Journal of Network Intelligence ©2024 ISSN 2414-8105 (Online)
Taiwan Ubiquitous Information Volume **9**, Number **1**, February **2024**

A Time Series Deep-Learning-Based Abnormality Detection Model in Power Consumption

Jingxiang Li*, Hao Lai, Yanhui Shi, Jinhai Wang

EHV Power Transmission Company, CSG, Guangzhou 510006, China
ljx.hust@163.com, 254901390@qq.com, zzbingshi@126.com, wangjinhai@im.ehv.csg

*Corresponding author: Jingxiang Li

Received October 27, 2023, revised December 27, 2023, accepted January 15, 2024

Abstract. *In modern electricity power systems, smart meters have been widely used to monitor the systems. Time series data is one of the most common types of data in these systems, such as household electricity power consumption, equipment running log etc. Thus, detecting abnormal behaviour according to these time series data is a key mechanism for power system management. In this paper, we focus on the problem of detecting abnormality by a time series household power consumption dataset. Although many researchers have provided methods for abnormality detection in electricity power systems, they still face several challenges. Traditional abnormality detection methods based on algebraic theory and signal-processing techniques are hard to deal with a large-scale dataset as the data collected from the power systems are increasing sharply and diverse. Deep-learning-based abnormality detection methods can deal with large and diverse data if the training data has been labeled between abnormal elements and normal ones. However, how to label a large-scale dataset is also another challenge. With unsupervised deep-learning-based models, some studies can avoid the problem of labeling data. Nonetheless, accurately defining output and detection accuracy are the new challenges for these approaches. To tackle the above challenges, we present a novel deep-learning-based abnormality detection model that combines an unsupervised clustering model and a recurrent neural network. The proposed model was trained on real power consumption time series data. The result shows that the proposed model not only can get rid of the trouble of labeling data but can also achieve higher accuracy than several existing recurrent neural networks.*

Keywords: Abnormality Detection, Deep Learning, Time Series, Power Consumption

1. Introduction.

Household electricity behaviour analysis plays a critical role in the management of power grids for utility companies [1]. Abnormal electricity behaviours may cause unfair electricity supply, and economic losses and even affect the safety of power grid operation [2]. Therefore, abnormality detection in power consumption has drawn great interest in the literature and industry [3–5]. It has previously been observed that typical abnormal electricity behaviours include electricity theft, meter failures and power systems attacks. Data from several studies suggest that household electricity consumption data is the key data in abnormality detection with the advent of smart meters [6]. Therefore, we focus on the abnormality detection based on the time series data of household power consumption in this paper. Traditionally, studies for abnormality detection use differential-algebraic theory along with signal-processing techniques to model the power system [7, 8]. Although such approaches are built on well-established theories, they have failed to address large computations caused by the modern power system with increasing size. Recently, deep-learning-based abnormality detection has become popular since its good performance in learning information from the input data including the time series data of household power consumption [9].

However, potential challenges are raised when using those deep-learning-based methods [10]. Most of the conventional deep learning methods are trained in a supervised way which means the training dataset should be classified between normal and anomaly elements. However, the real power consumption data usually doesn't have information about which one is the normal one or the outlier. Even in some cases with information that can indicate the outliers, it is still time-consuming to label large-scale data and the methods may be unable to detect new types of faults. Therefore, semi-supervised or unsupervised learning methods were present. These methods are able to find the difference between the normalities and outliers without the existence of the labeled data or with a small amount of labeled data. In contrast, unsupervised learning approaches also face a few disadvantages. It might require human intervention to understand the patterns. The sorting and output are difficult to accurately define. Furthermore, the results often have less accuracy.

This study set out to explore a novel deep-learning-based model for detecting abnormality in power systems which tries to tackle the aforementioned challenges. We obtain an unlabeled dataset with measurements of individual household electric power consumption. It's a representative time series dataset in a power system. Thus, the proposed method which can utilize this kind of data has a large potential promotion and application value. The main idea of the proposed method is to combine an unsupervised clustering model with a recurrent neural network (RNN) [11]. The unsupervised clustering model is used to give a feature of cluster number for each element of the dataset. Then the recurrent neural network is used to give a prediction of the whole trend of the electric power consumption. Finally, those elements of the dataset that are incompatible with the trend will be identified as outliers. In this way, the proposed model can address the aforementioned challenges. Specifically, our contributions are summarized as follows:

- We propose a novel deep-learning-based model to detect the abnormality in power consumption with time series data. The model combines an unsupervised clustering model with a recurrent neural network. We conducted the experiments on a real dataset of household electric power consumption. Compared with the well-known time series deep learning models, the proposed model shows a better performance for detecting the abnormality.
- As we have mentioned, there are two key challenges in the existing deep learning models. One is to label the boundary between the normal elements and abnormal ones in a large-scale dataset before training a supervised model. The other one is to have a better abnormality detection accuracy by using an unsupervised model. With an unsupervised clustering model and the assumption that the outliers should off-track the trend of the whole dataset, the proposed model can address these two problems well.
- The proposed model deals with time series data which is representative of a power system and we randomly insert some manual outliers to examine its capacity to detect new faults. As a result, the proposed model shows great potential ability to be applied in more scenarios with time series data.

The remaining parts of the paper are arranged as follows. Section 2 describes the related studies on abnormality detection. Section 3.2 shows the preprocessing of the dataset. Section 3 introduces the construction of the proposed model. Section 4 gives the experiment result of the proposed model. Finally, we conclude the work in Section 5.

