Hindawi
Mathematical Problems in Engineering
Volume 2018, Article ID 9464971, 10 pages
https://doi.org/10.1155/2018/9464971

Research Article

Text Case-Based Reasoning Framework for Fault Diagnosis and Predication by Cloud Computing

Zhiwang Zhong,[1] Tianhua Xu ,[2] Feng Wang ,[2] and Tao Tang[2]

[1]*School of Electronic and Information Engineering, Beijing Jiaotong University, Beijing 100044, China*
[2]*State Key Laboratory of Traffic Control and Safety, Beijing Jiaotong University, Beijing 100044, China*

Correspondence should be addressed to Tianhua Xu; thxu@bjtu.edu.cn

Received 30 October 2017; Accepted 2 July 2018; Published 12 July 2018

Academic Editor: Ibrahim Zeid

In Discrete Event System, such as railway onboard system, overwhelming volume of textual data is recorded in the form of repair verbatim collected during the fault diagnosis process. Efficient text mining of such maintenance data plays an important role in discovering the best-practice repair knowledge from millions of repair verbatims, which help to conduct accurate fault diagnosis and predication. This paper presents a text case-based reasoning framework by cloud computing, which uses the diagnosis ontology for annotating fault features recorded in the repair verbatim. The extracted fault features are further reduced by rough set theory. Finally, the case retrieval is employed to search the best-practice repair actions for fixing faulty parts. By cloud computing, rough set-based attribute reduction and case retrieval are able to scale up the Big Data records and improve the efficiency of fault diagnosis and predication. The effectiveness of the proposed method is validated through a fault diagnosis of train onboard equipment.

1. Introduction

Discrete Event System (DES), such as railway onboard system, produces a large amount of text data by recording the maintenance process, which consists of symptoms corresponding to faulty parts, observed failure modes, and repair actions taken to fix the faults. Hundreds of thousands of such repair verbatim are collected and are used for health status estimation, fault detection, and fault diagnosis and predication. However, the overwhelming size of the repair verbatim data restricts an ability of its effective utilization in the process of fault diagnosis and predication. The main reason is that the complex task requires large computational efforts in order to identify the fault reason and optimal maintenance actions. This has stimulated researchers for advanced computing frameworks aimed at reducing the complexities of fault diagnosis and predication by using of Artificial Intelligence technologies and distributed processing architectures [1]. Through in-depth study and data mining to these Big Data, the life cycle of DESs is obtained and intelligent maintenance decision support including fault detection, fault location, and repair advices is provided. In this paper, we take a railway signalling system as an example of DES.

However, the task of automatic discovery of knowledge from the repair verbatim is a nontrivial exercise mainly due to the following reasons:

(1) Unstructured repair verbatim: in maintenance documents, the repair records are typically written in unstructured text, which also includes noisy textual data resulted from synonym, abbreviation, and error records. This presents a big challenge for text process.

(2) Massive repair verbatim: in maintenance documents, there are hundreds of thousands of repair verbatim collected during the diagnosis episodes.

(3) High-dimension data: in maintenance documents, there are tens of thousands of distinct terms or tokens in the view of text mining. After elimination of stop words and stemming, the set of features is still too large for many machine learning algorithms.

In order to tackle the above challenges, text mining based on knowledge discovery from historical datasets has recently been proposed. Text mining [2–4] is a knowledge-intensive task, which is gaining a wider attention in several Original Equipment Manufacturing (OEM) industries, for example,

aerospace, automotive, power plants, medical, biomedicine, manufacturing, and sales and marketing divisions. In fault diagnosis domain, Rajpathak et al. [5] propose an ontology based text mining method for automatically constructing and updating a D-matrix by mining hundreds of thousands of repair verbatim. Rajpathak et al. [6] also present a real-life reliability system by fusing the field warranty failure data with the failure modes extracted from unstructured repair verbatim data by using the ontology based natural language processing technique to facilitate accurate estimation of component reliability. Wang and Xu et al. [7] present a bilevel feature extraction-based text mining that integrates features extracted at both syntax and semantic levels with the aim of improving the fault classification performance for railway onboard equipment, considering the fact that, in a maintenance situation, the operators always search solution of fault diagnosis and predication problems that could be very similar to other states, which have been previously processed. In these cases, the corresponding fault diagnosis and predication solutions are expected to be correlated to these similar system states. Hence, the fault diagnosis and predication problem can be quickly computed by a Case-Based Reasoning (CBR) module, which tries to infer from historical information the hidden relationship between system states and the corresponding historical solutions [8, 9]. CBR is an effective technique for problem solving in the fields in which it is hard to establish a quantitative mathematical model, such as fault diagnosis, health management, or industrial systems [10]. Following this idea, He [11] proposes a framework to use text mining and Web 2.0 technologies to improve and enhance CBR systems for providing better user experience. He also suggests that text mining and Web 2.0 are promising ways to bring additional values to CBR and they should be incorporated into the CBR design and development process for the benefit of CBR users. In order to tackle the vast amount of maintenance records, cloud computing is recently used in big maintenance process. Cloud computing is "a distributed computing technology that provides dynamically scalable computing resources including storage, computation power, and applications delivered as a service over the Internet" [12, 13]. It has several advantages such as location independence, cost effectiveness, maintenance, and scalability [14]. Bahga et al. [15] present a cloud computing framework, CloudView, for storage, processing, and analysis of massive machine maintenance data, collected from a large number of sensors embedded in industrial machines. A CBR approach is adopted for machine fault predication, where the past cases of failure from a large number of machines are collected in a cloud. Case-base creation jobs are formulated using the MapReduce parallel data processing model. Yang and Xu et al. [16] prescribe an agent-based heterogeneous data integration and maintenance decision support for high-speed railway signal system, in which ontology and CBR are integrated for fault diagnosis.

