IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 17, 2020, accepted February 26, 2020, date of publication March 2, 2020, date of current version March 12, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2977362

Smart Contract Classification With a Bi-LSTM Based Approach

GANG TIAN[1], QIBO WANG[1], YI ZHAO[2], LANTIAN GUO[3], ZHONGLIN SUN[1], AND LIANGYU LV[1]

[1]School of Computer Science and Engineering, Shangdong University of Science and Technology, Qingdao 266590, China
[2]School of Mathematics and Computer Science, Guangdong Ocean University, Zhanjiang 524088, China
[3]School of Automation and Electronic Engineering, Qingdao University of Science and Technology, Qingdao 266044, China

Corresponding author: Gang Tian (gtian@sdust.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61702305, Grant 11971270, and Grant 61903089, in part by the China Postdoctoral Science Foundation under Grant 2017M622234, and in part by the Science and Technology Support Plan of Youth Innovation Team of Shandong higher School under Grant 2019KJN2014.

ABSTRACT With the number of smart contracts growing rapidly, retrieving the relevant smart contracts quickly and accurately has become an important issue. A key step for recognizing the related smart contracts is able to classify them accurately. Different from traditional text, the smart contract is composed of several parts: source code, code comments and other useful information like account information. How to make good use of those different kinds of features for effective classification is a problem need to be solved. Inspired by this, we proposed a smart contract classification approach based on Bi-LSTM model and Gaussian LDA, which can use a variety of information as inputs of the model, including source code, comments, tags, account and other content information. Bi-LSTM is utilized to capture grammar rules and context information in source code, while Gaussian LDA model is employed to generate comments feature where the semantics of the comments are enriched by embeddings. We also use attention mechanism to focus on the more relevant features in smart contracts for tags and fuse account information to provide additional information for classification. The experimental results show that the classification performance of the proposed model is superior to other baseline models.

INDEX TERMS Smart contract classification, Bi-LSTM, attention mechanism, Gaussian LDA, account information.

I. INTRODUCTION

Blockchain technology has exhibited exciting prospects in games, lottery, medical service, finance and other aspects. The typical blockchain technology like Bitcoin implements a stack-based programming language for applications such as verifying public keys and signatures. However, Bitcoin's programming language has defects including Turing completion, blockchain data missing, state missing, and client incompatibility [1].

In this context, platforms such as Ethereum have developed a programming language Solidity to solve the Turing completion problem. Users can use Solidity to write specific smart contracts. Smart contract is defined as an event-driven, stateful program that runs on a replicated, shared ledger and is able to hold assets on the ledger [2]. Compared with traditional financial contracts, smart contracts have lower legal overhead and transaction overhead, and are convenient for users to use [3]. Smart contracts have been successfully implemented on many blockchain platforms such as Ethereum, Fabric and Corda [4]. In this paper, we focus on the smart contract of Ethereum.

Alongside the continuous development and application of blockchain technology, the number of smart contracts deployed on the blockchain platform has grown exponentially [5]. There are more than 200 smart contract applications on the Ethereum platform, and the average number of smart contracts released per month is close to 100,000 [6]. For users, how to quickly and accurately retrieve the right application service in a large number of smart contract applications is an important issue that needs to be solved. The classification of smart contracts is a means to effectively improve the

The associate editor coordinating the review of this manuscript and approving it for publication was Yongqiang Zhao.

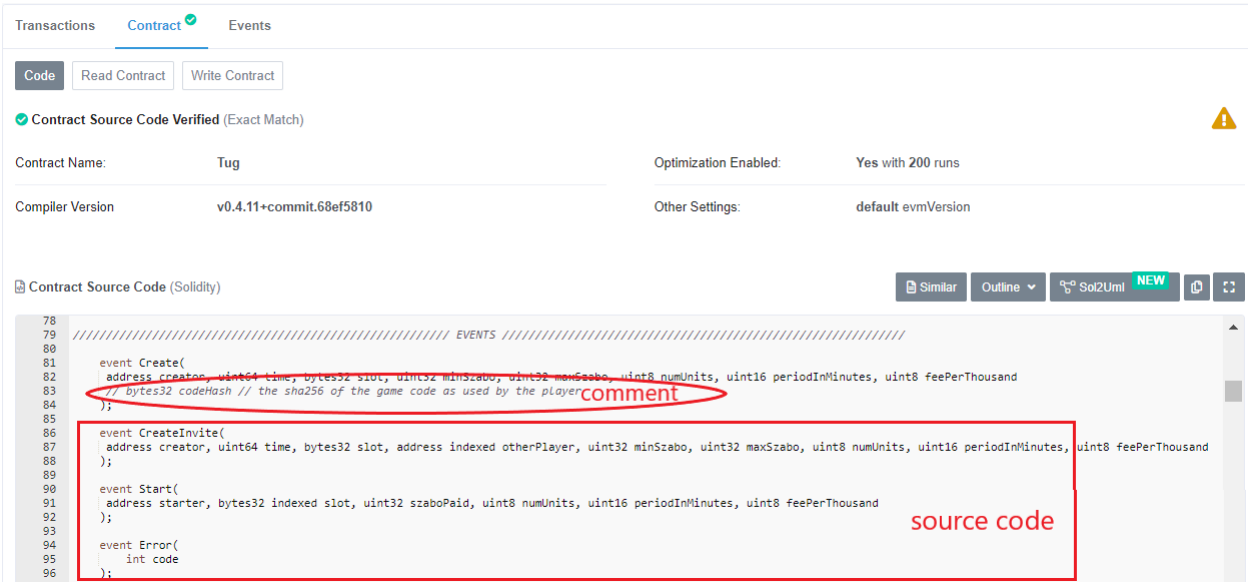

FIGURE 1. The composition of the smart contract.

search of smart contracts [1]. Therefore, it is necessary to design an effective classification model to realize effective classification of smart contracts.

However, to achieve effective classification of smart contracts, the following problems are mainly confronted with:

1) As shown in Fig. 1, a smart contract contains source code and comments. The grammar rules and context information in the source code need to be captured, and comments contain semantic information, but they are short and lack of statistical information.
2) Smart contracts are distinguished from traditional text. Traditional text feature representation ignores grammatical rules and contextual behavior information in the source code [7].

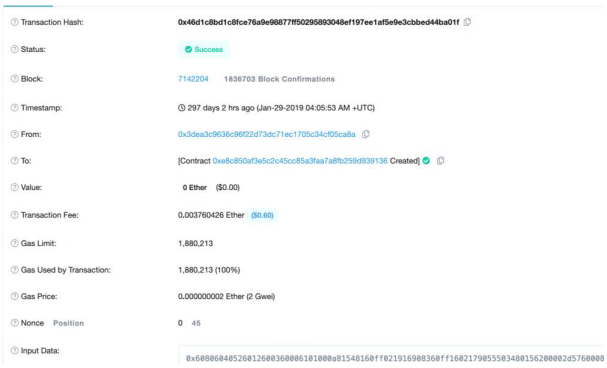

FIGURE 2. Account information.

