2023 IEEE International Conference on Data Mining Workshops (ICDMW) | 979-8-3503-8164-1/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICDMW60847.2023.00077

Automatic Component Identification Based on Time Series Classification for Intelligent Devices

Mingsen Du*, Yanxuan Wei*, Yupeng Hu†✉, Xiangwei Zheng*, Cun Ji*✉
*School of Information Science and Engineering, Shandong Normal University, Jinan, China
jicun@sdnu.edu.cn
†School of Software, Shandong University, Jinan, China
huyupeng@sdu.edu.cn

***Abstract*—Advances in manufacturing technology have made it possible to replace some components anywhere to meet the needs of different functions with multi-function devices. However, remote servers are typically not informed when workers replace device components. In these scenarios, there is a mismatch in the data models between the device components and the servers. To this end, this study proposes an automatic component identification method based on time series classification for intelligent devices. First, the component identification problem is transformed into a time series classification problem. Then, some classifiers are trained based on the time series data generated by the models. In the next step, the classifiers match the collected data with the component models. Finally, the components are identified by a fusion decision, respectively. The experimental results show that the proposed method has a high component identification accuracy.**

***Index Terms*—Internet of Things, Automatic component identification, Model matching, Time series classification, Deep learning**

I. Introduction

As the Internet of Things (IoT) [1]–[3] develops, sensors are becoming widely used in intelligent devices. So these devices are capable of collecting data and transmitting it to the server in the cloud, providing a convenient and efficient way to store and manage the data. This provides great convenience for monitoring [4], [5], anomaly detection [6], [7] or fault diagnosis [8], [9] the intelligent devices.

In general, the monitoring system is a powerful tool for intelligent device management. It is capable of assessing the performance of devices based on their data models, and can detect any potential issues or faults that may arise. It can also diagnose these problems and provide solutions that can help to ensure that the devices continue to run efficiently. The monitoring system is an invaluable asset for any organization that relies on intelligent devices, as it can help to ensure that they remain in peak condition.

With the development of manufacturing technology [10], [11], one intelligent device can integrate multiple functions. In this case, multi-function devices can replace some of their components to achieve a specified function. For example, the same excavator can dig in the ground or drill on hard pavement by simply changing the mechanical arms [12]. However, remote servers are typically not informed when workers replace device components. This will lead to a mismatch in the data models between the device components and the servers.

To address this situation, some manufacturers have attempted to add identification information to device components [13]. Another researchers identified components using additional hardware [14], [15]. However, these approaches faces the following challenges: 1) **New components required**. Most of the current components do not have identification information on them. To add identification information, new components had to be developed on top of the original ones. 2) **Significant updating of the monitoring system**. To accept the identification information, the monitoring system in the cloud needs to be significantly updated. Only then can the monitoring system build a new device data model based on the component identification information. 3) **Narrow range of customisation**. Components that have been sold cannot be modified. Adding identification information will not work for them. These methods is only suitable for new production intelligent device.

To address these challenges, it is desirable to automatically identify the components of a device based on its data. In other words, we should automatically identify device components based on the collected data [12], [16], [17]. The device usually collects observations periodically [18]. The collected data are naturally time series [19]. Therefore, this study proposes an automatic component identification method based on time series classification for intelligent devices. First, the component identification problem is transformed into a time series classification problem. Then, some classifiers are trained based on the time series data generated by the models. In the next step, the classifiers match the collected data with the component models, respectively. Finally, the components are identified by a fusion decision.

The main contributions of this paper can be summarized as follows:

- We have described the scenario of automatic component identification for intelligent devices. In this scenario, multi-function devices can replace some of their components to achieve a specified function. And we should automatically identify device components based on data.
- We transformed the component identification problem into a time series classification problem. In this way, we

✉Corresponding author: Yupeng Hu, Cun Ji.

2375-9259/23/$31.00 ©2023 IEEE
DOI 10.1109/ICDMW60847.2023.00077

can identify device components by predicting the label of the collected time series of the component.
- A fusion method for automatic component identification based on time series classification is proposed.
- Experiments are conducted to show the effects of the proposed method. The experimental results show that the proposed method has a high component identification accuracy.

The rest of this paper is structured as follows. Section II discusses some related work. In Section III we describe the motivation of the problem. The proposed method is presented in Section IV. Experimental results are presented in Section V. And our conclusions are given in Section VI.

II. Related Work

A. Automatic Component Identification

The proliferation of the Internet of Things [20], [21] and advances in manufacturing technology [22], [23] have enabled the development of devices that can easily replace a number of components. In this scenario, automatic component identification is helpful for remote control, detection or diagnosis of these devices.

A simple solution for automatic component identification is for components to actively upload their identification information. For example, Motamedi and Hammad [13] managed the components through their radio frequency identification tags. However, adding identification information to components may require updating the operating system of the intelligent device. In addition, the monitoring system in the cloud needs to be significantly updated to accept the identification information.

Some researchers identified components using additional hardware. Gao et al. [14] recognized electrical components using some cameras. Sánchez et al. [15] identified devices based on behavioural fingerprinting of the embedded components. However, the additional hardware is limited in adaptation scenarios.

Recently, more and more researchers have focused on identifying components of devices based on the collected data [12]. Giuliani et al. [16] proposed some hybrid techniques based on some field data for automatic model matching. Marchal et al. [17] identified the types of components by analyzing their periodic communication.

Due to the convenience of data analysis, this paper focuses on component identification based on the collected data.

B. Time Series Classification Methods

In general, the components usually collect observations periodically [18]. In other words, the collected data of the components is in the form of time series [19]. Therefore, we transform the component identification problem into the time series classification problem (see section III).

Due to the widespread existence of time series classification, it has attracted the attention of a large number of researchers [24], [25], and hundreds of time series methods have been proposed in recent decades [26]–[28]. These methods can be mainly divided into three categories [29]: 1) Distance-based methods focus on the distance measures between time series [26], [30]. They predict the label of a new time series as the nearest time series. 2) Feature-based methods classify time series based on features, such as: statistical features [31], [32], structural features [33], [34], the 22 canonical features (catch22) [26], [30], temporal features [35], [36] and word frequency features [37]. 3) Model-based methods assume that a model produces time series in the same class [38]. Models used in time series classification include autoregressive models, Markov models and deep learning models.

