Cybersecurity

RESEARCH Open Access

Exploring best-matched embedding model and classifier for charging-pile fault diagnosis

Wen Wang[1*], Jianhua Wang[3], Xiaofeng Peng[1], Ye Yang[1], Chun Xiao[2], Shuai Yang[2], Mingcai Wang[1], Lingfei Wang[1], Lin Li[3] and Xiaolin Chang[3*]

Abstract

The continuous increase of electric vehicles is being facilitating the large-scale distributed charging-pile deployment. It is crucial to guarantee normal operation of charging piles, resulting in the importance of diagnosing charging-pile faults. The existing fault-diagnosis approaches were based on physical fault data like mechanical log data and sensor data streams. However, there are other types of fault data, which cannot be used for diagnosis by these existing approaches. This paper aims to fill this gap and consider 8 types of fault data for diagnosing, at least including physical installation error fault, charging-pile mechanical fault, charging-pile program fault, user personal fault, signal fault (offline), pile compatibility fault, charging platform fault, and other faults. We aim to find out how to combine existing feature-extraction and machine learning techniques to make the better diagnosis by conducting experiments on realistic dataset. 4 word embedding models are investigated for feature extraction of fault data, including N-gram, GloVe, Word2vec, and BERT. Moreover, we classify the word embedding results using 10 machine learning classifiers, including Random Forest (RF), Support Vector Machine, K-Nearest Neighbor, Multilayer Perceptron, Recurrent Neural Network, AdaBoost, Gradient Boosted Decision Tree, Decision Tree, Extra Tree, and VOTE. Compared with original fault record dataset, we utilize paraphrasing-based data augmentation method to improve the classification accuracy up to 10.40%. Our extensive experiment results reveal that RF classifier combining the GloVe embedding model achieves the best accuracy with acceptable training time. In addition, we discuss the interpretability of RF and GloVe.

Keywords Charging-pile, Fault diagnosis, Machine learning classifier, Word embedding

Introduction

Recently, with the acceleration of global warming, human beings have realized that unrestricted use of fossil energy is harmful to the earth. Electric vehicles (EVs), with the advantage of environment-friendliness and energy efficiency, are considered to replace traditional fuel vehicles (Yan et al. 2019). With the increasing number of EVs, many distributed charging piles are among the essential infrastructures (Chen et al. 2020). Generally, a large number of charging piles locate in the wild with uncontrollable environmental factors, causing frequent charging-pile faults. Therefore, it is crucial to maintain the effectiveness of charging piles (Zhang et al. 2022; Wei et al. 2021).

Charging-pile service companies have been bringing a series of measures into force, with the aim to guarantee the effectiveness of charging piles. For example, when the customers encounter problems, they offer a service hotline and WeChat (Hao et al. 1087) mini program to publish emergency work orders. We now explain why it is necessary for a service provider to predict charging-pile faults to improve the efficiency of repairing service. The

*Correspondence:
Wen Wang
wangwen@evs.sgcc.com.cn
Xiaolin Chang
xlchang@bjtu.edu.cn
[1] State Grid Electric Vehicle Service Company, Ltd., 1 Baiguang Road, Xicheng District, Beijing, China
[2] State Grid Shanxi Marketing Service Center, 10 Wuluo Street, Tanghuai Garden, Taiyuan, Shanxi, China
[3] Beijing Key Laboratory of Security and Privacy in Intelligent Transportation, Beijing Jiaotong University, 3 Shangyuancun, Beijing 100044, China

occurrence of charging-pile work orders may be due to a mechanical fault or cyber security. We can imagine a scenario of mechanical fault: (a) a customer describes a fault of the charging pile using the service hotline; (b) the staff receives the fault work order, records the fault description, and dispatches maintenance workers to repair piles; (c) maintenance workers finish the work order and submit the fault category to the service system. However, dispatching maintenance workers will waste human and material resources if the fault is in the software platform or online electric system. Moreover, from the aspect of cyber security, security analysis and protection mechanisms must be conducted in order to improve the communication security between EVs and charging piles (Li et al. 2021). These discussions emphasize the importance of predicting charging-pile faults.

Recently, machine learning (ML) or deep learning (DL)-based techniques play a crucial role in charging-pile fault diagnosis (Shuai et al. 2022; Du et al. 2021) and abnormal detection (Li et al. 2021). Especially, Li et al. (2021) utilized Random Forest (RF) classifier to implement abnormal detection. However, existing studies on charging-pile fault diagnosis focus on the mechanical log data or sensor data streams (Gao et al. 2020, 2018; Wang et al. 2021; Yong and Ji 1650), while we concentrate on work order fault description data recorded by staff (different from mechanical log data and sensor data streams) and classify 8 types of faults, including installation error fault, charging-pile mechanical fault, charging-pile program fault, user personal fault, signal fault (offline), pile compatibility fault, charging platform fault, and other faults.

Figure 1 presents a simplified workflow of our paper. We firstly collect the raw data from the real-world electric service work orders to build a fault record dataset. Then, we conduct data preprocess by utilizing *Jieba* (Junyi 2022) tokenizer to tokenize the Chinese fault description. After that, we extract fault features based on fault description by adopting the extensively used word embedding models, such as N-gram (Suen 1979), Word2vec (Mikolov et al. 2013), GloVe (Pennington et al. 2014), and BERT (Devlin et al. 2018). At last, we utilize 10 ML or DL classifiers, including RF (Breiman 2001), Support Vector Machine (SVM) (Cortes and Vapnik 1995), K-Nearest Neighbor (KNN) (Sebastiani 2002), Multilayer Perceptron (MLP) (Rumelhart et al. 1986), Recurrent Neural Network (RNN) (Elman 1990), AdaBoost (AB) (Freund and Schapire 1997), Gradient Boosted Decision Tree (GBDT) (Friedman 2001), Decision Tree (DT) (Breiman et al. 2017), Extra Tree (ET) (Geurts et al. 2006), and VOTE, to classify the word embedding features for fault diagnosis.

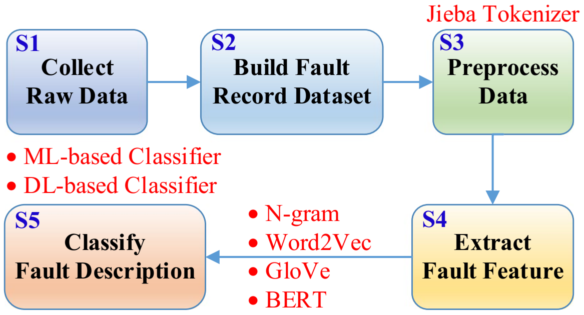


Fig. 1 The flow of paperwork

We summarize the following main contributions:

- We create a dataset of realistic charging-pile faults. Specifically, we collect original long-term real-world electric service work orders from June to December 2021. Moreover, we select fault description and category to build a structured fault record dataset. "Fault record dataset" section details the building of the dataset.
- We carry out extensive experiments to explore the best-matched combination between 4 fault description feature extraction models and 10 classifiers for effective fault diagnosis. To the best of our knowledge, we are the first to achieve all types of charging-pile fault diagnoses using fault descriptions ("Experimental result and discussion" section).

