Learning Regular Expressions for Interpretable Medical Text Classification Using a Pool-based Simulated Annealing Approach

Chaofan Tu
School of Computer Science
University of Nottingham
Ningbo, China
chaofan.tu@nottingham.edu.cn

Menglin Cui*
School of Computer Science
University of Nottingham
Ningbo, China
menglin.cui@nottingham.edu.cn

***Abstract*—In this paper, we propose a rule-based engine composed of high-quality and interpretable regular expressions for medical text classification. The regular expressions are auto-generated by a constructive heuristic method and optimized using a Pool-based Simulated Annealing (PSA) approach. Although existing Deep Neural Network (DNN) methods present high-quality performance in most Natural Language Processing (NLP) applications, the solutions are regarded as uninterpretable "black boxes" to humans. Therefore, rule-based methods are often introduced when interpretable solutions are needed, especially in the medical field. However, the construction of regular expressions can be extremely labor-intensive for large data sets. This research aims to reduce the manual efforts while maintaining high-quality solutions. The Pool-based Simulated Annealing method is proposed to automatically optimize the performance of machine-generated regular expressions without human interference. The proposed method is tested on real-life data provided by one of China's largest online medical platforms. Experimental results show that the proposed PSA method further improves the performance of initial machine-generated regular expressions compared with other meta-heuristics such as Genetic Programming. We also believe that the proposed method can serve as a vital complementary tool for the existing machine learning approaches in text classification applications when high levels of interpretability of the solutions are required.**

***Index Terms*—simulated annealing, regular expression, medical text classification**

I. Introduction

With the prevalence of modern computerized technologies in auxiliary medical diagnosis, the doctor/hospital operational efficiency has been greatly enhanced. Text mining is becoming a necessity in the field of intelligent healthcare sector since the amount of online medical data is explosively expanding. Some state-of-the-art approaches such as Convolutional Neural Networks (CNNs) [1] and Recurrent Neural Networks (RNNs) [2]–[5] have demonstrated excellent performance in clinical data mining tasks while no human effort is required once a suitable model has been built. However, the main drawback is the lack of interpretability in such "black box" approaches. Additionally, it may even require complete retraining if the scenario changes or the predefined categories are updated. Therefore, a system with interpretable solutions that can be understood by human is usually preferred in the real-world medical practice.

Our work is based on the collaboration with an online medical consultation platform with an average number of above 370,000 daily consultation requests. In order to assign specialized doctors across the different medical departments (such as pediatrics, gynecology, *etc*.) to different patients with specific needs, a medical text classifier is desired to classify the sentence-level patients' inquiries to several predefined medical templates (such as "pediatric diarrhea," "pregnancy," *etc*.). Since any mistakes in the classification process will lead to doctor miss-assignment and as a result reduce the system overall efficiency, we therefore value the precision more than the recall in our evaluation metric. For the sake of not only reducing the number of employees who should be available 24/7 handling the medical reception but also enhancing the platforms operational efficiency, it is very important to implement an automated text classification system in our online guidance scenario.

Regular expressions are widely used text-matching techniques that are fully interpretable compared to learning-based models. However, its main advantage is its shortcoming: the manual construction of the rules requires domain knowledge and a considerable amount of human work. In order to reduce human efforts while ensuring great interpretability of the solution, we propose a fully automated system for learning regular expressions in the medical text classification tasks. The system is trained to capture relevant words and synonyms, discard irrelevant words, and form word sequences for good readability and further revision. The contributions of this paper are summarized as follows:

- A specially designed regular expression structure is proposed to reduce the complexity whilst maintaining high flexibility of the solution;
- A Pool-based Simulated Annealing (PSA) method is proposed, and the word-vector model is embedded to enhance the interpretability of the auto-generated regular

* corresponding author

978-1-7281-6929-3/20/$31.00 ©2020 IEEE

expressions;
- Impacts of parallel and iterative strategies for tasks of learning regular expressions have been intensively explored by comparing two extended versions of PSA.