Due to its excellent performance, more and more researchers have been classifying time series based on deep learning models [39]. Zhao et al. [40] proposed a novel convolutional neural network (CNN) to generate features for time series classification. Wang et al. [41] applied the deep MultiLayer Perceptrons (MLP), Fully Convolutional Networks (FCN) and Residual Networks (ResNet) to time series classification. They suggested that FCN can be used as a simple but powerful baseline for time series classification. Dempster et al. [42] classified time series using RandOm Convolutional KErnel Transformation (ROCKET). On this basis, Dempster et al. [43] used a small fixed set of kernels to replace the random kernels in order to improve the training speed of ROCKET. Tan et al. [44] added multiple pooling to ROCKET.

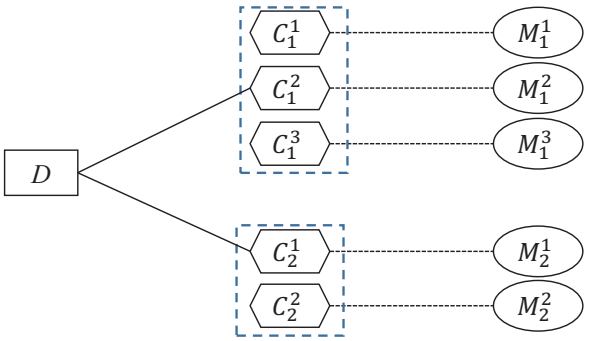


Fig. 1: Relations among device, components, and models. In this figure, D represents one intelligent device, C represents one component, and M represents the data model of the corresponding component.

III. Problem Motivation

A. Problem Scenario

multi-function devices can replace some of their components to achieve a specified function. This paper focuses on the following scenario: some components of the devices can be changed by the users. A demo of this scenario is shown in Fig. 1. In Fig. 1, D represents a smart device, C represents a component, and M represents the data model of the corresponding component. In this scenario, users can swap components within the same dashed box.

To study the problem more conveniently, we make the following assumptions for this scenario:

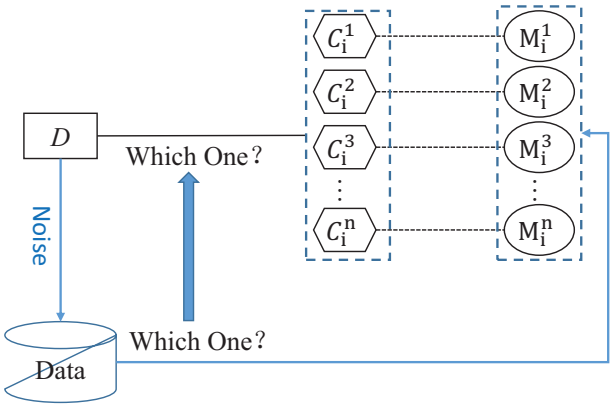


Fig. 2: Identify the current components through matching data to one model.

- *Assumption 1: Only one parameter is associated with a device component.* Based on this assumption, a multi-parameter component can be spit out as several single-parameter components.
- *Assumption 2: Each component corresponds to only one data model.* If components have multiple work modes, each work mode is considered a component. Switching work modes is also considered to be a replacement of components.
- *Assumption 3: The parameters between different components are independent.* In the future we will study components with associated parameters.
- *Assumption 4: The component periodically generates observations of parameters, and the device uploads the data in a period to the server in batches.* Based on this assumption, each period's data is in time series format and can be represented as an Eq. (1). In Eq. (1), $V_1, V_2, V_3, \cdots$ are observations in chronological order.

$$\{V_1, V_2, V_3, \cdots\} \tag{1}$$

B. Problems

Under this scenario, this paper is committed to solving the following problems:

Problem 1: Identify the current components based on time series classification.

As described earlier, workers typically do not inform remote servers when they replace device components. The remote server must identify the current component. As shown in Fig. 2, we can identify the current by matching the component data with models.

As in Assumption 4, the component data is in time series format. So, **time series classification can be used to match the component data with models**. In other words, we need to predict time series to a label of component models.

Problem 2: Get the conversion relationships between components.

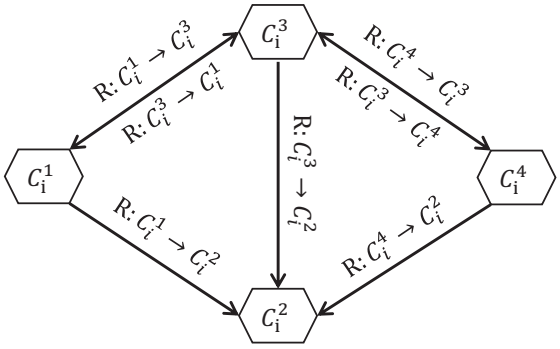


Fig. 3: Conversion relationships among components.

There may be some conversion relationships between components. As Fig. 3 shows, Component C_i^1 can be replaced by Component C_i^2 or Component C_i^3. In this case we want to get the conversion relations between the components in Fig. 3. In Fig. 3, the arrow represents the direction of transformation. For a one-way arrow, only the component to which the arrow points can replace the component at the other end. For two-way arrows, the two components can replace each other. The Component C_i^m replaced by the Component C_i^n can be expressed by the equation (2).

$$R : C_i^m \rightarrow C_i^n \tag{2}$$

To determine the conversion relationship, the front and rear components must be correctly identified. In fact, the component must be correctly identified twice. So **the key to solving the second problem is still time series classification**.

IV. The Proposed Method

To solve the mismatch in data models between device components and servers, this work proposes a fusion method based on time series classification to automatically identify components. As shown in Fig. 4, there are four steps in the proposed method:

1) **Data generating**. This step generates several data items for each model.
2) **Classifier training**. This step trains some classifiers based on the generated data items.
3) **Classification**. This step matches the collected data to the corresponding model by each classifier trained in Step 2.
4) **Fusion decision**. This step fuses the classification results to obtain the final matched component.