The left paper is organized as follows. "Preliminary" section overviews word embedding approaches and classifiers. "Fault record dataset" section gives the fault-record dataset. Experimental results and discussion are provided in "Experimental result and discussion" section. "Conclusion" section presents the conclusion.

Preliminary

Word embedding vector is a crucial feature extraction approach and benefits calculating the cumulative sentence embedding to conduct ML operation. This section first introduces 4 word embedding approaches to be investigated in this paper, including TF-IDF N-gram, Word2vector, GloVe, and BERT. Then 10 ML/DL classifiers are presented.

Word embedding approaches

Four word embedding approaches are discussed.

N-gram (Suen 1979)

It is a distinguished language feature extraction method. Due to its outstanding performance in dealing with sequence information, N-gram has been used in text feature extraction and classification fields and also achieved great success. N-gram utilizes a sliding window to divide a sequence into n-slice parts. After counting the term frequency-inverse document frequency (TF-IDF) and One-Hot embedding, we obtain a sequence embedding. As illustrated in Fig. 2, the red box is a sliding window whose sizes are 2, 3, and 4. Then the Chinese Word (CW) sentence of our corpus will be mapped into a vector.

Word2vec (Mikolov et al. 2013)

It is a neural network-based algorithm for training word vectors. It has two types of architecture. One is the Continuous Bag-Of-Words (CBOW) model, and the other is the continuous skip-gram model. CBOW is similar to Feedforward Neural Net Language Model (Bengio et al. 2000), where the non-linear hidden layer is removed, and the projection layer is shared for all words. After the training converges, words with similar meanings are mapped to a similar position in the vector space (illustrated in Fig. 3).

GloVe (Pennington et al. 2014)

It was proposed as a global vector for the word embedding model in 2014. This model combines the advantages of global matrix factorization and local context window methods and efficiently leverages the statistical information of a large corpus. After training on the non-zero elements in the word-word co-occurrence matrix, GloVe will produce a vector space with meaning in a fixed dimension. Figure 4 discloses the flow of GloVe training. We put corpus as input. Then we count CW term frequency and compute the co-occurrence matrix to train GloVe using proper hyper-parameters. At last, we obtain the word embedding result with a specific dimension.

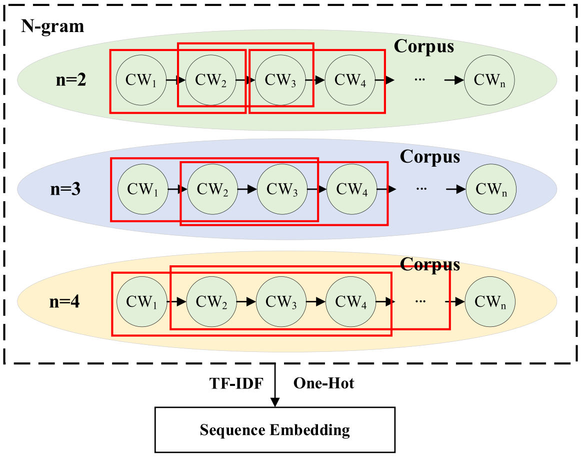


Fig. 2 The flow of TF-IDF N-gram embedding

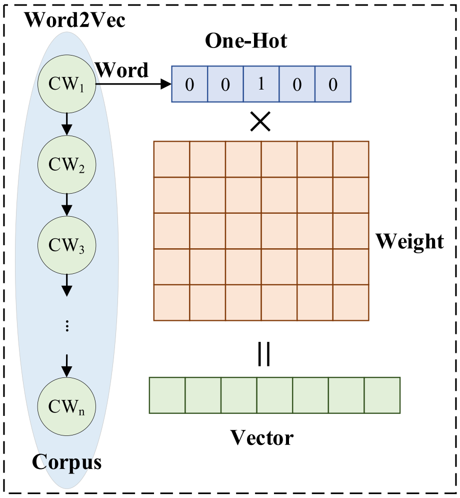


Fig. 3 The flow of Word2vec embedding

BERT

Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al. 2018) considers the bidirectional contexts and achieves denoising autoencoding-based model pre-training. It performs better than pre-training methods based on autoregressive language modeling (Yang et al. 2019). As illustrated in Fig. 5, if we input our corpus, each CW will obtain a token embedding, a sentence embedding, and a position embedding. Then all of them have to be put in two layers bidirectional transformer. After that, the contextual representation will be output as a specific dimension vector for the following training.

