Intelligent Automation & Soft Computing

Tech Science Press

DOI: 10.32604/iasc.2023.034750
Article

Power Information System Database Cache Model Based on Deep Machine Learning

Manjiang Xing*

Kyonggi University, Suwon, 449-701, Korea
*Corresponding Author: Manjiang Xing. Email: di25712376@163.com
Received: 26 July 2022; Accepted: 13 December 2022

Abstract: At present, the database cache model of power information system has problems such as slow running speed and low database hit rate. To this end, this paper proposes a database cache model for power information systems based on deep machine learning. The caching model includes program caching, Structured Query Language (SQL) preprocessing, and core caching modules. Among them, the method to improve the efficiency of the statement is to adjust operations such as multi-table joins and replacement keywords in the SQL optimizer. Build predictive models using boosted regression trees in the core caching module. Generate a series of regression tree models using machine learning algorithms. Analyze the resource occupancy rate in the power information system to dynamically adjust the voting selection of the regression tree. At the same time, the voting threshold of the prediction model is dynamically adjusted. By analogy, the cache model is re-initialized. The experimental results show that the model has a good cache hit rate and cache efficiency, and can improve the data cache performance of the power information system. It has a high hit rate and short delay time, and always maintains a good hit rate even under different computer memory; at the same time, it only occupies less space and less CPU during actual operation, which is beneficial to power The information system operates efficiently and quickly.

Keywords: Deep machine learning; power information system; database; cache model

1 Introduction

With the advent of the era of big data, the informatization level of electric power enterprises has maintained an upward trend, and various business systems have emerged as the times require. Power information systems are facing huge system access pressure and massive data storage pressure [1,2]. Only a database with powerful data storage and read and write capabilities can improve the efficiency of power information systems [3]. Not only does the infrastructure need to remain the same, but the database needs to be accessed more efficiently. Data caching can reduce the application program's access to physical data sources, improve the efficiency of database access, and even the entire power information system.

The current analysis of database cache mainly focuses on data blocks or data sets [4–6]. Literature [7] proposed a database cache model based on multiple storage environments. This model uses data compression techniques to optimize caching. However, this method takes up a lot of memory and is not suitable for TB-level data caches. Literature [8] proposes an information database cache model based on the Rsdis tool. The model uses the Rsdis tool to build the cache framework and optimize the cache. However, the certainty of the results returned by this method is poor. If the query result set is large, memory blockage will occur, and the hit rate will drop rapidly.

Aiming at the above-mentioned problems of slow running speed and low database hit rate, this paper studies the database cache model of electric power information system based on deep machine learning. The cache model includes program cache, SQL preprocessing and core cache modules. Build predictive models using boosted regression trees in the core cache module. A series of regression tree models are generated using machine learning algorithms for analyzing resource occupancy in power information systems. According to the dynamic adjustment of the voting threshold of the prediction model, the re-initialization of the cache model is realized.

2 Database Caching Model of Power Information System Based on Deep Machine Learning

First, the database cache model structure of the overall power information system is constructed based on deep machine learning. The cache model includes program cache, SQL preprocessing and core cache modules. Build predictive models using boosted regression trees in the core cache module. Generate a series of regression tree models using machine learning algorithms. At the same time, according to the dynamic adjustment of the voting threshold of the prediction model, the re-initialization of the cache model is realized.

2.1 Overall Model Structure

The buffer module, buffer core module and SQL preprocessing module in power information system make up the buffer model of power information system database based on deep machine learning. The overall model structure is shown in Fig. 1.

(1) Program Cache Module

The program cache module is embedded in the relevant background programs of the power information system. This module is mainly composed of data sharing pool, concurrency control sub-module and individual identification sub-module. The idea of module editing is aspect-oriented, and it does not modify the original business system logic while monitoring the module persistence layer. Paste the time sequence number and individual identification in the corresponding query command submitted by each user. Appropriate concurrent access strategies are required to ensure a specific level of transaction isolation. Shared query result set uses data sharing pool [9–11].

(2) SQL Preprocessing Module

SQL preprocessing module is used when parsing and optimizing query statements. The way to improve the efficiency of statements is to adjust the operation of multi-table joins and keywords replacement in SQL optimizer. SQL parsing using the parsing engine module reference data dictionary reference information, ensure that suitable for intelligent forecasting module and business learning to complete processing work [12].

