Hindawi
Computational Intelligence and Neuroscience
Volume 2020, Article ID 7132072, 12 pages
https://doi.org/10.1155/2020/7132072

Research Article

A Method to Extract Causality for Safety Events in Chemical Accidents from Fault Trees and Accident Reports

Junwei Du, Hanrui Zhao, Yangyang Yu, and Qiang Hu

School of Information Science and Technology, Qingdao University of Science and Technology, Qicngdao 266061, China

Correspondence should be addressed to Qiang Hu; huqiang200280@163.com

Received 24 February 2020; Revised 26 May 2020; Accepted 3 June 2020; Published 19 June 2020

Academic Editor: Mario Versaci

Chemical event evolutionary graph (CEEG) is an effective tool to perform safety analysis, early warning, and emergency disposal for chemical accidents. However, it is a complicated work to find causality among events in a CEEG. This paper presents a method to accurately extract event causality by using a neural network and structural analysis. First, we identify the events and their component elements from fault trees by natural language processing technology. Then, causality in accident events is divided into explicit causality and implicit causality. Explicit causality is obtained by analyzing the hierarchical structure relations of event nodes and the semantics of component logic gates in fault trees. By integrating internal structural features of events and semantic features of event sentences, we extract implicit causality by utilizing a bidirectional gated recurrent unit (BiGRU) neural network. An algorithm, named CEFTAR, is presented to extract causality for safety events in chemical accidents from fault trees and accident reports. Compared with the existing methods, experimental results show that our method has a higher accuracy and recall rate in extracting causality.

1. Introduction

In recent years, the chemical industry has made tremendous contributions to economic and social development. However, a series of safety accidents have occurred frequently along with the enormous economic benefits brought by chemical enterprises. For example, seventy-eight people died in the explosion of Yancheng Chemical Industrial Park on March 21, 2019. After the accident, sixteen chemical enterprises in this industrial park were closed [1]. The occurrence of chemical accidents has brought enormous economic losses to enterprises and individuals, made irreparable damage to the environment, and even caused heavy casualties [2]. Therefore, accident prevention and emergency treatment have become the focus of daily production in the chemical industry.

On the ascending scale of production in the chemical industry and abundance of chemical products, the production process is becoming more and more complex, and the risk factors in all aspects of production are also increasing [3]. Controlling and early warning the unsafe factors such as high temperature, high pressure, flammability, explosion, and poisoning in chemical production can effectively prevent future chemical accidents [4]. Versaci presented a fuzzy method to achieve the detection/classification of defects. It considered classes of defects to a certain depth characterized by typical ranges of fuzzy similarities [5]. The literature [6] covers the practical implementation of ultrasonic NDT techniques in an industrial environment, discussing several issues that may emerge and proposing strategies for addressing them successfully. Many of the technologies it provides can be applied to the detection of hazardous chemical production information.

By analyzing the causes of accidents, excavating the potential factors, evolution rules, and protective measures, we can decrease accident rates, reduce accident losses, and improve the safety management level and emergency disposal ability of chemical enterprises. Fault tree analysis is one of the most frequently used methods in safety analysis, prevention, and emergency disposal [7]. Fault trees can describe the causes of accidents and their temporal logic relationship [8]. So, we can find out the key events in

chemical accidents and predict potential hazards in chemical production from the existing fault trees.

However, only the evolution process of an accident can be obtained from fault trees. Time, location, and environmental state of these accidents were not described for the concise structure of fault trees. Thus, it may result in a lack of important information while analyzing the cause of an accident. Since most of the fault trees were constructed based on expert's accident analysis experience, there may be semantic ambiguity, incomplete information, mixed information, and information hybridity in the fault trees. Meanwhile, for the complex fault trees, we may face high complexity and incomplete evolutionary information while analyzing the event evolution mechanism. It is also very difficult to accurately locate or match event evolution sequences.

To compensate for the abovementioned deficiencies of the fault trees in safety analysis, early warning, and emergency disposal, event evolutionary graph (EEG) is introduced to model the event evolution process in chemical accidents in this paper. EEG is a type of graphic information carrier developed on the basis of knowledge graphs [9]. Illustrated as a digraph, it describes the causal relationship and temporal dependence in the event chains of an accident. An EEG which described the evolution process of chemical accidents is called chemical event evolutionary graph (CEEG). By traversing the CEEG, we can easily obtain the evolution sequences of events in chemical accidents. We can also predict the potential events in an accident by means of evaluating the event causality and transfer probability.

A scenario about the event of the "volatile explosion of oil and gas" is illustrated by the CEEG in Figure 1. The first node that is "valve leakage" says that the event starts from valve leakage of storage tanks. Then "oil and gas evaporate" and "explosive gas converge" occur sequentially. When the concentration of explosive gas exceeds a certain amount, it will cause an explosion. Explosion requires some triggering conditions. So, we can see that the node "explosive gas converge" connects with three succeeding nodes. "Explosion with fire," "explosion with thunder," and "explosion with static electricity" represent the explosion caused by the fire, thunder, and static electricity, respectively. "Explosion shock" and "fire breaking" are two main destruction scenarios. Thus, the nodes "explosion shock" and "fire breaking" are linked with three types of explosion nodes separately. Since the events in an accident are organized by their temporal or causal relationships, we can easily achieve the event traceability and early warning with a CEEG.

It is a complicated and challenging task to build the CEEG. Event identification, event relation extraction, and event entity link are the main tasks in the process of constructing the EEG. In this study, we build CEEG based on the existing fault trees and accident reports. Most events in the chemical accidents are with the causal relationship, and the causality is also the main link relation between safety events in the CEEG. So, we concern about how to identify events and extract causal relationships between these events. The main contributions of this study are as follows:

(1) We propose an effective method to extract event elements by combining fault tree with accident reports. The combination of fault tree and accident report greatly reduces the complexity of event extraction based on NLP.

(2) We obtain explicit causality by analyzing hierarchical structure relations of event nodes and logic gates in fault trees. Implicit causality is generated based on BiGRU neural network by feeding internal structural features of events and semantic features of event sentences. The accuracy and efficiency of extracting causality are improved by dividing causality into explicit causality and implicit causality.

(3) We have conducted several rounds of experiments to verify the effectiveness of the proposed method. In view of accuracy and recall rate, experimental results show that our model and method are superior to the state-of-the-art methods in extracting causality.

The rest of this paper is structured as follows. In Section 2, we introduce the formal definitions of fault tree and EEG. Section 3 presents a method to achieve event identification. How to extract causality between safety events is elaborated in Section 4. In Section 5, experiments are presented to show the effectiveness of our method. We conclude our work in Section 6.

2. Related Definitions

The main purpose of this paper is to provide an effective method of finding potential events and their causality. After getting events and their causality, we can build CEEG and then apply accident analysis, reasoning, and early warning. To accurately and automatically acquire the knowledge in building CEEG, we proposed a method to extract the events and causality from fault trees and accident reports. We will present the definitions of fault tree and EEG so as to better illustrate our method in the following sections.

Fault trees are frequently used to analyze the risks related to safety and they can describe the temporal logic of the events involved in a safety accident [10]. There are two types of nodes: events and gates in a fault tree. Events in a fault tree are used to represent the main events leading to accidents and they can be classified into three types: basic events, intermediate events, and top events. Gates represent how events propagate through the system while the edges were employed to express the occurring order relations of these events [11].