II. Related Work

Text classification assigns text documents to pre-defined classes. This is a typical supervised machine learning task with training text documents with predefined class labels. Automated text classification usually includes steps such as pre-processing, feature selection using statistical or semantic approaches, and text modeling [6]. Until late 1980s, text classification task was based on Knowledge Engineering (KE), where a set of rules were defined manually to encode the expert knowledge on how to classify the documents given the categories [7]. Since there is a requirement of human intervention in knowledge engineering, researchers in the 1990s have proposed many machine learning techniques to automatically manage and organize the textual documents [8]. The accuracy of those machine learning techniques is comparable to that of human experts and no artificial labor work from either knowledge engineers or domain experts is needed for the construction of a document management tool [8].

In order to perform text classification, a proper text representation is needed. Bag of Words (BoW) model is one of the most commonly used methods to represent a document. By using a fixed global vocabulary, the BoWs use a vector to represent a text document based on the frequency count of each term in the document. This kind of method of text representation is called Vector Space Model (VSM) [9]. Unfortunately, BoW/VSM representation scheme has its limitations such as very high dimensionality of the representation, loss of correlation with adjacent words, and absence of semantic relationship [10]. Another VSM-based method is a neural network model named word2vec, which was proposed by Mikolov *et al*. in 2013 [11], [12]. This kind of fixed-length vector representation (often hundreds of dimensions) trained by deep learning model has shown the ability to carry semantic meanings. This technique can be used in various NLP tasks such as text classification, speech recognition, and image caption generation [13].

After text representation, word embeddings or numerical representations for text feature extraction can be fed into classifion models such as naïve Bayes, decision tree, Support Vector Machine (SVM), neural networks, *etc*. [14]. The naïve Bayes classifier is the simplest probabilistic classifier used to classify the text documents into predefined labels. Decision trees are the most widely used inductive learning methods. Decision trees robustness to noisy data and their capability to learn disjunctive expressions seem suitable for document classification. Support Vector Machine is a supervised classification algorithm that has been extensively and successfully used for text classification tasks. Neural network based text classifiers are also prevalent in the literature, where the input units are denoted as feature terms, the output unit(s) is the category or categories of interest, and the weights on the edges connecting units form dependence relations [14].

With a simple convolutional neural network built on top of word-vector models, a series of experiments of sentence-level text classification problems suggest that unsupervised pre-training of word vectors is an important ingredient in deep learning for NLP [13]. Neural network based approaches are strong in terms of precision and recall but usually less interpretable because those "black box" models cannot be logically explained [15]. In addition, those "black box" approaches cannot be quickly modified except retraining the whole neural network models [16]. For those difficult issues, some related work has shown that regular expressions can be effectively used to solve text classification problems in an interpretable way. A novel regular expression discovery (RED) algorithm and two text classifiers based on RED were designed to automate both the creation and utilization of regular expressions in text classification [16]. The proposed RED+ALIGN method correctly classifies many instances that were misclassified by an SVM classifier. A novel transformation-based algorithm, called ReLIE, was developed to learn such complex character-level regular expressions for entity extraction tasks and related experiments demonstrated that it is effective for certain classes of entity extraction in text documents [17].

Automated regular expression generation can also be viewed as a data-driven optimization problem. In this paper, a well-known simulated annealing hyper-heuristic [18] has been adapted for learning regular expression based classifiers for text classification. The choice of this approach is based on the fact that there are naturally multiple neighbor operators available for generating regular expression variants and hyper-heuristics can learn to orchestrate the selections of different operators to achieve high performance across various problems. It has been shown that specially designed neighbor operators of SA will lead to better performance [15].

III. Scenario

In this section, we describe the medical text classification problem in detail, and introduce the regular expression structure that is globally applied in the construction of regular expressions.

A. Problem Description

Formally the problem can be defined as follows: given a set of predefined classes C (or medical templates in the context of our application) and a set of text inquires Q, the problem is to classify each inquiry $q \in Q$ to one of the classes $c \in C$ based on a set of previously labeled samples by medical experts. Table I gives a list of examples, where text inquiries are usually narrative texts provided by users, describing the medical conditions or problems; the classification task is to select the most appropriate medical template for this inquiry.