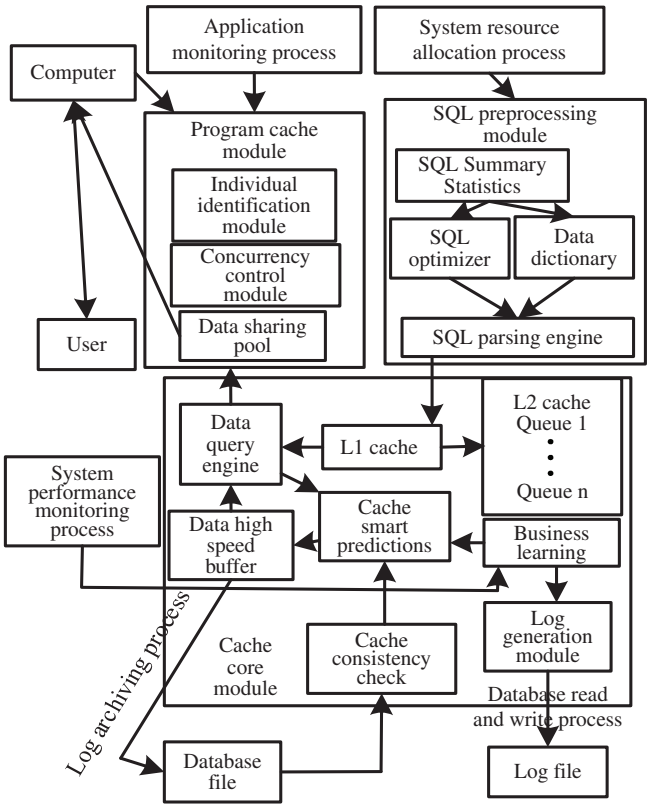

Figure 1: Overall model structure

(3) Cache Core Module

The SQL statements introduced in the first-level cache are called by the data query engine. When dividing different queues in the secondary cache, we need the individual identity document (ID) of the previous model. Based on the regression tree, the lifting regression tree is used to construct the original prediction model to calculate the residual of training set. Machine learning algorithm is used to learn the residual of training set. In the Business Learning Module, the Caching Intelligent Prediction Submodule searches the same operation node to cache possible queries [13]. The data cache reads the result set. The cache consistency checking submodule is responsible for synchronizing the database and cache using fixed algorithms.

(4) Business Process

In the process of system resource allocation, the cache core module is allocated to several terminals for simultaneous execution using Hadoop technology. The system performance monitoring process is mainly responsible for the dynamic adjustment of the relevant indicators. Power information system database cache principle flowchart is shown in Fig. 2.

After the implementation of the database to determine whether the database language belongs to the DML language. If the result is negative, the database is operated directly. If the result is affirmative, the next step will be taken. Is the target data in the cache or, if not, in the database and return the result. If it belongs to the buffer area, the operation is performed in the buffer area and the result is returned. After completing the above operations and after the results are returned, use intelligence to predict the

next action. The target data item is transmitted to the cache to determine whether the cache is full or not. If not, end the cache. If full, end the cache after replacing the data with the LRU algorithm.