The fault tree in Figure 2 described a scenario of an "oil tank explosion." We can see that the basic events "hollow appeared in plate of the tank" and "crack appeared in plate of the tank" are connected with the OR gate O_1. It means that the event "deformation or break occurred in the tank" will be triggered if one of the above basic events has happened. For the AND gate A_1, the events "20 Tons diesel oil filled in the tank," "deformation or break occurred in the tank," and "storage tank overdue maintenance" are its input events, and "diesel leakage from the tank" is its output event. So, only all the input events appear simultaneously, and the output

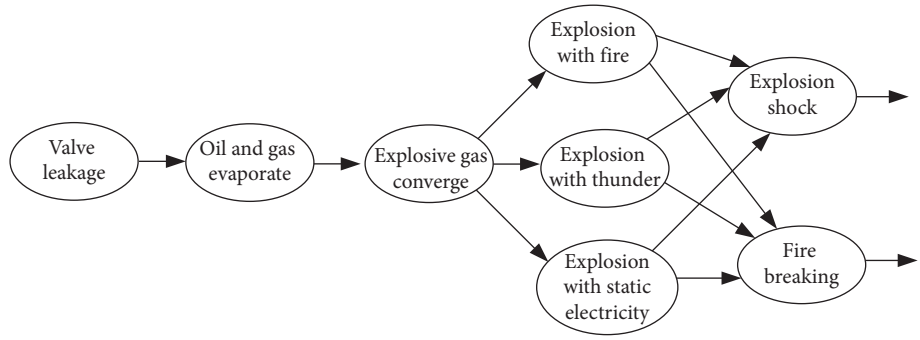

Figure 1: An event evolutionary graph under the scenario of the "volatile explosion of oil and gas."

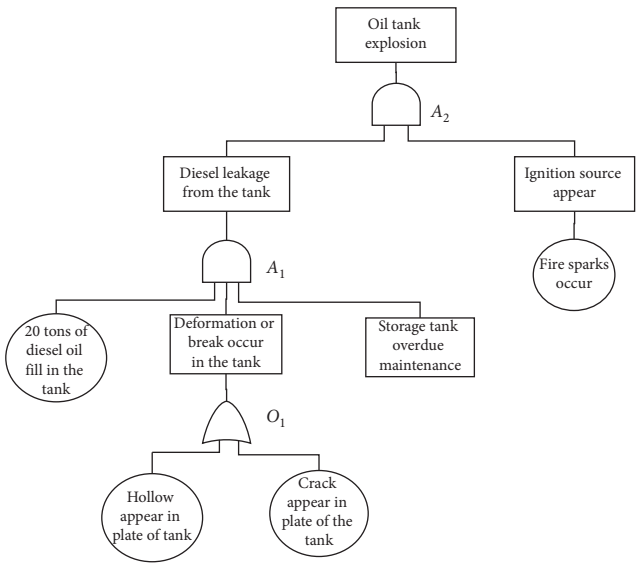

Figure 2: A fault tree under the scenario of "oil tank explosion."

event can occur. Similarly, we can deduce the sequence of events for "fire sparks occur," "ignition source appear," and "oil tank explosion."

2.1. Definition 1 (Fault Tree). A fault tree is a 4-tuple $FT = (V, G, E, v_0)$, consisting of the following components:

(1) V is the set of nodes in FT; each node v in V is used to represent an event

(2) G is the set of logic gates. $\forall g \in G$, $T(g)$ is a function that describes the type of each gate

(3) E is the set of arcs in FT, $E \subseteq V \times G \cup G \times V$

(4) v_0 is the root node of FT

There are three types of nodes in fault trees: root node, leaf nodes, and intermediate nodes. Root node v_0 represented the top event. $V_L = \{v \in V \wedge (\nexists g \in G, \text{ s.t. } (v, g) \in E)\}$; $\forall v \in V_L$, v is a leaf node, and it is used to denote a basic event. $V_M = \{v \in V \wedge (\exists g \in G, \text{ s.t. } (v, g) \in E)\}$; $\forall v \in V_M$, v is an intermediate node, and it is used to denote an intermediate event.

To easily obtain the input events and output event for a logic gate, we present two functions: (1) I: $G \longrightarrow \Psi(E)$ describes the input event of each gate; (2) $O: G \longrightarrow \Gamma(E)$ describes the output event of each gate.

From the example in Figure 1, we can see that an event evolutionary graph is a digraph. Nodes in event evolutionary graph are used to denote the events, and the arcs are adopted to represent the dependencies among these events. Now, we give the definition of EEG.

2.2. Definition 2 (Event Evolutionary Graph). Event evolutionary graph $(EEG) = (V, E)$. Here, V is a set of nodes; $\forall v_i \in V$, v_i is an event, and it is represented by abstract, generalized, and semantic complete verb phrase. E is a set of arcs; $\forall e_{ij} \in E$, it denotes that there exists dependency between the event v_i and v_j.

There are two kinds of dependencies between the events: sequential relation and causality. The sequential relation between two events refers to their partial temporal orderings. Causality is the relation between one event (the cause) and a second event (the effect), where the second event is understood as a consequence of the first [9]. In this study, we used the symbol "$\longrightarrow$" to represent causality. For two events e_i and e_j, $e_i \longrightarrow e_j$ means that e_i is the cause of e_j. It is obvious that the causal relation between events must be sequential. To find causality between two events is a more difficult and challenging work.

3. Event Identification

Event identification, also called event recognition or event extraction, is the process to find the component elements (factors) of an event from various information sources. In a recent study, Skarlatidis addressed the issue of uncertainty in logic-based event recognition by extending the Event Calculus with probabilistic reasoning [12]. Chen introduced a word-representation model to capture meaningful semantic regularities for words. He adopted a framework based on a dynamic multipooling convolutional neural network (DMCNN) to capture sentence-level clues and reserve crucial information [13]. Feng developed a language-independent neural network to capture both sequence and chunk information from specific contexts and used them to train an event detector for multiple languages without any manually encoded features [14]. Liao proposed a new event recognition method based on positive and negative weighting

proposed by constructing a trigger table [15]. Hogenboom gave a summarization of event extraction techniques for textual data, distinguishing between data-driven, knowledge-driven, and hybrid methods, and presented a qualitative evaluation of these methods [16].

In this study, we will extract events and investigated their causal relationship in chemical accidents. The information source of our event identification is fault trees and accident reports. Now, we give the formal structure of the event used in this paper.

3.1. Definition 3 (Event). An event in an accident is formally defined as a 4-tuple $e = \{o, v, p, t\}$, where o, v, p, and t are used to represent the event participants, event trigger word, location, and timestamp of event occurrence, respectively.

