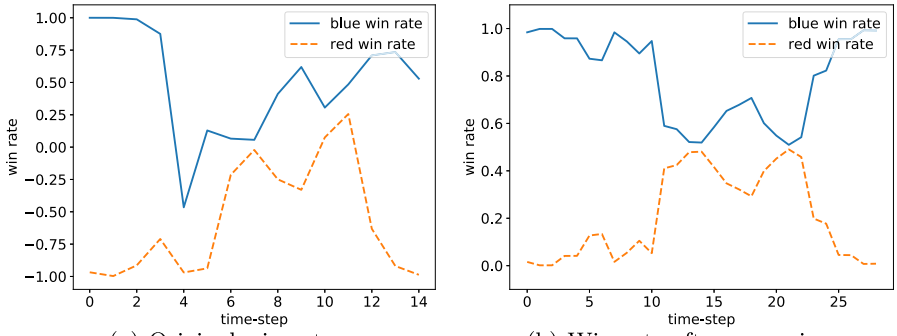
At the t-th time-step of one game, the state of both sides is $s_t = [s_t^r, s_t^b]$. From the aforementioned neural network, the estimated win rate of the current state v_t can be obtained by $(p_t, v_t) = f_\theta(s_t)$. Moreover, the features d_t can be obtained through the feature extraction process. We then take d_t as input and v_t as output to train a neural network to evaluate the win rate based on the situation.

Figure 7(a) is the estimation of the win rate v_t from the red and blue sides in one game. It can be seen that their estimations are not always the same. That is, for example, in some consecutive time-steps, both sides may "think" they are going to lose.

Therefore, we average the win rate estimations for two consecutive rounds. If the red side attacks at time-step t, the win rate estimation is denoted as w_t^r, then the estimation at the next time-step is denoted as w_{t+1}^b. Combining these two time-steps, the win rate of the red side at time-step t is

$$\widetilde{w}_t^r = \frac{w_t^r + 1}{w_t^r + w_{t+1}^b + 2} \tag{6}$$

and the win rate of the blue side is $\widetilde{w}_t^b = 1 - \widetilde{w}_t^r$. After the processing, the win rate of both sides in Fig. 7(a) is converted into Fig. 7(b). We can then take $(d_t, \widetilde{w}_t^r)$ as a sample.



(a) Original win rate (b) Win rate after averaging

Fig. 7. Win rate in one game. (Color figure online)

We build a neural network with seven fully connected layers with dropouts, where the activation function is Relu, the output layer is Sigmoid to ensure the output is in $[0, 1]$, the loss function is MSE, and the optimizer is Adam. The network's input is the 30 features of both sides d_t, and the output is the predicted win rate of the red side $\widetilde{w}_t^r$.

We use the data of 300 games, which contains a total of 12126 samples, to train and test the neural network. We use 9700 samples as the training set and others as the test set. The MAPE on the test set is 10.3%, and the MSE is 0.0107.

4.3 Analysis of Variance

In the previous part, we use all the 30 features to predict the win rate. However, it may be challenging to obtain all the situation features on the actual battlefield for various reasons. Therefore, it is necessary to study using fewer features to predict the win rate.

Analysis of variance is widely used to examine the influence of the independent variable A on the dependent variable Y. It partitions the sum of squares (SS) of the dependent variable Y into the sum of squares between groups (SS_A) and the sum of squares within groups (SS_E). SS_A represents the deviation of Y when A takes different values, and SS_E represents the random error of Y when A takes a fixed value. If A is a continuous variable, the values of A can be divided into several levels. ANOVA is a form of hypothesis testing, and the null hypothesis is that all values of factor A have the same effect on Y. If the null hypothesis is true, (7) can be proven, where MS is mean square, k is the total number of possible values A can take, n is the total number of cases, and $F_{k-1,n-k}$ is a F-distribution with $(k-1, n-k)$ degrees of freedom.

$$\frac{MS_A}{MS_E} \sim F_{k-1,n-k} \tag{7}$$

If $MS_A/MS_E > F_{k-1,n-k}(\alpha)$, reject the null hypothesis; otherwise, accept the null hypothesis, where a is the significance level.

The results of the ANOVA for the 30 features are shown in Table 3, which is sorted in descending order over the F value. A larger F value means the feature has a greater influence on the win rate.

We use the first i and last i features in Table 3 ($i = 1, 2, ..., 10$) as the input of the neural network to predict the win rate. The MSE between the predicted value and true value on the test set are shown in Fig. 8. It can be seen that the MSE of prediction using features with larger F-values is relatively smaller. Furthermore, as the number of input features increases, the difference of the MSE between the best and the worst tends to shrink. The MSE of the first ten features is 0.0117, which is very close to using all the 30 features, i.e., 0.0107.