2. **Related Work.**

In 2013, Alam et al. [12] surveyed the different fault detection techniques proposed in the literature. Dealing with large-scale data in abnormality detection in power systems had become a challenge at that time. Since the widespread of smart meters in electricity systems, Wang et al. [5] provided an approach for clustering electricity consumption behaviour in 2016. In this study, the time-based Markov model [13] and symbolic aggregate approximation are used to transform the large dataset to transition matrixes and tackle the challenges of big data. Later, due to the outstanding performance in learning hidden information from a raw dataset, deep-learning-based methods like convolutional neural network (CNN) [14] for abnormality detection are present. In particular, a long short-term memory (LSTM) model is widely used to deal with time series data [15].

3. **Our Proposed Model.**

In this section, we will introduce the construction of the proposed model. The proposed model combines two classical deep learning models. Thus, before we discuss the details of the proposed model, we will briefly describe these sub-models used in our model.

3.1. **Overview of the Proposed Model.**

As is shown in Figure 1, the proposed model contains a K-means clustering model, a GRU model and a fully connected layer. After the data preprocessing, we have a dataset in which each row is a one-hour power consumption data in a day. One row of data has 8 columns. The first column is the global active power that our model tries to fit and predict. The remaining 7 columns are taken as the features corresponding to the global active power. And because the dataset hasn't labeled which rows are outliers, we cannot construct a supervised deep learning model directly to train and detect the abnormality. Therefore, we are trying to construct a novel mode to detect the outliers. The proposed model is designed with the following steps:

- **Step 1.** As with most practices in deep learning, we divide the dataset into a training dataset and a test dataset. Then input the training dataset into a K-means clustering model. The purpose is to get a trained K-means clustering model to assign a cluster number for each hour's consumption data. Thus, an hour's consumption data will contain one global active power and 8 features. Based on the analysis of the raw dataset in Section 3.2, we believe that the variant trend in the power consumption data has something to do with the time series. It means the power consumption in different hours or days may follow a distinct trend. For example, power consumption on Sunday night will follow the same trend, while that on Monday morning will display another distinct trend. Thus, we want to use an unsupervised algorithm to cluster the data and give each hour's data a cluster tag as a feature.
- **Step 2.** We then set 24-hour data as one group to predict the next hour's power consumption. The original training dataset will get a cluster number for each row after the K-means clustering model. Then the new training dataset with cluster numbers will be input into a GRU model. Thus, the shape of one input data becomes $(24, 8)$.
- **Step 3.** To fit the global active power, we connect the output of the GRU model with a fully connected layer that has only one output. The output is a prediction of the next hour's global active power. The loss function is the mean absolute error [16]. Finally, we compare the predicted data with the real data, and real data that has numerous deviations from the predicted data will be identified as an outlier. The criteria to define the deviation is defined by the metric of the threshold of abnormal power consumption in section 4.

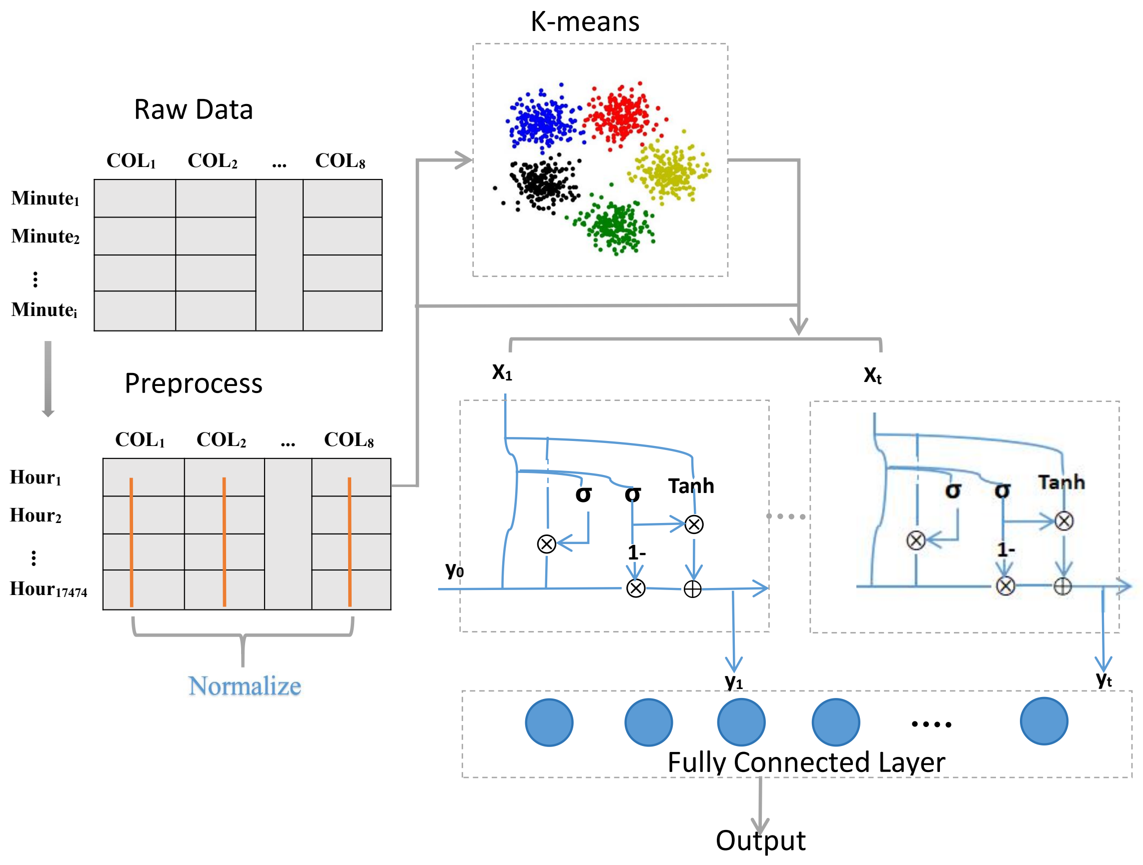

FIGURE 1 The Proposed Model

3.2. **Dataset Preprocessing.** The dataset we used in this paper is collected from the individual household electric power consumption in Sceaux (7km of Paris, France) [17]. The elements of the dataset are given in every minute. And each element or power consumption contains 9 variable information. The variable information is given in Table 1. The whole dataset contains 2075259 elements between December 2006 and November 2010 (47 months).