To concisely demonstrate an evolutionary process, fault trees were normally designed with summary information of events. We cannot find a detailed description of the information about the time, location, and environment state. Such information is elaborated in the accident reports. So, we can acquire these event elements by the natural language processing technology from fault trees and accident reports. The extraction of event elements includes the following work: corpus segmentation, part-of-speech tagging, semantic role labeling (SRL), semantic dependency parsing (SDP), and dependency parsing (DP) [17,18]. For each node in a fault tree, we can obtain event elements by the following steps:

(1) Participant ⟵ SRL (fault tree node)

(2) Trigger-word ⟵ SRL (fault tree node)

(3) Place ⟵ SDP (event sentences) and (Place.semantic-dependency (Trigger-word) = LOC)

(4) Time ⟵ SDP (event sentences) and (Time.-semantic-dependency (Trigger-word) = Time)

(5) Subject ⟵ DP (event sentences) and (Subject.dependency-parsing (Trigger-word) = SBV)

(6) Object ⟵ DP(event sentences) and (Object.dependency-parsing (Trigger-word) = VOB)

SRL is first used to identify the trigger words and participants of events in a fault tree. Timestamp and position for an event can be obtained by SDP technology from trigger words. The whole information about the event will be generated after the "subject-predicate-object" structure was parsed by DP. The aforementioned processing functions (SRL, SDP, and DP) were normally encapsulated as APP services. Here, the open-sourced natural language processing system developed in the Research Center for Social Computing and Information Retrieval of Harbin Institute of Technology was invoked in our study to parse event sentences [19].

In Figure 2, there is a node labeling "jet fuel spilled out" in a fault tree. The event sentence of this node in the corresponding accident report is "At 11 o'clock, jet fuel in pipeline spilled out." The processing results of SDP, SRL, and DP are shown in (a), (b), and (c) of Figure 3. We can see that "jet fuel" is the event participant while "spilled out" is an event trigger word.

SDP can identify semantic roles and their relationships in event sentences. The main relations between different roles include the agent relationship, the patient relationship, and the experiencer relationship. The result of SDP in Figure 3(b) shows that the participant "jet fuel" and trigger word "spilled out" are with the experiencer relationship. "In pipeline" and "at 11 o'clock" are of semantic dependence with a trigger word. The roles of "in pipeline" and "at 11 o'clock" were position and time, respectively. Therefore, the participant in this sentence is "jet fuel," the trigger word is "spilled out," the occurrence time is "at 11 o'clock," and the place of occurrence is "in pipeline." The relations of different words in the sentence were illustrated in Figure 3(c) by DP. So far, we can get all event elements of the sentence and the 4-tuple e = {jet fuel, spilled out, in pipeline, at 11 o'clock}.

4. Extraction of Event Causality

A fault tree is a kind of logical causality digraph including the symbols of events, logic gates, and transitions. It can show the variety of system states by the logical evolution of basic events. Event causality in a fault tree can be divided into two categories: explicit causality and implicit causality.

4.1. Extraction of Explicit Causality. Explicit causality can be extracted by analyzing the hierarchical structure relations of event nodes and the semantics of component logic gates. There are various types of logic gates in fault trees. Normally, the following three types of logic gates, namely, AND gate, OR gate, and VOT (k/N) gate, are the fundamental gates. By the combination of the above logic gates, we can get the semantic logic of all the other gates used in fault tree [11].

Let F be a fault tree and let BE represent the set of basic events in F. The semantics of F is a function π_F: $\Psi(BE) \times E \longrightarrow \{0,1\}$ where $\pi_F(S, e)$ indicates whether e fails given the set S of failed BE. It is defined as follows:

(1) For $e \in BE$, $\pi_F(S, e) = e \in S$

(2) For $g \in G$ and $T(g) = \text{AND}$, let $\pi_F(S, g) = \bigwedge_{x \in I(g)} \pi_F(S, x)$

(3) For $g \in G$ and $T(g) = \text{OR}$, let $\pi_F(S, g) = \bigvee_{x \in I(g)} \pi_F(S, x)$

(4) For $g \in G$ and $T(g) = \text{VOT}(k/N)$, let $\pi_F(S, g) = \sum_{x \in I(g)} \pi_F(S, x) \geq k$

From the semantics of logic gates, we know that events in lower-level nodes are the cause of events in upper-level nodes. Figure 4 illustrates a basic structure in a fault tree. Two events e_i and e_j were connected by the logic gate AND, and the event e_m is located in the upper-level node. So, we can get two explicit causality rules: $e_i \longrightarrow e_m$ and $e_j \longrightarrow e_m$. For a given fault tree, we can obtain the explicit causality rules by traversing all the logic gates.

4.2. Extraction of Implicit Causality. Explicit causality can be easily discriminated from the hierarchical structure of

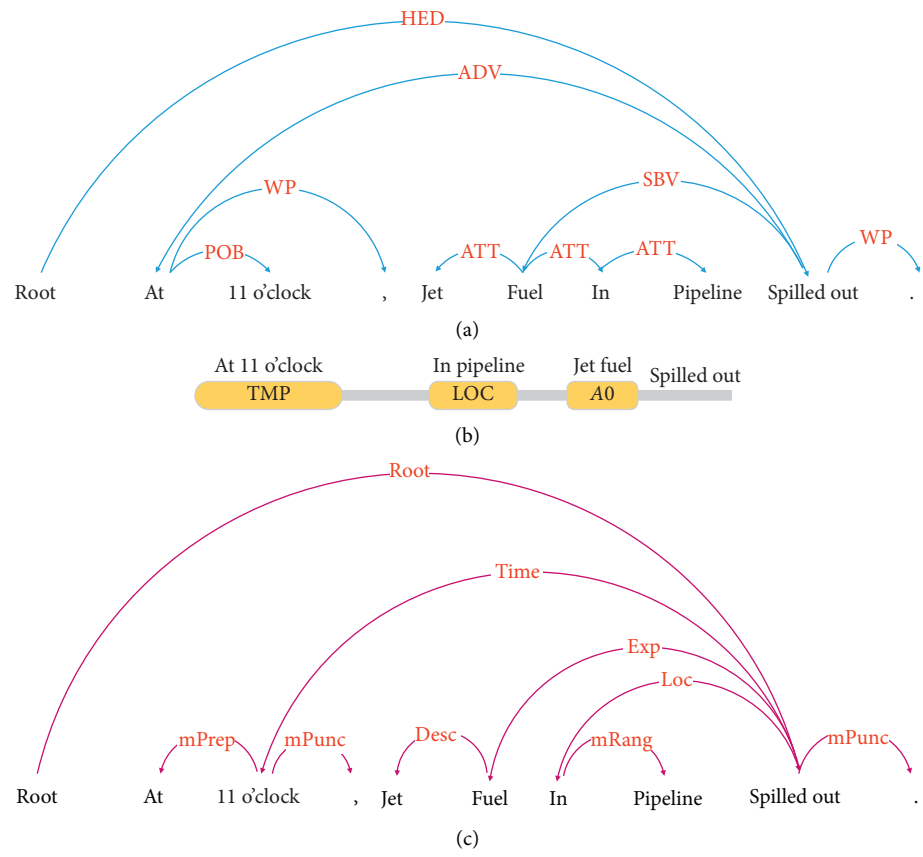

Figure 3: An example of event identification.