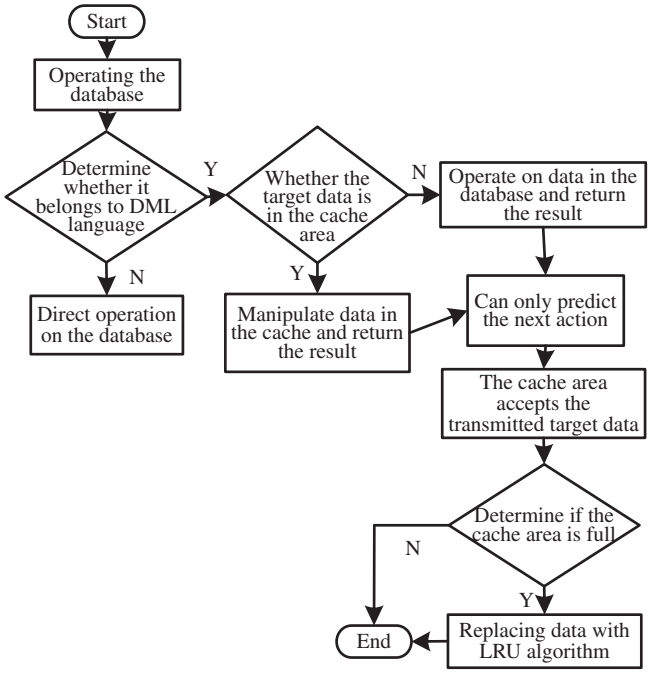

Figure 2: Database cache principle flowchart

2.2 Database Cache Model Based on Boosting Regression Tree

Regression tree is a tree-structured algorithm that describes the basis and belongs to a nonparametric model. A set of training data specified by the electric power information system is continuously divided into a subset without repetition by using a divide and conquer, top-down learning strategy. After the division is completed, the output is set as the mean value of the sample responses in each subset to construct a prediction model. The mathematical model of the regression tree is as follows:

$$f(x) = \sum_{m=1}^{M} z_m I(x \in K_m) \tag{1}$$

In Eq. (1), M and K_m respectively represent the number of subsets in the regression tree model and the divided subsets. z_m and $I(x \in K_m)$ represent the corresponding mean and indicative function of the data samples in each subset, respectively. The value is 1 if $x \in K_m$, and 0 if $x \notin K_m$.

Identify a training dataset in a power information system. The training data set N contains samples (x_i, y_i), $i = 1, 2, \cdots, N$. Each sample contains 1 output y_i and p inputs x_i. When building a regression tree model, it is necessary to find a split point s and a variable $x^{(j)}$ in all input variables of the database cache model to ensure that the set can be divided into two subsets. At the same time with minimal error. Eq. (2) is the process of exploring split points and split variables:

$$\min_{j,s}\left[\min_{c_1}\sum_{x_i \in K_1(j,s)}(y_i - z_1)^2 + \min_{c_2}\sum_{x_i \in K_2(j,s)}(y_i - z_2)^2\right] \tag{2}$$

$K_1(j,s)$ and $K_2(j,s)$ respectively represent the data samples in subset 1 and subset 2 when the set is divided. z_1 and z_2 represent the mean responses of data samples in subset 1 and subset 2, respectively. Repeat the above process until the termination conditions are met, and finally complete the regression tree model. Each subset contains less than a fixed number of data samples as a termination condition.

Because the single recursion tree model of power information system has limited prediction accuracy. Boosting algorithm is proposed to improve the prediction accuracy of regression tree model. The addition model is used to combine the final model and the forward distribution algorithm is used to fit a series of regression trees [14]. That is to say, each regression tree model needs a previous model as its basis. Eq. (3) is the boosted regression tree model:

$$f_T(x) = \sum_{t=0}^{G} h(x; a_t) \tag{3}$$

T, $h(x; a_t)$, and a_t represent the number of regression trees constructed, a single regression tree model, and the parameters of each regression tree model, respectively.

The prediction model of regression tree was constructed based on regression tree and ascending regression tree. The machine learning algorithm generates a series of regression tree models and combines the output averagely.

There is an independent relationship among the regression tree models obtained by machine learning algorithm. Use Eq. (4) to express regression tree prediction model:

$$f_{RF}(x) = \frac{1}{N} \sum_{i=0}^{G} h(x; a_i) \tag{4}$$

The number of regression tree models is represented by N.

The stochastic attributes are introduced into the stochastic sampling feature space and sample space when constructing the belonging models. Therefore, the relativity between recursive tree models is reduced, and the generalization of the model is reduced, which improves the robustness, efficiency and accuracy of the model.

The most important model parameter when building a model is the number of regression tree models N. As the number of regression trees increases, the generalization error of machine learning converges to the limit and reduces the fitting phenomenon. Increasing the number of regression tree models can suppress a large amount of noise that may exist in the data and ensure that the model has good performance. Usually the number of regression trees ranges from hundreds to thousands.

Based on the regression tree prediction model, through the performance detector, the regression tree prediction model can obtain the operation status of power information system and analyze the resource occupancy rate. Dynamic adjustment of voting options for regression trees [15].