TABLE 1 Variable Information

Variable Name	**Description**
Date	In format dd/mm/yyyy
Time	In format hh:mm:ss
Global_active_power	Household global minute-averaged active power
Global_reactive_power	Household global minute-averaged reactive power
Voltage	Minute-averaged voltage
Global_intensity	Household global minute-averaged current intensity
Sub_metering_1	Energy sub-metering No. 1 corresponding to the kitchen
Sub_metering_2	Energy sub-metering No. 2 corresponding to the laundry room
Sub_metering_3	Energy sub-metering No. 3 corresponding to an electric water-heater and an air-conditioner

To train the proposed model and get a good performance, we need to preprocess the dataset. Firstly, we should transfer the minute elements into hourly elements. As is shown in Algorithm 2, we sum the value of the variables of every 60 minutes. For the missing value, we take the mean value of the other elements in the same hour as its value. The data preprocessing is given in Algorithm 1. At the beginning, we delete the incoherent elements which is the data in the same hour contains less than 60 elements. For example, the data in the hour of 2006-12-15 17:00 which only contains 30 elements from 2006-12-15 17:30 to 2006-12-15 17:59 will be deleted. Then we combine the variables of Date and Time as one variable. Next, we invoke Algorithm 2 in Line 13 and 15 to calculate the sum value for every variable in an hour. Note that we take the mean value of the Voltage variable in an hour instead of the sum value. At the end, we normalize the data in Line 22 to 25 with a function of MinMaxScaler which is a data preprocessing technique that transforms numerical features by scaling them to a given range. It is often used as an alternative to standardization, which scales features to have zero mean and unit variance. The MinMaxScaler works by applying the following formula to each feature:

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}} \times (max - min) + min$$

where x is the original value, x_{min} and x_{max} are the minimum and maximum values of the feature, and min and max are the desired range. After preprocessing, the shape of the dataset becomes 17474×8 which means it contains 17474 elements or rows and each element contains 8 variables.

Algorithm 1 Data Preprocessing

```
 1: reader = open('raw_dataset')
 2: writer = create('new_dataset')
 3: Delete the incoherent elements
 4: HourlyData = []
 5: repeat
 6:    Read a row from reader
 7:    HourlyData.append(row)
 8:    if len(data) == 60 then
 9:       newline = [HourlyData[0][0],HourlyData[0][1]]
10:       for i in range(2, len(HourlyData[0])) do
11:          col = [row[i] for row in HourlyData]
12:          if 4 == i then
13:             newline.append(sumlist(col)/60)
14:          else
15:             newline.append(sumlist(col))
16:          end if
17:       end for
18:       writer.insert(newline)
19:       HourlyData = []
20:    end if
21: until The End of the dataset
22: dataset = read('new_dataset')
23: dataset = dataset.iloc[:,2:]
24: scaler = MinMaxScaler(feature_range=(0, 1)).fit(dataset)
25: scaled = scaler.fit_transform(dataset)
26: return scaled
```

Algorithm 2 sumlist

```
Require: a list of data including 60 elements
 1: sum = 0
 2: numMiss = 0
 3: for item in list do
 4:    if item is a miss value then
 5:       numMiss += 1
 6:    else
 7:       sum += float(item)
 8:    end if
 9: end for
10: if 0 == (60-numMiss) then
11:    return 0
12: end if
13: sum = (sum/(60-numMiss))*numMiss+sum
14: return sum
```

3.3. **Clustering.** The basic idea of the K-means clustering model was first provided by MacQueen et al. [18] in 1967 and then various extensions were proposed in the literature [19]. It is an unsupervised way of grouping data into *k* clusters based on their similarity. As is shown in Algorithm 3, the clustering method works by finding *k* points, called centroids, that represent the center of each cluster. Then, it assigns each data point to the nearest centroid, forming *k* clusters. This process will be repeated until the centroids do not change much or a maximum number of iterations is reached.

Algorithm 3 K-Means Clustering

```
Require: Data set $X = \{x_1, x_2, ..., x_n\}$, number of clusters $k$
Ensure: Cluster assignments $S = \{S_1, S_2, ..., S_k\}$, cluster centroids $C = \{c_1, c_2, ..., c_k\}$
 1: Initialize $C$ by randomly selecting $k$ data points from $X$
 2: repeat
 3:    Assign each data point $x_i$ to the nearest centroid $c_j$, forming $k$ clusters:
          $S_j = \{x_i \in X : dist(x_i, c_j) \leq dist(x_i, c_l) \text{ for all } l \neq j\}$
 4:    Update each centroid $c_j$ by computing the mean of all data points in $S_j$:
          $c_j = \frac{1}{|S_j|} \sum_{x_i \in S_j} x_i$
 5: until $C$ does not change or a maximum number of iterations is reached
 6: Return $S$ and $C$
```

3.4. **LSTM and GRU Training.** Except for the K-means clustering model, we also use recurrent neural networks (RNNs) [11] in our model. That is because the household power consumption data is typically time series data and RNNs are a type of neural network that can well handle sequential data. GRU and LSTM are two types of RNNs. They both use a gating mechanism to control the flow of information and avoid the problems of vanishing or exploding gradients that affect standard RNNs.