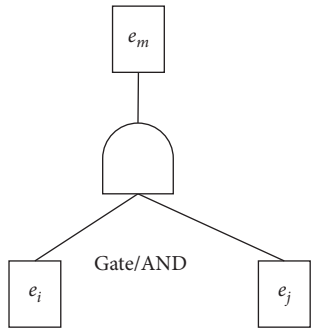

Figure 4: An example of a basic structure with AND gate.

event nodes in fault trees. However, there may be some hybrid information in the event nodes. Meanwhile, multiple events were occasionally described in one event node. Thus, some implicit causality may be hidden in the events of fault trees. Implicit causality should be extracted so as to build a correct CEEG. There are two steps in finding implicit causality. One is to investigate whether there is a causal relationship between two events and the other is to determine causal direction. The causal direction is used to describe which event is a cause and which one is the result. In this study, every two events in the fault tree nodes were assembled as candidate event pairs. By analyzing the internal structure of the events and semantic features of event sentences, we can identify the causal relationship and its direction with the help of our causal classifier.

Liu proposed an experience-based causality learning framework. Compared to traditional approaches, which attempt to handle the causality problem relying on textual clues and linguistic resources, they are the first to use experience information for causality learning [20]. Riaz focused on identifying and employing semantic classes of nouns and verbs with a high tendency to encode cause or noncause relations [21]. Zhao designed an abstract causality network and a dual cause-effect transition model. It is effective for discovering high-level causality rules behind specific causal events [22]. Zhao and Liu presented a new Restricted Hidden Naive Bayes model to extract causality from texts. It can cope with partial interactions among features so as to avoid overfitting problems on the Hidden Naive Bayes model, especially the interaction between the connective category and the syntactic structure of sentences [23]. A framework that combines intuitionistic fuzzy set theory and expert elicitation was proposed to enable quantitative analysis of temporal fault trees of dynamic systems with uncertain data [24].

In recent years, various types of neural networks and deep learning models have provided favorable support for the popularization and application of machine learning. For example, Deng proposed an improved quantum-inspired differential evolution method to construct an optimal deep

belief network, which is further applied to propose a new fault classification [25]. An improved ant colony optimization algorithm based on the multipopulation strategy, coevolution mechanism, pheromone updating strategy, and pheromone diffusion mechanism is proposed to balance the convergence speed and solution diversity and improve optimization performance in solving large-scale optimization problem [26]. Similar work about improved coevolution ant colony optimization algorithm with Multistrategies is presented in the literature [27]. Zhao extended a broad learning system based on the semisupervised learning of manifold regularization framework to propose a semi-supervised broad learning system. It can achieve higher classification accuracy for different complex data and takes on fast operation speed and strong generalization ability [28]. These methods are of great significance for us to mine and optimize causality by using neural networks.

In this study, we present a new method to obtain implicit causality by transforming the causality extraction into a binary classification problem. Four steps including internal structural features extraction of events, semantic features extraction of event sentences, feature fusion, and softmax classification were adopted to find implicit causality in a fault tree.

As shown in Figure 5, word (term) vector is first employed to express the lexical sequence feature of the event sentence. Then, BiGRU neural network is used to mine the context semantic features of the event sentence. To improve the accuracy of context semantic, we add the attention mechanism into the BiGRU model at the level of word and sentence. Finally, both semantic features and internal structure characteristics are input into softmax classifier to determine whether there are a causal relationship and the causal relationship direction between the given events.

4.2.1. Extraction of Internal Structure Features for Events. Internal structure features of events refer to the relationship characteristic of component elements in event pairs. Let $e_i = \{o_i, v_i, p_i, t_i\}$ be an event, where $0 \le i <= n$. $E = \{e_i\}$ is a set of events. $\forall e_i$ and $e_j \in E$, $<e_i, e_j>$ can form an event pair. Three internal structure features of event pairs were investigated in this section:

(1) Appearing probability: $P(e_i)$ was employed to represent the appearing probability of e_i. $Pc(e_i, e_j)$ is defined as the cooccurrence probability of e_i and e_j. Furthermore, $Pc(e_i \longrightarrow e_i)$ is the cooccurrence probability of e_i and e_j with the condition that e_i is the cause while e_j is the result. For the event elements, we present a group of appearing probability. $P(e_i.o)$ is used to express the appearing probability participant $e_i.o$. Similarly, $P(e_i.v)$, $P(e_i.p)$, and $P(e_i.t)$ are the appearing probability of trigger word, location, and timestamp of event, respectively.

(2) Pointwise mutual information: pointwise mutual information (PMI) is usually used to calculate the semantic similarity between two words [29]. The basic idea for PMI is to count the probability of two words simultaneously appearing in the text. Normally, two words are concluded with a high correlation for their higher PMI. Thus, PMI of events and their elements can be used to determine the correlation degree between two events. Definition of PMI for the event pairs and event elements can refer, respectively, to

$$\mathrm{PMI}(e_i, e_j) = \log \frac{P(e_i, e_j)}{P(e_i) * P(e_j)}, \tag{1}$$

$$\mathrm{PMI}(e_i.f, e_j.f) = \log \frac{P(e_i.f, e_j.f)}{P(e_i.f) * P(e_j.f)}, \quad f \in \{o, v, p, t\}. \tag{2}$$

(3) Position relevancy between events: events contained in fault tree nodes may exist in different sentences. Two sentences are generally considered with more dependence or causality if they are located closely. The distance between sentences is inversely proportional to the degree of relationship between the sentences. Paragraph sentences containing events are numbered sequentially from zero. Let TS be the total number of sentences in an accident report. $SP(e_i)$ is used to represent the number of the sentence including e_i. Relative position for an event pair $<e_i, e_j>$, namely, SPe_{ij}, is assigned as $SP(e_i) - SP(e_j)$. Position relevancy is defined as PRe_{ij}, $PRe_{ij} = 1 - SPe_{ij}/TS$.

We build a 19-v vector $ISFe_{ij}$ to express the internal structure features for event pair $<e_i, e_j>$. Here, $ISFe_{ij} = (P(e_i), P(e_j), P(e_i.o), P(e_j.o), P(e_i.v), P(e_j.v), P(e_i.p), P(e_j.p), P(e_i.t), P(e_j.t), Pc(e_i, e_j), Pc(e_i \longrightarrow e_j), Pc(e_j \longrightarrow e_i), PMI(e_i.o, e_j.o), PMI(e_i.v, e_j.v), PMI(e_i.p, e_j.p), PMI(e_i.t, e_j.t), PMI(e_i, e_j), PRe_{ij})$.

4.2.2. Extraction Semantic Feature in Event Sentences

(1) BiGRU Neural Network. Semantic dependence of two events can be obtained from event sentences. Semantic features of event sentences were taken as one of the features to identify event relations in our study. The tool "Word2vec" was used to train word embedding for the terms in the corpus of chemical accidents [30]. Then, event sentences can be expressed by the word embedding sequences. The word vectors were derived from the text training set of accident reports and some Internet accident news after denoising. Given a sentence consisting of n words, every word w is represented by a real-valued vector, and the vector of the sentence is represented as $S = (w_1, w_2, \ldots, w_n)$.