As shown in Fig. 4, these four steps can be divided into two stages: 1) the training stage, including data generating and classifier training, and 2) the identification stage, including classification and fusion decision.

In the following subsection, we will describe each step of the proposed method in more detail.

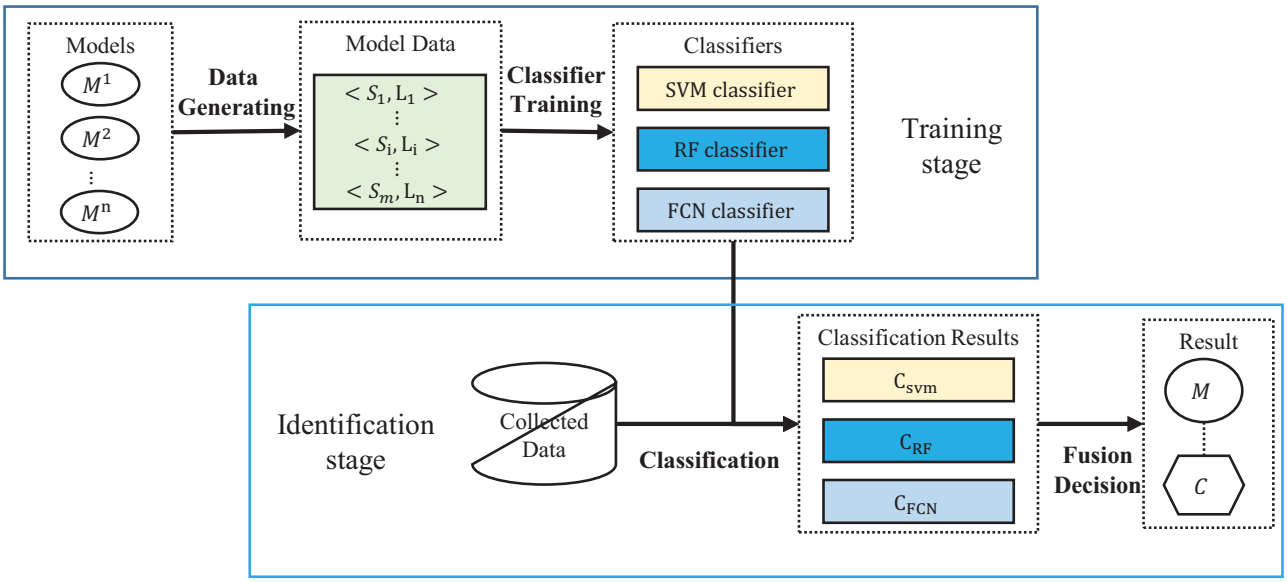


Fig. 4: The framework of proposed method.

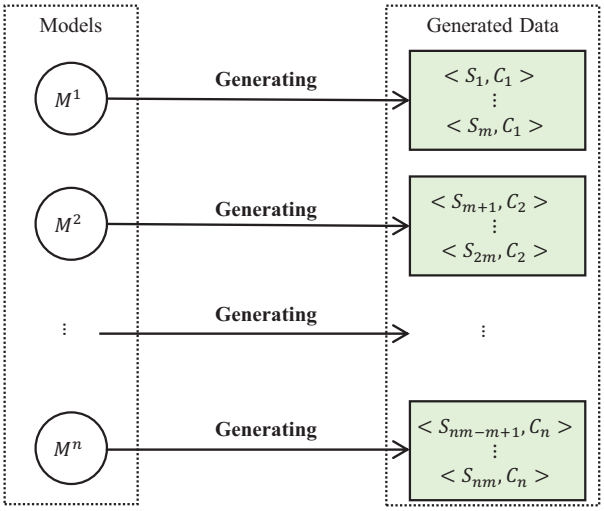


Fig. 5: Data generating

A. Data Generating

First, each model is used to generate some data item. Fig. 5 describes the data generating process. As shown in the left part of Fig. 5, there are n models in this scenario. For each model, m data items are generated in this step. The generated data items are shown in the right part of Fig. 5. Each data item can be expressed as $< S_i, C_i >$, where C_i is the model label and S_i is a time series generated by the corresponding model. The format of S_i is shown in Eq. (1).

TABLE I: Optional parameter values of the SVM classifier

Parameters	Optional values
Kernel type	'linear', 'rbf', 'poly', 'sigmoid'
Regularization parameter	0.1, 1, 10, 100
Kernel coefficient	0.001, 0.01, 1, 10, 100

B. Classifier Training

The process of classifier training is shown in Fig. 6. As depicted in Fig. 6, the generated data is divided into two parts: training data and validation data. This step adopts the strategy of 5-fold cross-validation. In other words, the generated data is split into five equal parts: four parts are used as training data, and the remaining part is used as validation data. The spit data are then used to train three classifiers: a Support Vector Machine (SVM) classifier [45], a Random Forest (RF) classifier [46], and an FCN classifier [41].

1) SVM classifier: The SVM classifier divides the observed data into different categories by maximizing the interval in the feature space [47]. There are two steps in the proposed method to train the SVM classifier: (1) SVM build. In this step, the parameters of the SVM are initialized. (2) SVM parameter selection. This step exhaustively searches the specified parameter values from the Table I. The SVM classifier with the optimal parameters is obtained through these two steps. As shown in Fig. 6, the expected accuracy acc_{SVM} of the SVM classifier is obtained at the same time.