LSTM: Long Short-Term Memory(LSTM) was introduced by Hochreiter et al. [20] in 1997. LSTM can learn long-term dependencies and capture the context of the input sequence. LSTM can also generate new sequences that are similar to the training data, such as text, music, or images. The basic unit in an LSTM is constructed as follows:

$$\text{input gate: } i_t = \sigma(W_i[y_{t-1}, x_t] + b_i) \tag{1}$$

$$\text{forget gate: } f_t = \sigma(W_f[y_{t-1}, x_t] + b_f) \tag{2}$$

$$\text{output gate: } o_t = \sigma(W_o[y_{t-1}, x_t] + b_o) \tag{3}$$

$$\text{candidate state: } \tilde{c}_t = \tanh(W_c[y_{t-1}, x_t] + b_c) \tag{4}$$

$$\text{cell state: } c_t = f_t \bigotimes c_{t-1} + i_t \bigotimes \tilde{c}_t \tag{5}$$

$$\text{hidden state: } y_t = o_t \bigotimes \tanh(c_t) \tag{6}$$

where σ is the sigmoid function, $\bigotimes$ is the element-wise multiplication, W and b are the weight matrices and bias vectors, x_t is the input vector at time step t, y_t is the hidden state vector at time step t. Typically, a unit has three gates, namely the input gate, the forget gate, and the output gate. The input gate decides how much of the new input to add to the cell state, the forget gate decides how much of the previous cell state to keep or forget, and the output gate decides how much of the current cell state to output as the hidden state. The hidden state is the output of the network, and the cell state is the internal memory of the network. The subscript t represents the index of each unit. All the units combine sequentially as a whole network.

GRU: Gated Recurrent Unit (GRU) as another type of RNNs can also handle sequential data and time-series data well. A comparison among different types of recurrent units in RNNs was given in 2014 [21]. Although LSTM was present earlier than GRU, LSTM is more expressive and flexible than GRU. However, LSTM has more parameters and is slower to compute. In contrast, GRU is simpler and faster than LSTM. And on some tasks, GRU can perform even better than LSTM. In our experiment, we find GRU outperforms LSTM as a sub-model of our model. Both LSTM and GRU use a gating mechanism to control the flow of information and avoid the problems of vanishing or exploding gradients. The main difference between LSTM and GRU is that LSTM has three gates (input, forget, and output) and two states (hidden and cell), while GRU has two gates (update and reset) and one state (hidden). The basic unit in a GRU is constructed as follows:

$$\text{update gate: } z_t = \sigma(W_z[y_{t-1}, x_t] + b_z) \tag{7}$$

$$\text{reset gate: } r_t = \sigma(W_r[y_{t-1}, x_t] + b_r) \tag{8}$$

$$\text{candidate state: } \tilde{y}_t = \tanh(W_a[r_t \bigotimes y_{t-1}, x_t] + b_a) \tag{9}$$

$$\text{hidden state: } y_t = (1 - z_t) \bigotimes y_{t-1} + z_t \bigotimes \tilde{y}_t \tag{10}$$

where z_t, r_t are the gate vectors at time step t. The update gate decides how much of the previous hidden state to keep or forget, and the reset gate decides how much of the previous hidden state to use for computing the new candidate state. GRU can also learn long-term dependencies, capture the context of the input sequence, and is easier to modify and extend.

4. Experiment Analysis.

4.1. Experiment Environment and Metrics for Assessment.

Our model is trained on a personal computer with a memory of 16GB, dual CPU of Intel(R) Core(TM) i7-10510U 1.80GHz 2.30 GHz, 64bit OS of Ubuntu 18.04.6 LTS and without any GPU. We use Python and Tensorflow to implement the model. To assess the performance of the proposed model, we use several metrics defined as follows:

Mean Squared Error (MSE). Mean Squared Error [22] is a statistical metric that evaluates how well a predictor approximates the true value of a quantity. In our model, the final output value is the predicted household power consumption of the next hour. Thus, the MSE is calculated by taking the average of the squared differences between the predicted power consumption values and the real values. The lower the MSE, the better the fit. The MSE can be expressed mathematically as:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{11}$$

where n is the number of examples of the training or test dataset, y_i is the real power consumption values, and $\hat{y}_i$ is the predicted values.

Threshold of Abnormal Power Consumption. We define this metric in our model to identify whether household power consumption is an abnormal value. Firstly, the trained model will output its prediction of household power consumption in the next hour according to the former 24-hour data. Then, we calculate the deviation between the predicted value and the real value as:

$$\text{MAE} = \frac{|y_i - \hat{y}_i|}{\hat{y}_i + \delta} \tag{12}$$

where y_i is a real power consumption value, $\hat{y}_i$ is a corresponding predicted value and δ is a minimal positive number to avoid the denominator becomes 0. Then we identify a household power consumption as an abnormal consumption when its MAE exceeds the threshold of abnormal power consumption. According to the analysis of our experiment, we set the threshold be 0.4 to get the best performance. The soundness of this abnormality detection assumes that the normal household power consumption should follow the trend of the whole consumption dataset and the predicted values follow this trend. Then if a real consumption value has a huge deviation from a predicted value, it is probably an outlier.