Let R be a regular expression designed for the classification of class C (we denote $|C|$ as the number of inquiries in class C), and let $M(R, Q) \in Q$ be the set of all medical texts matched by R in Q; Denote $M_p(R, Q) = \{q \in M(R, Q) : q$

TABLE I
EXAMPLES OF MEDICAL TEXT CLASSIFICATION

Text inquiries	Medical template
My daughter is three years old, She always coughs and does not have a fever.	*Cough: 1-3 years old child*
I have been suffered from pain in my lower abdomen for 3 weeks.	*Adult bellyache*
The acne grows on the back, and recently it is a little itchy near it.	*Folliculitis*
I have a serious hair loss. How to deal with it?	*Hair loss*

is an instance of $C\}$ to the set of all correctly matched entries (medical text inquires) and denote $M_n(R,Q) = \{q \in M(R,Q) : q$ *is not an instance of* $C\}$ is the set of all mismatched entries (medical text inquires). Like many classification problems, this problem also has two performance indicators, which are precision and recall as below:

$$precision(R,Q) = \frac{|M_p(R,Q)|}{|M_p(R,Q)| + |M_n(R,Q)|}, \quad (1)$$

$$recall(R,Q) = \frac{|M_p(R,Q)|}{|C|}. \quad (2)$$

The well-known F-measure (also called F-score) can be a better single metric when compared to precision and recall. With a non-negative real β for users' preference, it can be expressed as:

$$F_\beta(R,Q) = (1+\beta^2) \cdot \frac{precision(R,Q) \cdot recall(R,Q)}{\beta^2 \cdot precision(R,Q) + recall(R,Q)}. \quad (3)$$

The problem of automated learning of regular expression based classifiers for medical text in this paper can be formally expressed as an optimization problem for regular expression R. Let S be the solution space of R, for a given class of C and labelled dataset W which can be divided into a positive part and a negative part, the problem is to find a solution with the optimal objective function F_β from the solution space S. So this problem can be defined as:

$$R_{target} = argmax_{R \in S} F_\beta(R,Q). \quad (4)$$

B. The Regular Expression Structure

In this research, we hang on to the regular expression structure proposed in our previous work [19]. Each solution is encoded as a vector of m regular expressions $< R_1, R_2, ..., R_i, ..., R_m >$. Each regular expression follows a global structure of two parts P_i and N_i concatenated by the *NOT* function #_#, which is:

$$R_i = P_i \cdot (\#\_\#(N_i)), \quad (5)$$

where the positive part P_i tries to match all positive inquiries and the negative part N_i is then used to filter out the list of falsely matched inquiries by P_i.

For checking whether a particular inquiry belongs to a class (or template), the regular expressions in the vector are executed one by one sequentially in the same order of the vector for the inquiry under consideration. If the inquiry is matched by any of regular expressions, the inquiry is said to be in the class, otherwise, it is not in the given class.

IV. METHODOLOGY

A. Solution Pool Mechanism

According to the problem description and regular expression structures defined above, the medical text classification problem in this paper is transformed into a combinatorial optimization problem. Simulated Annealing is a method for approximating the global optimum of large-scale combinatotial problems. An improved Simulated Annealing method – Pool-based Simulated Annealing – is applied, which employs a well-designed population based mechanism to enhance the diversity of solutions shown in Fig. 1.

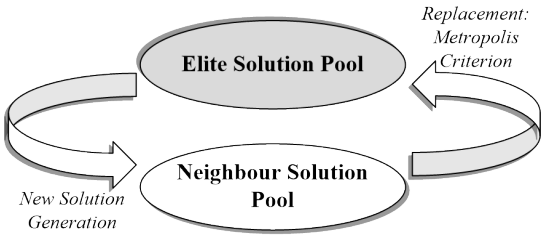

Fig. 1. Solution pool mechanism

The number of solutions in the elite solution pool is set to a fixed value, and the same amount of new solutions are transformed from an initial solution for the initialization of the elite solution pool. The number of solutions in the neighbor solution pool is the same as the elite solution pool. Each solution in the elite solution pool will produce a new solution in each iteration during the entire period, and then an updated neighbor solution pool can be formed from those all newly-generated solutions. The elite solution pool will also be updated in each iteration during the entire evolution. For each solution in the updated neighbor solution pool, one solution in the elite solution pool will be randomly selected for comparison and replacement. The acceptance criterion for replacement adopts Metropolis criterion based on Simulated Annealing algorithm. Every time the best solution in the elite solution pool will always be retained during the whole period. The details of the proposed solution pool mechanism are shown in algorithm 1.