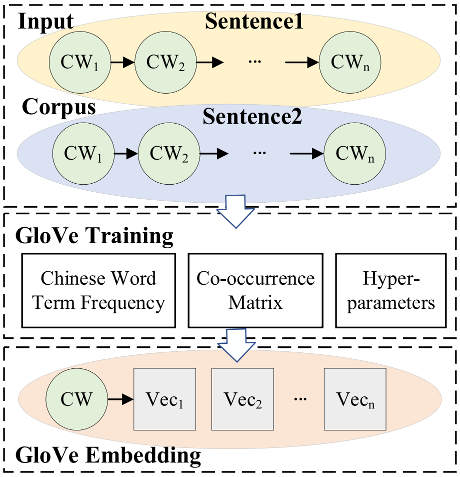


Fig. 4 The flow of GloVe embedding

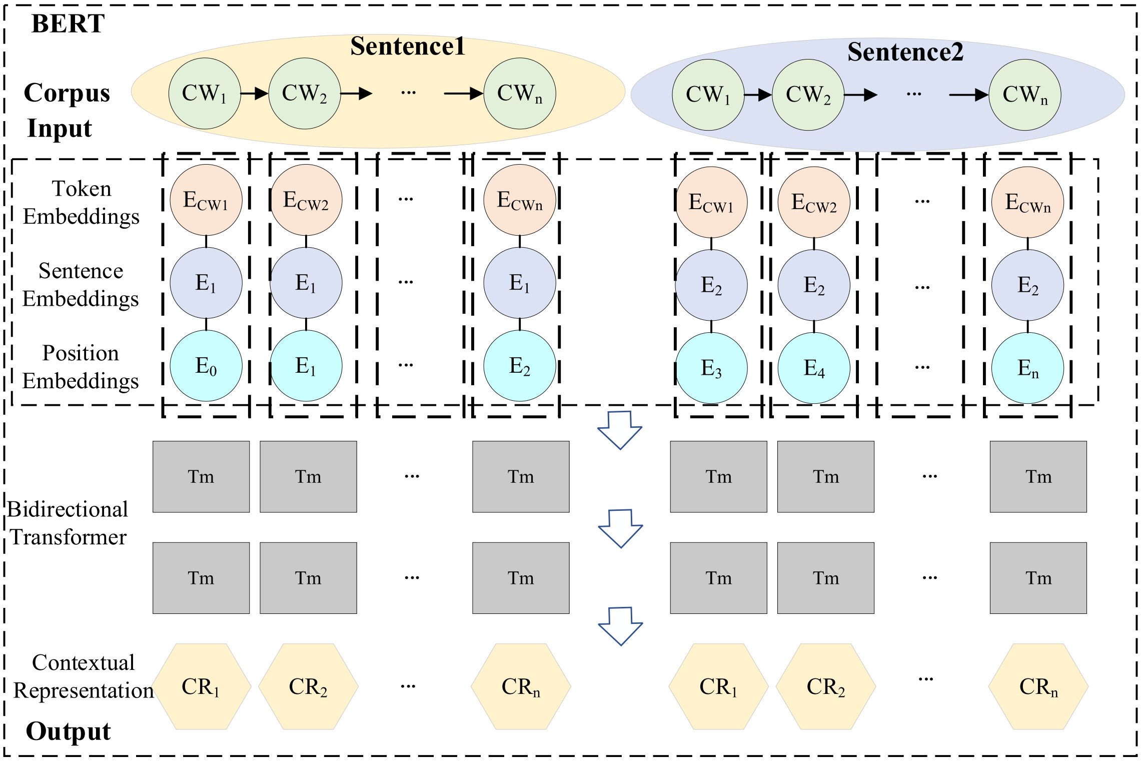


Fig. 5 The flow of BERT embedding

Classifiers

RF (Breiman 2001)

This classifier is based on ensemble learning and involves many independent decision trees. It uses bootstrap to extract samples as input and combines each decision tree classification result. Then RF gains the classification result via majority voting. In fact, it overcomes the over-fitting of a single tree by taking the average of multi predictions.

SVM (Cortes and Vapnik 1995)

SVM maps input vectors non-linearly to high dimension feature space, which builds a hyperplane. It aims at maximizing the margin between the two sides of a separating hyperplane.

KNN (Sebastiani 2002)

KNN is a widely used text classifier due to its simplicity and efficiency. It computes the nearest neighbors of each point by majority vote to classify.

MLP (Rumelhart et al. 1986)

MLP is a feedforward artificial neural network model. Given a set of features, MLP can learn a non-linear function approximator for classification.

RNN (Elman 1990)

RNN is a kind of neural network and is effective in processing sequence text data classification. Unlike feedforward neural networks, RNN can recurrent in the self-network to obtain a better sequence representation.

AB (Freund and Schapire 1997)

A new weak classifier is added in each AB training round until the predetermined error rate is reached. Each training sample is assigned a weight indicating the probability that it is selected into the training set by a classifier.

GBDT (Friedman 2001)

GBDT classifier is composed of multiple decision trees, and the conclusion of all trees adds up to the final classification result. Notably, the previous decision tree's residual is taken as the next decision tree's input.

DT (Breiman et al. 2017)

DT is a non-parametric supervised learning method used by the classifier. It utilizes a set of if-else decision rules to learn from data. Therefore, DT is simple and easy to understand and interpret.

ET (Geurts et al. 2006)

This classifier implements many randomized decision trees on various sub-samples and uses averaging to improve the predictive accuracy and control over-fitting.

VOTE

The VOTE classifier is an ML model that trains on an ensemble of numerous models and predicts an output based on the highest probability of chosen class as the output. It will simply aggregate the result of each classifier and predict the output based on the highest majority of voting. Instead of creating separate dedicated models and finding the accuracy for each classifier, VOTE will create a single model which trains by these models and predicts output based on their combined majority of voting for each output.

Fault record dataset

In this section, we first introduce one example of raw data. Then, we conduct raw data analysis, including work order source, top 10 cities or provinces of fault recordings, and the relationship between month and fault record amount. At last, we build a fault record dataset for subsequent studies.

Raw data sample

We collect the 8,481 raw data from an actual Internet of Vehicles platform service center from June to December 2021. Intuitively, we give one example of raw data in Table 1, which includes pile number, work order source, work date, work city, work order number, client type, fault description, work order state, fault category, and fault reason. Notably, we use 'xxx' to represent the actual number considering data privacy.

Raw data analysis

We analyze the raw data. We observe that 53.8% of work orders are sourced from the national service hotline, 21.9% from the EV fixed line, and 8.1% from WeChat mini program. The detailed plot is given in Fig. 6, indicating that more charging-pile users feedback faults via traditional trouble calls.

Moreover, we analyze the source of the work order. The top 10 provinces of fault records are denoted as P1, P2,..., and P10, respectively. As shown in Fig. 7, we can obtain the relationship between the region and fault records. For instance, more fault records demonstrate that more charging piles of EVs are deployed in a specific region. P1–P5 are all developed areas of China and possess more EVs than other growing provinces.

We collect fault records from June to December 2021 (demonstrated in Fig. 8), and we observe that with the increase of the month, more fault records have been reported.

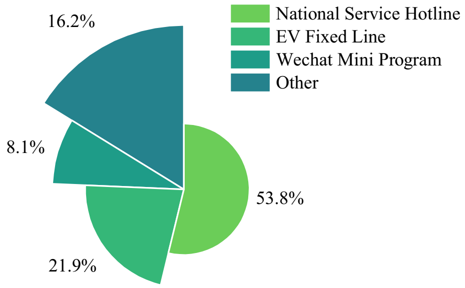


Fig. 6 Work order source

Table 1 One example of raw data

Data item	Data sample
Pile number	00xxxxxxxxxxxxxxxxxx16
Work order source	WeChat Mini Program
Date	6/22/2021
City/province	P1 (P = Province)
Work order number	2xxxxx1*
Client type	Private person
Fault description	The customer complained that there was something wrong with the display time of charging orders of individual piles. Now the time of charging orders is displayed on January 1, 2020
Work order state	Done
Fault category	Charging-pile program fault
Fault reason	Program time disorder

*Due to privacy reasons, we use 'xxx' to take the place of the actual number

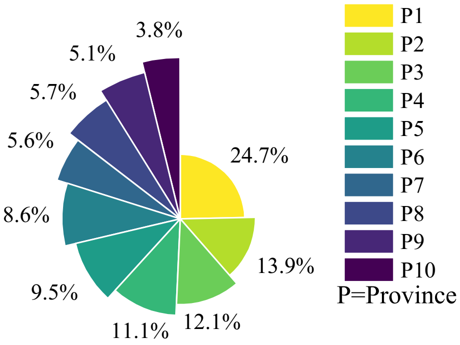


Fig. 7 Top 10 provinces of fault records

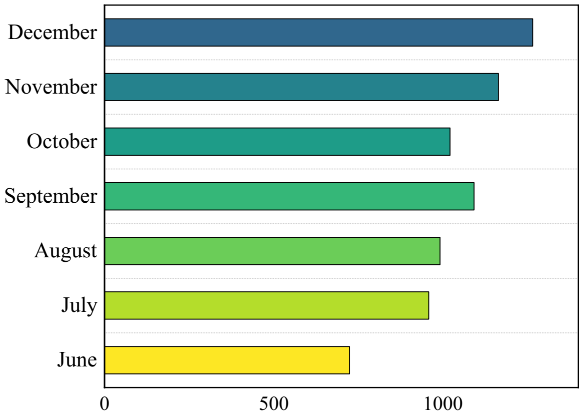


Fig. 8 The fault record number of each month

Fault record dataset

After raw data analysis, we explore the concrete usage using fault records. Hence, we establish a fault record dataset containing the label, fault category, and fault description. As stated in Table 2, labels 0 to label 7 respectively correspond to the different fault categories, including installation error fault, charging-pile mechanical fault, charging-pile program fault, user personal fault, signal fault (offline), pile compatibility fault, charging platform fault, and other faults. Moreover, we give one fault description sample for each label and category in Table 2.