Accuracy. Once the model has been trained, we input the test dataset to get a group of predicted power consumption values. Then we compare the real values with these predicted values to find those whose MAE exceeds the threshold of abnormal power consumption. Note the set of these abnormal values as A and let n_1 be the total number of A. Since the original dataset doesn't label which hour of the household power consumption is an abnormal value, we randomly choose one hour in every 24 hours in the test dataset and halve its consumption value. Then, these consumption values are thought to be outliers. Note the set of these manual halved values as B and let n_2 be the total number of B. Finally, we input the new test dataset which includes the manual halved values into the model and get a new group of predicted values. Note those values that exceed the threshold of abnormal power consumption in the new group of predicted values as C. Let D be the power consumption set whose index belongs to $C \cap (A \cup B)$. Then we can calculate the abnormality detection accuracy as follows:

$$\text{Accuracy} = \frac{d}{n_3} \tag{13}$$

where d is the total number of D.

Efficiency. This metric is the percentage of outliers that can be detected. It indicates the efficiency of abnormality detection and can be computed as follows:

$$\text{Efficiency} = \frac{d}{n_1 + n_2 - m} \tag{14}$$

where m is the total number of the data with a common index in A and B.

Relative Ratio. Let a_1 be the total number of the power consumption set whose index belongs to $C \cap (A - (A \cap B))$ and a_2 be the total number of the power consumption set whose index belongs to $C \cap (B - (A \cap B))$. Thus, a_1 represents the number of the original outliers while a_2 represents the number of the manually generated outliers. The relative ratio can be computed as follows:

$$\text{RR} = \frac{a_2 \times n_1}{a_1 \times n_2} \tag{15}$$

Therefore, the relative ratio indicates what kind of outliers are easier to be detected by our model. Since the original outliers were detected based on the assumption that the normal household power consumption should follow the trend of the whole consumption dataset, we think the bigger the value of RR the better the model to detect the unknown outliers.

4.2. Hyperparameters Tuning.

In the proposed model, there are six key hyperparameters that affect the result of the performance or metrics for assessment. To tune the hyperparameters, we selected and changed one of them in a range while fixing the value of the other hyperparameters. Then we compared the results in different values of each selected hyperparameter. Since there are four metrics (Mean Squared Error, Accuracy, Efficiency, and Relative Ratio) for assessing the model, the trade-off among the results of these metrics should be taken into account. Specifically, the tuning process is as follows:

The Number of Epochs. The hyperparameter of the epoch in a deep learning model indicates the number of times that the learning algorithm will work through the entire training dataset. As is shown in Figure 2, the accuracy and efficiency decrease when the number of epochs increases. Whereas, the relative ratio and MSE get a better performance at the beginning. Thus, we suggest the number of epochs is 25.

The Learning Rate. The hyperparameter of learning rate represents the speed at which the model learns and controls the amount of apportioned error that the weights of the model are updated with each time. As is shown in Figure 3, all the metrics get the best performance at 0.002. Thus, 0.002 is the recommended value for the learning rate.

The Batch Size. The batch Size is the number of samples processed in each iteration of the training process. As is shown in Figure 4, all the metrics except MSE get a better performance at 32 or 64. Since the magnitude of the MSE is much less than the other metrics, the difference of MSE in the figure seems not so obvious. However, from the specific results, we can find that the MSE gets the biggest value at 64. Thus, 32 is the recommended value for the batch size.

The Number of Units. This is the number of GRU cells in each layer of the network. It determines the size of the hidden state and the cell state of each cell. Obviously, 24 is the best choice for this hyperparameter as we can see from Figure 5.

The Number of Clusters. This is a hyperparameter of the K-means sub-model we used. It determines how the data points are grouped into distinct regions based on their similarity. There is no definitive answer to how to choose the best number of clusters for a K-means model. As is shown in Figure 6, when choosing 170 clusters, the model gets a better performance comprehensively.