However, the measured variables from railway signalling system monitoring system compose a large number of subsystems, such as track, switch, power, and interlocking subsystems, in which there exists of large number of operational and maintenance logs. In addition, after the case retrieval, the number of rows of the knowledge base increases dynamically in order to have a comprehensive set of historical solutions, and the number of data samples could be of the order of several thousand for realistic railway signalling system, which makes the overall problem intractable. Therefore, there is an urgent need to combine a practical text mining system with cloud computing that can quickly analyze such data to maintain the equipment safety and reliability. In this paper, a cloud computing-based computing framework with text case-based reasoning (TCBR) is proposed, which borrows the ideas from [15, 17–19]. The main idea is to extract fault features by text mining, reduce attributes by rough set theory [20–25], and solve the fault diagnosis and predication problem by deploying a CBR module based on the Hadoop platform with MapReduce framework [26, 27], which is a computing paradigm for Big Data management created at Google. This computing paradigm is able to scale up the computing to thousands of processors and terabytes (or petabytes) of data. A fault diagnosis and predication of train onboard equipment is presented to demonstrate its efficiency.

The rest of the paper is organized as follows. In Section 2, the system framework of integration of rough set theory and TCBR is proposed. In Section 3, the methodology of TCBR for fault diagnosis/predication is presented. In Section 4, the effectiveness of the proposed method is analyzed by application to a railway onboard system, and Section 5 draws the conclusion of the paper.

2. System Framework

Borrowing ideas from [15], a TCBR framework for fault diagnosis and predication by cloud computing in railway signalling system is presented in Figure 1. The proposed framework allows massive data collection and analysis in a computing cloud, with the benefit of real-time diagnosis and predication of equipment failure by information on cases related to faults. The proposed framework has capability to analyze the maintenance records gathered from field devices (e.g., switches, track circuits, and cables in railway signalling systems) from a number of railway signalling maintenance sectors (e.g., Changsha, Guangzhou, and Chenzhou in Figure 1). By processing these data, a case-base (CB) is created, which is used for fault diagnosis and predication by dispatching CB to maintenance sectors. These sectors carry on fault diagnosis and predication by comparing the gathered fault symptom with the cases in CB.

In the proposed cloud computing framework, HDFS and MapReduce are utilized to store and process large numbers of monitoring data. HDFS stores files across a collection of nodes in a cluster. Large files are split into blocks and each block is written to multiple nodes (default is three) for fault tolerance. MapReduce is a parallel data processing model which has two phases: Map and Reduce. In the Map phase, data are read from a distributed file system (such as HDFS), partitioned among a set of computing nodes in the cluster, and sent to the nodes as a set of key-value pairs. The Map tasks process the input records independent of each other and produce intermediate results as key-value pairs. When

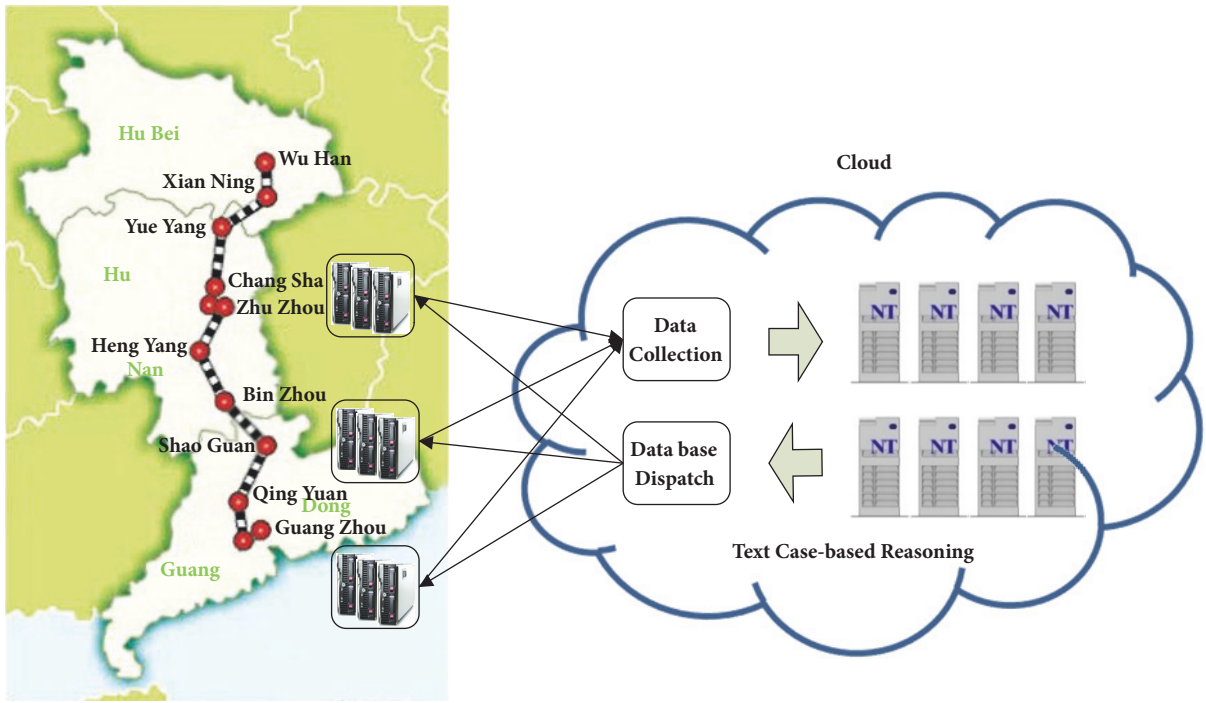

Figure 1: System framework.

all the Map tasks are completed, the Reduce phase begins in which the intermediate data with the same key is aggregated. In this paper, the task of fault feature reduction and case retrieval involving massive data analysis is done by the cloud computing using MapReduce jobs.

3. Methodology

The proposed framework comprises four main steps: data acquisition, feature extraction, rough set-based attribute reduction, and cloud computing-based case retrieval.