B. Initialization

From the above description, an existing initial solution is a precondition for the initialization. In order to balance speed and readability, we proposed a constructive heuristic method to generate initial regular expression solutions S_{init} by taking into account the word frequency, similarity, and co-occurrence in our previous study [19]. Specific steps are described below:

Algorithm 1 Pseudo-code of the proposed mechanism

Set the capacity of a solution pool to be N_{pool};
Define a set of each elite solution $S_{e\_i}(i = 1, 2, ..., N_{pool})$ as the elite solution pool P_e;
Define a set of each neighbor solution $S_{n\_j}(j = 1, 2, ..., N_{pool})$ as the neighbor solution pool P_n;
Set the best solution in P_e as S_{e_best}.
Solution Replacement:
$j = 1$
while $(j < N_{pool})$ **do**
  select S_{n_j};
  select S_{e_i} randomly from P_e;
  let δ be the difference in the evaluation function between S_{e_i} and S_{n_j};
  if the Metropolis criterion of simulated annealing is satisfied by δ **then**
    $S_{e\_i} = S_{n\_j}$;
  end if
  $j++$;
end while
add S_{e_best} into P_e temporarily;
sort P_e in descending order by the evaluation function of each solution $S_i(i = 1, 2, ..., N_{pool} + 1)$;
$P_e = \{S_{e\_i} \mid i = 1, 2, ..., N_{pool}\}$;
$S_{e\_best} = S_{e\_1}$;

1) For a given class c, divide the training data is into positive and negative sets based on labels.
2) Calculate the comparative frequency of each word in the two sets. If the comparative frequency exceeds a predetermined parameter, the word is selected as a feature word.
3) Regular expressions are generated based on the calculation of similarities and co-occurrence between words by predefined filtering mechanisms.
4) The iterative process stops when predefined evaluation metrics are satisfied, *i.e.*, with fitness score above a certain threshold or no more additions to the existing solutions can be found.

C. Neighbor Operators

In our proposed method, six specially designed neighbor operators are used for the generation of new solutions.

O1: **Adding *OR*** is an operator to add a word to the *OR* structure. First randomly select 10 words from the set of candidate words. Then randomly select an existing word from the *OR* structure, and calculate the cosine similarity $Sim_i(i = 1, 2, ..., 10)$ between the existing word and the other words based on pre-trained word vector model. The word with the highest probability $Prob_i$ is selected to add to the *OR* structure. $Prob_i$ is calculated as below:

$$Prob_i = \frac{Sim_i}{\sum_1^{10} Sim_i} \tag{6}$$

O2: **Removing *OR*** is an operator to randomly delete a sub-expression that makes up the *OR* structure, as a inverse operation of O1.

O3: **Adding *AND*** is an operator to extend the *AND* (or Adjacency) structure in the sub-expression. Randomly pick a word to insert into an existing *AND* (or Adjacency) structure or form a new *AND* structure with an existing word.

O4: **Removing *AND*** is an operator to randomly delete one sub-expression e or word w that makes up one *AND* (or Adjacency) structure, as an inverse operator of O3.

O5: **Swapping** is an operator to exchange the positions of any two sub-expressions or words in the *AND* (or Adjacency) structure.

O6: **Changing distance** is an operator to randomly change the maximum distances between two expressions or words based on a given *Distance Table*. So the *AND* function can be considered as an Adjacency structure with unrestricted distance.

D. Solution Decoding and Evaluation

Each regular expression R_i in the solution pool should be decoded to a valid regular expression that can be passed through the general regular expression matching engine. It is worth noting that the logical symbols defined in this paper are not exactly the same as the symbolic system of regular expressions. The *NOT* function defined in our system is not included in regular expression matching engine, so the positive and negative parts of R_i need to be separately handled. The performance of each regular expression based solution or classifier will be evaluated based on the F-measure introduced in Section II.

E. The Overall Algorithm

In this paper, a modified Simulated Annealing method, called pool-based simulated annealing, or PSA, is proposed to derive regular expressions to form a rule-based classification engine automatically. Figure 2 demonstrates the overall framework of the proposed PSA method.