3) As shown in Fig. 2, each smart contract corresponds to one account, and the account information contains relevant information. If only the source code, comments and tags are used during the establishment of the classification model, some important information of the smart contract will be lost, which will affect the classification effect [7].
4) As shown in Fig. 3, the smart contract dataset contains tags. How to use tag information to improve the classification effect is a challenge.

For problem 1, LSTM can capture long-term memory of the input and can be used to discover the internal structure and dependencies. In addition, LSTM can handle inputs of different lengths very well. The sentence representation generated by LSTM captures the dependency structure between sentences and the similarity between words, so it can better represent the semantics of sentences.

RNN based models such as GRU or MinimalRNN have fewer parameters and are easier to achieve convergence. When the amount of data is small, LSTM and GRU have almost the same effect. But for our dataset, the amount of data is large. At this time, LSTM has more parameters and it can achieve better performance.

RNN based algorithms such as LSTM-RNN can capture long-term dependencies well, and can selectively memorize or forget some unimportant information through training. But it cannot capture the semantic information from back to front, which will have a greater impact on the accuracy of classification.

Shi *et al.* [8] proposed a service recommendation approach based on text expansion and deep model. They first used a probabilistic topic model to expand services' description at sentence level. The model learns words' topics, sentences and descriptions in a stratified fashion. Then they proposed a LSTM-based model with two attention mechanisms to recommend services. One of the attention mechanisms can take tags as functional prior to get function-related features of services and Mashups. The other one considers Mashup requirements as application scenario to find best services.

FIGURE 3. Smart contract tags.

Besides, Cao *et al.* [9] used Bi-LSTM to learn the feature representations of Web services. After training the web service documents to get topic vectors, they used the topic attention strengthening processing for service feature representations and obtained the weight of the words in document. Finally, they input the service feature representations into neural network to predict the category of web services.

Therefore, we use Bi-LSTM to obtain the association between context information and non-continuous words in smart contracts for capturing semantic dependence.

For problem 2, there exist many approaches to enrich semantic representation of short text by transmitting external information. For instance, A transfer learning method is proposed by Jin *et al.* [10], which clusters short texts with auxiliary long texts. A short text clustering method based on world knowledge is proposed by Hu *et al.* [11]. Above works generally enrich the representation of service descriptions based on an implicit assumption that the auxiliary information and object text are semantically related. However, the assumption is not realistic as it is difficult to find such auxiliary information in real text. Besides, Many of the service discovery methods (such as LDA, LSA, etc) involved in traditional information retrieval (IR) models usually use vector spaces as feature representations. These methods may suffer from dimensionality curse due to the sparse representation of short text [12], [13].

To address this problem, we use an integrated word embedding and Gaussian LDA (GLDA) model to improve the performance of comments modeling [14]. The semantic features of words in text can be obtained by word embedding technology. Words with similar semantic and syntactic properties are often close in embedded vector spaces [15]. Thus, this characteristic allows for efficient modeling of contextual information such as word co-occurrence patterns that are used to enrich the semantics of comments in a smart contract. GLDA is a high-level topic model, which regards the input document as a set of embedded representations, and regards the learning topic as a multiple Gaussian distribution in the embedded space [16]. Therefore, using GLDA model, we can effectively model comment representation as topic representation.

For problem 3, we extract features of account and content information. Putting them into the model to improve the classification effect.

For problem 4, we use attention mechanism to focus smart contracts on tag information. Tang *et al.* [17] use an attention based LSTM network, where the attention mechanism enables the model to focus on key parts of the sentence that modulate the sentiment of the aspects. Hazarika *et al.* [18] input aspect information into the attentional process by concatenating the previous word with the aspect representation. Finally, we propose a mechanism of attention, we concatenate word embeddings and tag information into the attention process. Also, this allows the attention mechanism to focus on the relevant word in smart contracts for tags. Using attention mechanism to capture the key parts and give them higher weights in contracts.

We combine the above four methods to propose a new model based approach, called SCC-BiLSTM(Bi-LSTM for Smart Contract Classification).

In this paper, the main contributions as follows:

1) Comments are pre-trained into embeddings and represented as classification features by Gaussian LDA which utilizes embeddings to enrich the semantics of comments.
2) The contract account information that contain relevant contract address, balance, related transactions, data storage and other information is extracted and added into the model for classification.
3) Using Bi-LSTM with attention mechanism to obtain the association between context information and non-continuous words in smart contracts.
4) We compare our proposed model with other baseline methods through experiments, the proposed model has better performance than other baseline methods in *Precision,Recall and F-score*

II. RELATED WORK

A. SMART CONTRACT CLASSIFICATION MODEL

At present, there are few relevant research works on smart contract classification. Huang et al. proposed a method

that incorporates smart contract and account information. This method first obtains global vector representation using LSTM, whose inputs are word embeddings trained by smart contracts. Then, the generated global vector representation and account information are inputted into a feedforward neural network to obtain classification results [1].

To obtain the maximized feature information of the smart contract, Wu et al. employed a Bi-LSTM network to model the smart contract source code and account information simultaneously. In the feature learning process, the attention mechanism is used from the word level and the sentence level respectively, focusing on capturing words and sentences that are important for the classification of smart contracts. Finally, connect contract features and account features to generate the document-level feature representation of the smart contract, and finally get the classification result with the Softmax layer [7].

Gao designed a smart contract automatic classification system based on blockchain technology. According to the six-layer structure of the system, the weights of the term items in the smart contract are normalized, and the feature extraction method is used to filter out the contract terms with poor performance in the smart contract, and the representative feature terms in the contract are selected. Then calculate the similarity of the smart contract, automatically identify the smart contract type according to the content of the smart contract, and complete the intelligent contract automatic classification system [19].

Luu et al. discussed three kinds of security bugs: Transaction-Ordering Dependence, Timestamp Dependence, Mishandled Exceptions. They used a symbolic execution tool which analyses current Ethereum smart contracts to detect bugs [5]. And in [20], Omohundro et al. analyzed the relationships of smart contracts and study the smart contract clustering issues.

B. ETHEREUM ACCOUNT INFORMATION

However, there are two types of accounts on the Ethereum platform: externally owned accounts and contract accounts. Both accounts can be used to check balances, send transactions, etc. Nevertheless, the account is generated after creating the smart contract and controlled by it. Account information including Nonce, Banlance, etc. is associated with smart contract [21].