Notably, as shown in Table 3, in this paper, we focus on Chinese text classification and prediction to preserve the original data characteristics.

Experimental result and discussion

In this section, we concentrate on experimental settings and results. Firstly, data preprocessing and data splitting are given. Then, we introduce the ML or DL classifiers and experimental dependency used in this paper. In addition, the metrics of the experiment are represented. At last, we give the experimental result and interpretability discussion.

Data preprocessing

For extracting features of the Chinese text, we utilize the *Jieba* (Junyi 2022) as the tokenizer to cut the whole sentence of Chinese text into several segmentations. As stated in Table 4, we count the ten most frequent words in our dataset.

After that, we use N-gram, GloVe, Word2vec (CBOW model), and BERT as embedding approaches to convert Chinese Word segmentation (CW) to a multi-dimension vector. In Table 5, we give a sample of word2vec. In addition, we split our dataset into the training and testing sets for classifier training. The training set occupies 80% of all data, and the testing set possesses a 20% dataset, as described in Table 6. At last, we convert the CW of our corpus into 300 dimensions in GloVe, 20 in Word2Vec, and 768 in BERT. We set different dimensions to discuss

Table 2 Fault record dataset

Label	Fault category	Fault description*
0	Installation error fault	The customer said that he had asked the power supply staff to check the problem, and the staff said that the charging-pile installation personnel connected the ground wire wrong and please deal with it
1	Charging-pile mechanical fault	The customer found that when charging yesterday, the charging gun was hot, the car's charging port was hot, and the switch was hot. Power supply to charging pile about 5 m
2	Charging-pile program fault	The customer complained that there was something wrong with the display time of charging orders of individual piles. Now the time of charging orders is displayed on January 1, 2020
3	User personal fault	The indicator light of the charging pile is not on. This pile is transferred to the customer by others
4	Signal fault (offline)	The customer reported that the personal order pile failed to charge, and the yellow light was steady
5	Pile compatibility fault	The customer reported that the charging could not be started, the charging timeout occurred, the charging stopped, the background check was offline, and the field signal was excellent
6	Charging platform fault	After logging in to the mobile APP, the customer could not find the orderly pile and his orderly charging pile
7	Other faults	The circuit breaker leak protection trip of the upper section of the charging pile has been installed for three or four months

*The fault description we preprocessed is recorded in Chinese, not English

Table 3 Data amount of fault record dataset

Label	Fault category	Number of data sample
0	Installation error fault	706
1	Charging-pile mechanical fault	1712
2	Charging-pile program fault	1022
3	User personal fault	2690
4	Signal fault (offline)	1707
5	Pile compatibility fault	48
6	Charging platform fault	351
7	Other faults	245

Table 4 The most 10 frequent words in dataset

Chinese word segmentation	Term frequency
Charging	8246
Customer	6628
Pile	6546
Reflect	3410
With order	3091
Offline	2565
Personal	2291
Unable	2160
Fault	1856
User	1142

the relationship between the model performance and the word embedding dimension.

Data augmentation

As aforementioned in "Raw data sample" section, we collect 8,481 raw samples. However, a limited data scale will cause a higher error rate for ML models. The paraphrasing-based method is one of the effective data augmentation approaches in NLP (Geurts et al. 2006). In this paper, we utilize python library *synonyms* (Bengio et al. 2000) to find and replace the synonym of tokenizing fault description. Totally, we utilize 16,962 samples for ML training.

To be specific, we obtain all replaceable words for each sample, and randomly select a few words to replace. The more similar to the original word, the more likely it is to be selected. Plenty of synonym examples will be revealed in Table 7. Notably, our fault description is recorded in Chinese, so we give the Chinese version of the synonym to demonstrate the high similarity between the token and the synonym.

Experimental goal

In this paper, we utilize 4 embedding models, including N-gram, GloVe, Word2vec, and BERT, and use 10 ML or DL classifiers, including Random Forest (RF), Support Vector Machine (SVM), K-Nearest Neighbor (KNN), Multilayer Perceptron (MLP), Recurrent Neural Network (RNN), AdaBoost (AB), Gradient Boosted Decision Tree (GBDT), Decision Tree (DT), Extra Tree (ET), and VOTE, to classify our corpus.

We implement extensive experiments to explore the best-matched combination between word embedding models (N-gram, GloVe, Word2vec, and BERT) and classifiers. In general, we need to base the following goals.

Goal 1: The training time (including embedding training time and classifier training time) must be controlled in several seconds.

Goal 2: The combination of the embedding model and the classifier can achieve high accuracy in a real-world dataset.

Goal 3: The embedding model and classifier should be interpretable.

Experimental configuration

In this subsection, the experimental configuration of our experiments is given. Our experiments run in AMD R7 5800X platform with 32 GB of RAM, which is eight cores CPU, and we run RNN using NVIDIA GeForce RTX 3080 for accelerating neural network.

As described in Table 8, we use python 3.7.13 with a lot of python libraries to help model training. In addition, we utilize standard GloVe (Yang et al. 2019) in Ubuntu 18.04 LTS to train the word vectors using our corpus. Similarly, we give the hyper-parameters of each classifier in Table 9. Note that, since hyper-parameters have a large impact on each model, we try to choose the default parameters in Scikit-learn.

