Data-Driven Regular Expressions Evolution for Medical Text Classification Using Genetic Programming

Jiandong Liu
School of Computer Science
University of Nottingham Ningbo China
Ningbo, China
jiandong.liu@nottingham.edu.cn

Ruibin Bai
School of Computer Science
University of Nottingham Ningbo China
Ningbo, China
ruibin.bai@nottingham.edu.cn

Zheng Lu
School of Computer Science
University of Nottingham Ningbo China
Ningbo, China
zheng.lu@nottingham.edu.cn

Peiming Ge
Techonology Department
Ping An Health Cloud Company Limited China
Shanghai, China
gepeiming649@jk.cn

Uwe Aickelin
School of Computing and Information Systems
University of Melbourne
Melbourne, Australia
uwe.aickelin@unimelb.edu.au

Daoyun Liu
Techonology Department
Ping An Health Cloud Company Limited China
Shanghai, China
liudaoyun035@jk.cn

***Abstract*—In medical fields, text classification is one of the most important tasks that can significantly reduce human workload through structured information digitization and intelligent decision support. Despite the popularity of learning-based text classification techniques, it is hard for human to understand or manually fine-tune the classification for better precision and recall, due to the black box nature of learning. This study proposes a novel regular expression-based text classification method making use of genetic programming (GP) approaches to evolve regular expressions that can classify a given medical text inquiry with satisfaction. Given a seed population of regular expressions (randomly initialized or manually constructed by experts), our method evolves a population of regular expressions, using a novel regular expression syntax and a series of carefully chosen reproduction operators. Our method is evaluated with real-life medical text inquiries from an online healthcare provider and shows promising performance. More importantly, our method generates classifiers that can be fully understood, checked and updated by medical doctors, which are fundamentally crucial for medical related practices.**

***Index Terms*—text classification, genetic programming, co-occurrence matrix**

I. Introduction

Given imminent proliferation of medical data on the Internet and the increasing needs for online medical service of current society, applying AI algorithm in medical service has become one of the most active research topics in the last few decades. Text classification, as a problem in general being studied for many years, has been introduced to medical domain to improve service performance [1]–[3]. Classifying relevant medical data into clinical informative categories such as symptoms or disease could vastly reduce human labor cost in medical services. The applications of medical text classification not only can help doctors improve service quality by providing more structured information, but also build the foundation for more advance application such as automated/intelligent diagnosis. During the past several decades, many Machine Learning (ML) based solutions have been proposed for text classification problems. For example, Support Vector Machine (SVM) has been used to classify text documents [4]. To use SVM, a feature vector is extracted from text by bag-of-words (BOW) model. Although widely used, BOW only has limited semantic representation power. To improve this, word embeddings are introduced to automatically learn semantic relation among words [5]. By using word embeddings, neural network based methods such as Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) have brought significant advances in text classification [6], [7]. With help of convolutions of features, language patterns in texts can be learned without explicitly analyzing semantics. Despite great advancement in terms of precision and recall, text classification still struggles to incorporate domain knowledge into classifiers for better performance and more importantly better understandability. This is especially true for medical text classification. In NLP tasks, ML based methods are often criticized about their "black box" nature, impenetrable and inexplicable workings process [8], because solutions cannot be interpreted and modified by human experts with their domain

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

knowledge when classification performance is not satisfying. In contrast to ML methods, rule-based approaches attempt to simulate human-like manners by constructing rules according to domain experts through techniques such as decision trees. Expert system is a typical application of simulating human-like decision manners and has been adopted in many text process tasks since 1990 [9], [10]. While easy to understand with reasonable performance, rule-based approaches are considered to be highly tied to specific domains and therefore adapting to new domains usually requires reconstruction.

In medical domain, text classification applications require not only very high requirements, but also high reliability, verifiability, and ability to be manually updated if necessary. In such scenarios, medical experts, usually physicians, prefer to be able to verify the working process using their own domain knowledge and do not blindly trust a "black boxes" system that sometimes do not provide correct classification performance and even worse almost impossible to improve manually. As a result, in many cases rule-based approaches are preferred by these medical experts compared to ML based approaches, for its interpretability even though it requires additional human efforts for initial preparations and later system updates. In recent years, there have been many deep learning-based algorithms applied to medical text classifications [11], [12] and their performance has reached a very high level statistically. But still, the "black box" nature of deep learning methods are not preferable for direct use in practice [1].