Figure 2 outlines the steps of presented approach and related components. First, in the data acquisition and feature extraction module, the maintenance historical records are analyzed by lexical analysis and diagnosis ontology for annotating key terms recorded in the repair verbatim. The annotated terms are extracted, which are used to identify fault features, such as fault mode, fault phenomena, fault part, and corrective actions. The extracted fault features are further reduced by rough set theory. Then a CB is generated by collecting the reduced fault features. When new maintenance log is entered, the case retrieval is conducted by similarity calculation, in which the diagnosis ontology is used to compute the similarity between different concepts. Finally, the results of CBR are used for fault diagnosis or predication. The task of rough set theory based fault feature reduction and case retrieval involved massive data analysis is conducted by the cloud computing.

3.1. Data Acquisition and Feature Extraction. In the event of railway signalling malfunctioning, the diagnostic trouble symptoms are generated and transmitted to the monitoring center database by wired/wireless communications. After every diagnosis episode a repair verbatim is recorded, which consists of a textual description of the mixture of fault symptom (i.e., fault terms), e.g., 'faults', a fault symptom associated with a specific part, e.g., 'SDU (Speed Distance Unit)', failure modes (i.e., fault classes), and finally corrective actions, e.g., 'replaced SDU' taken to fix of its faults. In railway industry, millions of such repair verbatims are generated every year. Table 1 [7] gives a simple example with two verbatims. They provide useful data from which the knowledge must be discovered for efficient fault diagnosis and handling of the similar cases in the future. From repair verbatim data, text mining techniques can be used to establish the associations between fault terms and fault classes such that these associations can be used to improve the precision of fault diagnosis and predication.

This data acquisition component consists of collecting unstructured information from maintenance records; conducts fault feature extraction including lexical analysis, fault mode, fault phenomena, fault part, and corrective actions, and synonym recognition; and then inputs the fault feature reduction module.

In order to realize the fault term/feature extraction, we here utilize the ontology technology. The term "ontology" is defined as the study of the existence of knowledge and has been widely applied in different information systems. It is an explicit and formal specification of a conceptualization and advanced knowledge organization technique. Informally speaking, an ontology is able to be a conceptual model that specifies the terms and relationships between the concepts explicitly and formally, which in turn represents the knowledge for a specific domain. In railway fault diagnosis and

Table 1: Two examples of railway signaling maintenance records.

Date (Year/MM/DD)	Fault phenomenon	Fault class
2008/06/05	A train with ID G1002 had been unable to switch to CTCS-3 mode since departing from Guangzhou South Station. After the driver restarted ATP (Automatic Train Protection) at Changsha South Station, the train can switch to CTCS-3 mode.	Train-ground communication fault
2008/08/31	A train with ID G1026 reported that Cab Signal code could not be received after departing Zhuzhou West Station. Till arriving at Wuhan Station, the code received normally.	Specific Transmission Module (STM) related fault

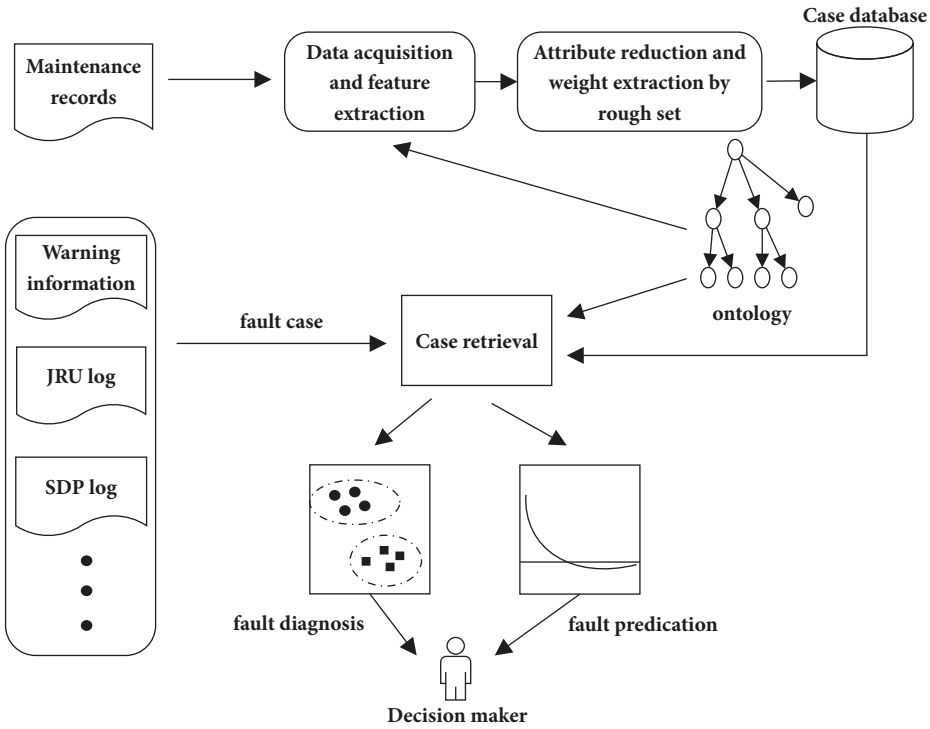

Figure 2: The working process.

predication, the ontology stores information about solutions for previous fault diagnosis and related fault symptom, fault part, and corrective actions, which is shown in Figure 3. Ontology is used to describe knowledge for fault diagnosis and maintenance cases. In the figure, there are concepts, such as System, Subsystem, Component, Fault, Symptom, and Maintenance Support, to describe the knowledge of railway signalling maintenance. The terms, such as train control system, onboard equipment, SDU, are the instances corresponding to the concepts, System, Subsystem, and Component, respectively.

At abstract level, the railway fault ontology is a structure of the form: $Ont_R = (C_i, C_{iSubconcept}, I_{C_i}, R_{C_i \longrightarrow C_j})$ [16]. The C_i represents concepts in the railway fault diagnostic domain. More specific concepts are represented by formalizing them in terms of the concept-subconcept hierarchy $C_{iSubconcept}$. The instances, I_{C_i}, formalize the domain specific concepts in terms of the data associated with the objects in real world; e.g., the concept fault mode can be instantiated by defining the instances, such as SDU failure or SDU relevant failures. These instances provide the knowledge base which can be used for annotating the fault features in the repair verbatim.