2) RF classifier: The RF classifier is an ensemble classifier based on some decision trees on different subsamples [48]. Similar to the SVM classifier, there are two steps to train the RF classifier: (1) RF build. This step initializes the parameters of RF. (2) RF parameter selection. This step exhaustively

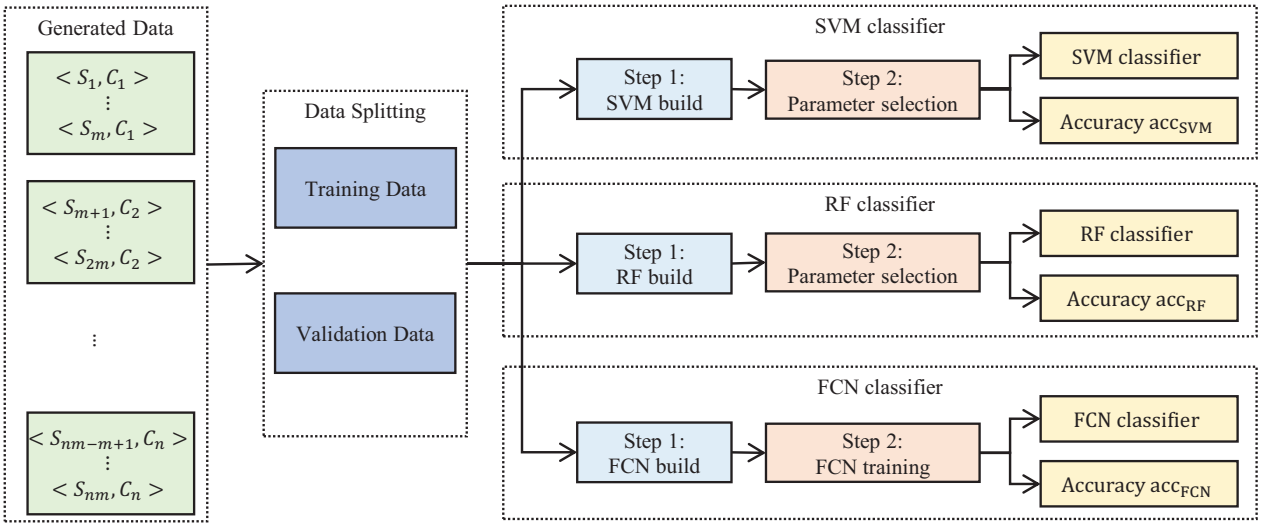


Fig. 6: Processes of classifier training.

searches the specified parameter values from Table II. This yields the RF classifier with the optimal parameters. The expected accuracy acc_{RF} of the RF classifier is also obtained.

3) FCN classifier: Recently, the FCN classifier has achieved excellent performance for time series classification [29]. In this study, the FCN classifier is constructed with five layers: First, the training data are received by the input layer. Then the data is processed by three hidden layers. Finally, the class label is predicted by an output layer. The FCN classifier is trained to minimize the categorical cross-entropy loss. After training, the FCN classifier and the expected accuracy acc_{FCN} are obtained.

C. Classification

Fig. 7 illustrates the classification diagram. As shown in Fig. 7, these three classifiers independently predict the class label of the collected data. Three classification results C_{SVM}, C_{RF} and C_{FCN} are obtained.

D. Fusion Decision

In this step, the three classification results from the previous step were fused to produce the final results. Fig. 8 shows the diagram of the fusion decision. Each classification result of the Subsection IV-C is given a corresponding weight. As shown in Fig. 8, the weight value is the expected accuracy of the corresponding classifier, and the value is obtained in Subsection IV-B. The weight of class i can be calculated as in Eq. (3), where C_p is the classification result of the SVM, RF or FCN classifier, w_p is the corresponding weight of the result, C_i is the class label of the model M^i. Then, the class with the maximum weight is the class of the matched model. Finally, the component corresponding to the matched model is identified as the final result.

$$w_i = \sum_{C_i = C_p} w_p \tag{3}$$

V. Experiments

A. Experimental Setting

(1) Experimental data.

In our experiments, 1000 items are generated for six components for each experiment. The six components and their corresponding models are shown in Table III. As Table III shows, the models of these components are divided into two types:

- One type is normal distribution models $X \sim N(\mu, \sigma^2)$. The data distribution function of a model in this kind is
$$f(x) = \frac{1}{\mu\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}, \tag{4}$$
where μ is the mean value, and σ is the standard deviation.
- Another type is uniform distribution models $X \sim U(a, b)$. The data distribution function of such a model is
$$f(x) = \begin{cases} \frac{1}{b-a}, & a < x < b \\ 0, & otherwise \end{cases} \tag{5}$$
where a is the minimum value, and b is the maximum value.

Due to the complex working environment, each observation data is obtained as Eq. (6). In Eq. (6), V_m is the value generated by the models, N_g is the Gaussian noise, and r is the scale of reduction and enlargement. In Eq. (6), r is a random value between $-r_{max}$ and r_{max}.

$$V = (V_m + N_g) * (1 + r) \tag{6}$$

(2) Evaluation metrics.

TABLE II: Optional parameter values of the RF classifier

Parameters	Optional values
Number of trees in the forest	50, 100, 150, 200, 250
Maximum depth of the tree	1, 3, 5, 7, 9, 11, 13
Whether bootstrap samples are used?	True, False
The quality measure function of a split	'gini', 'entropy'

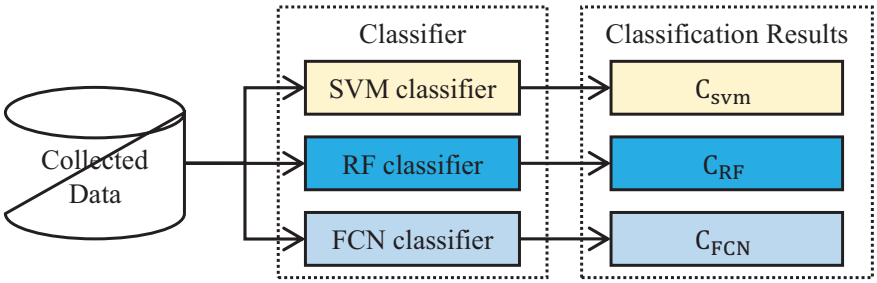


Fig. 7: The diagram of classification.

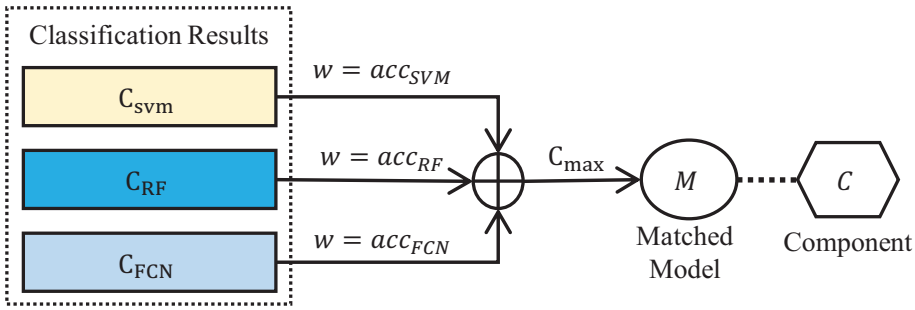


Fig. 8: The diagram of fusion decision.