Among many rule-based approaches, regular expression is one of the most popular techniques to provide an interpretable solution to text classification [2]. Regular expression-based approach is a classic text classification technique that uses predefined pattern matchings to provide binary text classifications. The technique has been used in many Natural Language Processing (NLP) studies in medical domains [3], [13]. Our best knowledge, in most of such studies, regular expressions are constructed manually by experts with domain knowledge. Such human efforts are very labor intensive and error-prone. In addition, those regular expressions are normally constructed by a group of domain experts instead of a single one. Hence, the resultant regular expressions often lack required standards and the quality of each individual one highly depends on the particular expert's experience and skills. To overcome this drawback, there have been a few research focus on automatic generation of regular expressions from big data [1], [2], [14], [15]. [1] proposed an iterative constructive heuristic method to generate regular expressions which can serve as drafts for manual improvement. However, like many greedy heuristics, the quality of this constructive heuristics is not guaranteed. None of these studies can produce classification performance that is comparable to those from manually generated regular expressions. This is caused by the fact that searching space for automatically generated regular expressions are enormous due to the huge feature words and their combinations using various regular expression operators. Searching for the optimal regular expressions in the entire solution space is extremely time consuming if even possible.

Furthermore, most previous automatic regular expression generation methods are designed to increase accuracy instead of facilitating the interpretability of their output regular expressions. While regular expressions are meant to be readable, those automatic generated ones are often long and making little sense to human experts and very hard for further manual improvement. We argue that, in real world scenario, clinical text especially those narrative ones produced by patients usually contain typos, abbreviations and non-standard terminologies. Automatically generated regular expressions based on such data can often result in overfitting if without final fine-tuning by human experts.

To address those problems, we study an automatic regular expression generation method to classify clinical text to support clinical diagnosis process with informative medical guidance. Our method is based on a Genetic Programming (GP) framework that is specifically modified to satisfy needs for evolving regular expressions from real world applications. Our model has been tested with large size real world clinical data and experiments prove that our model produces very satisfactory results in medical text classification task. More importantly, the regular expressions generated by our method are more interpretable and structured friendly for human experts fine-tuning and checking.

The main contributions of this paper are summarized as follows:

1) A data driven GP with controlled tree depth was proposed specifically to learn pattern of medical narrative texts aiming at both better classification performance and interpretability of solutions. Our work can be considered as a novel attempt that uses optimization algorithm to automatically learn regular expression from medical domain data and provides much guidance value in real world application.

2) A hybrid model combining machine learning and regular expression approaches are proposed for better performance from both worlds. This hybrid model uses regular expression classifiers as complementary part for machine learning algorithm to improve the overall classification performance.

II. RELATED WORK

Although machine learning approaches have shown promising performance in text classification tasks, practitioners often criticize their lack of interpretability, which provides human with no way of fine-tuning. As mentioned in previous section, regular expressions are often adopted in real world scenario to generate solutions in the situation where interpretability is essential. In general, regular expressions are mainly adopted in NLP tasks such as text classification [2], [15], text extraction [16], etc. In order to reduce labor cost in construction regular expressions, many works exploited various ways of automatic generation of regular expression solutions [17], [18]. While producing promising results, these works usually focus on data with limited complexity and most of their regular expression solutions are syntactically simple. In our opinion, such regular expression solutions are not capable to solve NLP tasks with complex data, such as those presented in this work. In addition,

those works have not been thoroughly applied to specific domain with large amounts of real-world data [19]. Thus, the robustness and suitability of these techniques for solving real world problem is still questionable. In order to test suitability, attempts of learning regular expression from large size of real domain data were conducted in domains detecting spam emails [20] and HTML detection [21]. In medical research field, regular expression based approaches are applied to text extraction from clinical records [22], symptom classification [2], and semantic recognition [23]. While those applications in real world problems demonstrate that learning regular expression from large size of data is feasible, solutions remain in simple architecture and have limited learning capability in dealing with long sequence of text. The drawback of those learning based approaches call for systems that are sophisticated enough to cope with long text. Along with the need for more sophisticated learning models, the solutions should be complex enough to find hidden pattern in training data. However, such models make output solutions much harder to be understood by human, which leave no easy way for further fine-tuning.