The binary relations $R_{C_i \longrightarrow C_j}$ are used to represent the association in railway domain. These relations are used in CBR to verify the associations between the fault features, which are extracted by lexical analysis and fault diagnosis ontology.

One of problems we must face is the synonym identification, which is regarded as data noise and plays a very important role in the following TCBR. Figure 4 shows the process for synonym identification.

Let W be the synonym term under consideration; the context of W in an instance is formed by terms surrounding W. This is done by feature selection in

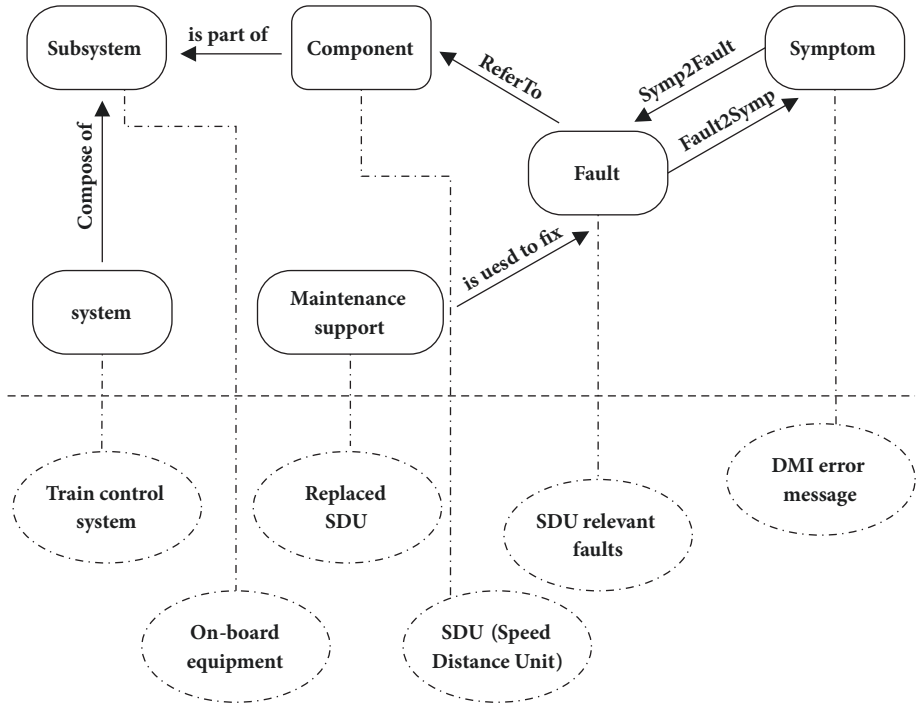

Figure 3: Ontology framework.

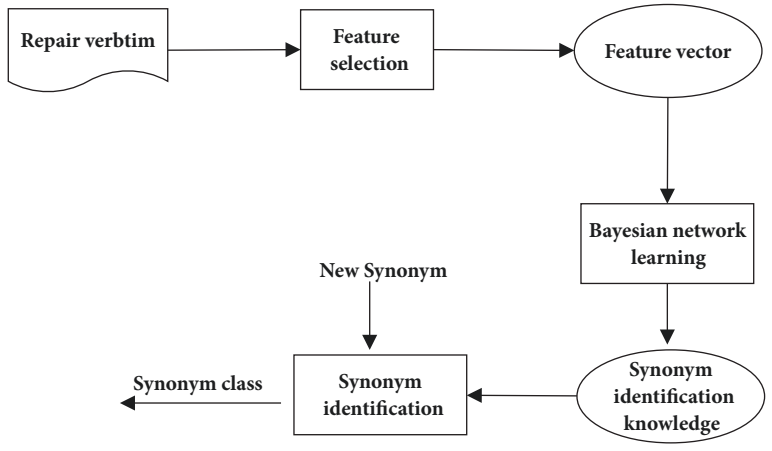

Figure 4: The process for synonym identification.

Figure 4. The context is then mapped to a feature vector $((f_1, v_1), (f_2, v_2), \ldots, (f_n, v_n))$, where f_i is a feature and v_i is its corresponding value. Surrounding words of \$W\$ in a fixed window size are used in the universe. Obviously, large values of window sizes capture dependencies at longer range but also dilute the effect of the words closer to the term. Leacock et al. [28] used a window size of 50, while Yarowsky [29] argued that a small window size of 3 or 4 had better performance. A small window size has an advantage of requiring less system space and running time [30]. Here, we use a small window size of 4 to conduct the synonym term identification, which is similar to [31]. In order to extract features in a window size of 4, a document annotation by ontology is needed. The main aim of document annotation is to attach the metainformation to each repair verbatim by highlighting the key terms recorded in it. Firstly, the concept instances from the fault ontology are matched with the terms/phrases written in each repair verbatim. The document annotation helps us extract relevant terms (i.e., fault attributes) for the following TCBR. A number of document preprocessing steps are taken to reduce the document dimensions: tokenization (a process of breaking a stream of text into the words and phrases), stop word deletion, and lexical matching. The nondescriptive stop words (for example, 'a', 'an', and 'the') are deleted to reduce the noise.

