Comput Syst Sci & Eng (2018) 4: 267–274
© 2018 CRL Publishing Ltd

International Journal of
Computer Systems
Science & Engineering

Analysis and Application of the Spatio-temporal Feature in Wind Power Prediction

Ruiguo Yu[1,2], Zhiqiang Liu[1,2], Jianrong Wang[1,3], Mankun Zhao[1,2], Jie Gao[1,3], Mei Yu[1,3,*]

[1] *School of Computer Science and Technology, Tianjin University, Tianjin 300350, China*
{rgyu; tjubeisong; wjr; zmk; gaojie; yumei}@tju.edu.cn
[2] *Tianjin Key Laboratory of Cognitive Computing and Application, Tianjin 300350, China*
[3] *Tianjin Key Laboratory of Advanced Networking, Tianjin 300350, China*

The spatio-temporal feature with historical wind power information and spatial information can effectively improve the accuracy of wind power prediction, but the role of the spatio-temporal feature has not yet been fully discovered. This paper investigates the variance of the spatio-temporal feature. Based on this, a hybrid machine learning method for wind power prediction is designed. First, the training set is divided into several groups according to the variance of the input pattern, and then each group is used to train one or more predictors respectively. Multiple machine learning methods, such as the support vector machine regression and the decision tree, are used in the proposed method. Second, all the trained predictors are adopted to make predictions for a sample, and the results generated from these predictors will be combined by an optimized combination method based on the variance. The experimental results based on the NREL dataset show that the method adopted in this paper can achieve a better performance than the stage-of-the-art approaches

Keywords: spatio-temporal feature; power wind prediction; variance; grouping; multi-predictors

0.1 Introduction

Short-term wind power prediction, aiming at forecasting the wind power of a turbine or turbine groups after a few minutes, is a key to ensure the stability of the power grid after wind power generation systems are connected to the grid. With the concept of sustainable development, the wind power, a type of the important clean renewable energy, has drawn an increasing amount of attention in recent years [1].

Machine learning methods are popular in the problem of short-term wind power prediction. The other two commonly used methods are mathematics and physical based methods, and statistics based methods [2]. The problem of wind power prediction is considered as a regression problem in the machine learning [3, 4], and the time series measurements of turbines in the past times are used to train a model. Generally, some measurements which are close to the target time are set as features to make predictions.

*Corresponding Author. E-mail: yumei@tju.edu.cn

Poloczek J, Treiber N A, Kramer O. proposed a novel method to extract the features, in which the spatio-temporal information is concluded into the input pattern [5]. We will call the feature with spatio-temporal information as *FST* in this paper.

The *FST* feature of a turbine can present the wind energy of other turbines in the past few times within a certain range. The performance of this feature is superior to the traditional ones in the experiment, and it is theoretically possible to make a further improvement on the accuracy of wind prediction because it contains a more complete dynamic change process of the wind energy.

Nils André Treiber et al. took advantage of the *FST* feature in their work [6], however, they simply replaced the traditional features with the *FST* features. In fact, the variance of the *FDT* feature itself reflects the stability situation of a certain region's wind energy output when the information of temporal-spatial information is included. Hence, the *FST* feature becomes a key factor in predicting the wind energy. Experiments show that when the variances of the features of the datasets to be predicted

remains within a range and the variances of the features of the training set also remains within this range, it can achieve a higher prediction accuracy, as shown in Fig 1.

To use the multiple models to solve the same problem is an effective way to get a better result [7], and in this case, how to integrate the results of the multiple models has a significant impact on the final prediction effect. Therefore, the weighted average is a simple and effective method to deal with this problem. However, there must be a way to calculate the weights of each model. Heinermann J et al. proposed a method in their work [8] and applied it to the wind energy prediction. They evaluate the mean squared error (*MSE*) of each base model, and then use the reciprocal of *MSE* as the weight of the base model, as (1) shows.

$$w = \frac{1}{MSE} \tag{1}$$

In this paper, the method of combining multi-predictors with (1) is called the *RW* (use reciprocal as the weight) method. Compared with the traditional methods, *RW* method performs well, but it still needs to improve.