C. TRADITIONAL TEXT CLASSIFICATION METHODS

Traditional text classification mainly uses methods based on machine learning [22] and deep learning methods. The text classification method based on dictionary or machine learning mainly extracts, generates and constructs feature sets from texts in combination with prior knowledge, and then uses the feature information as input data to train a classifier for text classification. Text classification methods based on neural networks mainly use LSTM to classify texts [23]. But smart contracts are distinguished from traditional text. A smart contract contains source code and code comments which both contain semantic information. But the comments are short and have a problem of semantic sparsity.

D. SEMANTIC ENHANCEMENT METHODS

Therefore, the integration of external information has been widely used to enrich the semantic representation of short texts. For example, Hu *et al.* [11] obtained improved short text clustering results obtained by world knowledge. Through transferring knowledge from auxiliary long text, Jin *et al.* [10] developed a short text clustering method based on transfer learning. These methods can partially alleviate the problem of semantic sparsity.

LDA [24] is a generative topic model that represents the hidden structure of a document collection. It is assumed that each document has a topic distribution in the corpus in LDA, the discrete topic distribution in corpus is extracted from the symmetrical Dirichlet distribution.

And [25] proposed Gaussian LDA (GLDA) model which is different from the LDA model. GLDA can model the words in a document as embedded word sequences. In GLDA [25], the words of input are converted to continuous vectors. Therefore, each generated topic can be represented as a multivariate Gaussian distribution. By analyzing the semantic similarity of embedded word vectors, it is proved that gaussian parameterization is reasonable [25]. This can help merge word embedded sets, effectively improving the performance of topic modeling. Thus, we use GLDA instead of LDA to take advantage of word embedding and probability model.

E. ATTENTION MECHANISM

Attention mechanism is an integral part of compelling sequence modeling and transduction models in various tasks, allowing modeling of dependencies without regard to their distance in the input or output sequences.An attention function can be described as mapping a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors. The output is computed as a weighted sum of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key [26].

Attention mechanism has been widely applied to and attained significant improvement in various tasks in natural language processing, including sentiment classification, text summarization, etc. Attention mechanism can guide attention to the correct location using latent clues within the context [27].

As for tags, tags can provide meaningful object descriptions and allow users to organize and index their contents. Tagging data have been proven useful in many fields, such as information retrieval(IR), data mining, etc [28]. An approach proposed by Zhou et al. to polarity classification based on sentiment tags. By building Sentiment-Topic model, to extract the sentiment tags of the text. The experimental results prove that sentiment tags have the effect of improving classification accuracy [29].

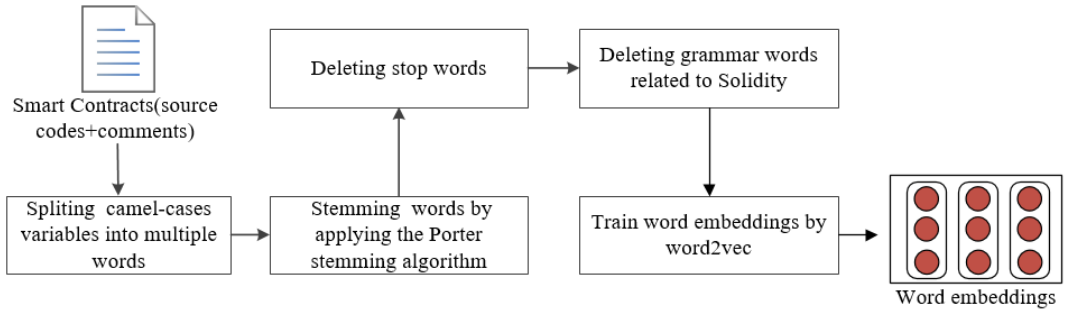

FIGURE 4. Training source codes and comments of smart contracts into word embedding.

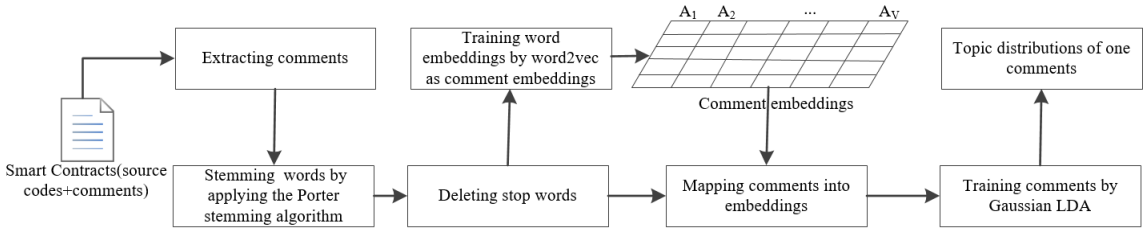

FIGURE 5. The Process of GLDA Modeling.

III. FRAMEWORK

In this section, the framework of our approach is proposed first in section III-A. Then, comments representation using GLDA are described in Section III-B. In Section III-C, the attention mechanism is utilized to focus on relevant segments in the smart contract with respect to the tags.

A. FRAMEWORK

The proposed smart contract classification model consists of three main steps: data preprocessing, GLDA modeling and model training.

1) DATA PREPROCESSING

The data crawled on the State of the Dapps[1] website contains tags, but the data crawled on the Ethernum[2] website do not involve tag information. So we need to tag recommendation for these smart contracts without tags. We first summarize all the tags in the same class of smart contract crawled on the State of the DAPPS as original tags, and use TF-IDF to calculate the frequency of these tags in each smart contract of the same class that does not contain tags, and select 4-6 tags with the highest frequency as smart contracts original tags without tags. Then using tag-co-occurrence to generate candidate tags based on original tags [28], and obtaining the recommended tags by using some tag ranking strategies [30]. Finally, we take the original tag and the tag generated through tag recommendation as the tags of a smart contract.

The account information we selected includes: smart contract balance, smart contract Ether value, transaction fee, timestamp, block number, number of smart contract transactions and status. In addition, we also extract content information from smart contracts. The content information include: the length of the smart contract, the number of lines, the contract name, and the frequency of the grammar words (e.g., ''function'' or ''public'').

As shown in Fig. 4, for each smart contract, we first split camel-cases variables into multiple words. For example, the variable named ''receiveApproval'' is divided into ''receive'' and ''Approval''. Then after stemming words and deleting stop words, we use Word2Vec[3] to train smart contracts into word embeddings $S \in R^{128*n}$.