The pre-processing step includes dividing the training data set into positive and negative sets, performing Chinese word segmentation, removing stop words, and pre-training the word2vec model. After the initial solution has been generated, the elite solution pool will be filled with new solutions transformed by the initial solution.

According to the Metropolis criterion, solutions in the elite solution pool may be replaced by solutions in the neighbor solution pool. Parameters such as the temperature of classic simulated annealing would be updated in each iteration. The program terminates when the total iterations are achieved or the stop criterion is met.

To further explore the impacts of different running strategies in computational time and performance, we design and implement an extended version called PSA-I for iterations strategy and another extended version called PSA-P for parallelism.

- PSA-I: It is considered to improve the recall by learning regular expressions iteratively, that is, before learning the

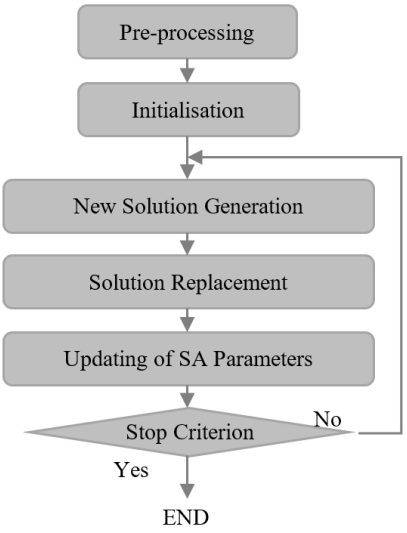

Fig. 2. The overall framework of PSA method

next regular expression, the text inquiries in the training set that are matched by existing regular expressions would be filtered out.

- PSA-P: This method pre-divides the positive training set for parallel acceleration. When the last parallel task is terminated, all the sub-solutions are merged as one solution as a whole. Pre-dividing the training set is based on semantic clustering and a random division is also set as a baseline.

V. Experiments

A. Data and Parameter Settings

The experiments in this paper are based on large-scale and high-quality training and test data collected from an online healthcare platform. The numbers of real-life medical text inquiries in the training, validation, and test sets are $2,000,000$, $500,000$, and $500,000$, respectively. If not stated separately, the following parameters are used in this paper.

- For the neighbor operators: *Distance Table* $= [0, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 100]$;
- For pool-based simulated annealing: starting temperature $T_S = 0.5$; stopping temperature $T_E = 0.05$; solution pool capacity $N_{pool} = 10$; Total iteration $K = 1000$;
- For F-measure: $\beta = 0.5$.

B. Evaluation of Solution Pool Mechanism

In this experiment, we control the variables N_{pool} and K for learning one regular expression for the same template C_1 to evaluate the solution pool mechanism. The N_{pool} of groups 1 to 3 are set to 1, 10, 50 and the N_{pool} of group 4 is set to 1 to represent the classic simulated annealing without the solution pool mechanism. The total number of new solutions in group 4 is set as the same as group 2. In Table II, the results of groups 1 to 3 show that the higher the N_{pool}, the more time consumption and the better the performance of F-measure. The comparison between group 2 and 4 shows that the solution pool mechanism not only enhances performance and but also reduces time cost. This is because the solution update process of group 4 makes its solution more and more complex to increase the evaluation time and this mechanism in group 2 can improve the diversity of solutions.

TABLE II
A Comparison of Pool Capacities

Group	N_{pool}	K	F_β	**Time**(min)
1	1	1000	0.60	8
2	10	1000	0.76	59
3	50	1000	0.77	364
4	1	10000	0.72	130

C. Result Interpretation

We have tested the proposed PSA and its two extended versions of PSA-I, PSA-P on 6 different medical text classes C_1 to C_6. For further explorations, PSA-P version adapted two different methods, namely the clustering division and random division. All solutions contain three regular expression based classifiers. One PSA-P group applies the widely used k-means clustering method to divide the training set into three different subsets for parallel processing, while another PSA-P group applies the random division as a comparison baseline. In Table III, the PSA-P with k-means clustering shows the highest precision and the least time cost; the PSA-I shows the highest recall, while the original PSA presents the most time cost with the highest F_β.