GRU neural network is a popular variant of the LSTM neural network. Compared with LSTM, GRU is with a more succinct structure [31]. GRU has only two control gates: update gate and reset gate. The information dissemination in GRU can be described as follows:

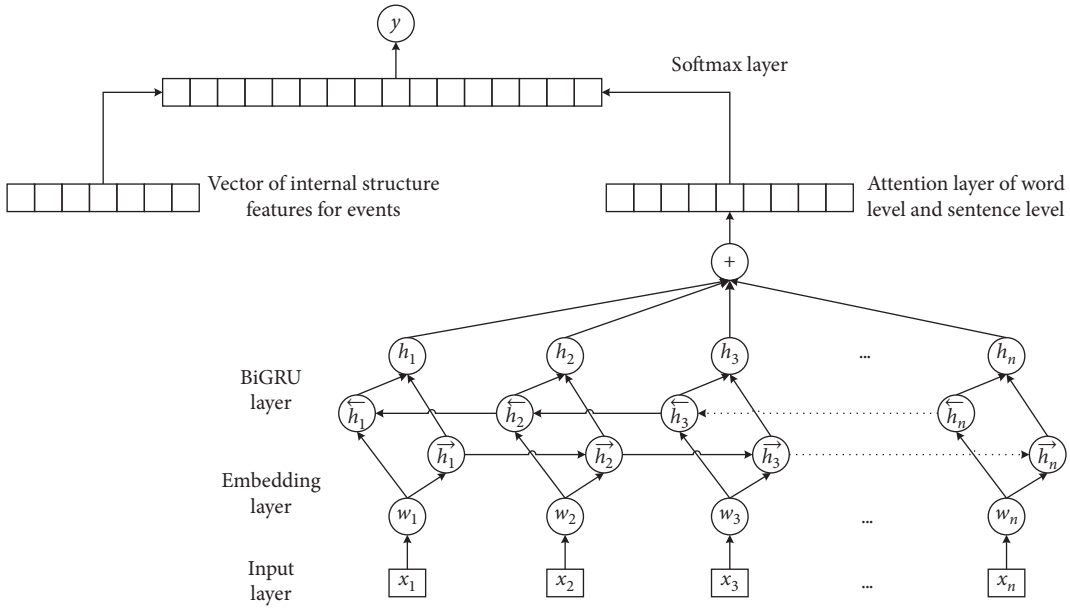

Figure 5: Extraction process of implicit causality.

(1) Update gate: the update gate z_t (see formula (3)) is used to control the extent to which the state information of the previous moment is brought into the current state. The larger the value of the update gate is, the more the state information of the previous moment can be brought in:

$$z_t = \sigma\left(W_z * \left[h_{t-1}, x_t\right]\right). \tag{3}$$

(2) Reset gate: reset gate r_t (see formula (4)) is used to control the degree of ignoring the state information of the previous moment. The smaller the value of reset gates is, the more the state information of the preceding moment is ignored:

$$r_t = \sigma\left(W_r * \left[h_{t-1}, x_t\right]\right). \tag{4}$$

Get a new hidden state; z_t and r_t jointly controlled how to obtain new hidden state $h_{t-1}$from the previously hidden state h_t as follows:

$$\widetilde{h_t} = \tanh\left(W * \left[r_t * h_{t-1}, x_t\right]\right), \tag{5}$$

$$h_t = \left(1 - z_t\right) * h_{t-1} + z_t * \widetilde{h_t}. \tag{6}$$

Compared with LSTM, GRU has the advantages of simple structure, fewer parameters, and fast training speed. It has shown a superior performance than LSTM. We use the accident text set to train the neural network. Event sentences in accident reports were first obtained according to the fault tree. Vectors of these event sentences were then input to the neural network to extract the semantic features of event sentences.

One-way neural network propagates from front to back, which can only contain the transmission of the previous information. The reverse transmission of the latter information cannot be propagated. The bidirectional neural network consists of two neural networks to train sequence forward and backward, respectively, and outputs two result sequences containing complete context information [32]. Here, we use the element-wise sum to combine the forward and backward pass outputs:

$$\overrightarrow{h_t} = \overrightarrow{\mathrm{GRU}}\left(wt, \overrightarrow{h_{t-1}}\right), \tag{7}$$

$$\overleftarrow{h_t} = \overrightarrow{\mathrm{GRU}}\left(wt, \overleftarrow{h_{t-1}}\right), \tag{8}$$

$$H = \overrightarrow{h_t} \oplus \overleftarrow{h_t}. \tag{9}$$

(2) Attention Mechanism. Attentive neural networks have recently demonstrated great success in a wide range of tasks, such as question answering, machine translations, and image recognition. We can apply attention computation for any two words in a sentence by introducing a self-attention mechanism. Thus, the dependence relationships of words in sentences can be learned more precisely. Word-level attention mechanism proposed by Zhou et al. [33] and sentence-level attention mechanism proposed by Lin et al. [34] for text representation have been widely concerned. In this section, we combine the above two methods to generate vectors for sentences.

In general, for an event pair $<e_p, e_q>$, e_p and e_q were located in different sentences. Assume that there are L sentences between the event e_p and e_q. The L sentences form a set Se_{pq}. Given a sentence Se_i in Se_{pq}, T_i was the number of words in sentence Se_i. w_{it} with $t \in [1, T_i]$ represents the t^{th}

word in Se_i. We obtain an annotation for a given word w_{it} by concatenating the forward hidden state and backward hidden state $hit = \overrightarrow{hit} \oplus \overleftarrow{hit}$. Once every word is assigned with weight, we can give an annotation for the sentence.

An activation function $\tan h(x)$ in formula (10) was used to handle hit. Then, we measure the importance of the word with a trained parameter vector W_1 and get a normalized importance weight α_{it} through a softmax function. Sentence vector S_i can be obtained by using a weighted sum of all the word annotations with their weight by the following:

$$\alpha_{it} = \frac{\exp\left(W_1^T * \tan h\left(h_{it}\right)\right)}{\sum_t \exp\left(W_1^T * \tan h\left(h_{it}\right)\right)}, \tag{10}$$

$$s_i = \sum_t \alpha_{it} * h_{it}. \tag{11}$$

Here, $h_{it} \in R^{\mathrm{d^W} * \mathrm{t}}$, $\mathrm{d^W}$ is the dimension of the word vectors, W_1 is a trained parameter vector, and w_i^T is a transpose, $W_1 \in R^{1 * d^W}$, and $s_i \in R^{\mathrm{d^W} * 1}$.

We first feed the word annotation of S_i into a one-layer MLP so as to get u_i as a hidden representation of S_i. Formula (13) was adopted to compute the weight of a sentence. We compute the vector v_s for Se_{pq} that summarizes all the information of sentences containing the event pairs:

$$\mu_i = \tan h\left(W_s * s_i + b_s\right), \tag{12}$$

$$\alpha_i = \frac{\exp\left(u_i * r\right)}{\sum_i \exp\left(u_i * r\right)}, \tag{13}$$

$$v_s = \sum_i \alpha i * s_i. \tag{14}$$

(3) Layer Normalization. During the training process of a deep learning network, parameter changes will lead to the distribution variation of input data in the subsequent network. To solve the problem of data distribution variation in the training process of the middle layers, Ioffe proposed the BN algorithm [35]. For each batch, the sum input distribution is used to calculate the mean and variance, which are used to normalize the sum input of neuron in each training sample. This method significantly reduces the training time of the precursor neural network. However, the effect of batch normalization depends on the size of minibatch. It is necessary to count the first-order and second-order statistics of each minibatch in the running process, which cannot be widely used in RNN networks. Therefore, Ba et al. proposed the concept of layer normalization (LN), which reduces training time by calculating the mean and variance of the sum input on one-layer neurons [36]:

$$h_t = f\left[\frac{g}{\sigma^t + \zeta} \odot \left(a^t - \mu^t\right) + b\right]. \tag{15}$$

Here, at is the input parameters of each layer, μ^t is the average value of input data, and σ^t is the input variance. g and b are bias constants, f is a linear transformation, and ζ is a regularization parameter. In this study, the LN method was introduced into formulas (4)–(6) to improve the training speed of the GRU neural network.