2) GLDA MODELING

We extract comments starting with ''/*'' and ending with ''* /'' or starting with ''//'' and ending with line break ''\n'' from the smart contract. As shown in Fig. 5, after stemming words and removing stop words, these comments are used as input to the Word2Vec model and are trained to get the word embedding, we call it comment embedding. Alternatively, pre-trained word embedding set, such as trained word embedding set using general large scale corpus (i.e.Wikipedia[4]) can also be introduced directly. Once the comment embedded set is obtained, the words in the comment can be mapped to the embedded representation and further used as input to GLDA. What GLDA do is to model each word in comments as the topic representation through latent factors.

[1]https://www.stateofthedapps.com/
[2]https://etherscan.io/
[3]https://code.google.com/archive/p/word2vec/
[4]https://dumps.wikimedia.org/enwiki/latest/enwiki-latest-pagesarticles.xml.bz2

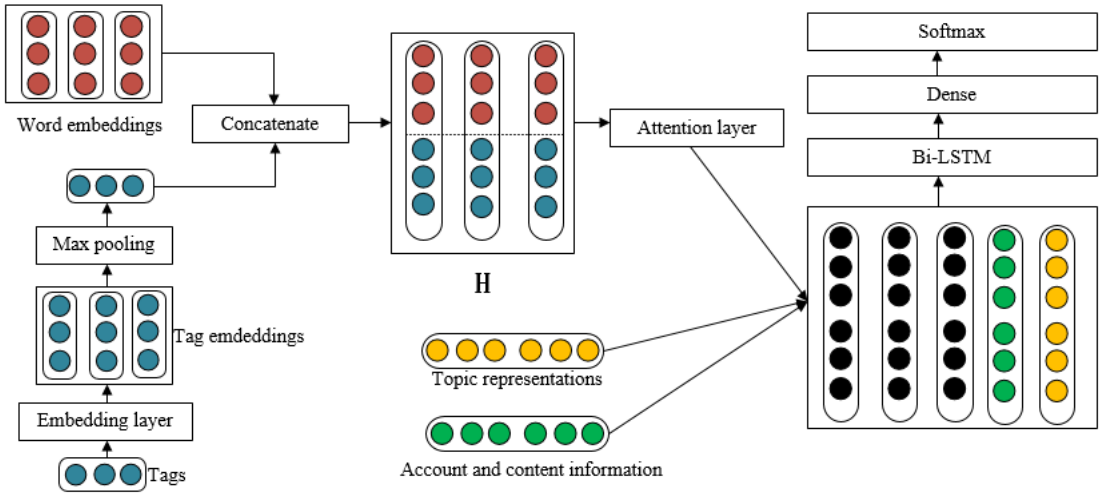

FIGURE 6. The Framework of Smart Contract Classification.

3) MODEL TRAINING

As shown in Fig. 6, we combine word embeddings, tags, topic representations, account and content information as input of our model which outputs classification result by Softmax classifier. The final output is the class of the smart contract and its corresponding probability. The loss function is shown in Equation 1

$$Loss = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{Z} y_{i,j} log p_{i,j} \tag{1}$$

where N is the size of train set and Z is the number of categories. $y_{i,j}$ represents the j-th element in the i-th label's one-hot vector(e.g.,0 or 1). $p_{i,j}$ denotes the probability associated with the category j.

B. GLDA

The reason we use GLDA instead of RNN is to enhance the semantics of comments in contracts.The comments of smart contracts are short and lack of statistical information which lead to poor recall.To address this issue, external information is introduced to enhance the semantic of comments. The proposed approach uses GLDA, which integrates word embedding seamlessly.The word embeddings have been shown to capture lexico-semantic regularities in language,which are external information for semantic sparsity.

In addition, GLDA model can also encode a prior preference for semantically coherent topics since Gaussian distributions capture a notion of centrality in space and semantically related words are localized in space [25].

Using the GLDA model, all comments in the same contracts can be represented as random mixes of potential topics, and from the Dirichlet Prior to extract its proportion. The process through GLDA to get topic representations of all comments in same smart contracts is shown in Fig. 7. And the GLDA generation process of contracts is as follow:

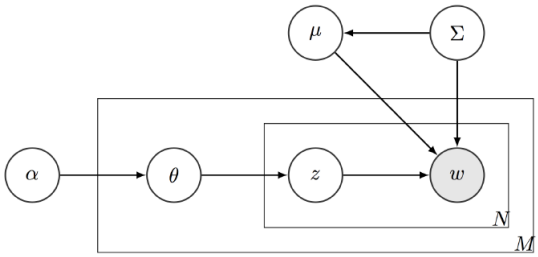

FIGURE 7. Graphical Model of GLDA.

1) for topic $k = 1$ to K
   (a) Draw topic covariance $\sum_k \sim W^{-1}(\psi, v)$
   (b) Draw topic mean $\mu_k \sim Normal(\mu, \frac{1}{k}\Sigma_k)$
2) for each c contract in contract C
   (a) Draw topic covariance $\theta_c \sim Dir(\alpha)$
   (b) for each word index i from 1 to N_c
      [i] Draw a topic $z_{(c,i)} \sim Multinomial(\theta_c)$
      [ii]Draw a embedding $v_{(c,i)}$ with a probability $\frac{v_{(c,i)}}{z_{(c,i)}}, u_{1..K}, \sum_{1..K} \sim Normal(\mu_{z_{(c,i)}}, \sum_{z_{(c,i)}})$

where k is denoted as a multivariate Gaussian with mean u_k and covariance $\sum_k$, c denotes each contract,$Dir(\alpha)$ is the Dirichlet distribution with parameter α. As shown in Fig. 8, a hierarchical representation model for smart contracts is constructed after using GLDA. An embedding e for each word w in a smart contract c is associated with a topic z. Based on above two distributions, two layers the Contract-Topic and the Topic-Embedding are generated as work [24].

After training with the Word2vec model, each word w(e.g.,"game" and "code") is represented as a vector of a fixed length. For instance, the word "player" can be represented as a vector with a size of 128-[0.28 -0.32 -0.16...-0.07]. Prior to the use of GLDA, each word in contracts could be represented by a vector of Word2Vec training. Therefore, the whole contract corpus can be mapped to a matrix of 128 dimensions. Secondly, as the input of GLDA,

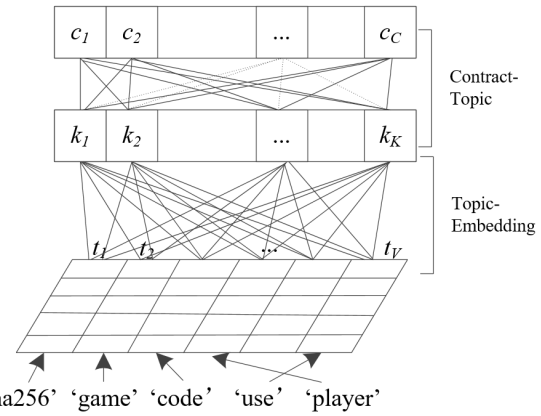

FIGURE 8. GLDA based Hierarchical Modeling for contract comments.

the matrix representing the whole contract corpus obtains two distributions, including: topic distribution and topic embedding distribution. Contract topic distribution is originated from the parameter $\theta(\theta \in |contracts| * |topics|)$ as a traditional LDA model.