Genetic programming (GP) [20] is another evolutionary algorithm widely applied to many areas such as grammar evolution in the medical field [21], regular expressions inference [22]–[24], and stochastic regular expressions for pattern matching [25]. In this subsection, we employ the GP optimization algorithm and compare the performance of our proposed PSA method with that of GP as a baseline. We set the population size to 100, survival rate to 30%, crossover rate to 20%, and mutation rate to 80%. The performance of INIT, GP, and GP methods are demonstrated in Table IV, where INIT represents the initial regular expression solution generated by our previous work [19].

Although GP provides notable enhancements to the result of the initial solution, the performance PSA is comparably better and more balanced. For example, PSA improves 5% more recall than GP on C_3 and increases the precision to more than 90% on C_6, while GP maintains the same performance as the initial solution. The reason is that the evolutionary operations in GP, such as crossover, might break the original structure and semantic relations between words and to some extent reduce the quality and interpretability of the new solution. In contrast, PSA embeds the solution pool mechanism to the traditional simulated annealing method, which maintains the structure of the solution in the process of population evolution. Therefore, the proposed PSA method is more efficient in finding better, or even optimized solutions.

TABLE III
A COMPARISON OF PSA, PSA-I, PSA-P CLUSTERING, AND PSA-P RANDOM METHODS

Class	PSA				PSA-I				PSA-P (Clustering)				PSA-P (Random)			
	Prec.	*Recall*	F_β	*Time (min)*	*Prec.*	*Recall*	F_β	*Time (min)*	*Prec.*	*Recall*	F_β	*Time (min)*	*Prec.*	*Recall*	F_β	*Time (min)*
C_1	**0.89**	0.63	**0.87**	266	0.76	**0.76**	0.76	220	0.87	0.42	0.83	**86**	0.86	0.52	0.84	110
C_2	0.69	0.41	**0.68**	335	0.56	**0.50**	0.56	300	**0.72**	0.29	0.68	**119**	0.62	0.26	0.59	125
C_3	**0.71**	0.11	0.58	417	0.52	**0.25**	0.50	372	**0.71**	0.11	**0.59**	**138**	0.66	0.09	0.54	144
C_4	**0.93**	0.33	**0.87**	394	0.81	**0.78**	0.81	333	0.92	0.37	**0.87**	**133**	0.92	0.31	0.86	128
C_5	0.84	0.69	0.83	405	0.87	**0.83**	0.87	345	**0.89**	0.54	**0.87**	**136**	0.86	0.51	0.84	134
C_6	**0.93**	0.61	**0.91**	396	0.84	**0.65**	0.83	239	**0.93**	0.48	0.89	**108**	0.92	0.48	0.89	124
AVG	0.83	0.46	**0.81**	369	0.73	**0.63**	0.72	302	**0.84**	0.37	0.80	**120**	0.81	0.36	0.77	128

TABLE IV
A COMPARISON OF INIT, GP, AND PSA APPROACHES

Class	INIT			GP			PSA		
	Precision	*Recall*	F_β	*Precision*	*Recall*	F_β	*Precision*	*Recall*	F_β
C_1	0.85	0.34	0.80	**0.93**	0.34	0.87	0.89	**0.63**	**0.88**
C_2	0.78	0.22	0.71	**0.81**	0.27	**0.75**	0.69	**0.41**	0.67
C_3	0.73	0.05	0.48	**0.74**	0.06	0.52	0.71	**0.11**	**0.59**
C_4	0.9	0.63	0.89	0.9	**0.64**	**0.89**	**0.93**	0.33	0.87
C_5	**0.85**	0.44	0.82	0.85	**0.47**	0.82	0.84	**0.69**	**0.83**
C_6	0.77	**0.89**	0.77	0.77	0.89	0.77	**0.93**	**0.61**	**0.91**
AVG	0.81	0.43	0.75	0.83	0.45	0.77	**0.83**	**0.46**	**0.79**

D. Practicality Evaluation

The regular expression based classifier can be used as a secondary verification for fully interpretable medical data processing. Compared with the "black box" machine learning approaches, medical experts are able to understand and modify the regex rules to get better solution. We design the experiment to evaluate the interpretability of regular expressions generated by different approaches. Specifically, 50 manually composed regular expressions, 50 auto-generated regular expressions by GP, and 50 auto-generated regular expressions PSA are randomly selected for third-party practicality blind evaluation. The scores of the three methods is shown in Table V.