First, the function image of is shown in Fig 2, which reflects the relationship between the *MSE* of the base model and its weight in the RW method. It can be seen that when the *MSE* of the base model is distributed within an interval with small values, the weight decreases rapidly with the increase of *MSE*. Whereas when the *MSE* is distributed within an interval with big values, the weight changes slowly, In the practical applications, the *MSE* of the base model is greater than 5 in most cases, which makes the weights among the models become too close, and cannot make the well-performed models account for higher weights and more important status.

Second, the reciprocal of the error is set as the weight, so the weight is linearly related to 1/MSE, however, the more adaptive non-linear function is superior to the linear function, which can better express the relationship between them. Thus, there are at least two points where the RW method can be improved.

Based on the basic regression models in the machine learning, this paper proposes a hybrid wind power prediction method called *ML_HWP*, in which *FST* is used.

First, we divide the training data into groups according to the variance of the input feature, so that the variance of each data set remains in a fixed interval. In the process of grouping, each example is endowed with an attribute, *type*, which indicates the serial number of the group containing it.

Second, we take each set of training data to train one or more models. The different machine learning algorithms are adopted to train multiple models. After the model training is completed, the effect of each model on each type of data is estimated through the cross validation. Finally, a number of trained models are acted on each example when making prediction, and the weighted average is used to combine results they produced. This paper adopts an improved method based on *RW* to calculate the weights. Comparing with five methods, including support vector machine regression, k-nearest neighbors regression, decision tree, artificial neutral networks and the RW method, ML_HWP can improve the prediction accuracy to 3.975%, 11.484%, 16.613%, 4.74% and 3.882% respectively.

The rest of this paper is organized as follows: section 2 introduces the commonly used machine learning methods in the wind power prediction and the extraction method of temporal-spatial features. Section 3 describes the model construction process and the important algorithms of ML_HWP method. Section 4 verifies the proposed algorithms with a large number of experiments. We conclude the paper in section 5

1. RELATED WORK

1.1 Machine learning methods for wind power prediction with time series measurements

Wind power values recorded from of a turbine with a certain time interval is called the *time series measurements*. *Time series measurements* are the base of the most machine learning methods for the wind power prediction [9], which are represented as $M=\{m_i\}$. Generally, the problem of short-term wind power prediction is described to be: predicting the value of m_t with given $m_0 \sim m_{t-h+1}$, where h is the so-called *time horizon*, and t is the target.

The machine learning methods which are commonly used in this field contains k-nearest neighbors regression (k-NN), support vector machine regression (SVR), decision tree (DT) [10] and artificial neutral networks (ANNs) [11]. SVR and k-NN are considered to be stage-of-the-art methods. k-NN regression is one of the simplest regression algorithm, which predicts the numerical targets based on a similarity measurement. It has been widely used for its simplicity and high efficiency [12]. Vladimir N. Vapnik et al. proposed SVR in 1996 [13], which maps the input data to the higher dimensional space through the kernel function so as to build a linearly decision function to achieve the linear regression in the higher dimensional space. DT is a tree structure, in which each internal node of the tree represents a test on an attribute, and the outcomes of the test can generate branches. Each leaf of DT indicates a label, and the path from the root to the leaf shows the regression rules. ANNs are usually a graph, in which the nodes are divided into several layers and each layer only connects with the its neighbors. A typical ANNs consists of three layers, including the input layer, hidden layer and output layer. The prediction rules are represented in the weights of edges.

WindML, developed by Kramer, O et al., is a python based machine learning framework that focuses on the wind power prediction [9]. The framework can download the data of national renewable energy laboratory (NREL), which is collected during 2004∼2006 with a time interval of 10 minutes [14]. Based on this framework, the regular works in the wind power prediction, such as data acquisition and feature extraction, can be easily achieved. Thus, we have achieved our method based on this framework.

1.2 Temporal-spatial feature extraction

Features in machine learning methods for wind power prediction are extracted from the wind power series measurements. Fig. 3 (a) shows a common method of feature extraction. The successive measurement points are set as a feature, and the point after the interval is regarded as the corresponding output of the feature [11].