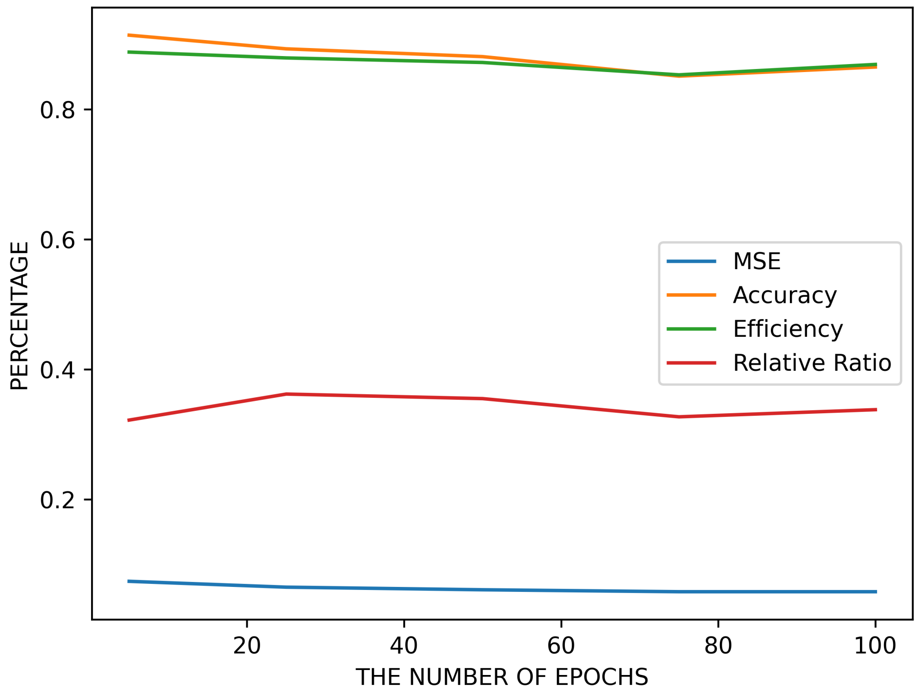

FIGURE 2 Results in different numbers of epochs

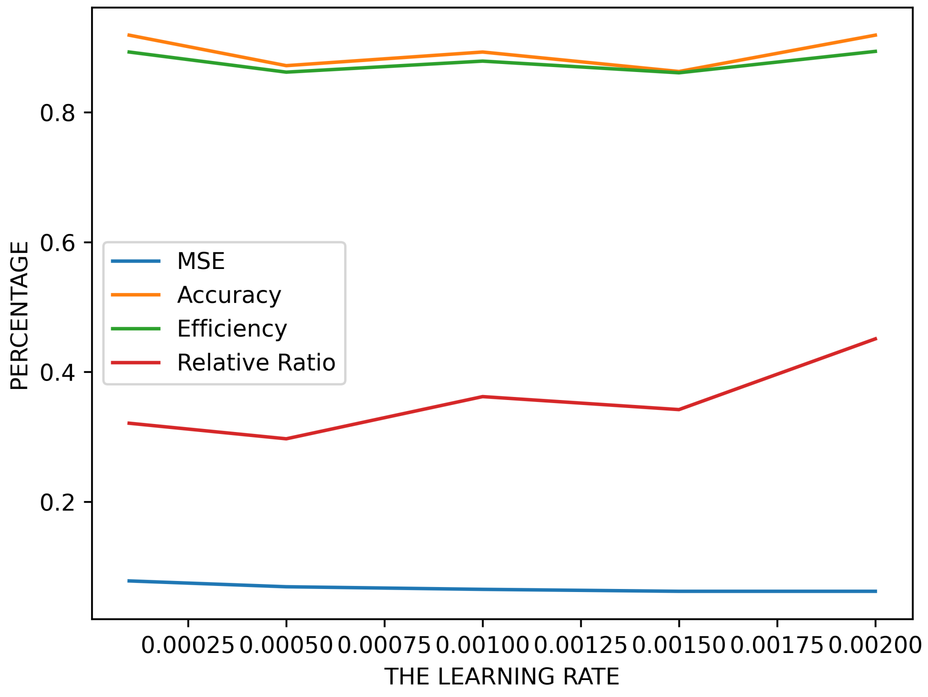

FIGURE 3 Results in different learning rate

The Training Ratio. This is the proportion of the available data that is used for training the model. Apparently, in a ratio of 80 percentage, as is shown in Figure 7, the model gets the best performance at all the assessment metrics.