TABLE III: Components and corresponding models

Component ID	Corresponding model	Type
C^1	$X \sim N(0,1)$	
C^2	$X \sim N(0,2)$	Type 1
C^3	$X \sim N(1,1)$	
C^4	$X \sim U(0,1)$	
C^5	$X \sim U(0,1.2)$	Type 2
C^6	$X \sim U(-0.2,1)$	

Component identification accuracy (acc_C) and conversion relationship accuracy (acc_R) are used as the evaluation metrics.

Acc_C can be calculated as Eq. (7). In Eq. (7), C_{total} is the total number of items collected, , and $C_{accuracy}$ is the number of the number of items correctly identified.

$$acc_C = \frac{C_{accuracy}}{C_{total}} \tag{7}$$

Acc_R can be calculated as Eq. (8). In Eq. (8), R_{total} is is the total number of conversion relations, and $R_{accuracy}$ is the number of correct conversion relationships. Since a conversion relationship is between two adjacent data items, R_{total} is equal to C_{total} minus one.

$$acc_R = \frac{R_{accuracy}}{R_{total}}, \tag{8}$$

TABLE IV: Parameters setting

Parameters	Value
m: the generated number for each model	50
σ: the standard deviation of Gaussian noise	0.1
r_{max}: the reduction and enlargement scale	0.1

(3) Experimental environment.

Experiments were run in Python on a 3.40 GHz Intel Core i5 CPU with 16GB, 2667 MHz internal memory. Unless otherwise stated, the parameters are set as Table IV. For reproducibility, we have released our codes and parameters on Github[1]. The results can be reproduced independently.

B. Comparison with Baselines

We contract our method with the four baselines: CNN [40], ROCKET [42]–[44], TDE [49] and catch22 [37]. The

[1]Our codes: https://github.com/Ji-Cun/ModelMatching

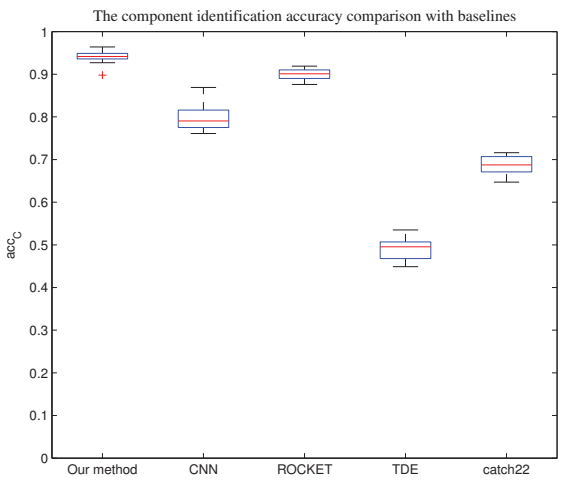


Fig. 9: Component identification accuracy comparison with baselines.

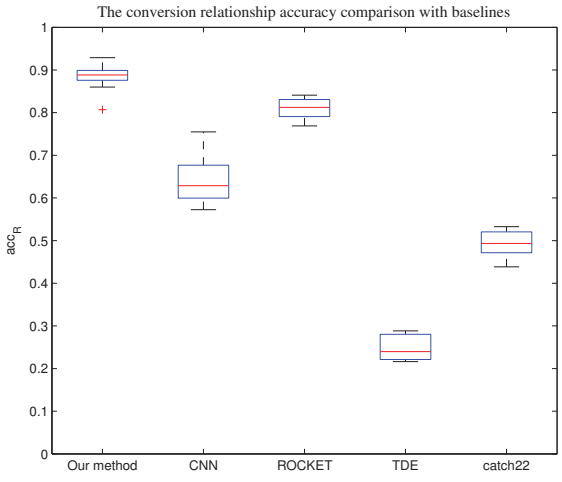


Fig. 10: Conversion relationship accuracy comparison with baselines.

comparison results of these methods are plotted in Fig. 9 and Fig. 10.

Among them, the component identification accuracy rates of these methods are depicted in Fig. 9. As shown in Fig. 9, our method has the highest component identification accuracy among these methods.

At the same time, the conversion relationship accuracy rates of these methods are depicted in Fig. 10. As shown in Fig. 10, our method has the highest conversion relationship accuracy.

Overall, these results in Fig. 9 and Fig. 10 show that the proposed method is better than the baselines.

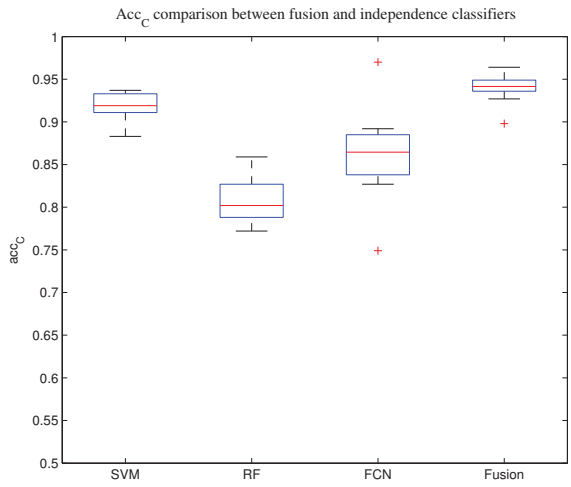


Fig. 11: Component identification accuracy comparison between fusion and independence classifiers.

C. Sensitivity Analysis

(1) Fusion analysis.

As mentioned in Section IV, the proposed methods fused three independent classifiers: SVM, RF and FCN. This set of experiments compares the fusion and independence classifiers.

The component identification accuracy rates of the fusion and independence classifiers are depicted in Fig. 11. Fig. 11 shows that using classifiers in fusion has a higher component identification accuracy than using them in independence.