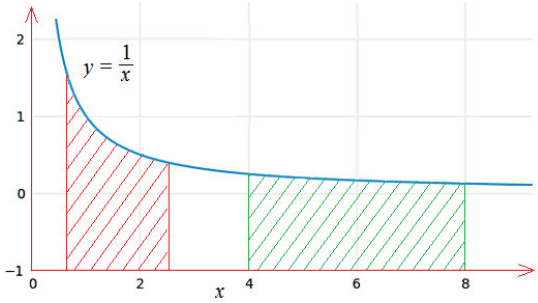

Figure 1 Graph of function $y = 1/x$.

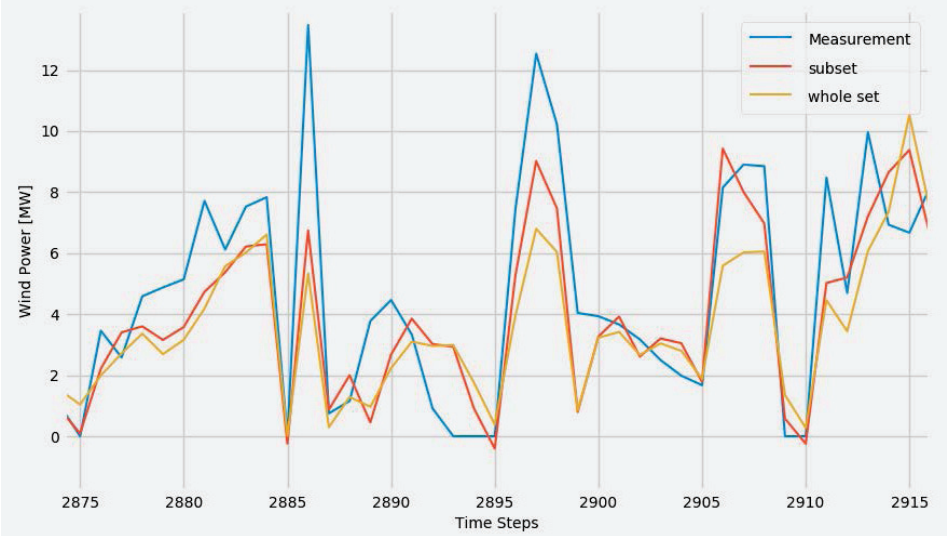

Figure 2 prediction results of the models trained by a certain subset and the whole set of training data respectively. SVR((kernel='rbf', epsilon=0.1, C=100.0, gamma=0.0001) is used as prediction. Time series of turbine #11637 in 2004 are used as training data, and that of 2005 are used as test data. The selected variance range is [2, 5].

Compared with the above method of feature extraction, the temporal-spatial feature extraction shown in Fig. 3 (b) takes the spatial information into account. To explain thoroughly, we set the target turbine as the center and gather the surrounding turbines together as a group. Then, we extract the feature of each turbine with the traditional method and then combine them together as a new feature. It should be noted that the feature contains the temporal and spatial information, but they are implied in the specific data. Thus, how to make use of the information depends on the specific prediction algorithm.

2. WIND POWER PREDICTION WITH MULTI-PREDICTORS

In this chapter, we plan to discuss the basic ideas and the main processes of the wind power prediction with multi-predictors (ML_HWP).

2.1 Framework

The main processes of the *ML_HWP* are shown as follows:

1. transform the time series measurements into training dataset and test dataset.

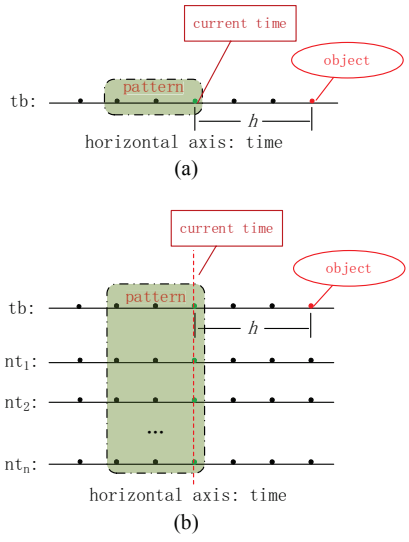

Figure 3 Transforming time series into features and corresponding responses. The word "tb" means the object turbine, and nt_i means the ith neighbor of the object. The red point both in (a) and (b) are outputs of the features.

2. Divide the training set into several groups according to the variance of input pattern from small to large, and then endow each example of both training set and test set the attribute, *type*. The value of *type* in the training set equals to

the serial number of group that contains it., and also equals to the virtual serial number of group that contains in in the test data if the test data is divided into groups with the same variance thresholds during the process of dividing the training sets.