To infer the posterior distribution of topics in the contract and the topic assignment of each word, we derive a folded Gibbs sampler to resample the distribution or assignment iteratively until their state converges. The updating rules of iteration are shown in Equation 2

$$p(z_{(c,i)} = k|z_{-(c,i)}, V_c, \zeta, \alpha) \propto (n_{(k,c)} + \alpha_k) * t_{v_k - M+1}(v_{(c,i)}|\mu_k, \frac{\kappa_k + 1}{\kappa_k}\Sigma_k) \quad (2)$$

where $z_{-(c,i)}$ denotes the presently assigned topic for each embedding, excluding a contract which is at the $i-th$ place of contract text set $C.V_c$ denotes the embedding vector sequence for contract set $C.v_{(c,i)}$ denotes a vector sequence in a contract c at position $i.\alpha$ denotes Dirichlet prior. M denotes the length of individual word embedding vector. A tuple $\zeta = (\mu, \kappa, \sum, v)$ denotes the parameters of the prior distribution. In a M-dimensional space, each topic k is characterized as a multivariate Gaussian distribution with mean μ_k and covariance Σ_k. The t-distribution with freedom degree v, parameters μ,and parameters Σ are expressed as $t_v(v|\mu, \sum)$.

Note that the front unit of Equation 1 denotes the probability that the topic k would be assigned to the contract c. The unit and the LDA model are similar, which means the process of generating topics from the Contract-Topic distribution is similar. When running the GLDA, will use this unit to build the first layer.

During Gibbs sampling process when the assignment probability of topic k to $v_{(c,i)}$ are computing. After obtaining the parameters, above discussed topic embedding distribution can be simply conducted. The generating process was done as the work in [24].

C. ATTENTION MECHANISM

Using attention mechanism, it enables the attention mechanism to focus on relevant segments in the smart contract with respect to the tags. First, we convert the tag in each smart contract into a serialized vector $\tau \in R^{10*1}$ and input it into the Embedding layer to get the tag embedding, then use max-pooling to convert it into a tag vector $\tau' \in R^{32*1}$. Then we concatenate τ' with each word vector in word embedding S to get matrix $H \in R^{160*n}$. Attention mechanism is applied on H to get an attention matrix $A \in R^{160*n}$. The overall attention mechanism to generate A is summarized as Equation 3,4

$$H' = tanh(w * H + b) \quad (3)$$

$$m = softmax(\beta^T * H') \quad (4)$$

where $w \in R^{160*160}$,$b \in R^{160*1}$, and $\beta \in R^{160*1}$ are projection parameters to be learned during training. Finally, we multiply each element of m with each column of H to get matrix A.

IV. EXPERIMENTS

In this section, we first introduce experimental settings including the experimental platform and dataset in section IV-A. Then, the evaluation metrics is introduced in section IV-B. In section IV-C, we introduce the parameter settings in the proposed model and compare the proposed model with other baseline methods. Finally, in section IV-D, we analyze the experimental results to prove the performance of the proposed model.

A. EXPERIMENTAL SETUP

We experiment on a 3.9 GHz Intel i7 8750H CPU and 16 GB RAM computer. Python tool named keras is used on Windows 10 to train the model. As for GLDA, because of the authors of [25] have already implemented a GLDA model of Java, so we directly use the Java implementation in github[5] to obtain topic representations and save them as files. Then we use Python to read these files and build the smart contract classification model. Finally, we input the obtained topic representations combine with word embeddings, tag vectors, account and content information into the model implemented by Python and train the model.

More than 25,000 smart contract data are crawled on Ethernum and State of the Dapps website. After removing the duplicate data, we delete the data that the transaction is inactive and compiled incorrectly.

Finally, 15213 smart contracts remain. According to the classification of smart contracts by Mohanta *et al.* [6] and Wu *et al.* [7], we manually divide these smart contracts into Entertainment, Tools, Management, Finance, Lottery, Internet of Things(IOT), and Others seven categories. The number of smart contract for each category is shown in Table 1.

B. EVALUATION METRICS

Precision, *Recall* and *F-score* are employed as the evaluation metrics in this paper to measure the overall classification

[5]https://github.com/rajarshd/Gaussian LDA

TABLE 1. The number of smart contract for each category.

Category	Number	Category	Number
Entertainment	2093	Tools	1758
Management	1008	Finance	7852
Lottery	1382	IOT	467
Others	652		

performance [31]. They are defined as Equation 5,6,7.

$$Precision = \frac{TP}{TP + FP} \tag{5}$$

$$Recall = \frac{TP}{TP + FN} \tag{6}$$

$$F - score = \frac{2 * Precision * Recall}{Precision + Recall} \tag{7}$$

where, the TP indicates the number of positive smart contracts in the test set that are classified as positive. FP indicates the number of negative smart contracts that are classified as positive;FN indicates the number of positives that are negative.

C. PERFORMANCE COMPARISON

We select 60% of our dataset as train set, 20% as validation set and 20% as test set. The train set is the samples used for model training. The validation set is used to verify the parameters(e.g.,learning rate or epoch) of the model and monitor whether the model has overfitted. The test set is used to verify the final performance and the generalization ability of the model.

To figure out the parameters in the model, we use Grid Search with Cross Validation method, which is encapsulated in a Python tool named sklearn. We train the best parameters by using sklearn. The parameters in the proposed model are shown in Table 2.

TABLE 2. The parameters in the proposed model.