Table 5 The sample of Word2vec

Word	Word2vec (20 dimensions)
Customer	[– 0.02485732 0.14806207 – 0.35677359 – 0.57840854 – 0.35948697 – 0.9146083 – 0.50397265 2.2205336 0.29582977 1.1330733 1.2003825 – 0.5351351 – 1.7470182 0.63969433 0.6082744 – 1.0082941 3.0654325 – 0.41733867 – 0.3103616 – 1.3387867]

Table 6 Splitting result of dataset

Label	Training set	Testing set	Sum
0	565	141	706
1	1370	342	1712
2	818	204	1022
3	2152	538	2690
4	1366	341	1707
5	38	10	48
6	281	70	351
7	196	49	245
Sum	6786	1695	8481

Table 7 The synonym examples

Token		Synonym	
English	Chinese	English	Chinese
Call up	打电话	Call	来电
Customer	客户	User	用户
Fault	故障	Breakdown	损坏
Start	启动	Restart	重启
Blackout	断电	Powercut	停电
Insert	插入	Load	装入
Facilities	设施	Equipment	设备
Verify	核实	Confirm	确认
Repair	维修	After sales	售后

Metric

4 standard ML metrics, precision, recall, accuracy, and F1-score, to evaluate the performance of each model combination. For each sample in the dataset, there are four possible partitioning outcomes:

> TP (True Positive): Number of samples belonging to and classified as a positive class;

Table 8 The environment and corresponding libraries

Environment	Libraries
Python 3.7.13	Pandas 1.3.5
	Numpy 1.21.5 Jieba 0.42.1 Gensim 4.1.2 Imbalance-learn 0.9.0
	Scikit-learn 1.0.2 Pydotplus 2.0.2 Pytorch 1.12.0 Cudatoolkit 11.3.1 Transformers 4.20.1
GloVe 1.2	–
BERT (pre-trained)	Chinese_wwm_ext_pytorch
Word2vector	Gensim.models.word2vec (in gensim 4.1.2)

> FP (False Positive): Number of samples belonging to a negative category and classified as a positive category;
> FN (False Negative): Number of samples belonging to a positive category and classified as a negative category;
> TN (True Negative): Number of samples in the negative category and classified as negative.

Then the precision (Eq. (1)), recall (Eq. (2)), accuracy (Eq. (3)), and F1-score (Eq. (4)) of each class can be calculated respectively as follows:

$$precision = \frac{TP}{TP + FP} \tag{1}$$

$$recall = \frac{TP}{TP + FN} \tag{2}$$

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3}$$

Table 9 The hyper-parameters of classifiers

Classifier	Hyper-parameters
RF	n_estimators = 500, n_jobs = -1
SVM	kernel = rbf, decision_function_shape = ovr, max_iter = 1000
KNN	n_neighbors = 8
MLP	activation = relu, solver = adam, momentum = 0.9, learning_rate_init = 0.001, random_state = 1
RNN	loss = CrossEntropy, hidden_dim = 128, layer = 2, optimizer = adam
AB	n_estimators = 500
GBDT	n_estimators = 500, learning_rate = 0.1, max_depth = 1, random_state = 1
DT	criterion = gini
ET	criterion = gini
VOTE	voting = hard, including six classifiers; same hyper-parameters as RF, SVM, KNN, MLP, AB, and GBDT

Table 10 The time in training embedding models (s)

	N-gram (n = 2)	GloVe	Word2vec	BERT
Train time	1544.36	7.50	0.22	Pretrained Model

$$F1 - score = \frac{2 \cdot precision \cdot recall}{precision + recall} \tag{4}$$

Experimental results

In this section, we will follow the experimental goals (in "Data augmentation" section) to explore the best match of embedding models and classifiers. Firstly, we give the training time for different embedding models and classifiers. Then, we discuss the effectiveness of imbalance learning. In addition, we provide the accuracy, precision, recall, F1-score, and average train time of different combinations to evaluate the model combination performance. At last, we conduct the interpretability discussion to give the final analysis.

Training time comparison

Table 10 gives training time for 4 embedding models. We observe that the N-gram (n = 2) training time is 1544.36 s, while GloVe and Word2vec are 7.50 s and 0.22 s, respectively. We use the pre-trained model- 'chinese_wwm_ext_pytorch' in BERT.

We then compare the training time and accuracy of 16 combinations of 4 embedding models and 4 classifiers. The results are given in Tables 11, 12, respectively. The N-gram approach not only consumes more training time but also has lower accuracy in RF, KNN, and DT classifiers. The MLP + N-gram achieves an accuracy of 76%. However, the training time is unacceptable under **Goal 1**, and the accuracy rate cannot reach **Goal 2**. Therefore, we only select GloVe, Word2vec, and BERT in the following experiments in "Experimental results" section.

Imbalance learning comparison

Observing our fault record dataset (in Table 3), we find a big difference in the number of data samples for seven classes, which means the dataset is imbalanced. With this in mind, we try to utilize python library-imbalance learn to reduce the effect of the imbalance dataset. As described in Table 13, we record the accuracy under imbalance and non-imbalance learning in 10 classifiers.

After comparing the experimental results with and without imbalance learning in different classifiers and embedding models, we observe that the improvement of imbalance learning is little. Moreover, as recorded in Table 14, adopting imbalance learning consumes more training time. Hence, we only adopt non-imbalance learning processing in the following experiments to satisfy **Goal 1**.

Table 11 Training time of 16 combination of 4 embedding models and 4 classifiers (s)

Embedding models	RF	KNN	MLP	DT
N-gram (n = 2)	27.35	2.25	196.71	8.28
GloVe	0.91	0.23	4.01	0.10
Word2Vec	1.00	0.19	3.51	0.05
BERT	1.11	0.23	3.87	0.09

Performance comparisons

This subsection will give the overall performance comparisons from the perspective of accuracy rate, precision rate, recall rate, F1-score, and average training time.

Firstly, we provide the accuracy of different classifiers under GloVe, Word2vec, and BERT embedding models. As shown in Fig. 9, the four classifiers have better accuracy under three embedding models, including RF, RNN, DT, and ET. Especially, RF and RNN classifiers achieve the top 2 accuracy, for instance, RF + GloVe 79.67%, RF + Word2vec 81.26%, RF + BERT 80.32%, RNN + GloVe 82.91%, RNN + Word2vec 78.26%, and RNN + BERT 81.85%. However, to satisfy **Goal 1**, from the perspective of average training time, RNN reaches the highest time consuming, which more than 172 s. The detailed precision, recall, and F1-score results are shown in Table 15. Note that we bold metrics which are more than 79% to emphasize the performance of classifiers.

Hence, to satisfy **Goal 1** and **Goal 2**, we select RF as the most appropriate classifier for the fault diagnosis task. Figure 10 shows the confusion matrix of RF + Glove.

Performance with data augmentation

As mentioned in "Data augmentation" section, we implement data augmentation to expand our data scale for better ML performance. With more training samples, we improve our model performance. We illustrate the accuracy and average train time in Fig. 11. Compared with performance without data augmentation, we calculate

Table 12 Accuracy (%) of 16 combination of 4 embedding models and 4 classifiers

Embedding models	RF	KNN	MLP	DT
N-gram (n = 2)	71.48	31.47	75.84	58.40
GloVe	79.67	42.02	45.43	78.90
Word2Vec	81.26	43.25	33.88	77.90
BERT	80.32	42.72	41.31	78.43

Table 13 The accuracy result with and without imbalance learning (%)

Classifiers*		GloVe	Word2vec	BERT
RF	0	79.67	81.26	80.32
	1	78.96	80.44	80.26
SVM	0	27.58	31.23	30.47
	1	37.65	29.64	31.05
KNN	0	42.02	43.25	42.72
	1	42.02	42.07	41.78
MLP	0	45.43	33.88	41.31
	1	42.96	36.12	39.54
RNN	0	82.91	78.26	81.85
	1	81.91	80.02	81.38
AB	0	23.75	24.04	22.27
	1	24.75	26.69	27.28
GBDT	0	42.31	41.37	43.84
	1	41.37	40.42	43.08
DT	0	78.90	77.90	78.43
	1	76.72	77.67	76.90
ET	0	76.72	76.72	77.14
	1	77.14	76.78	76.84
VOTE	0	52.80	51.74	56.28
	1	54.10	51.97	57.22