3. Train predictors with each group of training data, and then calculate the *MSE* of the predictors on each group of the training set. The results form a matrix E, in which $E_{i,j}$ represents the *MSE* of the *j-th* training set on the *i-th* predictor, Actually, cross-validation is used in this step.

4. Predict with the multi-predictors and combine these predictors' results through the weighted average method to product the final results. The weights are calculated according to matrix E that is generated in the process 3.

SVR perfects well in most case, but the accuracy is hard to be further improved. So the hybrid method is a better choice. Others methods like k-NN are not as good as SVR, but when comparing algorithms with their best prediction times, which means the closest one to the real value among multi-predictors' results, as shown in table 1, there is not much difference. This phenomenon shows potential of other methods.

The framework of ML_HWP is shown in fig 4, which shows the main processes described above. Data is denoised before grouping for system robustness [15].

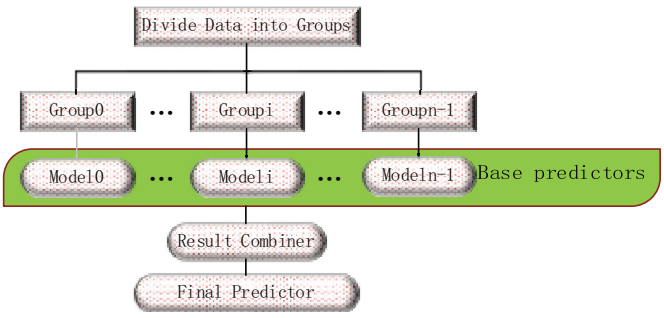

Figure 4 Framework of ML_HWP method. Data divider, Multi-predictors and result combiner are three components of the method.

Table 1 Best behaved times of for popular prediction algorithms on turbine #11637 of NREL dataset.

Turbine id	SVR	k-NN	ANNs	DT
11637	2033	3664	1890	3563
4155	2228	3115	2142	3187
3389	3631	2211	2098	2710
4115	1866	4461	1647	2575
30867	2184	3247	2234	2908

2.2 Model building

1. data preparation

   The first step of building our model is to extract features from the time series measurements. That is, to get the input pattern and the corresponding response. The way to achieve above function is inherited from work of Justin Heinermann et al. [11]. The training set after pretreatment is regarded as $\Re = \{(X_i, y_i)\}_{i=0}^{N-1}$, where X_i is the input pattern of the ith example and y_i is the corresponding desired response. N is the number of examples. Test set is represented as the similar form, $\Im = \{(X_i, y_i)\}_{i=0}^{M-1}$, where M is the number of examples in test set.

2. Calculate the variance v_i of each X_i, as (2).

$$v_i = \sum_{j=0}^{w-1} X_{i,j}^2 - \overline{X_i}^2 \tag{2}$$

   Where $\overline{X_i}$ is the average value of $X_{i,j}$. The accuracy of wind power prediction to a turbine is strongly related to the turbine's variances of its' input patterns. Generally, the smaller the average variance is, the lower *MSE* can be achieved.

3. Divide $\Re$ into several groups according to the variance of the input patterns, that is, $\Re$=$\{\Re_i\}_{i=0}^{n-1}$, where $\Re_i$ is a subset of $\Re$. There is a list of thresholds $H = \{H_i\}_{i=0}^{n}$, and each $\Re_i$ satisfies the rule in (3).

$$H_i < (\text{variance}(\pi.X) | \pi \in \Re_i) \leq H_{i+1} \tag{3}$$

   Note that $\Re$ is divided into n groups by $n+1$ thresholds, and we try to keep the members of each group being equal. Actually, each item of $\Re$ is given an attribute, *type = the serial number of the subset it belongs to* after the process of dividing. Attribute *type* is used in combining predictors' results when making prediction.

4. Training predictors with each $\Re_i$. In this step, each group of training set is served as an independent training set to train base models. base models are based on a single algorithm like SVR, k-NN and so on. In *ML_HWP.* each $\Re_i$ is used to train several predictors, but no two of them used the same base algorithm. The trained predictors are expressed as $P = \{P_i\}_{i=0}^{m-1}$, where m is the number of predictors.