Meanwhile, the conversion relationship accuracy rates of these methods the fusion and independence classifiers are depicted in Fig. 12. Fig. 12 shows that using classifiers in fusion has higher conversion relationship accuracy than using them in independence.

Overall, these results in Fig. 11 and Fig. 12 show that using classifiers in fusion can improve accuracy.

(2) Noise analysis.

We analyzed the effect of the reduction and enlargement scale and Gaussian noise, respectively.

First, we analyze the effect of the reduction and enlargement scale. In this group of experiments, the reduction and enlargement scale r_{max} is set to 0, 0.1, 0.2, $\cdots$ and 1. At the same time, the standard deviation σ is set to 0. The effect of the reduction and enlargement scale r_{max} is shown in Fig. 13. As shown in Fig. 13, acc_C and acc_R decrease slowly as r_{max} increases.

Secondly, we analyze the effect of Gaussian noise. In this set of experiments, the standard deviation σ of the Gaussian noise is set to 0, 0.1, 0.2, $\cdots$ and 1. At the same time, the reduction and enlargement scale r_{max} is set to 0. The effect of the Gaussian noise is shown in Fig. 14. As shown in Fig. 14, acc_C and acc_R decrease rapidly as r_{max} increases.

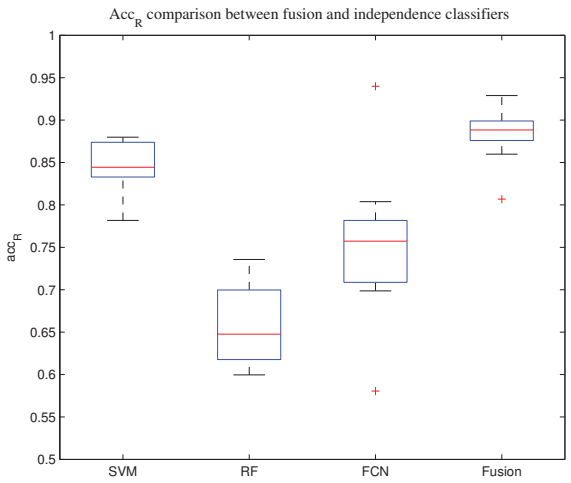


Fig. 12: Conversion relationship accuracy comparison between fusion and independence classifiers.

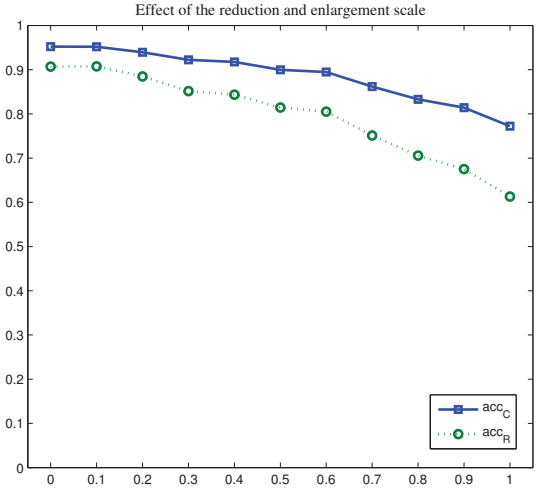


Fig. 13: Effect of the reduction and enlargement scale.

From Fig. 13 and Fig. 14, we can conclude that the proposed method is robust to the reduction and enlargement scale r_{max}. However, Gaussian noise has a significant impact on the accuracy.

(3) Generated number analysis.

The effect of the generated number m in Section IV-A is shown in Fig. 15. As shown in Fig. 15a, acc_C and acc_R increase with m in the beginning. When m is greater than 40, acc_C and acc_R remain at a high level. As shown in Fig. 15b, the training time increases linearly with m. Considering acc_C, acc_R and training time, we suggest setting m to 40 or 50.

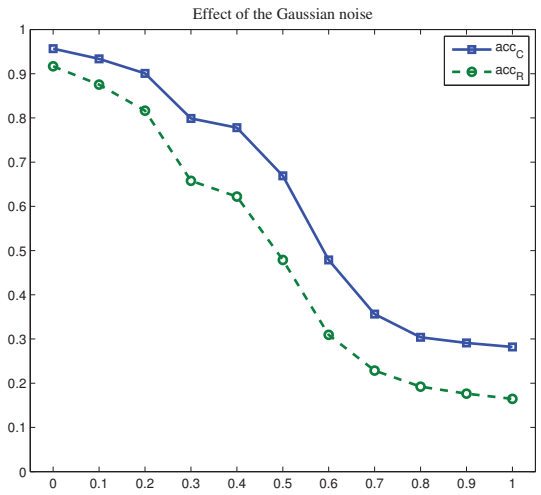


Fig. 14: Effect of Gaussian noise.

VI. Conclusion

Multi-function devices can replace some of their components to achieve a specific function. However, remote servers are typically not informed when workers replace device components. This leads to a mismatch in the data models between the device components and the servers. In this case, automatic identification of device components becomes a major challenge. To this end, this study proposes an automatic component identification method based on time series classification for intelligent devices. First, the component identification problem is transformed into a time series classification problem. Then, some classifiers are trained based on the time series data generated by the models. In the next step, the classifiers match the collected data with the component models. Finally, the components are identified by a fusion decision. The experimental results show that the proposed method has a high component identification accuracy. In the future, we will identify components for intelligent devices with fewer scene constraints.

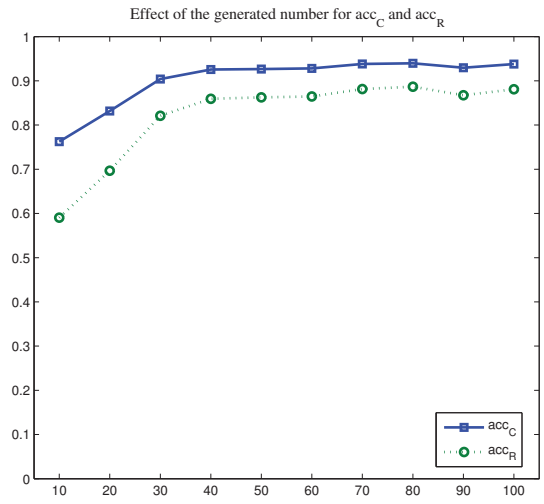
Acknowledgment

This work is partially funded by the Innovation Methods Work Special Project [grant number 2020IM020100], and the Natural Science Foundation of Shandong Province [grant number ZR2020QF112].