To provide better performance, the process of our framework takes the advantage of GP [24], which employs the Darwinian principle of evolution to find a solution. Usually, the applications of GP in classification use only precision and recall as evolving guidance as they are the main objectives. Obviously, such traditional GP is limited by the fact that it takes no other consideration into account. We argue that in order to generate interpretable solutions, our technique has to integrate interpretability as evolving guidance into GP system.

Ours is a novel framework of automatic learning regular expression via GP based method to compose effective and interpretable regular expressions.

III. THE PROBLEM AND SOLUTION ENCODING

In this work, the training data are manually labeled with a set of pre-defined categories by human experts.

A. Problem

Given a set of labeled inquires Q and a group of pre-defined categories C, classify each $q \in Q$ to one category $c \in C$. For each category c, a regular expression vector R_c is developed to match all the inquiries belonging to this category. Therefore, the solution to the problem is a list of n regular expression vectors:

$$(R_1, R_2, R_3, R_4, R_5, R_6 \ldots . R_n)$$

where vector R_c is designed for category c's classification and contains a number of regular expressions:

$$R_c =< r_c^1, r_c^2, ..., r_c^m >$$

To check whether a particular inquiry q belongs to a category c, the corresponding regular expressions in R_c are executed one by one sequentially. If the inquiry q is matched by any regular expression in R_c, the inquiry q is said to be in category c. Otherwise, the inquiry q does not belong to the given category. For each regular expression r_i, the following structure is adopted

$$ri =< P\#\_\#N >$$

where P and N are respectively the positive and negative parts and they are combined by NOT function to form a regular expression. A given regular expression containing NOT function matches strings which are matched by positive part P and not matched by negative part N.

B. Data

We collected online clinic consultation data from our collaborator for both training and testing. In this paper a total of 4,634,742 Chinese text records are collected within two weeks time. All collected records are labeled by medical experts from our collaborator. To standardize our framework, we use International Classification of Disease Ninth Revision (ICD-9) principle as labelling standards. Table I shows a sample of (translated) records that we collected.

TABLE I
A SAMPLE OF MEDICAL INQUIRY RECORDS WITH LABELS

Inquires	**Disease category**
Biphenyl double fat drop pill is gone, please prescribe for me.	Hepatopathy
Cannot sleep at night, too much dreams.	Insomnia
Always sleep talking, Feel stressed	Insomnia
I have a fever and coughs.	Pneumonia
My girlfriend gets a cold and sneezes a lot.	Upper respiratory infection
I started burping 2 days ago, it cannot stop.	Adult indigestion
I see a little blood on my underwear.	Vaginal bleeding

In our framework, there are 776 medical categories predefined by experts. We sorted categories according to their sizes of inquiries and found that the largest 30 categories make up to about 50% of the total inquires in our data as shown in figure 1.

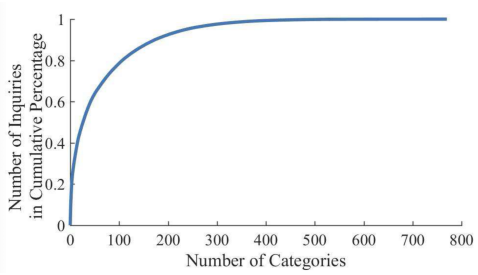

Fig. 1. Number of categories and inquiries in cumulative percentage. [1]

Thus, in our experiment, we randomly select records from the top 30 categories to test our proposed methods. In order to analyze the impact of size of the training data set on the performance of the algorithm, we formed 3 sub data sets with different size (2 million, 0.5 million and 0.2 million) and randomly select 80% and 20% of each sub data set to form training set and testing set.

C. Proposed regular expression syntax

In order to generate regular expression that can be fine-tuned, our model aims at achieving good performance as well as readability for human experts to understand and modify solutions. In this paper, we define a solution as interpretable only if it can be easily read and fine-tuned by experts. Thus, we proposed a novel regular expression syntax that guarantees good interpretability. In the default regular expression grammar, a regular expression is a character class or literals combined by operators. In general regular expression, there are 3 types of operators usually used: alternation, look around, quantifiers.

In this paper, we restricted our regular expressions to 3 new operators only on the basis of alternation and quantifiers. Other more sophisticated operators are not used in the automated expression generation but can be used by human experts during fine-tuning stage. The regular expressions from our system will contain only feature words which are connected by proposed functions (see Table II). In addition, Table III is the terminal set of words and expressions used to form regular expressions. With help of our newly introduced syntax, our model reduces search space for optimization and at same time improves interpretability of solutions.