5. Build a matrix $E^{m \times n}$, where $E_{i,j}$ means the MSE of P_i testing on $\Re_j$. In other words, use P_i as predictor and $\Re_j$ as test set to obtain the *MSE*. This is actually a Cross-validation method.

6. Build the weight matrix $W^{m \times n}$ based on matrix E, as shown in (4).

$$W_{i,j} = \frac{1.0}{(E_{i,j} - h \cdot \min(E_{,j}))^t} \tag{4}$$

   $E_{,j}$ means the *j-th* column of E, and h and t are two adjustable parameters, they are introduced to solving the problem described in chapter I. comparing with *RW* method, parameter h changes the Distribution interval of denominator, and t make the denominator into a non-linear function of *MSE*.

   Before transform E to W, an optional normalized operation on E can be adopted, as shown in (5).

$$E_{i,j} = \frac{E_{i,j}}{\max(E_{,j})} \tag{5}$$

7. For test data $\Im = \{(X_i, y_i)\}_{i=0}^{M-1}$, calculate the variance u_i of each X_i and then based on u_i to determine the value of attribute *type*, t_i,, which equals to k if and only if $H_k < u_i \leq H_{k+1}$.

8. Make prediction to X_i with each predictor. Suppose that the results from base predictors are $F_i = \{f_{i,j}\}_{j=0}^{m-1}$, then the final result is obtained as (6).

$$r_i = \frac{\sum_{j=0}^{m-1} \left(f_{i,j} \cdot W_{j,t_i}\right)}{\sum_{j=0}^{m-1} \left(W_{j,t_i}\right)} \tag{6}$$

2.3 Algorithm

The main processes of *ML_HWP* will be described through pseudocode in this chapter.

1. dividing the training set into groups.

   Firstly, calculate the variance v_i of X_i for each example in training set, and then, calculate the sorted index, *ids*, of $\Re$, that is, $v_{ids_i} < v_{ids_j}$ if $i < j$. when the number of groups n is given, the number of examples in each group is about $\ell = \|\Re\|/n$. We divide the *ids* into n continuous parts from left to right, and the variance of each split point will be viewed as thresholds. The first element of thresholds will be set to -1 and the last element will be set to an infinity number to conclude all the examples when thresholds are extended to test set. Code 3-1 shows the details of what we have described above.

code 3-1 data divider

```
Input: X, y, n
Return: Xs, ys, h
begin:
v = variance(X)
ids = argsort(v)
h = [−1]
members = len(X)/n, cur = 0
for i in (0, n):
if i < len(X)%n: next = cur+(members+1)
else: next = cur+members
Xs[i] = X[ids[cur∼next-1]], ys[i] = y[ids[cur∼next-1]]
cur = next
h.append(v[ids[cur-1]])
h[len(h)-1] = INF
return Xs, ys, h
end
```

   In code 3-1, *argsort()* is a built-in function in *Python* that can return *ids* that is described above.

2. training Multi-predictors

   One or more basic predictors can be built by using the data from each group. All basic predictors contain two functions, *fit* and *predict*, no matter what type of predictors they are. Function *fit* is used to train the model, while *predict* produces response to an input pattern.

code 3-2 training predictors

```
Input: Xs, ys
Return: predictors
begin:
predictors = []
for each (x, y) in (Xs, ys):
for each p in selected_base_predictors:
predictors.append(p.fit(x, y))
return predictors
end
```

   In code 3-2, *selected_base_predicors* means the base algorithms, including SVR. k-NN, DT and ANNs.

3. Cross-validation to obtain the matrix *E and W*

code 2-1 get the matrix E and W

```
Input: Xs, ys, predictors
Return: E, W
begin:
n=len(X), m=len(predictors)
E = zeros(m, n)
for i in range(0, m):
for j in range(0, n):
p = predictors[i]
E_{i,j} = MSE(p.predict(Xs[i]), ys[i])
W = zeros(m, n)
for i in range(0, m):
for j in range(0, n):
W_{i,j} = 1.0/(E_{i,j} − h.?max(E_{,j}))
end
```

3. EXPERIMENTS

3.1 Key parameters

The key adjustable parameters in *ML_HWP* method are the number of data groups, n, and the parameters h and t in (4).