We select the repair verbatim set, $(v_1, \ldots, v_n)$, consisting of the synonym term of W as training data. In order to avoid the feature sparse, four words on either side of W and the

contextual information specified in terms of the component $(Q_1, \ldots, Q_n)$, symptoms $(S_1, \ldots, S_n)$, and actions $(A_1, \ldots, A_n)$ cooccurring with W are collected. Under the independent assumption of component, symptoms, and actions, the synonym term identification is formulated:

$$C_w = \arg\max_{Q_n, S_l, A_r} P(W \mid Q_n, S_l, A_r) \tag{1}$$

$$= \arg\max_{Q_n, S_l, A_r} \frac{P(Q_n, S_l, A_r \mid W)}{P(Q_n)\, P(S_l)\, P(A_r)} \tag{2}$$

$$= \arg\max_{Q_n, S_l, A_r} P(Q_n \mid W) \cdot P(S_l \mid W) \cdot P(A_r \mid W) \tag{3}$$

By native Bayes assumption, symptoms and actions are independent of each other, and $P(Q_n^i \mid W)$ is calculated by

$$P(Q_n \mid W) = \prod_{Q_n^i \in Q_n} P\left(Q_n^i \mid W\right) \tag{4}$$

$$P\left(Q_n^i \mid W\right) = \frac{count\left(Q_n^i, W\right)}{count\,(W)} \tag{5}$$

where $count(Q_n^i, W)$ is the number of cooccurrences of Q_n^i and W.

After the Bayesian model for synonym term identification is done, the data noise resulting from synonym term may be solved.

3.2. Fault Case, Attribute Reduction, and Weight Calculation. Fault cases are defined as a fault detection, diagnosis, and fix situation in terms of fault symptom descriptions and related solutions. In a CBR method, a case is described by a set of attributes (fault features) or aims that identify the instance of a fault, its symptom, and its solution. It can be used for specification of a fault situation and its relevant attributes, which is used to facilitate the retrieval of suitable maintenance records. These features are viewed as the attributes of each fault record in the database.

The purpose of the attribute reduction has been employed to remove redundant conditional attributes from discrete-valued datasets, while retaining their information content [24]. Attribute or feature selection is to identify the significant features, eliminate the irrelevant or dispensable features to the learning task, and build a good learning model. It refers to choosing a subset of attributes from the set of original attributes. For reducing the dataset and assigning weights to the case feature attributes we use rough set theory [22, 23]. Rough set theory is a mathematical tool to deal with problems on vagueness and uncertainty. We now provide few key definitions of the rough set theory:

(1) Information System: an Information System is defined as a pair $S = (U, A)$, where U is a finite nonempty set called the universe that includes all the cases and A is a finite nonempty set of attributes. If we distinguish two disjoint classes of attributes, called condition and decision attributes, then the information system is called a decision table, $S = (U, C \cup D, V, f)$, where $A = C \cup D$, $C \cap D = \emptyset$. Each attribute $a \in A$ is associated with a set V_a of its values by function f, called the domain of a.

(2) Indiscernibility Relation: Indiscernibility Relation $IND(B)$ for any subset $B \subseteq A$ is defined as $IND(B) = \{(x, y) \in U \times U : \forall a \in B, f_a(x) = f_a(y)\}$. Two entities are considered indiscernible by the attributes in B if and only if they have the same value for every attribute in B. $IND(B)$ is an equivalence relation that partitions \$U\$ into equivalence classes which are denoted by $U/IND(B)$.

(3) Lower and upper approximations: for any case $X \subseteq U$ and attribute subset $R \subseteq A$, the lower approximation of X, $\mathrm{R}_*(X)$ is the set of objects of U that are surely in X, whereas the upper approximation of X, $\mathrm{R}^*(X)$ is the set of objects of U that are possibly in X.

(4) Positive region: the positive region of decision class $U/IND(D)$. With respect to condition attributes, C is denoted by $POS_C(D) = \bigcup \mathrm{R}_*(X)$. It is a set of objects of U that can be classified with certainty to classes $U/IND(D)$ by employing attributes of C.

(5) Reduct: A subset $R \subseteq C$ is said to be a D-reduct of C if $POS_R(D) = POS_C(D)$ and there is no $R' \subseteq R$ such that $POS_{R'}(D) = POS_C(D)$. In other words, a reduct is the set of attributes that can differentiate all equivalence classes.

(6) Core: Core is the set of attributes that are contained by all reducts, defined as $CORE_D(C) = \bigcap RED_D(C)$, where $RED_D(C)$ is the D-reduct of C. In other words, the Core is the set of attributes that cannot be removed without changing the positive region; i.e., all attributes present in the Core are indispensable.

The rough set attribute reduction by cloud computing is formulated as the following problem.

Theorem 1. *Zhang [25] has given a decision table $S = (U, C \cup D, V, f)$. Let $S = \bigcup_{i=1}^{m} S_i$, where $S_i = (U_i, C \cup D, V, f)$. It satisfies (1) $U = \bigcup_{i=1}^{m} U_i$; (2) $U_k \cap U_j = \emptyset, \forall j, k \in \{1, 2, \ldots, m\}$ and $j \neq k$. For any subset $B \subseteq C$, $U/B = \{E_1, E_2, \ldots, E_t\}$ and $\forall i \in \{1, 2, \ldots, m\}$, $U_i/B = \{E_{i1}, E_{i1}, \ldots, E_{ip}\}$. Let $E_{all} = \bigcup_{i=1}^{m} U_i/B = \{E_{11}, E_{12}, \ldots, E_{1p_1}, \ldots, E_{m1}, E_{m2}, \ldots, E_{mp_m}\}$. Therefore, for any $E_j \in U/B$, $j \in \{1, 2, \ldots, t\}$, we have $E_j = \bigcup\{F \in E_{all} \mid \overrightarrow{F_B} = \overrightarrow{E_{jB}}\}$.*

According to Theorem 1, each subdecision table can compute equivalence classes independently. At the same time, the equivalence classes of different subdecision tables can combine together if their information set is the same. Following [32], the attribute reduction by cloud computing is listed as Algorithms 1, 2, and 3.