Assuming you have hit the SQL statement, set it G_n to represent the node. The regression tree is pruned according to I/O time, memory usage and CPU utilization, and the child nodes obtained after pruning are represented as follows:

$$\begin{cases} G^{I/O} = \left\{g_1^{I/O}, g_2^{I/O}, \cdots, g_n^{I/O}\right\} \\ G^{M} = \left\{g_1^{M}, g_2^{M}, \cdots, g_n^{M}\right\} \\ G^{CPU} = \left\{g_1^{CPU}, g_2^{CPU}, \cdots, g_N^{CPU}\right\} \end{cases} \tag{5}$$

If the power information system can maintain normal operation and all resources are available, then the cache set predicted by regression tree prediction model is represented as $G^{CPU} \cup G^{I/O} \cup G^{M}$. If the

system central processing unit (CPU) usage exceeds the threshold, the cache collection is represented as $G^{I/O} \cup G^{M}$. According to the above method, the analogy is continued until no more idle resources can be applied. At this time, the least recently used (LRU) table in the cache area is emptied, and the cache model is reinitialized.

3 Experimental Analysis and Results

A certain electric power company is taken as the research object and its information system is called after negotiation. The company is located in a city in central China, mainly responsible for more than half of the city's commercial areas, residential areas of power supply and power consumption management. The actual work includes charging, managing the regional power supply system, power supply equipment maintenance and some regional line laying. Since its inception in 2009, the annual increase in turnover has become an important economic support enterprises in the region. The data required for the experiment is the data with user identification automatically recorded in the background of the company's power information system. About 8,000 access records were selected for 2 days.

The model is applied to the system to verify the performance of the model. The three comparison models are the database cache model based on multiple storage environments, the information database cache model based on rsdis tools and a novel adaptive database cache optimization algorithm based on predictive working sets in cloud environment. Two methods of comparison were derived from references [7–9].

The main metrics to analyze cache performance include latency, metric hit ratio, and byte hit ratio. The latency time represents the total amount of time the server downloaded the object until the client implemented the cache. Metric hit ratio represents the number of objects the user gets from the cache and the total number of objects obtained. The byte hit ratio represents the number of objects obtained from the buffered user and the total number of objects obtained.

Comparison of three models in the case of different cache file size, the results are shown in Table 1.

As you can see from Table 1, as the cache file size increases, the hit ratio of each model tends to increase. But the upward trend of the two comparison models was slower, when the cache file reached 3000KB. The hit ratios of various storage environment models , rsdis tool models and cloud environment models reached 34.16% ,49.23% and 43.98% respectively, while the model in this paper reached 80.14%. It can be seen that using this model can greatly reduce the redundancy of the database cache files, improve the hit rate and improve the overall performance of the power information data caching system.

Table 1: Metric hit ratio comparison (%)

Cache file size/KB	Multiple storage environment models	Rsdis tool model	Cloud environment model	Model of this paper
200	5.41	7.38	6.23	10.48
400	7.06	10.95	7.58	15.62
600	9.95	13.66	9.92	20.39
800	12.38	16.82	13.83	25.41

(Continued)

Table 1: Continued

Cache file size/KB	Multiple storage environment models	Rsdis tool model	Cloud environment model	Model of this paper
1000	14.27	19.34	15.36	30.68
1200	16.85	22.61	18.03	35.62
1400	18.64	25.74	20.74	40.19
1600	20.65	28.31	25.93	45.28
1800	22.37	31.05	28.50	50.33
2000	24.62	34.69	30.96	55.46
2200	26.85	37.09	32.75	60.84
2400	28.34	40.51	35.06	65.49
2600	30.16	43.59	38.54	70.31
2800	32.57	46.71	40.72	75.92
3000	34.16	49.23	43.98	80.14

The byte hit ratios of the three models are compared and the results are shown in Table 2.

As you can see from Table 2, similar to metric hit ratio, all four models increase byte hit ratio as the cache file size increases. This model has a higher byte hit ratio than the comparison model. So it can be proved that this model has better effect of eliminating redundancy.

Table 2: Byte hit ratio comparison (%)

Cache file size/KB	Multiple storage environment models	Rsdis tool model	Cloud environment model	Model of this paper
50	6.38	9.32	10.06	11.68
100	8.54	12.45	12.84	16.34
200	10.95	15.85	17.28	28.78
400	12.54	18.39	20.86	33.37
800	14.62	21.25	25.47	42.95
1600	16.82	24.94	28.21	49.85
3200	18.64	27.62	31.58	57.38
6400	20.61	30.19	33.43	61.67
12800	22.95	33.54	35.10	77.86
25600	24.28	36.98	39.66	86.95

Comparing the latency times of the three models with different cache file sizes, the results are shown in Fig. 3.