Parameter	Parameter Description	Value
dr	Dropout Rate	0.3
bs	Batch Size	32
e	Epochs	5
o	Optimizer	Adam
lr	Learning rate	0.0001

In order to prove the performance of the proposed model, we compare the proposed model with other baseline methods. These methods are as follows:

1) Bi-LSTM: In this method, source code and comments of smart contracts are trained into word embeddings by Word2Vec. The obtained word embeddings are combined with account and content information as input of Bi-LSTM. The neural unit is set to 128 in one direction.
2) TF-IDF+Bi-LSTM: In this method, comments of each contract are first represented by the TF-IDF algorithm. Then, the obtained word embeddings, account and content information, obtained topic representations are inputted into Bi-LSTM.
3) GLDA+Bi-LSTM: In this method, GLDA is first used to get topic representations of comments. Then obtained topic representations are combined with account and content information as input of Bi-LSTM.
4) CNN-BiLSTM: In this method, word embeddings, account and content information are employed as the input of Bi-LSTM, and tag vectors as the input of CNN respectively. Then, the results of CNN and BiLSTM are connected together, and output the classification results through a feedforward neural network and Softmax. The window size is set to 3 and the filter is set to 32.
5) GLDA+CNN-BiLSTM: In this method, word embeddings, topic representations obtained by GLDA, account and content information are additionally as input of Bi-LSTM, and tag vectors as the input of CNN respectively. Then, the results of CNN and BiLSTM are connected together, and output the classification results through a feedforward neural network and Softmax.
6) ATT-BiLSTM [32]:In this method, firstly, word embeddings are inputted into Bi-LSTM. Then, combining Bi-LSTM hidden layer with related word vectors, and generate final representation by the function of attention mechanism.

D. EXPERIMENTAL RESULTS AND ANALYSIS

1) VALIDATION OF EMBEDDING

As discussed above, the performance of smart contract classification can be improved by changing the Bag of Words model to a continuous embedded space to reflect more semantic knowledge. Fig. 9 shows the *F-score* performance of the proposed model with different dimensional word embedding trained by Word2Vec model using different corpus SAWSDL-TC3(TC3)[6] and Wikipedia.

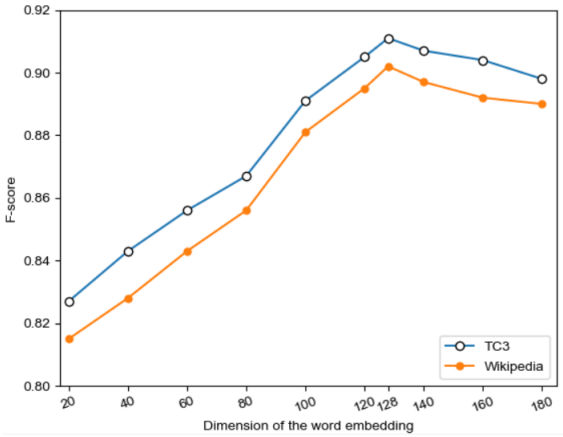

FIGURE 9. The performance of using different corpus.

As shown in Fig. 9, the proposed model using TC3 has better *F-score* performance than using Wikipedia. A possible explanation for this might be that some words extracted from comments do not appear enough times in the Wikipedia

[6]http://www.semwebcentral.org/projects/sawsdl-tc

corpus to be removed during training embedding, despite their usefulness [16]. Moreover, the proposed model has better *F-score* when the word embedding dimension is 128.

2) VALIDATION OF NUMBER OF TOPICS *K*

We conduct experiments on the effect of the number of topics. However, since no clear standard is provided for setting the number of topics, we experiment on the topic K of different numbers, used to the log-likelihood results of GLDA model for evaluation. Experimental procedures follow [33]. For all values of K, we run the GLDA model until the output of convergence. In that case, the log-likelihood values will eventually converge after hundreds of iterations.

The experiments are conducted under different values of $K \in [50, 150]$. The step length is set to 10, and the iteration number is set to 300. For the number of topics K for each candidate value, we conduct a set of independent experiments. Fig. 10 shows the performance of log-likelihood. Log-likelihood value increases when K changes from 50 to 110, and decreases from 110 to 150. When K is equal to 110, we obtain the best log-likelihood. Thus, experimental results show that the number of topics is set to 110.

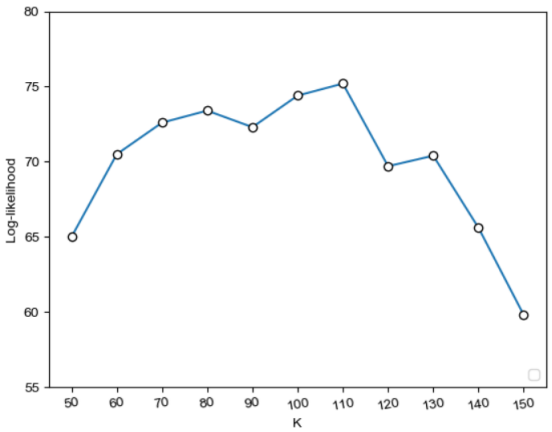

FIGURE 10. The performance of different number of topics.

3) THE PERFORMANCE OF DIFFERENT SEMANTICS ENHANCING APPROACHES

As shown in Fig. 11, In the absence of semantic enhancement of comments, the *F-score* of using only Bi-LSTM is lower. If use TF-IDF or GLDA to semantically enhance comments,*F-score* will increase by 0.016 and 0.027 respectively.Experimental result shows that using GLDA or TF-IDF to enhance comments semantics can improve the model effect. However,using GLDA has better performance than TF-IDF.

4) THE PERFORMANCE OF DIFFERENT TAG USAGE APPROACHES

As shown in Fig. 12, Adding tag information but not adding attention mechanism, the *F-score* of CNN-LSTM can reach

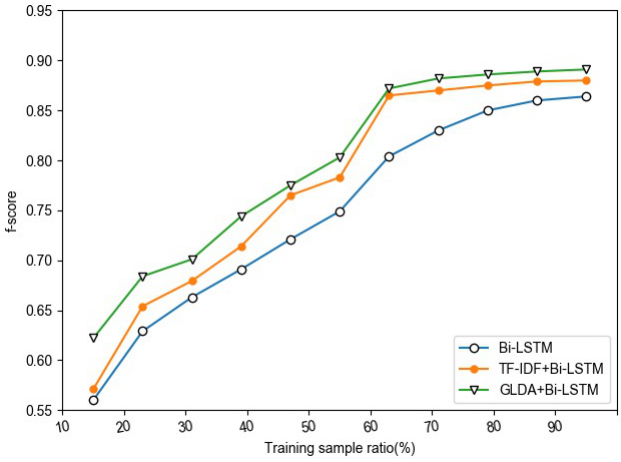

FIGURE 11. The performance of different semantics enhancing approaches.

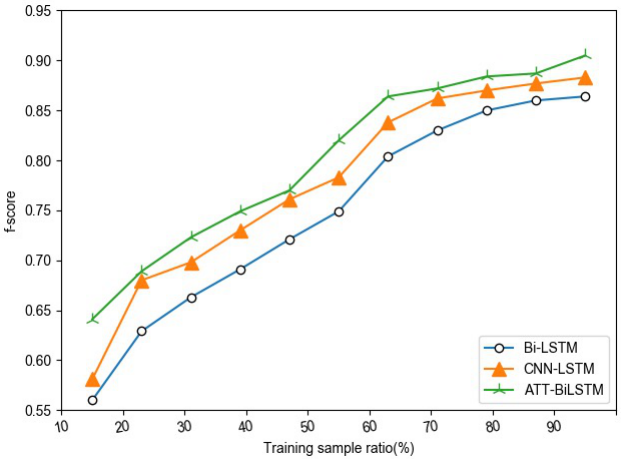

FIGURE 12. The performance of different tag usage approaches.