A problem before determining the best value of n is how to divide the data, whether using an arithmetic progression as thresholds or trying to keep the members of each group being equal. To solve this problem, the distribution of variance of the patterns is studied. As shown in **fig** 5.

The variances of input patterns have great influence on the accuracy. A test on turbine #11637 shows that the average *MSE* of test examples with a variance which is smaller than 2 can achieve 2.980. It accounts for 43.4% of the all test examples, while the average *MSE* of the rest examples reaches 20.922. From fig 5, we can see that the variances of examples are distributed in a large range of 0∼300. However, most of them are with small values, about half of them are smaller than 2, and only about 30% are larger than 10. If use an arithmetic progression as thresholds for dividing data, the members of some groups are too small in the case, thus, setting an arithmetic progression as threshold is not reasonable.

To explore the relationship between any two parameters, we set the third parameters with a default value, and then visualize the relationship between two variables by a surface, which demonstrates the change of *MSE* with two selected parameters. The results are shown in figure 6.

As shown in figure 6, we can conclude that:

1. The correlation among variables is not strong.
2. If the value of t is less than 1, the effect is very poor. In a certain range, *MSE* reduces quickly with the increase of t. if the value of t is greater than 2, which is the optimal value, *MSE* begins to increase slowly. In fact, the optimal value of t is different for different turbines, as shown in Figure 7. For some turbines, the optimal value of t is about 2, but about 4 for most of them. Generally, value 4 is a good choice.
3. If the parameter n value is 3 or 4, the effect is better. According to our method, it is true that the data is grouped according to the variance of the input feature can improve the prediction accuracy. However, the model will be difficult to have convergence when the number of groups is too large.
4. the change law of the parameter h is relatively simple. Generally, *MSE* gradually reduces with the increase of parameter h. However, taking the meaning of the formula (4) into account, when the value of h tends to be 1, the value of the function will become infinite, which is theoretically unreasonable. In the experiment, when the value of h is close to 1, *MSE* grows rapidly. When the value of h is more than 1, *MSE* is no longer regular, as shown in Fig 8. After lots of experiments, we find that the optimal value is usually between 0.8 and 0.9.

3.2 Using multiple types of base predictors

We trained multiple types of predictors with each group of training set and tries different combinations. The experimental results are shown in Table 2. The first four lines of the Table represents the results of using a single base algorithm, while the rest of lines demonstrates the results of using varieties of base algorithms. The table only lists some of the combinations because of the limited space, while the statistics in the table are able to prove the effectiveness of our method.

As table 2 shows, we can conclude that:

1. Different base methods show big difference when they work alone. SVR and ANNs are superior to k-NN and DT.
2. Hybrid predictors with multiple basic algorithms perform better than base predictors;
3. Performance of hybrid predictor depends on its base predictors. It is clear that SVR and ANNs are two best performed base algorithms, and the hybrid predictor with these two base algorithms is also the best performed hybrid predictor.
4. The proposed method is almost ineffective to the k-NN algorithm. In fact, k-NN algorithm makes decisions according to the nearest neighbors of each example during the process of making prediction. In addition, the similar examples will probably appear in the same group in the process of grouping. Thus, the data grouping has little effect on the k-NN algorithm.

3.3 Compare with other methods.

We design experiments to compare our proposed methods and existing algorithms. The existing methods include four commonly used machine learning methods such as SVR, k-NN, DT, ANNs and RW. In general, SVR and k-NN are usually considered as the stage-of-the-art methods. The comparative results are shown in table 3.

Table 2 MSE of types of combination.

Turbine id	11637	4155	3389	4115	DT
[1]	11.978	7.160	10.693	7.885	5.083
[2]	13.909	8.002	11.285	8.376	5.709
[3]	11.768	7.158	10.424	7.756	5.102
[4]	15.240	8.557	12.374	8.934	6.092
[1,3]	11.492	6.977	10.270	7.752	5.041
[1,2,3]	11.932	7.285	10.576	7.876	5.133
[1,3,4]	12.012	7.247	10.813	8.053	5.154
[1,2,3,4]	13.872	7.973	11.266	8.352	5.679

In this table, 1=SVR, 2=k-NN, 3=Networks and 4=DT. SVR and Networks are used in each type of combination for their output performance. Parameters, n=5, h=0.9, t=1.7. Due to the characteristics of the algorithm itself, the results of DT and ANNs have randomness, so we used the average of 10 runs result when involving the above two algorithms.