*1 = imbalance learning, 0 = non-imbalance learning

Table 14 The training time with and without imbalance learning (s)

Classifier*		GloVe	Word2vec	BERT
RF	0	0.88	1.00	1.12
	1	1.00	0.88	1.25
SVM	0	4.72	2.27	9.53
	1	5.86	2.89	11.75
KNN	0	0.23	0.18	0.24
	1	0.24	0.19	0.30
MLP	0	4.18	3.62	3.82
	1	3.36	2.82	5.40
RNN	0	133.86	42.28	340.02
	1	133.49	42.42	354.34
AB	0	7.88	2.78	18.25
	1	9.13	3.19	25.75
GBDT	0	35.85	17.30	83.37
	1	42.43	20.84	140.38
DT	0	0.08	0.04	0.11
	1	0.10	0.06	0.15
ET	0	0.02	0.01	0.04
	1	0.03	0.01	0.07
VOTE	0	53.92	27.40	122.15
	1	61.54	30.76	176.98

*1 = imbalance learning, 0 = non-imbalance learning

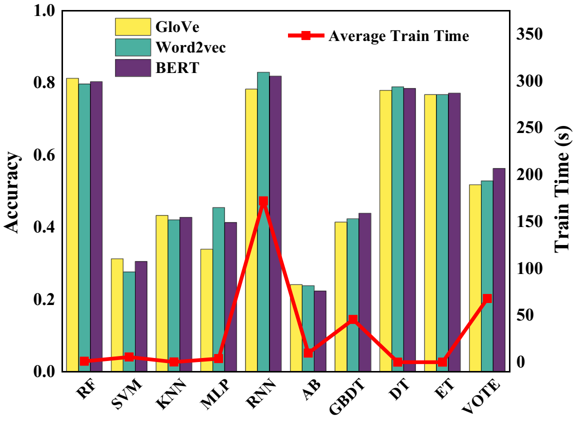
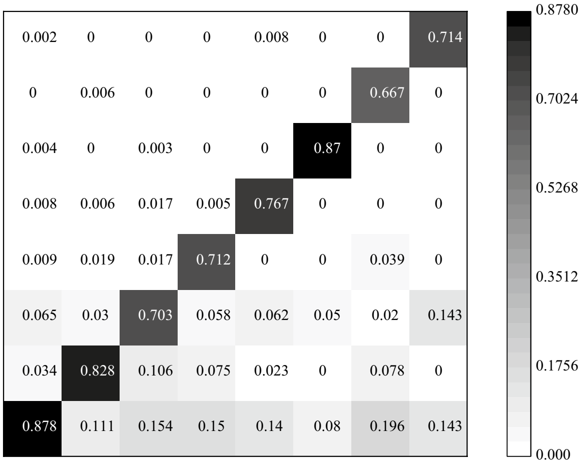


Fig. 9 The accuracy and average training time result from different combinations

Table 15 Precision, recall, and F1-score result (%)

Classifiers	Metric*	GloVe	Word2vec	BERT
RF	P	**80.34**	**82.21**	**81.41**
	R	**79.67**	**81.26**	**80.32**
	F	**80.00**	**81.73**	**80.86**
SVM	P	42.00	44.75	45.47
	R	27.58	31.23	30.47
	F	33.29	36.79	36.49
KNN	P	41.05	42.34	41.19
	R	42.02	43.25	42.72
	F	41.52	42.79	41.94
MLP	P	47.28	33.14	41.96
	R	45.43	33.88	41.31
	F	46.34	33.51	41.63
RNN	P	**82.98**	78.44	**82.06**
	R	**82.91**	78.26	**81.85**
	F	**82.94**	78.35	**81.96**
AB	P	32.94	31.66	28.89
	R	23.75	24.04	22.27
	F	27.60	27.33	25.15
GBDT	P	43.32	44.83	43.91
	R	42.31	41.37	43.84
	F	42.81	43.03	43.88
DT	P	78.96	78.01	78.77
	R	78.90	77.90	78.43
	F	78.93	77.96	78.60
ET	P	76.94	76.86	77.40
	R	76.72	76.72	77.14
	F	76.83	76.79	77.27
VOTE	P	61.23	58.60	60.94
	R	52.80	51.74	56.28
	F	56.70	54.95	58.51

*There are three metrics including precision, recall and F1-score. We use *P* to denote precision, *R* to denote recall, and *F* to denote F1-score



0.002	0	0	0	0.008	0	0	0.714
0	0.006	0	0	0	0	0.667	0
0.004	0	0.003	0	0	0.87	0	0
0.008	0.006	0.017	0.005	0.767	0	0	0
0.009	0.019	0.017	0.712	0	0	0.039	0
0.065	0.03	0.703	0.058	0.062	0.05	0.02	0.143
0.034	0.828	0.106	0.075	0.023	0	0.078	0
0.878	0.111	0.154	0.15	0.14	0.08	0.196	0.143

Fig. 10 The confusion matrix of RF + GloVe

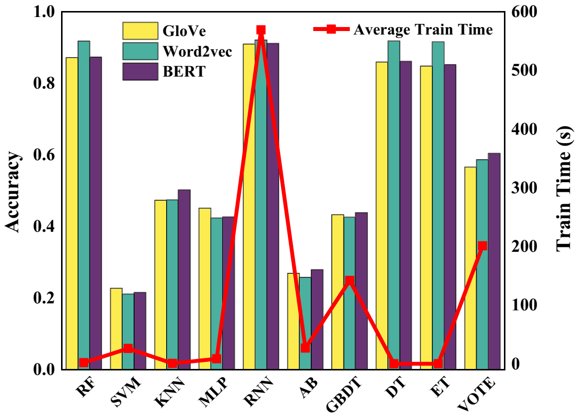


Fig. 11 The accuracy and average training time result with data augmentation

the statistical results in Table 16, where the positive improve average are in bold. Notably, all classifiers have been improved except SVM. The possible reason is that we maintain the same hyper-parameters as in Table 9 of each classifier, and SVM is more sensitive with proper hyper-parameters.

Interpretability discussion

To achieve **Goal 3**, we need to analyze the model's interpretability. Compared with black box DL models, such as RNN, most traditional ML models have better interpretability. In addition, Word2vec and BERT are neural network-based word embedding models, while GloVe utilizes the co-occurrence matrix and term frequency of corpus to train the embedding vector. In other words, GloVe does not involve a neural network and has better interpretability.