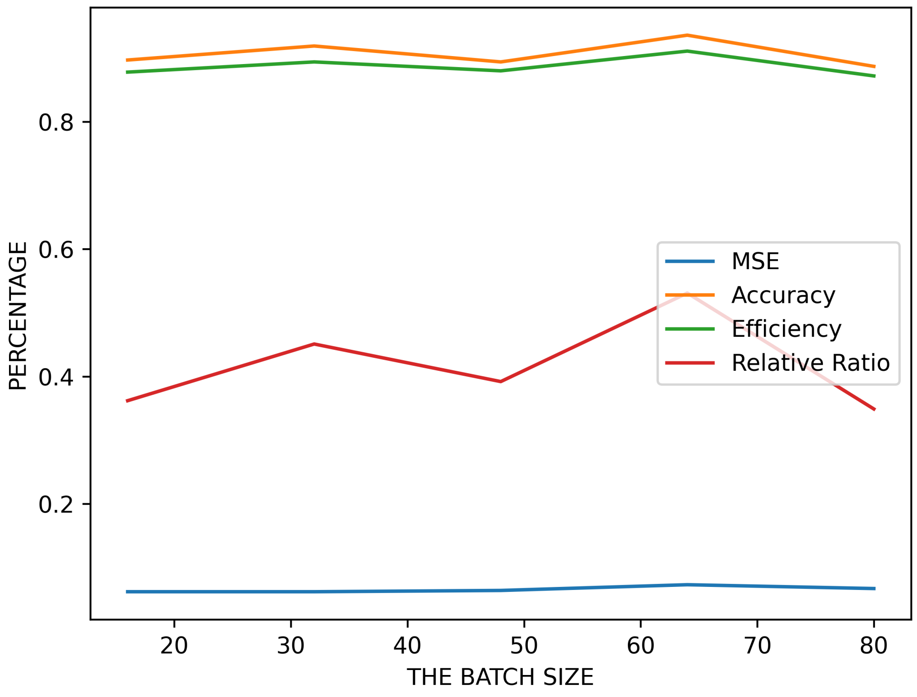

FIGURE 4 Results in different batch size

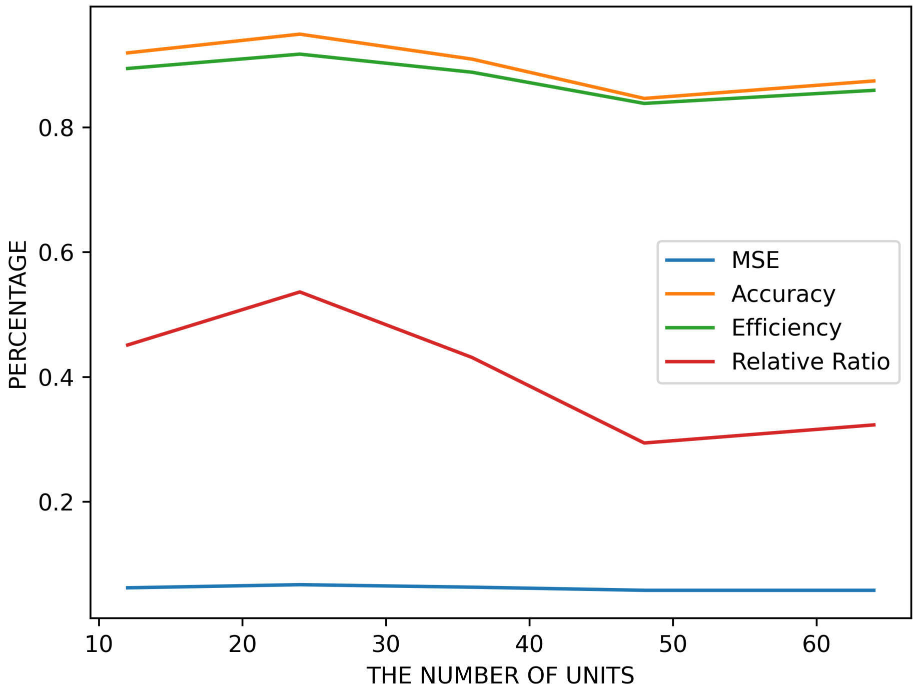

FIGURE 5 Results in different numbers of units

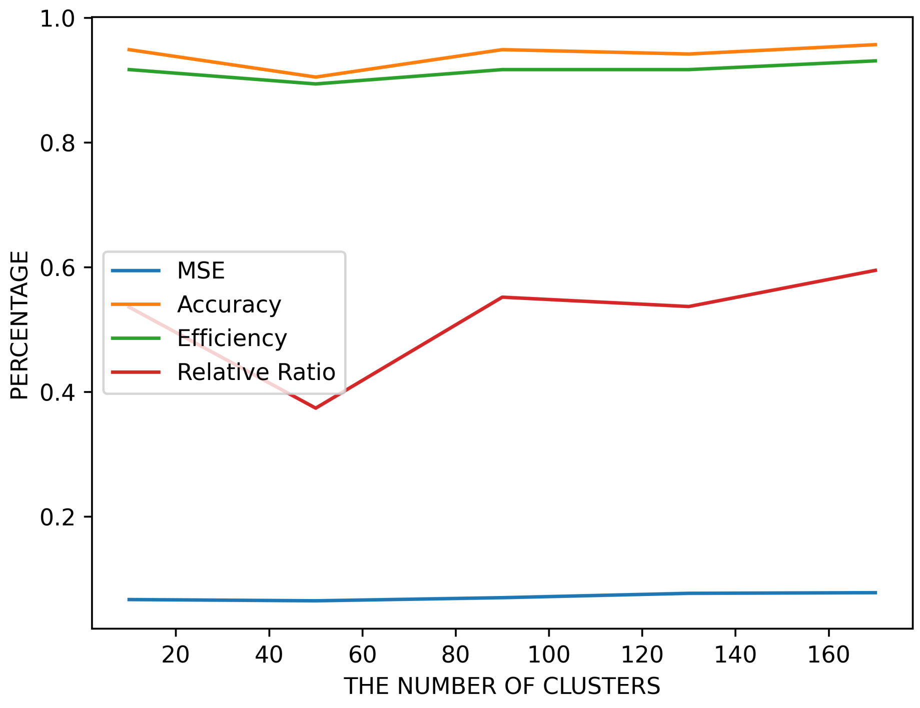

FIGURE 6 Results in different numbers of clusters

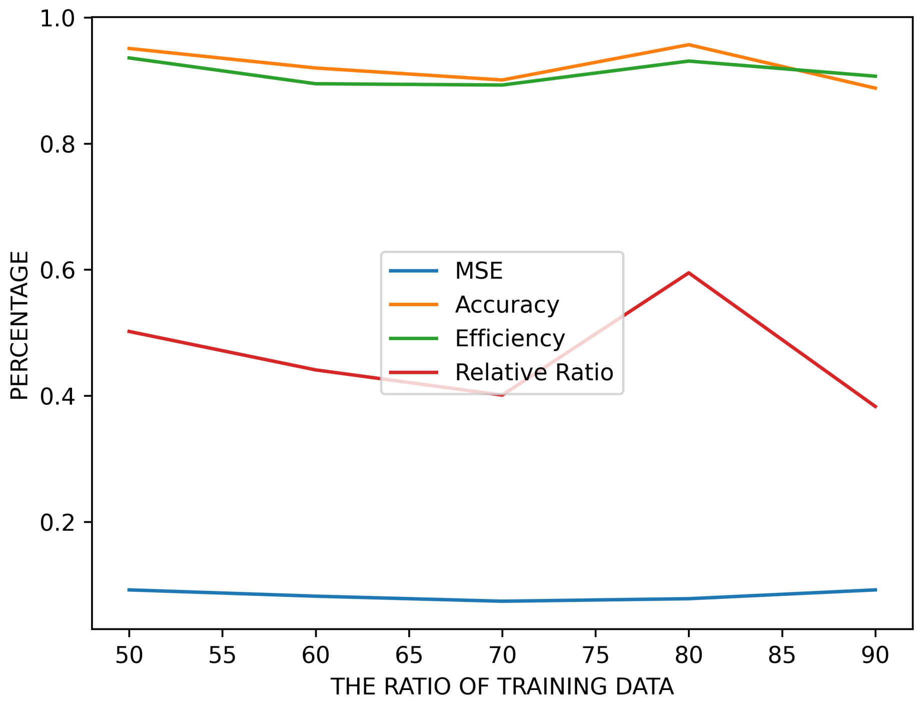

FIGURE 7 Results in different ratios of training data

Table 2 AUC COMPARISON WITH OTHER DEEP LEARNING MODELS

Models	**Accuracy**	**Efficiency**	**Relative Ratio**	**Mean Square Error (MSE)**
LSTM	0.93	0.909	0.446	0.0068
GRU	0.924	0.898	0.462	0.0068
Ours	**0.957**	**0.931**	**0.595**	**0.0078**

4.3. **Comparison with other Models.**

Eventually, we choose the best result of our model to compare with two well-known RNN models (LSTM and GRU) which are good at dealing with time series data. As is shown in Table 2, our model outperforms the other models comprehensively.