Table 3. Results of ANOVA

Feature	F	PR(>F)
red: value of non-missile units	2863.0286	0
red: number of non-missile units	1675.8144	0
blue: value of non-missile units	1373.2211	7.9974E−285
red: value of anti-ground missiles	375.53513	2.0574E−82
red: value of anti-air missiles	343.7655	1.0750E−75
red: number of missiles	276.7351	1.8566E−61
blue: total value	188.9768	1.1120E−42
blue: number of missiles	182.6833	2.5074E−41
blue: number of non-missile units	138.3295	9.1909E−32
red: value of ground units	130.6649	4.1827E−30
blue: value of anti-ground missiles	117.6721	2.7310E−27
red: value of air units	115.0895	9.9193E−27
blue: value of missiles	76.1177	3.0200E−18
blue: value of anti-air missiles	59.5973	1.2552E−14
red: number of anti-air missiles	56.2550	6.8097E−14
blue: number of anti-ground missiles	55.1852	1.1705E−13
red: total value	55.0848	1.2315E−13
blue: number of anti-air missiles	39.3230	3.7141E−10
blue: value of ground units	36.1119	1.9166E−09
red: number of air units	34.0109	5.6202E−09
red: number of ground units	31.8621	1.6922E−08
red: total defense	31.6575	1.8797E−08
red: value of missiles	24.9382	6.0022E−07
blue: number of ground units	22.9143	1.7139E−06
red: number of all units	22.6500	1.966E−06
red: number of anti-ground missiles	12.1402	4.9519E−04
blue: number of all units	11.4244	7.2711E−04
blue: number of air units	5.0296	0.0249
blue: total defense	3.2456	0.0716
blue: value of air units	0.4646	0.4954

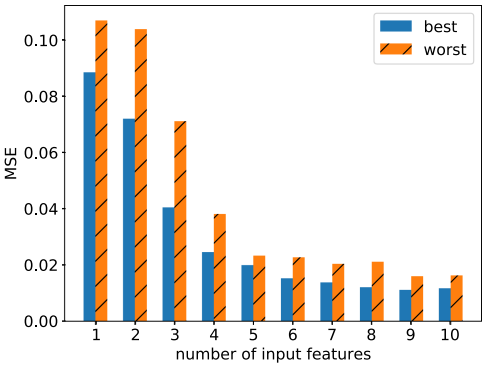


Fig. 8. MSE using different features for prediction.

5 Conclusion

In this paper, we first constructed an attack-defense model based on deep reinforcement learning and used it to generate a large amount of battle data. Next, the electromagnetic situation feature extraction method was proposed, 30 features were extracted from the original data to characterize the electromagnetic situation, and the electromagnetic situation prediction model based on LSTM was constructed. Third, we proposed a win rate evaluation model with situation features as input and win rate as output. We further used ANOVA to analyze the extracted features and showed that we could use a small number of the most relevant features to predict the win rate while maintaining high prediction accuracy.

References

1. Endsley, M.R.: Toward a theory of situation awareness in dynamic systems. Human Factors **37**(1), 32–64 (1995)
2. Cai, X., Song, J.: Analysis of complexity in battlefield electromagnetic environment. In: 2009 4th IEEE Conference on Industrial Electronics and Applications, pp. 2440–2442. IEEE (2009)
3. Shen, D., Jiang, B., Liu, T., et al.: Realizing of a battlefield electromagnetic situation system. In: 2017 36th Chinese Control Conference (CCC), pp. 10304–10309. IEEE (2017)
4. Yuan, H., Wenrui, D., Chunlei, L.: Assessment and prediction of complex electromagnetic environment based on Bayesian network. In: 2017 IEEE International Conference on Unmanned Systems (ICUS), pp. 120–125. IEEE (2017)
5. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. Commun. ACM **60**(6), 84–90 (2017)
6. Jozefowicz, R., Vinyals, O., Schuster, M., et al.: Exploring the limits of language modeling. arXiv preprint arXiv:1602.02410 (2016)

7. He, K., Zhang, X., Ren, S., et al.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
8. Minar, M.R., Naher, J.: Recent advances in deep learning: an overview. arXiv preprint arXiv:1807.08169 (2018)
9. Wei, C., Qi, L., Wu, R., Lin, Y.: Electromagnetic spectrum threat prediction via deep learning. In: Liu, S., Yang, G. (eds.) ADHIP 2018. LNICST, vol .279, pp. 433–442. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-19086-6_48
10. Shu, Y.F., Wei, X.C., Fan, J., et al.: An equivalent dipole model hybrid with artificial neural network for electromagnetic interference prediction. IEEE Trans. Microw. Theory Tech. **67**(5), 1790–1797 (2019)
11. Perla, P.P., McGrady, E.D.: Why wargaming works. Naval War Coll. Rev. **64**(3), 111–130 (2011)
12. Silver, D., Schrittwieser, J., Simonyan, K., et al.: Mastering the game of go without human knowledge. Nature **550**(7676), 354–359 (2017)
13. Cleeremans, A., Servan-Schreiber, D., McClelland, J.L.: Finite state automata and simple recurrent networks. Neural Comput. **1**(3), 372–381 (1989)
14. Bengio, Y., Simard, P., Frasconi, P.: Learning long-term dependencies with gradient descent is difficult. IEEE Trans. Neural Netw. **5**(2), 157–166 (1994)
15. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)