TABLE II
THE FUNCTION SET IN THE PROPOSED REGULAR EXPRESSION SYNTAX

Function	Description
$OR(w_1, ...w_n)$	Function to match texts containing $w_1$or ... or w_n. The element of OR function can be functions or feature words
$AD(e_1, e_2, \{a, b\})$	Function to match texts containing e_1,e_2 and e_1 is on the left of e_2. Distance between e_1, e_2 should lays in range {a, b}. e_1,e_2 are corresponding expression to each other.
$NOT(P, N)$	Function to match texts which are matched by positive part P and not matched by negative part N. Negative part N is formed by a group of negative expression N_i connected by OR function. Negative expression N_i can be feature words or functions.

TABLE III
THE TERMINAL SET IN THE PROPOSED REGULAR EXPRESSION SYNTAX

Symbol	Descriptions
w	A feature word in word dictionary* extracted from training set
e	An expression which is formed by a group of feature words connected by OR function

*Word dictionary will be explained later.

For the purpose of better readability of regular expression, the following constraints are also applied to each regular expression r_c^m. Later on, we will show, through experiments, that these constraints do not affect the quality of solutions but help reducing the search space of our GP significantly.

1) First structure of regular expression r_c^m is NOT function which connects P and N

2) Positive part P is formed by two expressions which connected by AD function.

3) Negative part N is formed by a group of negative expression N_i connected by OR function. N_i can be a feature word or a function.

4) Expressions of AD function should not contain any other nested functions except for OR function

5) Function OR in positive part should not contain any other nested functions except for itself.

Figure 2 is an example of regular expression formed by feature words and proposed syntax function.

IV. METHODOLOGY

In this section, we propose a data-driven genetic programming method to evolve regular expressions. Figure 3 illustrates the overall stages of this method, including pre-processing, feature extraction and solution searching.

A. Pre-processing

In pre-processing, we employ a popular Chinese word segmentation method, Jieba [25], to segment input text into words (note that our data is in Chinese as described in the previous section). For a given category c, all its inquiries q in Q_c are considered as positive set and the rest of inquiries (denoted by $Q_{\bar{c}}$) are considered as negative set. $Q_{\bar{c}}$ is the complementary set of Q_c given the universe set Q. After segmentation of positive set and negative set, positive word dictionary w_p and negative word dictionary w_n are obtained, together with the frequency of each word over the whole data set.

B. Feature extraction

Over-fitting is one of the common problems in machine learning when solving classification tasks, where model give well performance in training/validation data but poor performance in testing. To solve this problem, we introduced a measurement, average word frequency [1], to quantify uniqueness of feature words. Average word frequency f_w^c is defined as the frequency of a given word $w \in W$ existing in all inquiries in Q_c divided by the number of sentences in inquiries belonging to Q_c. The measurement is calculated as below:

$$f_w^c = \frac{\sum_{q \in Q_c} f_{w,q}^c}{\sum_{q \in Q_c} l_q} \tag{1}$$

where l_q is the number of sentences in inquiry q and $f_{w,q}^c$ is the number of times word $w \in W$ existing in an inquiry $q \in Q_c$. The average word frequency indicates how popular a given word is in each category. A pre-defined threshold will be used to select words whose average word frequency is higher than threshold to be included as feature words.

In addition to average word frequency, a co-occurrence matrix is also calculated. We argue that a regular expression with good interpretability should contain hidden feature pattern of text for a given category and human should be able to understand the matching objective by simply reading it. Thus, being able to extract hidden feature patterns from training data