Table 3 comparative results of types of methodds

Turbine id	11637	4155	3389	4115	DT
SVR	12.094	7.240	10.976	8.000	5.130
k-NN	13.797	7.972	11.264	8.405	5.705
DT	14.517	8.560	11.858	8.681	6.273
ANNs	12.028	7.423	10.834	8.062	5.326
RW	12.103	7.327	10.935	7.935	5.102
ML_HWP	11.492	6.977	10.270	7.752	5.041

4. CONCLUSIONS AND FUTURE WORKS

This paper investigates the spatio-temporal feature and focuses on the important role of its variance in wind power prediction. Based on this, a hybrid machine learning algorithm is designed. This paper studies the various variables involved in the algorithm and analyzes their influence on the prediction effect. Finally, we prove the effectiveness of the proposed method by comparing with the five existing machine learning algorithms.

In the future, we will further study and adjust the parameters in the method to achieve better predictive results.

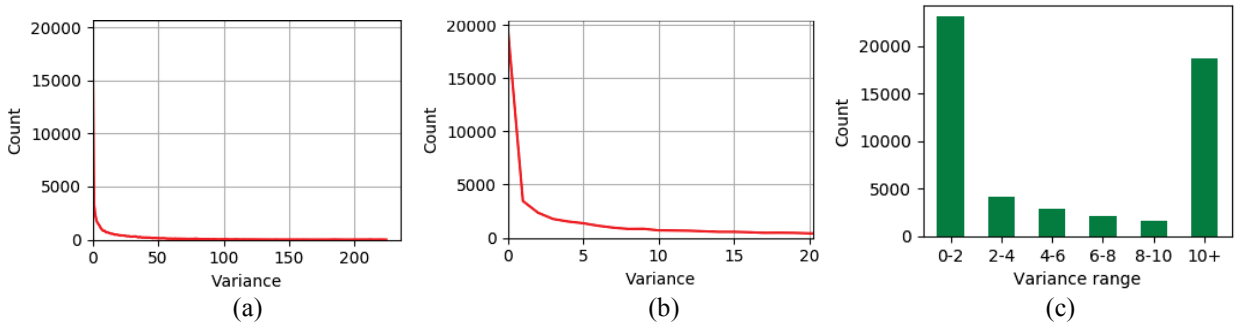

Figure 5 Variance distribution of turbine #11637's data. picture (a) shows the dataset's global distribution of variance, and figure (b) shows part of (a). figure shows the amount of data in each variance interval.

Figure 6 Parameters of *ML_HWP*. (a) shows the relationship of n and t; (b) shows the relationship of h and t; (c) shows the relationship of n and h. The experiments are adopted on turbine #11637. When training any two parameters, the third parameter is assigned with a default value: n=4, h=0.9 and t=2.0.

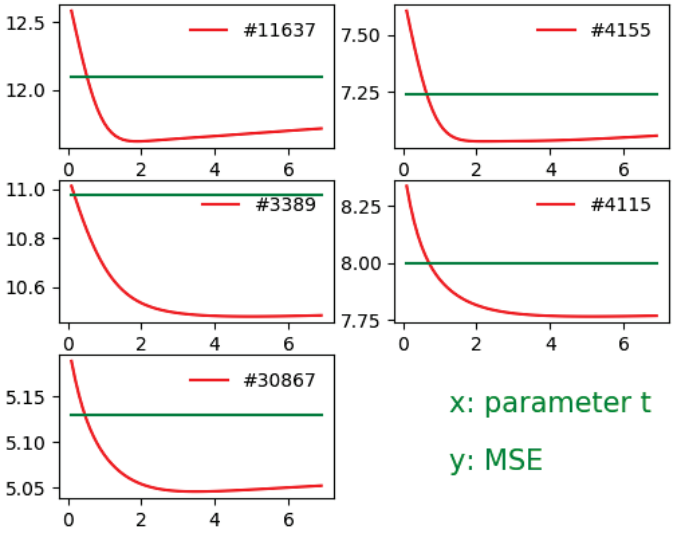

Figure 7 Experiment results of algorithm described in this chapter. The green line in each figure represents the MSE of a single SVR predictor, while the red line represents the change of MSE with parameters in formula 5-2.