In Algorithm 1, each sample x can be seen as a key/value pair: the key is $f_c(x)$ (c $\in C$), the value is $f_d(x)$ ($d \in D$). This step is executed in parallel by multiple Map operations. In Algorithm 2, the samples which have the same key are merged to generate $Sig^c_{POS}(A, D)$. In Algorithm 3, by means

```
Data: S =< U, A = C ∪ D, V, F >
Result: < Equivalentclass, (f_d(x), 1) >
begin
    for x ∈ S_i do
        for c ∈ C − A do
            output< Equivalentclass, (f_d(x), 1) >
        end
    end
end
```

Algorithm 1: Attribute reduction Map(key, value).

```
Data: < Equivalentclass, [(d_1, n_1), ..., (d_2, n_2), ...] >
Result: Attribute c, Sig^c_POS
begin
    for < d, n >∈ [(d_1, n_1), ..., (d_2, n_2), ...] do
        Compute counts of different decision value
        (n_{d_1}, n_{d_2}, ...)
        Compute
        Sig^c_POS(A, D) = |POS(D | A ∪ c) − POS(D | A)|
    end
    Output< Attribute c, Sig^c_POS >
end
```

Algorithm 2: Attribute reduction Reduce(key, value).

```
Data: S =< U, A = C ∪ D, V, F >
Result: Attribute Reduction R
begin
    Red = 0
    compute Sig_POS(C, D)
    while Sig_POS(C, D) ≠ Sig_POS(Red, D) do
        Start a job and compute, Sig^c_POS(C, D), c ∈ C − c
        by Algorithms 1 and 2
        Select c_l = max_{c∈C−c}{Sig^c_POS(C, D)},
        Red = Red ∪ c_l;
    end
end
```

Algorithm 3: Parallel rough set attribute reduction.

of Algorithms 1 and 2, the less significant attributes are removed and the attribute reduction is obtained.

In this weighting method we use the significant attribute dependence coefficient, computing the reduction of information. The significant dependence coefficient is computed as

$$Sig_{POS}(C, D) = \frac{card\left(POS_C(D) - POS_{(c-c_i)}(D)\right)}{card(D)} \quad (6)$$

where c_i is the ith attribute from which we are computing the weight; *card* is the cardinality; $POS_C(D)$ is the positive region of all relations (features) present in the reducts; and, finally, $POS_{(c-c_i)}(D)$ is the positive region of all relations present in the reducts extracting attribute c_i.

The weight of c_i is computed as

$$w_{c_i} = \frac{\mu_{c_i}}{\sum_{c_i \in C} u_{c_i}} \quad (7)$$

Considering that the calculation of attribute weights $Sig_{POS}(C, D)$ is similar to $Sig^c_{POS}(C, D)$, w_{c_i} can be computed by Algorithms 1 and 2. For simplicity, we omit it here.

3.3. Case Retrieval and Ranking. The case retrieval is one of the most important processes in TCBR design and also in TCBR component of the system. When a new failure situation occurs, the TCBR system retrieves, from a case-base, previous cases that are similar to the new failure situation.

Table 2: Case-base of train onboard equipment failures.

case	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}	d
1	0	0	0	1	0	0	0	0	0	1	1
2	0	1	0	0	0	0	0	0	1	0	1
3	0	0	0	0	0	1	0	0	1	0	1
4	0	0	0	0	1	0	0	1	0	0	2
5	0	1	0	0	0	0	0	0	0	1	1
6	0	0	1	0	0	0	0	0	0	1	1
7	0	0	0	0	0	1	0	0	0	1	1
8	1	0	0	0	0	0	0	0	1	0	2
9	0	1	0	0	0	0	1	0	0	0	3

```
begin
    Map(key, value);
    read new fault records with reduced fault attributes;
    compute a similarity value by the calculated attribute weight;
    find the most similarity value by comparison with CB;
    output <key=case index, value=similarity value>;
    Reduce(key, value);
    find the largest similar index;
    output <key=similarity value, value=case index>
end
```

Algorithm 4: MapReduce algorithm for case retrieval and ranking.

Therefore case retrieval and ranking are equal to searching the most similar cases in the generated CB. This job is easily parallelized in the cloud computing framework by assigning a partial task to each node in a cluster, in which k most similar cases are searched by Map function. After that, Reduce task is used to aggregate these similar cases and then finds most similar case, which is used for fault diagnosis, predication, and maintenance decision.

To retrieve similar cases from historical ones, a similarity measurement is commonly used in case retrieval. The value of similarity is between 0 (not similar) and 1 (most similar). The total similarity value is calculated by

$$S = 1 - D(x, y) = 1 - \sqrt{\sum_{i}^{n} w_{c_i} \times D\left(f_{c_i}(x), f_{c_i}(y)\right)^2} \quad (8)$$

where

(1) if c_i is a boolean attribute, and $f_{c_i}(x) = (y)$, then $D(f_{c_i}(x), f_{c_i}(y)) = 0$,

(2) if c_i is a boolean attribute, and $f_{c_i}(x) \neq (y)$, then $D(f_{c_i}(x), f_{c_i}(y)) = 1$,

(3) if c_i is continuous, then $D(f_{c_i}(x), f_{c_i}(y)) = |f_{c_i}(x) - f_{c_i}(y)|/(max_{f_{c_i}(x)} - min_{f_{c_i}(y)})$,

(4) if $f_{c_i}(x)$, $f_{c_i}(y)$ are concepts, the concept based similarity is calculated by ontology [8, 33], which will be described as semantic similarity.

Semantic similarity between two concepts is considered to be governed by the shortest path length as well as the depth of the concepts in ontology [33]; that is,

$$sim(A, B) = e^{-\alpha l} * \frac{e^{\beta h} - e^{-\beta h}}{e^{\beta h} + e^{-\beta h}} \quad (9)$$

where $\alpha \geq 0$ and $\beta \geq 0$ are parameters scaling the contribution of shortest path length l and depth h, respectively.

The case retrieval Algorithm 4 works as follows: we assume that case-base has stored the historical data related to a given fault situation. When a new fault record is input into the system, the case-base is searched to retrieve all cases with a similar profile. The similarity of the new problem to the stored cases is determined by calculating similarity value between case features. Then, once the most similar cases have been obtained from the case-base, they will be used in the fault diagnosis and predication to generate a maintenance decision.