0.883, compared to Bi-LSTM increased by 0.019. This proves that using tags can improve the performance of model. However, the *F-score* of ATT-BiLSTM with attention mechanism is 0.905, which is higher than using tags alone. This proves that using attention mechanism can capture the key parts and give them higher weights in smart contracts, which has better performance than the methods only using tags.

5) SENSITIVITY EXPERIMENTS

In order to figure out the components such as word embedding, tag embedding, topic representation, account and content information which one is more important to the performance. We remove each component that make up the input of SCC-BiLSTM and determine which part is important to the performance.As shown in Fig. 13. SCC-BiLSTM has the lowest F-score without word embeddings, followed by lack of tag embeddings. After that we remove account and content information. Finally we remove topic representations.

This shows that word embedding is most important to the performance. Tag embedding, account and content

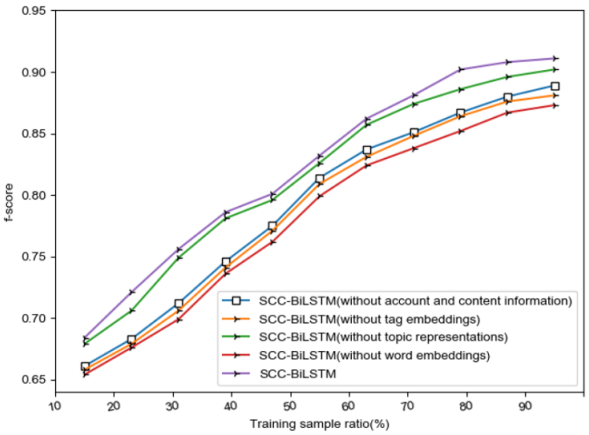

FIGURE 13. The result of sensitivity experiments.

information also have good effect on the performance of the proposed model. Besides, the experimental results prove that although topic representation is not the most important component for model performance, it still can improve the model effect.

Finally,Table 3 shows the experimental results for different methods. The classification result of each method were evaluated using *Precision*, *Recall*, and *F-score*.

TABLE 3. The performance of different methods.

Method	*Precision*	*Recall*	*F-score*
Bi-LSTM	0.878	0.852	0.864
TF-IDF+Bi-LSTM	0.886	0.874	0.880
GLDA+Bi-LSTM	0.902	0.882	0.891
CNN-BiLSTM	0.889	0.877	0.883
GLDA+CNN-BiLSTM	0.895	0.887	0.891
ATT-BiLSTM	0.901	0.910	0.905
SCC-BiLSTM (without account and content information)	0.893	0.886	0.889
SCC-BiLSTM (without tag embeddings)	0.885	0.877	0.881
SCC-BiLSTM (without word embeddings)	0.876	0.871	0.873
SCC-BiLSTM (without topic representations)	0.905	0.899	0.902
SCC-BiLSTM	0.917	0.906	0.911

Our proposed model which combines GLDA, attention mechanism and account information has better performance than the others, which illustrate the performance of the proposed method.

V. CONCLUSION

In this article, we take smart contracts in Ethereum platform as example. To address the effective classification of smart contracts, we propose a Bi-LSTM model based approach called SCC-BiLSTM. According to the characteristics of smart contracts and the programming language Solidity, GLDA and attention mechanism are used to improve the performance of the model. GLDA is used to solve the problem of sparse semantics of comments in contracts. And the attention mechanism can make full use of tags. Besides, combining the account and content information can effectively improve the classification effect.

Experimental results demonstrate that the proposed approach achieves superior effectiveness on smart contract classification, which demonstrates the feasibility of the proposed model.

In the future, we would like to dig deeper into the characteristics of the programming language Solidity to discover the useful characteristics as many as possible, and use the characteristics of smart contracts to extract more content information and semantic features. We also will seek better attention mechanisms to make better use of tags to improve the smart contract classification effect.

ACKNOWLEDGMENT

The authors would like to thank reviewers for the precious comments. *(Gang Tian and Lantian Guo are co-first authors.)*

REFERENCES

[1] B. T. Huang, Q. Liu, Q. M. He, Z. G. Liu, and J. H. Chen, "Towards automatic smart-contract codes classification by means of word embedding model and transaction information," *Acta Automatica Sinica*, vol. 43, no. 9, pp. 1532–1543, 2017.

[2] C. Z. He Haiwu and Y. An, "Survey of smart contract technology and application based on blockchain," *J. Comput. Res. Develop.*, vol. 55, no. 11, p. 2452, 2018.

[3] K. Delmolino, M. Arnett, A. Kosba, A. Miller, and E. Shi, "Step by step towards creating a safe smart contract: Lessons and insights from a cryptocurrency lab," in *Proc. Int. Conf. Financial Cryptogr. Data Secur.*, 2016, pp. 79–94.

[4] T. T. A. Dinh, J. Wang, G. Chen, R. Liu, B. C. Ooi, and K.-L. Tan, "BLOCKBENCH: A framework for analyzing private blockchains," in *Proc. ACM Int. Conf. Manage. Data*, 2017, pp. 1085–1100.

[5] L. Luu, D.-H. Chu, H. Olickel, P. Saxena, and A. Hobor, "Making smart contracts smarter," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*, New York, NY, USA, 2016, pp. 254–269.

[6] B. K. Mohanta, S. S. Panda, and D. Jena, "An overview of smart contract and use cases in blockchain technology," in *Proc. 9th Int. Conf. Comput., Commun. Netw. Technol. (ICCCNT)*, Piscataway, NJ, USA, Jul. 2018, pp. 1–4.

[7] Y. Z. D. Wu and T. Cai, "Hierarchical attention mechanism and bidirectional long short-term memory based neural network model for smart contract automatic classification," *J. Comput. Appl.*, to be published.

[8] M. Shi, Y. Tang, and J. Liu, "Functional and contextual attention-based LSTM for service recommendation in mashup creation," *IEEE Trans. Parallel Distrib. Syst.*, vol. 30, no. 5, pp. 1077–1090, May 2019.

[9] Y. Cao, J. Liu, B. Cao, M. Shi, Y. Wen, and Z. Peng, "Web services classification with topical attention based bi-LSTM," in *Proc. Int. Conf. Collaborative Comput., Netw., Appl. Worksharing*, 2019, pp. 394–407.