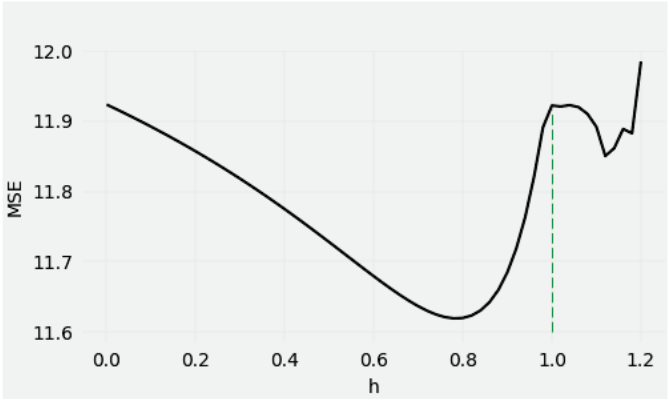

Figure 8 curve of MSE with parameter h. The experiment is carried out based on turbine # 11367. ANNs is the only base algorithm. Group number = 5, t = 1.7.

Acknowledgments

We thank Kramer, O. et al. for providing WindML framework.

REFERENCES

1. Da Z, Xiliang Z, Jiankun H, et al. Offshore wind energy development in China: Current status and future perspective[J]. Renewable and Sustainable Energy Reviews, 2011, 15(9): 4673–4684.
2. Foley A M, Leahy P G, Marvuglia A, et al. Current methods and

advances in forecasting of wind power generation[J]. Renewable Energy, 2012, 37(1): 1–8.
3. Colak I, Sagiroglu S, Yesilbudak M. Data mining and wind power prediction: A literature review[J]. Renewable Energy, 2012, 46: 241–247.
4. Kramer O, Gieseke F. Short-term wind energy forecasting using support vector regression[C]//Soft Computing Models in Industrial and Environmental Applications, 6th International Conference SOCO 2011. Springer, Berlin, Heidelberg, 2011: 271–280.
5. Poloczek J, Treiber N A, Kramer O. KNN regression as geo-imputation method for spatio-temporal wind data[C]//International Joint Conference SOCO'14-CISIS'14-ICEUTE'14. Springer, Cham, 2014: 185–193.
6. Treiber N A, Heinermann J, Kramer O. Wind Power Prediction with Machine Learning[M]//Computational Sustainability. Springer International Publishing, 2016: 13–29.
7. Giebel G, Badger J, Landberg L, et al. Wind power prediction using ensembles[R]. 2005.
8. Heinermann J, Kramer O. Machine learning ensembles for wind power prediction[J]. Renewable Energy, 2016, 89: 671–679.
9. Kramer O, Gieseke F, Heinermann J, et al. A framework for data mining in wind power time series[C]//International Workshop on Data Analytics for Renewable Energy Integration. Springer, Cham, 2014: 97–107.
10. Suresha M, Danti A, Narasimhamurthy S K. Decision trees to multiclass prediction for analysis of arecanut data[J]. Computer systems science and engineering, 2014, 29(1): 105–114.
11. Treiber N A, Heinermann J, Kramer O. Wind Power Prediction with Machine Learning[M]//Computational Sustainability. Springer International Publishing, 2016: 13–29.
12. Yoo S, Lee K Y, Kim M H. Fast k-NN search using pre-computed l-NN sets[J]. International Journal of Computer Systems Science & Engineering, 2011, 26(4): 231–240.
13. Drucker H, Burges C J C, Kaufman L, et al. Support vector regression machines[C]//Advances in neural information processing systems. 1997: 155–161.
14. Lew D, Milligan M, Jordan G, et al. How do wind and solar power affect grid operations: The western wind and solar integration study[C]//8th international Workshop on large scale integration of wind power and on transmission networks for Offshore wind farms. 2009: 14–15.
15. Lei J, Jiang T, Wu K, et al. Robust local outlier detection with statistical parameter for big data[J]. COMPUTER SYSTEMS SCIENCE AND ENGINEERING, 2015, 30(5): 411–419.