4.2.3. Fusion of Features and Classification for Events. We have presented a method to obtain the internal structure features for events and semantic features in event sentences. In this section, we achieve the fusion of features and classification of causality.

There are three kinds of classification results for softmax classifier, which indicate whether two events have causality and causality direction. v_s is a sentence vector obtained from formula (14), v_e is the vector of event structure feature, and W_f is the model training parameter. y (see formula (16)) is used to express the classification result of two types of feature fusion:

$$y = \arg\max\left(\mathrm{softmax}\left(W_f * \left(v_s + v_e\right)\right)\right). \tag{16}$$

Meanwhile, cross-entropy was introduced to serve as a training objective function (see formula (17)). In formula (17), n is the number of sentences and θ represents all the parameters in the model:

$$H(\theta) = \sum_{i=1}^{n} \log p\left(r_i \mid s_i, \theta\right). \tag{17}$$

4.3. Algorithm for Causality Extraction. In this section, we summarize the main operating steps of our proposed method. An algorithm, namely, CEFTAR, is presented to extract causality from fault trees and chemical accident reports.

In te Algorithm 1, we first construct three sets. They are the set of events (ES), event pairs (EPS), and event pairs with causality (ECS). All these sets are initialized as empty sets. From line (3) to line (4), we use the popular word segmentation tool "Jieba" to obtain all the words in the chemical accident reports. So, we can get a corpus based on these words. Meanwhile, the tool "Word2vec" is employed to generate vectors for the words in the corpus. By traversing all the fault trees in FTS, we can add all the events into the ES (see line (5) to line (8)). In line (9), event pairs are generated with any combination of events in ES. All the event pairs are added to EPS.

For an event pair, we first extract explicit causality (see line (11) to line (12)). If two events are located in different hierarchical structures and connected with the same logic gate, they have explicit causality. Implicit causality will be further investigated once they are not with explicit causality. After analyzing the internal structural feature for the event pair, we construct $ISFe_{ij}$ and use v_e to represent the vector of $ISFe_{ij}$. Then, semantic features of sentences including the even pair are obtained by the following steps. We get all the sentences between the two events and compute the vector for these sentences based on BiGRU neural network. Finally, the combination vector of the internal structural feature and the sentence semantic feature is sent to a softmax classifier to decide whether the two events have implicit causality (see line (13) to line (18)). ECS is returned by the CEFTAR algorithm as the final result of causality. The meanings of parameters in all the formulas and symbol abbreviations are presented in Table 1.

Input: the set of fault trees (FTS) and accident reports (ARS).
Output: the set of event pairs with causality (ECS).
(1) Construct the set of events (ES) and event pairs (EPS).
(2) ECS = EPS = ES = Φ;
(3) Achieve word segmentation for ARS by the tool "Jieba" and build the corpus CA;
(4) For each word w in CA, train a vector for w by "Word2vec";
(5) For each $ft \in$ FTS
(6) For each $n_e \in ft.E$
(7) {Identify the event e in the node n_e;
(8) ES = ES ∪ {e}; }
(9) For $\forall e_i$ and $e_j \in$ ES, build event pair $<e_i, e_j>$ and EPS = EPS ∪ {$<e_i, e_j>$};
(10) For each $<e_i, e_j> \in$ EPS
(11) If $\exists g \in ft.G$, s.t.: $e_i \in I(g) \wedge e_j \in O(g)$ or $e_j \in I(g) \wedge e_i \in O(g)$ then ECS = ECS ∪ {$<e_i, e_j>$};
(12) Else { construct $ISFe_{ij}$ and use ve represent the vector of $ISFe_{ij}$
(13) compute Se_{ij};
(14) For each Se_p sentence in Se_{ij}
(15) Build the vector sp for Se_p, $S_p = \Sigma \alpha_{it}{}^* h_{it}$;
(16) Generate the vector vs. for Se_{ij}; $v_s = \Sigma \alpha_i{}^* S_i$;
(17) $y = argmax(\text{softmax}(W_f^*(v_s + v_e)))$;
(18) if ($y == 1$) then ECS = ECS∪{$<e_i, e_j>$}; }
(19) Return (ECS)
(20) }

Algorithm 1: The CEFTAR algorithm.

Table 1: Notations and meanings.

Notation	Meaning
$FT\ (V, G, E, v_0)$	Fault tree, where V is the set of nodes, G is the set of gates, E is the set of edges, and v_0 is the root node
V_M, V_L	Set of intermediate nodes and set of leaf nodes
$EEG = (V, E)$	The expression of event evolutionary graph, where V is the set of nodes and E is the set of edges
$\Psi(.)$	The function that returns the input events for a given logic gate
$\Gamma(.)$	The function that returns the output event for a given logic gate
$e = \{o, v, p, t\}$	Event e, where o, v, p, and t are used to represent the event participants, event trigger word, location, and timestamp of event occurrence, respectively
$SRL(.)$	Semantic role labeling function
$SDP(.)$	Semantic dependency parsing function
$DP(.)$	Dependency parsing function
$\pi_F(S, e)$	The function to judge whether e fails given the set S of failed BE
$P(.)$	Probability function
$Pc(.,.)$	The cooccurrence probability function
PMI	Pointwise mutual information
z_t	The update gate of GRU unit
r_t	The reset gate of GRU unit
x_t	The input of GRU unit
h_t	The hidden layer information at the current moment
h_{t-1}	The hidden layer information at the previous moment
$\tilde{h}_t$	The candidate hidden layer information at the current moment
W	The weight matrix
σ	The sigmoid activation function
$\tanh$	The tanh activation function
$\oplus$	The vector concatenating function
α_{it}	The normalized word weight of sentence s_i
s_i	The sentence vector
u_i	The hidden representation of sentence vector s_i
α_i	The normalized sentence weight of sentence set Se_{pq}
v_s	The vector for Se_{pq}
μ^t	The average value of input data
Σ^t	The input variance
G, b	The bias constants
$f(.)$	The linear transformation function
Z	The regularization parameter
H(.)	The cross-entropy function

Table 2: Value of optimal parameters in the model.