4. Application to Fault Diagnosis of a Railway Onboard System

We now describe a typical use case of fault diagnosis using the proposed framework. Table 2 shows part of a case-base of train onboard equipment failures with data comprising ten different attributes after feature extraction (i.e., c_1-c_{10}) and the corresponding fault types d, where c_1 means ATP (Automatic Train Protection) failure, c_2 communication interruption between DMI (Driver Machine Interface) and

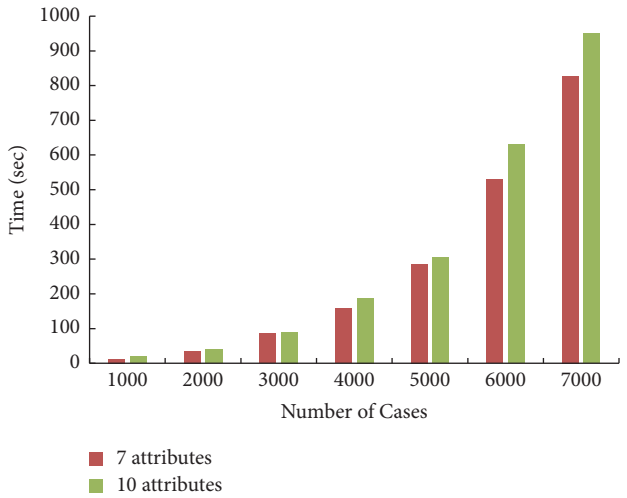

Figure 5: Comparison of case retrieve time with different number of attributes.

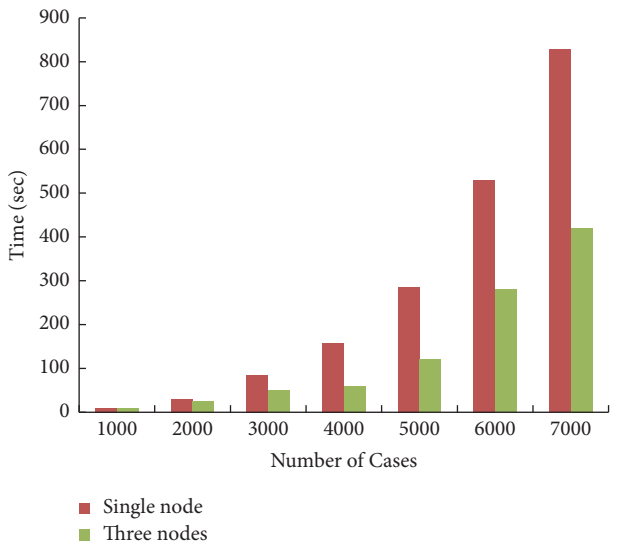

Figure 6: Comparison of case retrieve time with different number of nodes.

CPU, c_3 emergency brake fault, c_4 ATP startup failure, c_5 normal brake failure, c_6 brake test failure, c_7 bypass relay being invalid, c_8 relay interface of normal brake being invalid, c_9 redundant relay interface being invalid, c_{10} relay interface of emergency brake being invalid, "d=1" brake test failure, "d=2" normal brake failure, and "d=3" brake bypass failure.

To evaluate the performance of the proposed framework, we perform a number of experiments by varying the number of cases and attributes and compare the case retrieval times. The local node used for evaluations had an Intel(R) Pentium(R) CPU G630 2.7 GHz processor with 4 GB memory and 500G disk space. The local node contained the case-base in a text file database and case retrieval jobs are programmed in Eclipse platform using Java. Figure 5 illustrates the comparison of case retrieval times with different attributes. From it we can see that with the attribute reduction the case retrieval and fault diagnosis/predications can reduce the time consumption obviously. Figure 6 shows the comparison of case retrieval times with different nodes (i.e., compute units). We observe that even with a large case-base in the local node with up to 7,000 cases, the case retrieval and fault diagnosis/predications can be done in a timescale of seconds. For analysis of 7,000 cases with the proposed method, experiments show a speed up of up to 2 times using a computing cluster (with 3 compute units) as compared to a single node.

5. Conclusion and Future Research

This paper presents a text case-based reasoning framework for fault diagnosis and predication by cloud computing, which integrates the text mining, rough-based attributes reduction, and case retrieval by cloud computing. It is shown that the proposed approach has the following advantages: (1) text mining helps in processing unstructured maintenance records; (2) rough set-based attribute reduction is used to reduce the attributes of fault cases, which significantly decrease the complexity of case retrieval; (3) case retrieval by cloud computing is able to scale up a large amount of case bases and improves the efficiency of fault diagnosis and predication. The effectiveness of the proposed algorithm is demonstrated through its use in fault diagnosis and predication of a railway onboard system. Optimization of text mining and rough set reduction in cloud computing needs further research.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The work was supported by National Natural Science Foundation of China under Grant no. 61473029, Gansu Key Laboratory of Traffic Information Engineering and Control of Lanzhou Jiaotong University within the frame of the Project 20161101, the State Key Laboratory of Rail Traffic Control and Safety of Beijing Jiaotong University within the frame of the Projects RCS2014ZT05 and RCS2016ZT010, and the Fundamental Research Funds for the Central Universities (no. 2017YJS019).

References

[1] J. Baek, Q. H. Vu, J. K. Liu, X. Huang, and Y. Xiang, "A secure cloud computing based framework for big data information management of smart grid," *IEEE Transactions on Cloud Computing*, vol. 3, no. 2, pp. 233–244, 2015.

[2] M. A. Hearst, "Untangling text data mining," in *Proceedings of the the 37th annual meeting of the Association for Computational*

Linguistics (ACL '99), pp. 3–10, College Park, Maryland, June 1999.

[3] L. Huang and L. M. Yi, "Text Mining with Application to Engineering Diagnostics," in *Proceedings of the International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*, vol. 4031, pp. 1309–1317, Annecy, France.

[4] J.-M. Kim and S. Jun, "Graphical causal inference and copula regression model for apple keywords by text mining," *Advanced Engineering Informatics*, vol. 29, no. 4, pp. 918–929, 2015.