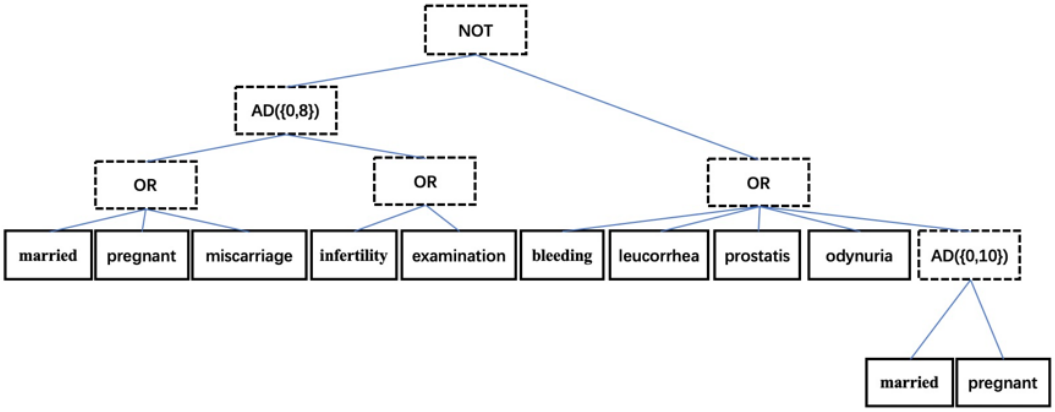

Fig. 2. An example GP tree of a regular expression

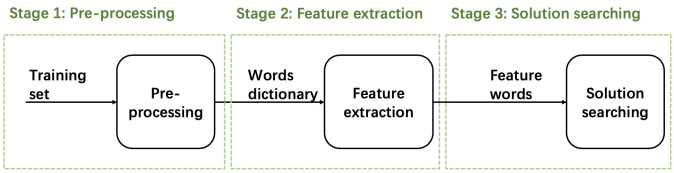

Fig. 3. Stages of proposed GP-based system

is vital in feature extraction process. In order to achieve that, we applied co-occurrence matrix that describes how words occur together because co-occurrence data help our framework to captures the relationship between feature words.

Formally, co-occurrence matrix in our framework describes the frequency of a pair of feature words existing in one inquiry in a given order. For each category c, two matrices are constructed for both positive and negative feature words dictionaries w_p, w_n extracted from Q_c and $Q_{\bar{c}}$, respectively. The co-occurrence formula is defined as follows:

$$M(i,j) = \sum_{q \in Q_c} \begin{cases} 1 & \text{if } pos_q(i) < pos_q(j) \\ 0 & \text{Otherwise} \end{cases} \quad (2)$$

where i and j represent feature words in inquires set and $pos_q(i)$ denotes the position of a given word i in a given inquiry q.

C. Solution searching

We use genetic programming (GP) to evolve regular expressions. The overview of the algorithm is given in Figure 4.

1) Initial population: The initial population can be given by human experts as guidance or randomly selected from positive words dictionary. In this study, categories are pre-defined by medical experts to represent groups of similar symptoms, which means the category names usually contain domain knowledge. In order to utilize the domain knowledge, we firstly form a regular expression by category names as shown in Table IV for examples. The distance element of AD function is randomly generated among range (1,10), and it will evolve during mutation operation. The negative part of every initial individual is w_n^1, the most frequent word in negative word dictionary. Then, we use the top (N-n) frequent words in positive words dictionary to form the rest of population in the same way, where N is the pre-defined population size and n is the number of feature words in category names.

TABLE IV
EXAMPLE OF INITIAL INDIVIDUAL GENERATED BY CATEGORIES NAME

Name of categories	Individuals generated by categories name
Flu consultation	NOT(AD(flu,flu,{0,8}),OR(w_n^1)) NOT(AD(consultation,consultation,{0,6}),OR(w_n^1))
Bitten by mammals	NOT(AD(bitten,bitten,{0,9}),OR(w_n^1)) NOT(AD(mammals,mammals,{0,4}),OR(w_n^1))

In our study, each individual starts with only one regular expression, but we will keep inserting regular expressions to the existing one to increase individual's complexity (and more importantly the fitness) until the end of evolution. In our experiment, for every 500 generations, a regular expression is formed by a feature word randomly selected from positive word dictionary and most frequent word of negative dictionary and the regular expression will be inserted to each individual in population.