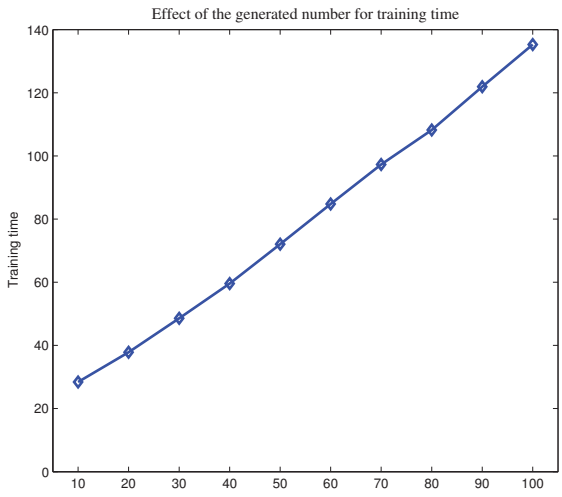
The authors would like to thank the anonymous reviewers and the editors for their insightful comments and suggestions, which are greatly helpful for improving the quality of this paper.

References

[1] M. Younan, E. H. Houssein, M. Elhoseny, and A. A. Ali, "Challenges and recommended technologies for the industrial internet of things: A comprehensive review," *Measurement*, vol. 151, p. 107198, 2020.



(a) acc_M and acc_R with different m



(b) Training time with different m

Fig. 15: Effect of the generated number

[2] P. Rajesh, F. H. Shajin, and G. Kannayeram, "A novel intelligent technique for energy management in smart home using internet of things," *Applied Soft Computing*, vol. 128, p. 109442, 2022.

[3] A. Koohang, C. S. Sargent, J. H. Nord, and J. Paliszkiewicz, "Internet of things (iot): From awareness to continued use," *International Journal of Information Management*, vol. 62, p. 102442, 2022.

[4] Y. Min, S. Shen, H. Li, S. Liu, J. Mi, J. Zhou, Z. Mai, and J. Chen, "Online monitoring of an additive manufacturing environment using a time-of-flight mass spectrometer," *Measurement*, vol. 189, p. 110473, 2022.

[5] Q. Jiang, X. Zhou, R. Wang, W. Ding, Y. Chu, S. Tang, X. Jia, and X. Xu, "Intelligent monitoring for infectious diseases with fuzzy systems and edge computing: A survey," *Applied Soft Computing*, p. 108835, 2022.

[6] S. Leroux and P. Simoens, "Sparse random neural networks for online anomaly detection on sensor nodes," *Future Generation Computer Systems*, 2022.

[7] H. Wang, J. Guo, X. Ma, S. Fu, Q. Yang, and Y. Xu, "Online self-evolving anomaly detection in cloud computing environments," *arXiv preprint arXiv:2111.08232*, 2021.

[8] Z. Meng, Z. Zhao, B. Zhu, and F. Fan, "Online diagnosis for rolling bearings based on multi-channel convolution and transfer learning," *Measurement Science and Technology*, vol. 33, no. 11, p. 115116, 2022.

[9] M. Elsisi, M.-Q. Tran, K. Mahmoud, D.-E. A. Mansour, M. Lehtonen, and M. M. Darwish, "Effective iot-based deep learning platform for online fault diagnosis of power transformers against cyberattacks and data uncertainties," *Measurement*, vol. 190, p. 110686, 2022.

[10] Z. Shi, Y. Li, and C. Liu, "Knowledge distillation-enabled multi-stage incremental learning for online process monitoring in advanced manufacturing," in *2022 IEEE International Conference on Data Mining Workshops (ICDMW)*. IEEE, 2022, pp. 860–867.

[11] J. Wang, C. Xu, J. Zhang, and R. Zhong, "Big data analytics for intelligent manufacturing systems: A review," *Journal of Manufacturing Systems*, vol. 62, pp. 738–752, 2022.

[12] C. Ji, S. Liu, C. Yang, L. Cui, L. Pan, L. Wu, and Y. Liu, "A self-evolving method of data model for cloud-based machine data ingestion," in *2016 IEEE 9th International Conference on Cloud Computing*. IEEE, 2016, pp. 814–819.

[13] A. Motamedi and A. Hammad, "Lifecycle management of facilities components using radio frequency identification and building information model," *Journal of Information Technology in Construction*, vol. 14, no. 18, pp. 238–262, 2009.

[14] J. Gao, H. Sun, J. Han, Q. Sun, and T. Zhong, "Research on recognition method of electrical components based on feyolov4-tiny," *Journal of Electrical Engineering & Technology*, pp. 1–11, 2022.

[15] P. M. Sánchez Sánchez, J. M. Jorquera Valero, A. Huertas Celdrán, G. Bovet, M. Gil Pérez, and G. Martínez Pérez, "Can evil iot twins be identified? now yes, a hardware behavioral fingerprinting methodology," *Now Yes, a Hardware Behavioral Fingerprinting Methodology*, 2021.

[16] M. Giuliani, L. Cadei, M. Montini, A. Bianco, F. Grimaccia, M. Mussetta, and A. Niccolai, "Hybrid artificial intelligence techniques for automatic simulation models matching with field data and constrained production optimization," in *International Petroleum Technology Conference*. OnePetro, 2020.

[17] S. Marchal, M. Miettinen, T. D. Nguyen, A.-R. Sadeghi, and N. Asokan, "Audi: Toward autonomous iot device-type identification using periodic communication," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1402–1412, 2019.

[18] A. A. Cook, G. Mısırlı, and Z. Fan, "Anomaly detection for iot time-series data: A survey," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6481–6494, 2019.

[19] M. Du, Y. Wei, X. Zheng, and C. Ji, "Multi-feature based network for multivariate time series classification," *Information Sciences*, vol. 639, p. 119009, 2023.