5. **Conclusion.** This paper develops a novel model for detecting abnormality in electricity systems. The model was trained on a real dataset of household power consumption. The chosen dataset is a kind of representative data in most electricity systems, and the purpose of choosing a time series dataset is to apply the proposed model in more scenarios in the future. The proposed model is a deep-learning-based model which combines the unsupervised K-means model with the GRU model. Prior to this study, high abnormality detection accuracy and the capacity to deal with a large-scale unlabeled dataset were the two key challenges. The proposed model can address these challenges well. The experiment result suggests that the proposed model outperforms several well-known RNN models. However, this current study is limited by only fitting time series data. A more general method can be found in the future.

REFERENCES

[1] Y. S. Akil, Z. Muslimin, I. Taufik *et al.*, "Household electricity consumption in rural area: An occupant's behavior analysis to increase energy saving," in *Journal of Physics: Conference Series*, vol. 1811, no. 1. IOP Publishing, 2021, p. 012023.

[2] X. Kong, X. Zhao, C. Liu, Q. Li, D. Dong, and Y. Li, "Electricity theft detection in low-voltage stations based on similarity measure and dt-ksvm," *International Journal of Electrical Power & Energy Systems*, vol. 125, p. 106544, 2021.

[3] L. Liang, "Abnormal detection of electric security data based on scenario modeling," *Procedia Computer Science*, vol. 139, pp. 578–582, 2018.

[4] W. Zhang, X. Dong, H. Li, J. Xu, and D. Wang, "Unsupervised detection of abnormal electricity consumption behavior based on feature engineering," *IEEE Access*, vol. 8, pp. 55 483–55 500, 2020.

[5] Y. Wang, Q. Chen, C. Kang, and Q. Xia, "Clustering of electricity consumption behavior dynamics toward big data applications," *IEEE Transactions on Smart Grid*, vol. 7, no. 5, pp. 2437–2447, 2016.

[6] S. Zhang, J. Liu, B. Zhao, J. Cao *et al.*, "Cloud computing-based analysis on residential electricity consumption behavior," *Power System Technology*, vol. 37, no. 6, pp. 1542–1546, 2013.

[7] F. Milano and R. Zárate-Miñano, "A systematic method to model power systems as stochastic differential algebraic equations," *IEEE Transactions on Power Systems*, vol. 28, no. 4, pp. 4537–4544, 2013.

[8] M. H. Bollen, *Understanding power quality problems.* IEEE press New York, 2000, vol. 3.

[9] O. A. Omitaomu and H. Niu, "Artificial intelligence techniques in smart grid: A survey," *Smart Cities*, vol. 4, no. 2, pp. 548–568, 2021.

[10] F. Zhou, G. Wen, Y. Ma, H. Geng, R. Huang, L. Pei, W. Yu, L. Chu, and R. Qiu, "A comprehensive survey for deep-learning-based abnormality detection in smart grids with multimodal image data," *Applied Sciences*, vol. 12, no. 11, p. 5336, 2022.

[11] L. R. Medsker and L. Jain, "Recurrent neural networks," *Design and Applications*, vol. 5, no. 64-67, p. 2, 2001.

[12] M. K. Alam, F. H. Khan, J. Johnson, and J. Flicker, "Pv faults: Overview, modeling, prevention and detection techniques," in *2013 IEEE 14th Workshop on Control and Modeling for Power Electronics (COMPEL).* IEEE, 2013, pp. 1–7.

[13] D. Niu, H. Shi, J. Li, and C. Xu, "Research on power load forecasting based on combined model of markov and bp neural networks," in *2010 8th World Congress on Intelligent Control and Automation*. IEEE, 2010, pp. 4372–4375.

[14] Y. Yang, Y. Wang, and H. Jiao, "Insulator identification and self-shattering detection based on mask region with convolutional neural network," *Journal of Electronic Imaging*, vol. 28, no. 5, pp. 053 011–053 011, 2019.

[15] L. Xia and Z. Li, "A new method of abnormal behavior detection using lstm network with temporal attention mechanism," *The Journal of Supercomputing*, vol. 77, pp. 3223–3241, 2021.

[16] "Mean absolute error," 2023. [Online]. Available: https://en.wikipedia.org/wiki/Mean_absolute_error

[17] G. Hebrail and A. Berard, "Individual household electric power consumption," UCI Machine Learning Repository, 2012, DOI: https://doi.org/10.24432/C58K54.

[18] J. MacQueen *et al.*, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1, no. 14. Oakland, CA, USA, 1967, pp. 281–297.

[19] K. P. Sinaga and M.-S. Yang, "Unsupervised k-means clustering algorithm," *IEEE Access*, vol. 8, pp. 80 716–80 727, 2020.

[20] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[21] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[22] "Mean squared error," 2023. [Online]. Available: https://en.wikipedia.org/wiki/Mean_squared_error