2) Adaptive genetic operators: After initial population is produced, genetic operators (crossover, mutation) are applied to all these individuals over generations. In this paper, we adopt single-point crossover and shrink mutation [26] methods to generate child individuals. we propose a self-adaptive genetic operation which can be described as follows:

$$P_c = Sigmoid[a_c(\frac{f_{avg} - f}{f_{max} - f_{avg}})] \quad (3)$$

$$P_m = Sigmoid[a_m(\frac{f_{avg} - f}{f_{max} - f_{avg}})] \quad (4)$$

where P_c, P_m are the probability of crossover, mutation and f is the fitness. f_{max}, f_{avg} are the highest and average fitness in current population and a_c, a_m are speed parameters of evolution. Those equations assign different probabilities of being operated to individuals according to their fitness

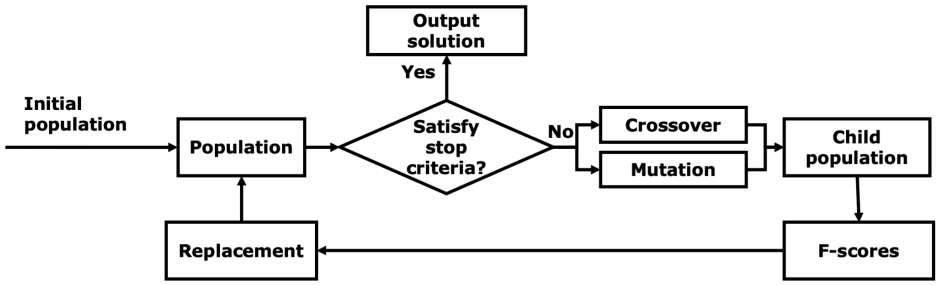

Fig. 4. Overview of the genetic programming algorithm for automated regular expression generation.

via sigmoid function. Individuals with lower fitness will be assigned higher probability, which will accelerate evolution of low fitness individuals.

3) Interpretability: In our framework, a regular expression is considered interpretable when it is easy for doctors to read and fine-tune for better performance. Thus, an interpretable regular expression should be capable to learn hidden language pattern in a given category, and the feature words forming regular expression should be logically correlated. In order to find the hidden pattern, co-occurrence matrix(defined in Section IV B) is used. Co-occurrence matrix is used as guidance of selecting words in mutation operation occurred on positive part. According the proposed syntax and format constraints, positive part of a regular expression is formed by two group of expression combined by AD function. When mutating feature words in AD function, a feature word $\bar{w}$ from expression $\bar{e}$ will be replaced by another feature word w selected from the word dictionary. Thus, calculating the correlation level of feature word w towards the corresponding expression e will be helpful in selecting words. In our work, we propose a measurement of correlation level as follow:

$$e = OR(w_1, w_2 \ldots, w_n)$$

$$C(w,e) = \sum_{i\,=\,1}^{n} \begin{cases} M(w,w_i) & \text{if } pos(\bar{e}) < pos(e) \\ M(w_i,w) & \text{Otherwise} \end{cases} \quad (5)$$

where $M(w, w_i)$ is the function that returns co-occurrence frequency of word w towards words w_i. $pos(e)$ denotes the position of a given word expression e in a given AD function.

The correlation measurement function will be calculated for each word in positive word dictionary when the mutation is operated towards corresponding expressions. After obtaining every word's correlation towards expressions, we generate a probability of being selected via normalization as follows.

$$positive\ word\ dictionary\ = (w_1, w_2 \ldots, w_m)$$

$$probability(w)\ =\ \frac{C(w,e)}{\sum_{i\,=\,1}^{m} C(w_i,e)} \quad (6)$$

By doing so, our system integrates co-occurrence matrix into mutation operation to produce interpretable solution via expressing hidden language pattern.

4) Fitness function: Similar to other works, we use F score as only fitness function in our method. For a binary classification problem, F-Score is defined as follows:

$$F_\beta = \frac{\left(\beta^2+1\right) Precision * Recall}{\beta^2 * Precision + Recall} \quad (7)$$

where β is the balancing parameter which decides preference between precision and recall.

5) Replacement: After offspring population is generated, a replacement operator is applied on them to select N individuals according to their fitness.

6) Stopping criterion: The process of evolution continues until there is no change in the highest F-score of population for consecutive generations.

V. EXPERIMENT

We conducted a comprehensive experiment to evaluate the performance of our algorithm on real-life data.

A. Regular Expression Classifier

As discussed in previous part, the number of regular expressions that every solution contains depends on the stopping criteria and evolving process. In order to observe the impact of training size, we test the model on 3 different sizes of data set. Table V is the performances of our model on 3 data sets. As can be seen from the table, the larger size of training data leads to better performance. Furthermore, we include a baseline results from [1], which developed a constructive heuristic approach to generate regular expression classifiers. Our paper and [1] both conducted experiments on same data set.