Table 16 Accuracy result (%)

Classifiers	Improve average	GloVe	Word2vec	BERT
RF	**8.36**	87.21	91.81	87.33
SVM	− 7.96	22.72	21.10	21.57
KNN	**5.66**	47.33	47.42	50.22
MLP	**3.17**	45.12	42.35	42.65
RNN	**10.40**	90.95	92.10	91.16
AB	**3.50**	26.88	25.76	27.94
GBDT	**0.74**	43.30	42.59	43.86
DT	**9.58**	85.97	91.87	86.15
ET	**10.37**	84.85	91.63	85.21
VOTE	**4.98**	56.62	58.68	60.45

We utilize python library pydotplus to visualize the RF classification. We provide one of 500 RF trees, in which the number of the training sample is 200, and the embedding model is GloVe in Fig. 12, to illustrate the interpretability of RF + GloVe.

In brief, to satisfy **Goal 1**, **Goal 2**, and **Goal 3**, we select the RF classifier and GloVe word embedding model to finish the fault diagnosis task.

Result discussion

Why the accuracy of the raw data with data preprocessing is low? We believe that they are two main reasons: little raw data scale and irregular manual fault description. As is known to all, a larger data scale will help the model learn more features. Besides, the real-world dataset is recorded by customer service staff which is irregular and casual, which will immensely affect the performance of classification. In fact, when we replace some tokens and make data augmentation using synonyms which are regular descriptions, we achieve greatly improving in accuracy metric.

What can we learn from the interpretability result? The interpretability of the model can reflect the logic of model classification and Fig. 12 shows the logic of one RF tree. Worse interpretability, such as RNN, is a black box for the whole training process and is unacceptable for critical infrastructures.

What can we conclude from the different performances of models? From the extensive experimental results, the RNN, RF, DT, and ET have superior model performance. Except for RNN which has worse interpretability, RF, DT, and ET are the tree-based methods, which indicates the tree structure has good performance for fault classification and fault diagnosis. Besides, compared with other classification models, the tree structure is easier to adjust the hyper-parameters and achieve the best result.

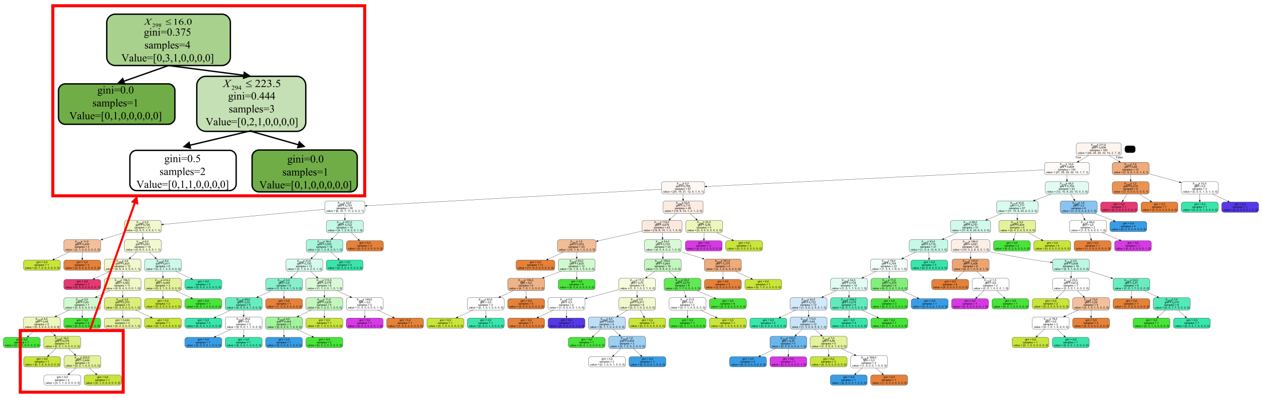


Fig. 12 One of the RF trees when the number of training samples is 200, and the embedding model is GloVe

Conclusion

With the development of electric vehicles (EVs), many charging piles as the supporting facility have been deployed. This paper mainly focuses on fault diagnosis to maintain the effectiveness of charging piles. Specially, we vectorize fault description of the real-world fault record dataset using N-gram, GloVe, Word2vec, and BERT embedding models. Then we utilize ten machine learning or deep learning classifiers, including Random Forest (RF), Support Vector Machine (SVM), K-Nearest Neighbor (KNN), Multilayer Perceptron (MLP), Recurrent Neural Network (RNN), AdaBoost (AB), Gradient Boosted Decision Tree (GBDT), Decision Tree (DT), Extra Tree (ET), and VOTE, to explore the best-matched embedding model and classifier for helping charging-pile fault diagnosis.

Our extensive experiments reveal that RF classifier working with the GloVe embedding model in the real-world dataset can achieve the best accuracy with low training time. At last, we discuss the interpretability of RF and GloVe.

Abbreviations

EV	Electric vehicle
ML	Machine learning
DL	Deep learning
TF-IDF	Term frequency-inverse document frequency
CW	Chinese Word Segmentation
RF	Random Forest
SVM	Support Vector Machine
KNN	K-Nearest Neighbor
MLP	Multilayer Perceptron
RNN	Recurrent Neural Network
AB	AdaBoost
GBDT	Gradient Boosted Decision Tree
DT	Decision Tree
ET	Extra Tree
CBOW	Continuous Bag-Of-Words

Acknowledgements

This work was supported by the State Grid Technology Project "Research on Interaction between Large-scale Electric Vehicles and Power Grid and Charging Safety Protection Technology" (5418-202071490A-0-0-00) from State Grid Corporation of China.

Author contributions

Drafting the manuscript: JW and XC. Revising the manuscript critically for important intellectual content: WW, XC, YY, CX, SY, MW, and LW. Experiments deployment: JW and LL. All authors read and approved the final manuscript.

Authors' Information

Wen Wang is currently working as the Deputy General Manager of State Grid Electric Vehicle Service Company. He is Professorial Senior Engineer; Expert of China National Key R&D Program Review Group; Secretary-General of IEEE PES EV Satellite Committee (China); Senior Member of National Electrical System Security Protection Expert Group; Specialist in the field of power system automation, power trading and information security.

Jianhua Wang received the B.S. degree and M.S. degree in Software engineering from Taiyuan University of Technology in 2017 and 2020. He now pursues for his PhD degree in Beijing Jiaotong University, major in Cyberspace Security. His research interests include adversarial machine learning and federated learning.

Xiaofeng Peng is currently working as the V2G department head of State Grid EV Service Co., Senior Engineer; His research interests include V2G, Load Aggregation Technology.

Ye Yang is currently working as the R&D scientist of State Grid EV Service Co., Senior Engineer; His research interests include AI, Block-chain, smart grid control technology.

Chun Xiao is currently working as the senior engineer of State Grid Shanxi Marketing Service Center, and the specialist in marketing service.

Shuai Yang is currently working as the senior engineer of State Grid Shanxi Marketing Service Center, and the specialist in marketing service.

Mingcai Wang is currently working as the senior engineer of State Grid Electric Vehicle Service Company, Ltd. He is the specialist in the field of power system automation.