Item	Value
Learning rate	0.001
Batch-size	50
Gru-size	128
Dropout	0.7
Bias constant in LN: g	0.001
Number of iterations	200
Embedding size	200
Layer number	4
Regularization parameter in LN: ζ	0.0001
Bias constant in LN: t	0.001

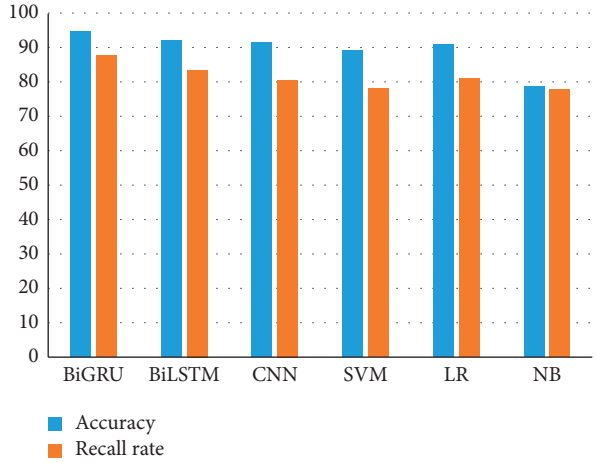

Figure 6: Accuracy and recall rate for different models.

5. Experiment and Analysis

In this section, we present experiments to validate the effectiveness of the proposed model and method. Our experiments were performed on the dataset which consists of 5867 accident reports and fault trees. Five experts in the domain of chemical accident analysis were employed to extract and annotate the causality in these reports and fault trees.

The hardware of the computer is as follows: CPU is i7-8700 with 3.2 GHz, six cores, and twelve threads. The memory is 16G. The Graphics card is GTX1060 with 6G. Tensorflow was adopted to implement the causal relationship extraction model in this study. Five rounds of experiments were performed and the average values were taken as the experimental results. A grid search algorithm is used to test the combination of different parameters to determine the optimal parameters for our model. The values of optimal parameters in our model are shown in Table 2.

We compared our model with other frequently used machine learning or neural network models to show its advantages. From Figure 6, we can see that our model is with higher accuracy and recall rate in extracting causality than BiLSTM, CNN, SVM, LR, and NB. We can see that the accuracy and recall rate of BiGRU, BiLSTM, and CNN are higher than those of SVM, LR, and NB. It is because the neural network model is superior to the traditional machine learning model for mining the hidden features. BiGRU and BiLSTM had higher accuracy and recall rate than CNN since LSTM networks can better capture context features for long text sequences, while CNN is suitable for capturing local features.

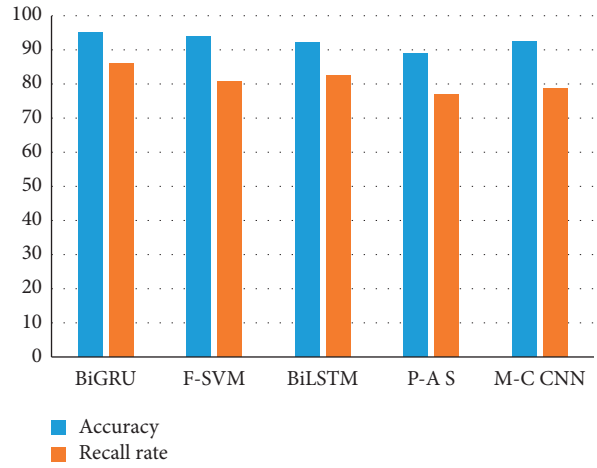

Figure 7: Accuracy and recall rate for different methods.

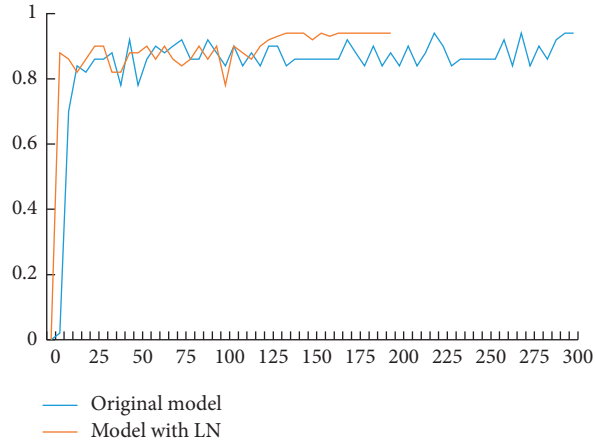

Figure 8: The effect of LN layer normalization on model performance.

Four state-of-the-art methods including Feature-SVM (F-SVM) [8], BiLSTM [37], pattern-argument semantics (P-A S) [38], and Multicolumn CNN (MCCNN) [39] were also executed on the same dataset to obtain causality. As shown in Figure 7, the accuracy and recall rate of our method is the highest one. Thus, experimental results show that our proposed model and method in extracting causality are superior to the existing methods.

Two data curves are shown in Figure 8, in which the abscissa is the number of running steps and the ordinate is the model accuracy. We can see that the LN layer normalization accelerated network convergence and reduced operation time and cost.

6. Conclusions

CEEG is an EEG describing the evolution process of chemical accidents. We can easily obtain evolution sequences of events in chemical accidents. Safety analysis, early warning, and emergency disposal can be performed based on these evolution sequences. To accurately and easily obtain the causality in building CEEG, a method to extract causality for safety events in chemical accidents from fault trees and accident reports is proposed in this paper.

We propose an effective method to extract events and their elements by combining fault tree with accident reports. Causality between these events is divided into explicit causality and implicit causality. We obtain explicit causality by analyzing hierarchical structure relations of event nodes and logic gates in fault trees. Implicit causality is generated based on BiGRU neural network by feeding internal structural features of events and semantic features of event sentences. Experimental results show that the proposed method conduces to better performance in accuracy and recall rate during the process of extracting causality.

In future work, more elements of events affecting chemical accidents will be taken into consideration, such as the environment, weather, and policy guidance factors. The accuracy will be further increased after more elements are adopted to model the events. Meanwhile, more cases of chemical accidents will be collected so as to enrich the training dataset. The proposed method will get better performance after adjusting the optimal model parameters with more abundant data available.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported the Key Research Program of Shandong Province under Grant 2018GGX101052, the Natural Science Foundation of China under Grant 61973180, and the Natural Science Foundation of Shandong Province under Grant ZR2019MF033.

References

[1] W. Wang, J. Bao, and S. Yuan, "Proposal for planning an integrated management of hazardous waste: chemical park, Jiangsu Province, China," *Sustainability*, vol. 11, no. 10, pp. 28–46, 2019.

[2] L. Fyffe, S. Krahn, J. Clarke, D. Kosson, and J. Hutton, "A preliminary analysis of key issues in chemical industry accident reports," *Safety Science*, vol. 82, pp. 368–373, 2016.

[3] M. Yazdi, S. Kabir, and M. Walker, "Uncertainty handling in fault tree based risk assessment: state of the art and future perspectives," *Process Safety and Environmental Protection*, vol. 131, pp. 89–104, 2019.

[4] T.-H. Lee, D.-J. Lee, and C.-H. Shin, "Characteristic analysis of casualty accidents in chemical accidents," *Fire Science and Engineering*, vol. 31, no. 1, pp. 81–88, 2017.

[5] M. Versaci, "Fuzzy approach and Eddy currents NDT/NDE devices in industrial applications," *Electronics Letters*, vol. 52, no. 11, pp. 943–945, 2016.

[6] P. Burrascano, S. Callegari, A. Montisci, M. Ricci, and M. Versaci, *Ultrasonic Nondestructive Evaluation Systems*, Springer, Berlin, Germany, 2015.