TABLE V
AVERAGE PERFORMANCE OF REGULAR EXPRESSION CLASSIFIER ON DIFFERENT DATA SET

Training size	200,000	500,000	2,000,000
Precision	0.8245	0.8649	0.9151
Recall	0.5642	0.6153	0.6353
Precision*	0.76	0.83	0.89
Recall*	0.45	0.51	0.57

*Baseline results provided by [1]

Table V gives the average performance of the algorithm across 30 categories. The algorithm's performance varies

among categories because of the diversity nature of different categories. Figure 5 describes randomly selected 10 categories performance results, which are capable to reflect the overall performance of algorithm. From the figure, we can see that the algorithm obtains 96% precision and 91% recall in category *preparing for pregnancy* whereas only achieve 85% precision and 43% recall in category *abnormal vaginal bleeding*. The standard deviation of precision and recall within categories are 6.3% and 18.7%, respectively. The reason behind such variation in terms of performance is largely due to the different concentration level of feature words in different categories. For category *preparing for pregnancy*, feature words are clustered around relevant terms about *baby* and *mother*. In contrast, the feature words of category *abnormal vaginal bleeding* are relatively diversified, ranging from *sexual behavior*, *menstruation* to *endocrine disorder*.

B. Solution Interpretability

Compared with other machine learning based approaches, our framework provides an more interpretable regular expression-based solution via following improvements.

1) Regular expression: In this paper, we adopted regular expression to form solutions, which provide experts with an easy way to read and understand solutions.

2) Interpretability guidance: During GP process, we involved co-occurrence matrix to provide an evolving direction. Co-occurrence matrix allows solutions to include words which are more relevant to current solutions in interpretability aspect.

3) Fine-tuned by experts: The reason we integrate interpretability into GP evolving is that our medical experts expect to fine-tune the regular expressions solutions when they cannot give satisfactory performance. Solutions after fine-tuned will be modified to be more consistent with experts experience. Thus, fine-tuned solutions will be capable to give better performance. Table VI shows an example of regular expression solution fine-tuned by our medical experts.

TABLE VI
EXAMPLE OF FINE-TUNED REGULAR EXPRESSIONS

Example of fine-tuned regular expressions
.*(pre-pregnancy|pregnant|expectant|cyetic|childing).{0,24}(examination |test| matters|caution|attention).*
.*(married|pregnant).{0,8}(infertility|examination). *#_#.*(bleeding| eucorrhea|prostatis|odynuria) .*
Fine-tuned Regular Expression
.*(pre-pregnancy|pregnant|expectant |cyetic|childing|**family way**| *conception*|**ha.{0,8}bab**).{0,24} (examination|test|matters| caution|attention). *
.*(married|pregnant|**miscarriage**). {0,8} (infertility|examination). *#_#.* (bleeding|leucorrhea|prostatis|odynuria|**period.{0,10}come**).*

C. Performance enhancement by combining with deep learning methods

In addition to human fine-tuning, we combine regular expression solutions with other machine learning model to improve classification performance. We selected Naïve Bayes, SVM, RNN and CNN as baselines and combined each of them with regular expression solutions. We applied the hybrid solution of machine learning and regular expression in our experiment. Specifically, if classification confidence of machine learning model is lower than 0.6, we employ regular expression classifiers on the first 5 predictions in orders. The results are summarized in Table VII. Note that, due to limited computing resources, the results in Table VII are based on 500,00 records data set.

TABLE VII
PERFORMANCE OF HYBRID CLASSIFIERS

Solution	**Precision**	**Recall**
Naïve Bayes	0.71	0.63
Naïve Bayes + Regex	0.82	0.68
Naïve Bayes + fine-tuned Regex	0.89	0.78
SVM	0.77	0.75
SVM + Regex	0.86	0.78
SVM + fine-tuned Regex	0.90	0.81
RNN	0.88	0.79
RNN + Regex	0.88	0.80
RNN + fine-tuned Regex	0.94	0.84
CNN	0.91	0.82
CNN + Regex	0.91	0.82
CNN + fine-tuned Regex	0.94	0.88