[20] S. Li, L. D. Xu, and S. Zhao, "The internet of things: a survey," *Information systems frontiers*, vol. 17, pp. 243–259, 2015.

[21] K. Rose, S. Eldridge, and L. Chapin, "The internet of things: An overview," *The internet society (ISOC)*, vol. 80, pp. 1–50, 2015.

[22] A. Stornelli, S. Ozcan, and C. Simms, "Advanced manufacturing technology adoption and innovation: A systematic literature review on barriers, enablers, and innovation types," *Research Policy*, vol. 50, no. 6, p. 104229, 2021.

[23] Y. Ye, S. Wan, S. Li, and X. He, "Mechanical wind sensor based on additive manufacturing technology," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–8, 2022.

[24] C. Ji, Y. Hu, S. Liu, L. Pan, B. Li, and X. Zheng, "Fully convolutional networks with shapelet features for time series classification," *Information Sciences*, vol. 612, pp. 835–847, 2022.

[25] L. Xi, Y. Liang, X. Huang, H. Liu, and A. Li, "Unsupervised multimodal domain adversarial network for time series classification," *Information Sciences*, vol. 624, pp. 147–164, 2023.

[26] A. Abanda, U. Mori, and J. A. Lozano, "A review on distance based time series classification," *Data Mining and Knowledge Discovery*, vol. 33, no. 2, pp. 378–412, 2019.

[27] A. P. Ruiz, M. Flynn, J. Large, M. Middlehurst, and A. Bagnall, "The great multivariate time series classification bake off: a review and

experimental evaluation of recent algorithmic advances," *Data Mining and Knowledge Discovery*, vol. 35, no. 2, pp. 401–449, 2021.
[28] A. Gupta, H. P. Gupta, B. Biswas, and T. Dutta, "Approaches and applications of early classification of time series: A review," *IEEE Transactions on Artificial Intelligence*, vol. 1, no. 1, pp. 47–61, 2020.
[29] H. Ismail Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, "Deep learning for time series classification: a review," *Data mining and knowledge discovery*, vol. 33, no. 4, pp. 917–963, 2019.
[30] D. S. Shen and M. Chi, "Tc-dtw: Accelerating multivariate dynamic time warping through triangle inequality and point clustering," *Information Sciences*, vol. 621, pp. 611–626, 2023.
[31] M. Baghizadeh, K. Maghooli, F. Farokhi, and N. J. Dabanloo, "A new emotion detection algorithm using extracted features of the different time-series generated from st intervals poincaré map," *Biomedical Signal Processing and Control*, vol. 59, p. 101902, 2020.
[32] A. Nanopoulos, R. Alcock, and Y. Manolopoulos, "Feature-based classification of time-series data," *International Journal of Computer Research*, vol. 10, no. 3, pp. 49–61, 2001.
[33] X. Wang, K. Smith, and R. Hyndman, "Characteristic-based clustering for time series data," *Data mining and knowledge Discovery*, vol. 13, pp. 335–364, 2006.
[34] H. Deng, G. Runger, E. Tuv, and M. Vladimir, "A time series forest for classification and feature extraction," *Information Sciences*, vol. 239, pp. 142–153, 2013.
[35] L. Ye and E. Keogh, "Time series shapelets: a new primitive for data mining," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2009, pp. 947–956.
[36] C. Ji, M. Du, Y. Hu, S. Liu, L. Pan, and X. Zheng, "Time series classification based on temporal features," *Applied Soft Computing*, p. 109494, 2022.
[37] C. H. Lubba, S. S. Sethi, P. Knaute, S. R. Schultz, B. D. Fulcher, and N. S. Jones, "catch22: Canonical time-series characteristics," *Data Mining and Knowledge Discovery*, vol. 33, no. 6, pp. 1821–1852, 2019.
[38] A. Kotsifakos and P. Papapetrou, "Model-based time series classification," in *International Symposium on Intelligent Data Analysis*. Springer, 2014, pp. 179–191.
[39] H. Xing, Z. Xiao, R. Qu, Z. Zhu, and B. Zhao, "An efficient federated distillation learning system for multi-task time series classification," *IEEE Transactions on Instrumentation and Measurement*, 2022.
[40] B. Zhao, H. Lu, S. Chen, J. Liu, and D. Wu, "Convolutional neural networks for time series classification," *Journal of Systems Engineering and Electronics*, vol. 28, no. 1, pp. 162–169, 2017.
[41] Z. Wang, W. Yan, and T. Oates, "Time series classification from scratch with deep neural networks: A strong baseline," in *2017 International joint conference on neural networks*. IEEE, 2017, pp. 1578–1585.
[42] A. Dempster, F. Petitjean, and G. I. Webb, "Rocket: exceptionally fast and accurate time series classification using random convolutional kernels," *Data Mining and Knowledge Discovery*, vol. 34, no. 5, pp. 1454–1495, 2020.
[43] A. Dempster, D. F. Schmidt, and G. I. Webb, "Minirocket: A very fast (almost) deterministic transform for time series classification," in *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, 2021, pp. 248–257.
[44] C. W. Tan, A. Dempster, C. Bergmeir, and G. I. Webb, "Multirocket: multiple pooling operators and transformations for fast and effective time series classification," *Data Mining and Knowledge Discovery*, pp. 1–24, 2022.
[45] C.-C. Chang and C.-J. Lin, "Libsvm: a library for support vector machines," *ACM transactions on intelligent systems and technology*, vol. 2, no. 3, pp. 1–27, 2011.
[46] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.
[47] D. M. Abdullah and A. M. Abdulazeez, "Machine learning applications based on svm classification a review," *Qubahan Academic Journal*, vol. 1, no. 2, pp. 81–90, 2021.
[48] N. M. Abdulkareem, A. M. Abdulazeez *et al.*, "Machine learning classification based on radom forest algorithm: A review," *International Journal of Science and Business*, vol. 5, no. 2, pp. 128–142, 2021.
[49] M. Middlehurst, J. Large, G. Cawley, and A. Bagnall, "The temporal dictionary ensemble (tde) classifier for time series classification," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2020, pp. 660–676.