TABLE V
INTERPRETABILITY EVALUATION

Score / Method	1 *(cannot be used)*	2 *(major revisions required)*	3 *(moderate revisions required)*	4 *(minor revisions required)*	5 *(can be used directly)*
GP	3.5				
PSA	3.9				
Experts	4.7				

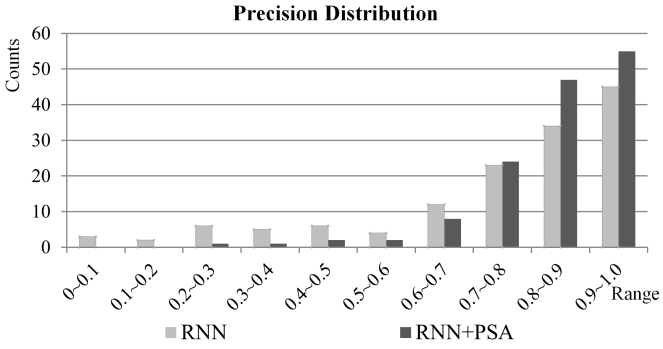

Fig. 3. Precision distribution of RNN and RNN+PSA methods

According to the third-party blind assessment, most auto-generated classifiers by PSA are well readable and can be applied to practical use after some or minor revisions, which benefits from both the proposed solution structures and the use of word vector model.

Experimental results in Fig. 3 also show that the performance of our method can be further improved by combining it with the state-of-the-art DNN approaches. Figure 3 demonstrates the performance of the RNN model and the combination of the RNN and the rule-based engine generated by PSA. We believe the proposed method can serve as an important complementary tool for text classification applications that require high levels of interpretability of the solutions. Through the auto-generated regular expression based classifiers, a fully interpretable rule-based engine is derived for medical text classification. On the real-world online medical inquiry data, the rule-based engine is used to perform secondary verification of the top 5 predictions from an RNN model. Results show that the rule-based engine contributes to an average 9.35% improvement (from 74.94% to 84.29%) in precision to the performance of the RNN model. It can be observed that the new method reduces the number of tasks with low precision of less than 60% and increases the number of tasks with high precision of more than 80%.

VI. CONCLUSION

In this work, the construction of regular expressions for medical text classification is transformed into a combinato-

rial optimization problem. A variant of Simulated Annealing method – Pool-based Simulated Annealing – is proposed, which combines the traditional Simulated Annealing with a deep learning word vector model (pre-trained word2vec model). The specially designed neighbor operation in the proposed PSA method fully takes into account the word vector information and aggregates the words with the same semantic meaning without destroying the structure of the solution in the process of population evolution. Compared with the traditional evolutionary algorithm such as simulated annealing and genetic programming, and "black box" machine learning models such as CNN and DNN, PSA classifier presents better performance and interpretability, as well as important practical value in the medical text classification.

In addition, iterative and parallel strategies have been explored for further improvement in the computational time. We also combine the regular expression classifiers with the state-of-art RNN classification model and test the system performance on real-life data. Experimental results show that the proposed method can be effectively used in conjunction with DNN methods to achieve a better overall system performance. In our future research, we plan to embed the existing expert rules into the knowledge graph to improve the local search efficiency. Furthermore, we will explore the use of deep learning based methods to automatically fine-tune the selection of different operators in order to improve the system performance.

REFERENCES

[1] C. Yao, Y. Qu, B. Jin, L. Guo, C. Li, W. Cui, and L. Feng, "A Convolutional Neural Network Model for Online Medical Guidance," *IEEE Access*, 2016.

[2] Ö. Uzuner, B. R. South, S. Shen, and S. L. DuVall, "2010 i2b2/VA challenge on concepts, assertions, and relations in clinical text," *Journal of the American Medical Informatics Association*, 2011.

[3] Z. C. Lipton, D. C. Kale, C. Elkan, and R. Wetzel, "Learning to diagnose with LSTM recurrent neural networks," *arXiv preprint arXiv:1511.03677*, 2015.