Lingfei Wang is currently working as the senior engineer of State Grid Electric Vehicle Service Company, Ltd. He is the specialist in power trading and control of electric power system.

Lin Li is currently an Associate Professor with the School of Computer and Information Technology, Beijing Jiaotong University. Her current research interests include cryptographic protocols, privacy preserving, and federated learning.

Xiaolin Chang (Member, IEEE) is a professor at the School of Computer and Information Technology, Beijing Jiaotong University. Her current research interests include Edge/Cloud computing, Network security, security and privacy in machine learning. She is a senior member of IEEE.

Funding

This work was supported by the State Grid Technology Project "Research on Interaction between Large-scale Electric Vehicles and Power Grid and Charging Safety Protection Technology" (5418-202071490A-0-0-00) from State Grid Corporation of China..

Availability of data and materials
Not applicable.

Declarations

Ethics approval and consent to participate
Not applicable.

Competing interests
No potential conflict of interest was reported by the authors.

Received: 21 October 2022 Accepted: 8 January 2023
Published online: 04 April 2023

References

Bengio Y, Ducharme R, Vincent P (2000) A neural probabilistic language model. In: Advances in neural information processing systems, vol 13. https://proceedings.neurips.cc/paper/2000/hash/728f206c2a01bf572b5940d7d9a8fa4c-Abstract.html. Accessed 29 July 2022

Breiman L (2001) Random forests. Mach Learn 45(1):5–32

Breiman L, Friedman JH, Olshen RA, Stone CJ (2017) Classification and regression trees. Routledge, New York. https://doi.org/10.1201/9781315139470

Chen T et al (2020) A review on electric vehicle charging infrastructure development in the UK. J Mod Power Syst Clean Energy 8(2):193–205. https://doi.org/10.35833/MPCE.2018.000374

Cortes C, Vapnik V (1995) Support-vector networks. Mach Learn 20(3):273–297. https://doi.org/10.1007/BF00994018

Devlin J, Chang M-W, Lee K, Toutanova K (2018) Bert: pre-training of deep bidirectional transformers for language understanding. ArXiv:181004805

Du J, An W, Zhou M, Mao W, Huang G, Deng S (2021) Research on fault diagnosis method of DC charging pile based on deep learning. In: 2021 11th international conference on power and energy systems (ICPES), pp 426–431. https://doi.org/10.1109/ICPES53652.2021.9683841

Elman JL (1990) Finding structure in time. Cogn Sci 14(2):179–211. https://doi.org/10.1207/s15516709cog1402_1

Freund Y, Schapire RE (1997) A decision-theoretic generalization of on-line learning and an application to boosting. J Comput Syst Sci 55(1):119–139. https://doi.org/10.1006/jcss.1997.1504

Friedman JH (2001) Greedy function approximation: a gradient boosting machine. Ann Stat 29(5):1189–1232

Gao D-X, Hou J-J, Liang K, Yang Q (2018) Fault diagnosis system for electric vehicle charging devices based on fault tree analysis. In: 2018 37th Chinese control conference (CCC), pp 5055–5059. https://doi.org/10.23919/ChiCC.2018.8482691

Gao D, Lv Y, Sun Y, Wang Y, Yang Q (2020) Remote mobile monitoring and fault diagnosis system for electric vehicle circular charging device based on cloud platform. In: 2020 Chinese control and decision conference (CCDC), pp 4146–4151. https://doi.org/10.1109/CCDC49329.2020.9164523

Geurts P, Ernst D, Wehenkel L (2006) Extremely randomized trees. Mach Learn 63(1):3–42. https://doi.org/10.1007/s10994-006-6226-1

Hao L, Wan F, Ma N, Wang Y (2018) Analysis of the development of WeChat mini program. J Phys Conf Ser 1087:062040. https://doi.org/10.1088/1742-6596/1087/6/062040

Junyi S (2022) jieba. Accessed 28 July 2022 https://github.com/fxsjy/jieba

Li Y, Ji X, Jiang D, Meng T (2021) Abnormal detection system design of charging pile based on machine learning. IOP Conf Ser Earth Environ Sci 772(1):012058. https://doi.org/10.1088/1755-1315/772/1/012058

Mikolov T, Chen K, Corrado G, Dean J (2013) Efficient estimation of word representations in vector space. ArXiv:13013781

Pennington J, Socher R, Manning CD (2014) Glove: global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pp 1532–1543

Rumelhart DE, Hinton GE, Williams RJ (1986) Learning representations by back-propagating errors. Nature 323(6088):6088. https://doi.org/10.1038/323533a0

Sebastiani F (2002) Machine learning in automated text categorization. ACM Comput Surv 34(1):1–47. https://doi.org/10.1145/505282.505283

Shuai C, Sun Y, Zhang X, Yang F, Ouyang X, Chen Z (2022) Intelligent diagnosis of abnormal charging for electric bicycles based on improved dynamic time warping. IEEE Trans Ind Electron. https://doi.org/10.1109/TIE.2022.3206702

"stanfordnlp/GloVe." Stanford NLP https://github.com/stanfordnlp/GloVe. Accessed 28 July 2022.

Suen CY (1979) n-gram statistics for natural language understanding and text processing. IEEE Trans Pattern Anal Mach Intell PAMI-1(2):164–172. https://doi.org/10.1109/TPAMI.1979.4766902

Wang Q, Lu X, Yang D (2021) Fault diagnosis of DC–DC module of V2G charging pile based on fuzzy neural network. IOP Conf Ser Earth Environ Sci 772(1):012027. https://doi.org/10.1088/1755-1315/772/1/012027

Wei S-Y, Zhu Q, Li X-M, Meng X-H (2021) Research on comprehensive evaluation of electric vehicle charging failures. In: 2021 6th international conference on intelligent computing and signal processing (ICSP), pp 1255–1259. https://doi.org/10.1109/ICSP51882.2021.9408966

Yan Y, Li Q, Chen W, Su B, Liu J, Ma L (2019) Optimal energy management and control in multimode equivalent energy consumption of fuel cell/supercapacitor of hybrid electric tram. IEEE Trans Ind Electron 66(8):6065–6076. https://doi.org/10.1109/TIE.2018.2871792

Yang Z, Dai Z, Yang Y, Carbonell J, Salakhutdinov RR, Le QV (2019) XLNet: generalized autoregressive pretraining for language understanding. In: Advances in neural information processing systems, vol 32. https://proceedings.neurips.cc/paper/2019/hash/dc6a7e655d7e5840e66733e9ee67cc69-Abstract.html. Accessed 29 July 2022

Yong X, Ji W (2020) Research on detection and fault diagnosis technology of electric vehicle charging facilities. J Phys Conf Ser 1650(2):022100. https://doi.org/10.1088/1742-6596/1650/2/022100

Zhang L, Gao T, Cai G, Hai KL (2022) Research on electric vehicle charging safety warning model based on back propagation neural network optimized by improved gray wolf algorithm. J Energy Storage 49:104092. https://doi.org/10.1016/j.est.2022.104092

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.