[5] D. G. Rajpathak and S. Singh, "An ontology-based text mining method to develop D-matrix from unstructured text," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 44, no. 7, pp. 966–977, 2014.

[6] D. Rajpathak and S. De, "A data- and ontology-driven text mining-based construction of reliability model to analyze and predict component failures," *Knowledge and Information Systems*, vol. 46, no. 1, pp. 87–113, 2016.

[7] F. Wang, T. Xu, T. Tang, M. Zhou, and H. Wang, "Bilevel Feature Extraction-Based Text Mining for Fault Diagnosis of Railway Systems," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 1, pp. 49–58, 2017.

[8] R. Ellis, T. Allen, and A. Tuson, "Ontology based CBR with jCOLIBRI," in *Applications and Innovations in Intelligent Systems XIV*, 162, p. 149, London, 2007.

[9] T. Virkki-Hatakka and G. L. L. Reniers, "A case-based reasoning safety decision-support tool: Nextcase/safety," *Expert Systems with Applications*, vol. 36, no. 7, pp. 10374–10380, 2009.

[10] S. Lee and Y. Ng, "Hybrid case-based reasoning for on-line product fault diagnosis," *The International Journal of Advanced Manufacturing Technology*, vol. 27, no. 7-8, pp. 833–840, 2006.

[11] W. He, "Improving user experience with case-based reasoning systems using text mining and Web 2.0," *Expert Systems with Applications*, vol. 40, no. 2, pp. 500–507, 2013.

[12] I. Arpaci, "Understanding and predicting students' intention to use mobile cloud storage services," *Computers in Human Behavior*, vol. 58, pp. 150–157, 2016.

[13] K. Stanoevska-Slabeva, T. Wozniak, and S. Ristol, *Grid and cloud computing: A business perspective on technology and applications*, Springer Publishing Company, Berlin, Heidelberg, 2010.

[14] T. Shon, J. Cho, K. Han, and H. Choi, "Toward advanced mobile cloud computing for the internet of things: Current issues and future direction," *Mobile Networks and Applications*, vol. 19, no. 3, pp. 404–413, 2014.

[15] A. Bahga and V. K. Madisetti, "Analyzing massive machine maintenance data in a computing cloud," *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 10, pp. 1831–1843, 2012.

[16] L. Yang, T. Xu, and Z. Wang, "Agent based heterogeneous data integration and maintenance decision support for high-speed railway signal system," in *Proceedings of the 2014 IEEE 17th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1976–1981, Qingdao, China, October 2014.

[17] L. Troiano, A. Vaccaro, and M. C. Vitelli, "On-line smart grids optimization by case-based reasoning on big data," in *Proceedings of the 2016 IEEE Workshop on Environmental, Energy, and Structural Monitoring Systems (EESMS)*, pp. 1–6, Bari, Italy, June 2016.

[18] B. K. Lee and E. H. Jeong, "A design of a patient-customized healthcare system based on the Hadoop with text mining (PHSHT) for an efficient disease management and prediction," *International journal of Software Engineering & Applications*, vol. 8, no. 8, pp. 131–150, 2014.

[19] J. Lin and C. Dyer, *Synthesis Lectures on Human Language Technologies*, Morgan and Claypool Publishers, 2010.

[20] Z. Pawlak, "Rough sets and decision analysis," *Information Sciences*, vol. 38, no. 3, pp. 132–144, 2000.

[21] S. Ji, S. Yuan, and S. Wang, "An Algorithm for Case-Based Reasoning Based on Similarity Rough Set," in *Proceedings of the 2008 Fifth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)*, pp. 226–230, Jinan Shandong, China, October 2008.

[22] N. Zhong, J. Dong, and S. Ohsuga, "Using rough sets with heuristics for feature selection," *Journal of Intelligent Information Systems*, vol. 16, no. 3, pp. 199–214, 2001.

[23] Z. Pawlak, "Rough sets," *International Journal of Computer & Information Science*, vol. 11, no. 5, pp. 341–356, 1982.

[24] Q. Shen and A. Chouchoulas, "Modular approach to generating fuzzy rules with reduced attributes for the monitoring of complex systems," *Engineering Applications of Artificial Intelligence*, vol. 13, no. 3, pp. 263–278, 2000.

[25] J. Zhang, T. Li, D. Ruan, Z. Gao, and C. Zhao, "A parallel method for computing rough set approximations," *Information Sciences*, vol. 194, pp. 209–223, 2012.

[26] Apache Hadoop, http://hadoop.apache.org, 2012.

[27] MapReduce Definition, http://en.wikipedia.org/wiki/MapReduce.

[28] C. Leacock, G. Towell, and E. Voorhees, "Corpus-based statistical sense resolution," in *Proceedings of the the workshop*, 265, pp. 260-260, Princeton, New Jersey, March 1993.

[29] D. Yarowsky, "Decision Lists For Lexical Ambiguity Resolution: Application to Accent Restoration in Spanish and French," in *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*, vol. 14, pp. 88–95, 2012.

[30] H. Liu, Y. A. Lussier, and C. Friedman, "Disambiguating ambiguous biomedical terms in biomedical narrative text: An unsupervised method," *Journal of Biomedical Informatics*, vol. 34, no. 4, pp. 249–261, 2001.

[31] D. G. Rajpathak, "An ontology based text mining system for knowledge discovery from the diagnosis data in the automotive domain," *Computers in Industry*, vol. 64, no. 5, pp. 565–580, 2013.

[32] J. Qian, D. Miao, and Z. Zhang, "Parallel Algorithm Model for Knowledge Reduction Using MapReduce," *Journal of Frontiers of Computer Science and Technology*, vol. 7, no. 1, pp. 34–45, 2013.

[33] Y. Li, Z. A. Bandar, and D. McLean, "An approach for measuring semantic similarity between words using multiple information sources," *IEEE Transactions on Knowledge and Data Engineering*, vol. 15, no. 4, pp. 871–882, 2003.