[10] O. Jin, N. N. Liu, K. Zhao, Y. Yu, and Q. Yang, "Transferring topical knowledge from auxiliary long texts for short text clustering," in *Proc. 20th ACM Int. Conf. Inf. Knowl. Manage. (CIKM)*, 2011, pp. 775–784.

[11] X. Hu, N. Sun, C. Zhang, and T.-S. Chua, "Exploiting internal and external semantics for the clustering of short texts using world knowledge," in *Proc. Proc. 18th ACM Conf. Inf. Knowl. Manage. (CIKM)*, 2009, pp. 919–928.

[12] T. Kenter and M. de Rijke, "Short text similarity with word embeddings," in *Proc. 24th ACM Int. Conf. Inf. Knowl. Manage. (CIKM)*, 2015, pp. 1411–1420.

[13] J. Xie, Z. Song, Y. Li, Y. Zhang, H. Yu, J. Zhan, Z. Ma, Y. Qiao, J. Zhang, and J. Guo, "A survey on machine learning-based mobile big data analysis: Challenges and applications," *Wireless Commun. Mobile Comput.*, vol. 2018, pp. 1–19, Aug. 2018.

[14] N. Zhang, J. Wang, Y. Ma, K. He, Z. Li, and X. Liu, "Web service discovery based on goal-oriented Query expansion," *J. Syst. Softw.*, vol. 142, pp. 73–91, Aug. 2018.

[15] O. Levy and Y. Goldberg, "Linguistic regularities in sparse and explicit word representations," in *Proc. 18th Conf. Comput. Natural Lang. Learn.*, 2014, pp. 746–751.
[16] G. Tian, S. Zhao, J. Wang, Z. Zhao, J. Liu, and L. Guo, "Semantic sparse service discovery using word embedding and Gaussian LDA," *IEEE Access*, vol. 7, pp. 88231–88242, 2019.
[17] D. Tang, B. Qin, X. Feng, and T. Liu, "Effective LSTMs for target-dependent sentiment classification," 2015, *arXiv:1512.01100*. [Online]. Available: http://arxiv.org/abs/1512.01100
[18] D. Hazarika, S. Poria, P. Vij, G. Krishnamurthy, E. Cambria, and R. Zimmermann, "Modeling inter-aspect dependencies for aspect-based sentiment analysis," in *Proc. Conf. North Amer. Assoc. Comput. Linguistics, Hum. Lang. Technol.*, vol. 2, Jun. 2018, pp. 266–270.
[19] G. Fei, *Design of Intelligent Contract Automatic Classification System Based on Blockchain Technology*. Beijing, China: Plateau Science Research, 2018.
[20] S. Omohundro, "Cryptocurrencies, smart contracts, and artificial intelligence," *AI Matters*, vol. 1, no. 2, pp. 19–21, Dec. 2014.
[21] (2017). *Ethereum White Paper*. [Online]. Available: https://github.com/ethereun/wiki/wiki/White-Paper
[22] D. Zhang, H. Xu, Z. Su, and Y. Xu, "Chinese comments sentiment classification based on word2vec and SVMperf," *Expert Syst. Appl.*, vol. 42, no. 4, pp. 1857–1863, Mar. 2015.
[23] D. Ma, S. Li, X. Zhang, and H. Wang, "Interactive attention networks for aspect-level sentiment classification," in *Proc. 26th Int. Joint Conf. Artif. Intell.*, Aug. 2017, pp. 1–7.
[24] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet allocation," *J. Mach. Learn. Res.*, vol. 3, pp. 993–1022, Mar. 2003.
[25] R. Das, M. Zaheer, and C. Dyer, "Gaussian LDA for topic models with word embeddings," in *Proc. 53rd Annu. Meeting Assoc. Comput. Linguistics*, 2015, pp. 795–804.
[26] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.
[27] D. Hu, "An introductory survey on attention mechanisms in NLP problems," in *Proc. SAI Intell. Syst. Conf.*, 2018, pp. 432–448.
[28] L. Chen, Y. Wang, Q. Yu, Z. Zheng, and J. Wu, "WT-LDA: User tagging augmented LDA for Web service clustering," in *Proc. Int. Conf. Service-Oriented Comput.*, 2013, pp. 162–176.
[29] M. Zhou and F. Zhu, "Polarity classification based on sentiment tags," *Acta Elect Ronica Sinica*, vol. 45, no. 4, pp. 1018–1024, 2017.
[30] B. Sigurbjörnsson and R. van Zwol, "Flickr tag recommendation based on collective knowledge," in *Proc. 17th Int. Conf. World Wide Web (WWW)*, 2008, pp. 327–336.
[31] F. Morstatter, L. Wu, T. H. Nazer, K. M. Carley, and H. Liu, "A new approach to bot detection: Striking the balance between precision and recall," in *Proc. IEEE/ACM Int. Conf. Adv. Social Netw. Anal. Mining (ASONAM)*, Aug. 2016, pp. 533–540.
[32] C. Du and L. Huang, "Text classification research with attention-based recurrent neural networks," *Int. J. Comput. Commun. Control*, vol. 13, no. 1, p. 50, Feb. 2018.
[33] W. Buntine, "Estimating likelihoods for topic models," in *Proc. Asian Conf. Mach. Learn.* Berlin, Germany: Springer 2009, pp. 51–64.

GANG TIAN received the Ph.D. degree in computer science and engineering from Wuhan University, China, in 2016. He is currently an Associate Professor with the School of Computer Science and Engineering, Shandong University of Science and Technology, China. His current research interests include web service discovery, transfer learning, feature engineering, and knowledge extraction.

QIBO WANG is currently pursuing the master's degree with the School of Computer Science and Engineering, Shandong University of Science and Technology, China. His research interests include web service discovery, natural language processing, and feature engineering.

YI ZHAO received the Ph.D. degree in computer science and engineering from Wuhan University, China, in 2018. He is currently an Assistant Professor with the School of Mathematics and Computer Science, Guangdong Ocean University, China. His current research interests include artificial intelligence, deep learning, and knowledge graph.

LANTIAN GUO was a Visiting Researcher with the School of Computing, Queen's University, Canada. He is currently an Assistant Professor with the School of Automation and Electronic Engineering, Qingdao University of Science and Technology, China. His current research interests include big data, recommendation systems, machine learning, and artificial intelligence.

ZHONGLIN SUN is currently a Professor with the Shandong University of Science and Technology. His research interests include software engineering and database management.

LIANGYU LV is currently pursuing the master's degree with the Shandong University of Science and Technology. Her research interest is natural language processing.