As you can see from Fig. 3, the latency of each model decreases as the cache file grows. In contrast, this model has the shortest latency, which shows that this model has a faster cache efficiency in caching database files.

The byte hit ratio and index hit ratio of each model are compared under different computer memory sizes. The experimental results are shown in Table 3.

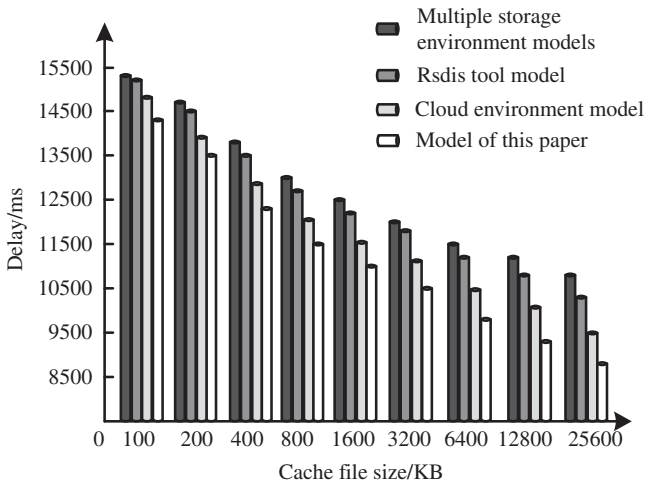

Figure 3: Latency comparison results

Table 3: Comparison of different memory hit ratios (%)

Hit rate	Computer memory size/GB	Multiple storage environment models	Rsdis tool model	Cloud environment model	Model of this paper
Byte hit rate/%	2	20.64	62.58	60.23	98.24
	4	25.71	65.16	64.84	98.28
	6	30.85	68.44	69.63	98.36
	8	35.48	71.24	73.19	98.38
	8	40.06	74.62	76.92	98.47
	10	45.84	77.85	79.14	98.49
	12	50.13	80.61	82.83	98.54
Indicator hit rate/%	2	28.64	63.59	84.99	98.25
	4	33.47	66.16	86.17	98.35
	6	38.95	69.85	87.39	98.51
	8	43.88	72.15	88.94	98.54
	8	48.62	75.62	89.47	98.61
	10	53.55	78.24	90.31	98.72
	12	58.92	81.68	92.63	98.79

Analysis of Table 3 shows that with the increase of computer memory, the hit rate of the three comparison models gradually increases. The computer memory size directly affects the model cache hit rate. Only computers with large enough memory can achieve a high cache hit rate; The byte hit rate and index hit rate of the method in this paper are both high. At the same time, the stability of the hit rate of the model in this paper is relatively high, and the size of the computer memory will not have a serious impact on the cache hit rate. It can ensure the smooth operation of the data cache of the power information system.

Compare the memory usage and CPU usage of the three models with different cache file sizes. The experimental results are shown in Table 4.

Table 4: Comparison results of memory and CPU utilization

Occupancy classification	Cache file size/KB	Multiple storage environment models	Rsdis tool model	Cloud environment model	Model of this paper
Memory usage/%	500	63.14	56.34	55.82	23.64
	1000	69.32	61.39	53.97	23.79
	2000	72.65	65.27	50.18	24.52
	4000	76.82	69.33	48.38	24.66
	8000	81.35	70.25	47.12	24.75
	16000	86.52	71.36	45.96	24.86
	32000	89.34	75.28	44.82	24.93
	64000	90.24	79.66	42.19	25.15
CPU usage/%	500	52.49	61.25	40.86	12.35
	1000	61.25	65.39	39.62	12.51
	2000	64.95	68.32	38.12	12.59
	4000	71.25	71.05	36.08	12.61
	8000	75.68	72.66	35.76	12.75
	16000	79.31	75.91	31.92	12.82
	32000	80.16	78.29	30.52	12.95
	64000	82.64	82.06	29.11	13.06

From the analysis of Table 4, it can be seen that compared with the three comparison models, the memory occupancy rate and CPU occupancy rate of the model in this paper are lower. It shows that the use of the model in this paper will not affect the operation state of the power information system, and even improve the operation effect of the system. It has a positive impact on the system and has broad application prospects.