[4] A. N. Jagannatha and H. Yu, "Structured prediction models for RNN based sequence labeling in clinical text," in *Proceedings of the conference on empirical methods in natural language processing. conference on empirical methods in natural language processing*, vol. 2016, p. 856, NIH Public Access, 2016.

[5] A. N. Jagannatha and H. Yu, "Bidirectional RNN for medical event detection in electronic health records," in *Proceedings of the conference. Association for Computational Linguistics. North American Chapter. Meeting*, vol. 2016, p. 473, NIH Public Access, 2016.

[6] M. K. Dalal and M. A. Zaveri, "Automatic Text Classification: A Technical Review," *International Journal of Computer Applications*, 2011.

[7] V. Mitra, C. J. Wang, and S. Banerjee, "Text classification: A least square support vector machine approach," *Applied Soft Computing Journal*, 2007.

[8] G. P. C. Fung, J. X. Yu, H. Lu, and P. S. Yu, "Text classification without negative examples revisit," *IEEE Transactions on Knowledge and Data Engineering*, 2006.

[9] G. Salton, A. Wong, and C. S. Yang, "A Vector Space Model for Automatic Indexing," *Communications of the ACM*, 1975.

[10] M. Bernotas, K. Karklius, R. Laurutis, and A. Slotkiene, "The peculiarities of the text document representation, using ontology and tagging-based clustering technique," *Information Technology and Control*, 2007.

[11] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *1st International Conference on Learning Representations, ICLR 2013 - Workshop Track Proceedings*, 2013.

[12] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations ofwords and phrases and their compositionality," in *Advances in Neural Information Processing Systems*, pp. 3111–3119, 2013.

[13] Y. Kim, "Convolutional neural networks for sentence classification," in *EMNLP 2014 - 2014 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, 2014.

[14] B. S. Harish, D. S. Guru, and S. Manjunath, "Representation and classification of text documents: A brief review," *IJCA, Special Issue on Recent Trends in Image Processing and Pattern Recogniton*, 2010.

[15] A. Moreo, E. M. Eisman, J. L. Castro, and J. M. Zurita, "Learning regular expressions to template-based FAQ retrieval systems," *Knowledge-Based Systems*, 2013.

[16] H. Dalianis, J. Sjöbergh, and E. Sneiders, "Comparing manual text patterns and machine learning for classification of e-mails for automatic answering by a government agency," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2011.

[17] Y. Li, R. Krishnamurthy, S. Raghavan, S. Vaithyanathan, and H. V. Jagadish, "Regular expression learning for information extraction," in *EMNLP 2008 - 2008 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference: A Meeting of SIGDAT, a Special Interest Group of the ACL*, 2008.

[18] R. Bai, J. Blazewicz, E. K. Burke, G. Kendall, and B. McCollum, "A simulated annealing hyper-heuristic methodology for flexible decision support," *4OR*, 2012.

[19] M. Cui, R. Bai, Z. Lu, X. Li, U. Aickelin, and P. GE, "Regular Expression Based Medical Text Classification Using Constructive Heuristic Approach," *IEEE Access*, vol. 7, pp. 147892–147904, 2019.

[20] J. Koza, *Genetic programming: on the programming of computers by means of natural selection*. MIT press, 1992.

[21] P. S. Ngan, M. L. Wong, K. S. Leung, and J. C. Y. Cheng, "Using Grammar Based Genetic Programming for Data Mining of Medical Knowledge," *Genetic Programming 1998: Proceedings of the Third Annual Conference*, 1998.

[22] A. Cetinkaya, "Regular expression generation through grammatical evolution," in *Proceedings of GECCO 2007: Genetic and Evolutionary Computation Conference, Companion Material*, 2007.

[23] W. B. Langdon and A. P. Harrison, "Evolving regular expressions for genechip probe performance prediction," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2008.

[24] A. Bartoli, G. Davanzo, A. De Lorenzo, M. Mauri, E. Medvet, and E. Sorio, "Automatic generation of regular expressions from examples with genetic programming," in *GECCO'12 - Proceedings of the 14th International Conference on Genetic and Evolutionary Computation Companion*, 2012.

[25] B. J. Ross, "Probabilistic Pattern Matching and the Evolution of Stochastic Regular Expressions," *Applied Intelligence*, 2000.