[7] C. Joshi, F. Ruggeri, and S. P. Wilson, "Prior robustness for bayesian implementation of the fault tree analysis," *IEEE Transactions on Reliability*, vol. 67, no. 1, pp. 170–183, 2018.

[8] W. Dai, L. Riliskis, P. Wang, V. Vyatkin, and X. Guan, "A cloud-based decision support system for self-healing in distributed automation systems using fault tree analysis," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 3, pp. 989–1000, 2018.

[9] Z. Li, S. Zhao, X. Ding, and T. Liu, "EEG: knowledge base for event evolutionary principles and patterns," in *Proceedings of the Chinese National Conference on Social Media Processing*, pp. 40–52, Beijing, China, September 2017.

[10] E. Ruijters and M. Stoelinga, "Fault tree analysis: a survey of the state-of-the-art in modeling, analysis and tools," *Computer Science Review*, vol. 15-16, pp. 29–62, 2015.

[11] A. Rauzy and C. Blériot-Fabre, "Towards a sound semantics for dynamic fault trees," *Reliability Engineering & System Safety*, vol. 142, pp. 184–191, 2015.

[12] A. Skarlatidis, G. Paliouras, A. Artikis, and G. A. Vouros, "Probabilistic event calculus for event recognition," *ACM Transactions on Computational Logic*, vol. 16, no. 2, pp. 1–37, 2015.

[13] Y. Chen, L. Xu, K. Liu, D. Zeng, and J. Zhao, "Event extraction via dynamic multi-pooling convolutional neural networks," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, (Volume 1: Long Papers)*, pp. 167–176, Beijing, China, July 2015.

[14] X. Feng, B. Qin, and T. Liu, "A language-independent neural network for event detection," *Science China Information Sciences*, vol. 61, no. 9, Article ID 092106, 2018.

[15] T. Liao, W. Fu, S. Zhang, and Z. Liu, "Event recognition oriented to emergency events and its application," in *International Conference on Applications and Techniques in Cyber Security and Intelligence*, pp. 1375–1384, Springer, Cham Switzerland, 2019.

[16] F. Hogenboom, F. Frasincar, U. Kaymak, F. De Jong, and E. Caron, "A survey of event extraction methods from text for decision support systems," *Decision Support Systems*, vol. 85, pp. 12–22, 2016.

[17] L. Mei, H. Huang, X. Wei, and X. Mao, "A novel unsupervised method for new word extraction," *Science China Information Sciences*, vol. 59, no. 9, pp. 92–102, 2016.

[18] E. Cambria and B. White, "Jumping NLP curves: a review of natural language processing research," *IEEE Computational Intelligence Magazine*, vol. 9, no. 2, pp. 48–57, 2014.

[19] http://www.ltp-cloud.com/.

[20] Y. Liu, S. Wang, J. Zhang, and C. Zong, "Experience-based causality learning for intelligent agents," *ACM Transactions on Asian and Low-Resource Language Information Processing*, vol. 18, no. 4, pp. 1–22, 2019.

[21] M. Riaz and R. Girju, "Recognizing causality in verb-noun pairs via noun and verb semantics," in *Proceedings of the EACL 2014 Workshop on Computational Approaches to Causality in Language (CAtoCL)*, pp. 1–10, Gothenburg, Sweden, 2014.

[22] S. Zhao, Q. Wang, S. Massung et al., "Constructing and embedding abstract event causality networks from text snippets," in *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining—WSDM '17*, pp. 335–344, Cambridge, UK, February 2017.

[23] S. Zhao, T. Liu, S. Zhao, Y. Chen, and J.-Y. Nie, "Event causality extraction based on connectives analysis," *Neurocomputing*, vol. 173, pp. 1943–1950, 2016.

[24] S. Kabir, T. K. Geok, M. Kumar, M. Yazdi, and F. Hossain, "A method for temporal fault tree analysis using intuitionistic fuzzy set and expert elicitation," *IEEE Access*, vol. 8, pp. 980–996, 2020.

[25] W. Deng, H. Liu, J. Xu, H. Zhao, and Y. Song, "An improved quantum-inspired differential evolution algorithm for deep belief network," *IEEE Transactions on Instrumentation and Measurement*, 2020.

[26] Wu Deng, J. Xu, and H. Zhao, "An improved ant colony optimization algorithm based on hybrid strategies for scheduling problem," *IEEE Access*, vol. 7, pp. 20281–20292, 2019.

[27] H. Deng, L. Peng, H. Zhang, B. Yang, and Z. Chen, "Ranking-based biased learning swarm optimizer for large-scale optimization," *Information Sciences*, vol. 493, pp. 120–137, 2019.

[28] H. Zhao, J. Zheng, W. Deng, and Y. Song, "Semi-supervised broad learning system based on manifold regularization and broad network," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 67, no. 3, pp. 983–994, 2020.

[29] F. H. Khan, U. Qamar, and S. Bashir, "SentiMI: introducing point-wise mutual information with SentiWordNet to improve sentiment polarity detection," *Applied Soft Computing*, vol. 39, pp. 140–153, 2016.

[30] D. Zhang, H. Xu, and Z. Su, "Chinese comments sentiment classification based on word2vec and SVMperf," *Expert Systems with Applications*, vol. 42, no. 4, pp. 1857–1863, 2015.

[31] C. Yu, S. Wang, and J. Guo, "Learning Chinese word segmentation based on bidirectional GRU-CRF and CNN network model," *International Journal of Technology and Human Interaction*, vol. 15, no. 3, pp. 47–62, 2019.

[32] R. Zhao, D. Wang, R. Yan, K. Mao, F. Shen, and J. Wang, "Machine health monitoring using local feature-based gated recurrent unit networks," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 2, pp. 1539–1548, 2017.

[33] P. Zhou, W. Shi, and J. Tian, "Attention-based bidirectional long short-term memory networks for relation classification," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 207–212, Berlin, Germany, August 2016.

[34] Y. Lin, S. Shen, Z. Liu, H. Luan, and M. Sun, "Neural relation extraction with selective attention over instances," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2124–2133, Berlin, Germany, August 2016.

[35] S. Ioffe, "Batch renormalization: towards reducing minibatch dependence in batch-normalized models," in *Advances in neural information processing systems*, pp. 1945–1953, American Institute of Physics, College Park, MD, USA, 2017.

[36] Ba J. L., Kiros J. R., Hinton G. E, Layer normalization, 2016.

[37] Y. Zhang, P. Li, and G. Zhou, "Classifying temporal relations between events by deep BiLSTM," in *Proceedings of the 2018 International Conference on Asian Language Processing (IALP)*, pp. 267–272, Bandung, Indonesia, November 2018.

[38] L. I. Pei-Feng, Z. Guo-Dong, and Z. Qiao-Ming, "Semantics-based joint model of Chinese event trigger extraction," *Journal of Software*, vol. 27, no. 2, pp. 280–294, 2016.

[39] C. Kruengkrai, K. Torisawa, C. Hashimoto, J. Kloetzer, J. Oh, and M. Tanaka, "Improving event causality recognition with multiple background knowledge sources using multi-column convolutional neural networks," in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, pp. 3466–3473, San Francisco, CA, USA, February 2017.