4 Conclusion

At present, the power information system often has a huge amount of data, and the concurrent access volume is high. A database caching model must be explored. It can greatly reduce the cache occupancy rate and also has a high de-redundancy and hit rate. Aiming at the above problems, this paper proposes a database caching model for power information systems based on deep learning. This paper uses the regression tree as the basis for the cache model of the power information system, and learns the residuals on the training set through machine learning. The actual use of the power information system as the research object to carry out practical verification, the method in this paper has a high hit rate and a short delay time. This method can always maintain a good hit rate even under different computer memory. At the same time, the method in this paper only takes up less space occupancy and less CPU occupancy during actual operation. It is conducive to the efficient and fast operation of the power information system.

Funding Statement: The author received no specific funding for this study.

Conflicts of Interest: The authors declare they have no conflicts of interest to report regarding the present study.

References

[1] N. C. Zhou, J. Q. Liao, Q. G. Wang, C. Y. Li and J. Li, "Analysis and prospect of deep learning application in smart grid," *Automation of Electric Power Systems*, vol. 43, no. 4, pp. 180–191, 2019.

[2] X. Gao, G. F. Chen and H. L. Zhao, "Network security situation assessment of power information system based on data mining," *Electrical Measurement & Instrumentation*, vol. 56, no. 19, pp. 102–106, 2019.

[3] T. N. Ma, C. Wang, L. L. Peng, X. F. Guo and M. Ming, "Short-term load forecasting of power system considering demand response and multi-task learning based on deep structure," *Electrical Measurement & Instrumentation*, vol. 56, no. 16, pp. 50–60, 2019.

[4] Z. C. Qu, L. Gao, B. L. Kang and G. Y. Shi, "A power system fault full information diagnosis model based on multi-source data," *Power System Protection and Control*, vol. 47, no. 22, pp. 59–66, 2019.

[5] L. Xi, L. Yu, X. Zhang and W. Hu, "Automatic generation control of ubiquitous power Internet of Things integrated energy system based on deep reinforcement learning," *Scientia Sinica Technologica*, vol. 50, no. 2, pp. 221–234, 2020.

[6] L. P. Chen, L. F. Yin, T. Yu and K. Y. Wang, "Short-term power load forecasting based on deep forest algorithm," *Electric Power Construction*, vol. 39, no. 11, pp. 42–50, 2018.

[7] J. C. Zhang, X. G. Liu and G. Wang, "Cache optimization for compressed databases in various storage environments," *Journal of Computer Applications*, vol. 38, no. 5, pp. 1404–1409, 2018.

[8] Z. Y. Sun, "Optimizing method of large-scale electronic information cache in compressed database," *Electronic Design Engineering*, vol. 28, no. 7, pp. 95–98+103, 2020.

[9] A. O. Thakare and P. S. Deshpande, "A novel adaptive database cache optimization algorithm based on predictive working sets in cloud environment," *IEEE Access*, vol. 7, no. 7, pp. 54343–54359, 2019.

[10] L. T. Tan, J. L. Li, B. Ren, Y. He, X. Gao *et al.,* "Health evaluation model of smart grid dispatch and control system based on RB-XG Boost algorithm," *Electric Power Automation Equipment*, vol. 40, no. 2, pp. 189–198, 2020.

[11] Y. Deng, M. F. Peng and J. W. Liu, "Power cyber-physical system modeling and information attack mechanism analysis," *Acta Power System and Its Automation*, vol. 33, no. 10, pp. 10–17, 2021.

[12] Z. Ruan, L. Lv, Y. B. Liu, J. Liu, D. G. Wang *et al.,* "Coordinated attack model of cyber-physical power system considering false load data injection," *Electric Power Automation Equipment*, vol. 39, no. 2, pp. 181–187, 2019.

[13] C. B. Sun, M. Cheng, K. Yuan, J. Sun, Y. Song *et al.,* "Real time simulation platform of power cyber-physical system based on node mapping model," *Power System Technology*, vol. 43, no. 7, pp. 2368–2377, 2019.

[14] S. Z. Chen, J. L. Liang, D. Y. Lu, Z. X. Zeng and X. R. Deng, "Multi-source fault data comprehensive analysis method for power system based on COMTRADE model," *Journal of Electric Power Science and Technology*, vol. 126, no. 3, pp. 94–102, 2019.

[15] X. L. Feng, H. Wei, H. B. Wei and Y. Zhang, "Distributed real-time database for dispatching automation system with micro-services," *Power System Protection and Control*, vol. 46, no. 21, pp. 138–144, 2018.