VI. CONCLUSION

Although regular expression-based approaches have been used in text classification task for its robustness and easiness to understand, most of related works rely on manually constructing regular expressions which is a very labor consuming process. In this paper, we propose a regular expression learning framework to solve text classification problem. In contrast to other regular expression learning approaches, we adopted a GP optimization technique to generate interpretable solutions with help of novel regular expression syntax and encoding. In our case, solutions generated by the proposed framework not only solve the "black box" problem of current popular machine learning algorithms in text classification but also provide an easy way for doctors to modify regular expression classifiers when the performance is not satisfactory. We tested our algorithm on real-life medical text inquires and the experimental results show that the proposed algorithm is able to obtain good quality solutions. Although our framework cannot produce solutions which are comparable to machine learning solutions, it can be combined with machine learning approaches to form a hybrid model for even better performance.

In the future, we will systematically investigate in the impact of the proposed regular expression encoding on the solution quality and the search efficiency. We will also look into more efficient ways to combine GP based regular expressions with deep neural networks in addressing other interpretable text classification problems.

REFERENCES

[1] M. Cui, R BAI, Z. Lu, X. Li, U. Aickelin, and P. Ge, "Regular Expression Based Medical Text Classification Using Constructive Heuristic Approach," vol. 7, 2019.

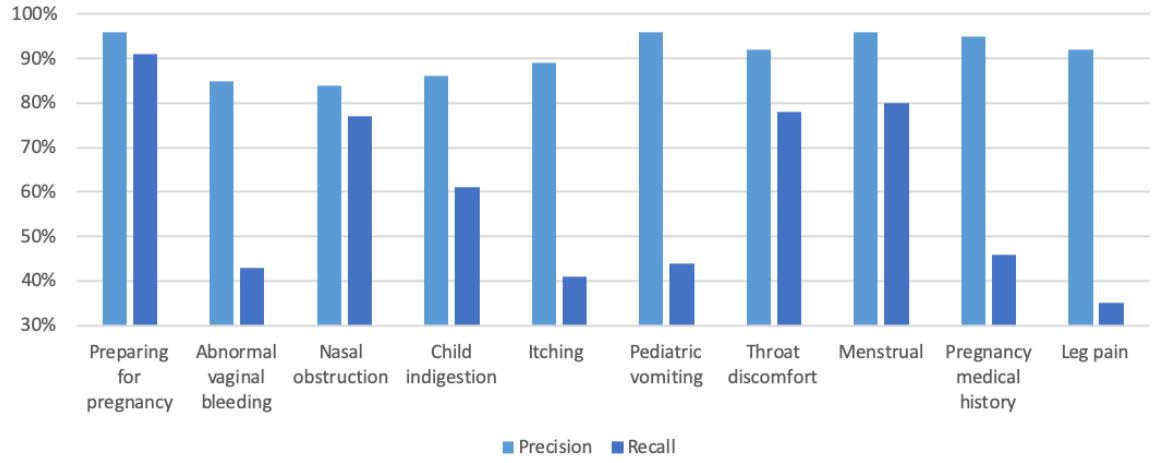

Fig. 5. Sample of 10 categories performance

[2] D. D. A. Bui and Q. Zeng-Treitler, "Learning regular expressions for clinical text classification," J. Am. Med. Informatics Assoc., vol. 21, no. 5, pp. 850–857, 2014.

[3] M. A. Murtaugh, B. S. Gibson, D. Redd, and Q. Zeng-Treitler, "Regular expression-based learning to extract bodyweight values from clinical notes," J. Biomed. Inform., vol. 54, pp. 186–190, 2015.

[4] T. Joachims, "Text categorization with support vector machines: Learning with many relevant features," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 1398, pp. 137–142, 1998.

[5] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 1st Int. Conf. Learn. Represent. ICLR 2013 - Work. Track Proc., pp. 1–12, 2013.

[6] P. Liu, X. Qiu, and H. Xuanjing, "Recurrent neural network for text classification with multi-task learning," IJCAI Int. Jt. Conf. Artif. Intell., vol. 2016-Janua, pp. 2873–2879, 2016.

[7] A. Hassan and A. Mahmood, "Convolutional Recurrent Deep Learning Model for Sentence Classification," IEEE Access, vol. 6, pp. 13949–13957, 2018.

[8] A. Hart and J. Wyatt, "Evaluating black-boxes as medical decision aids: issues arising from a study of neural networks," Med. Informatics, vol. 15, no. 3, pp. 229–236, Jan. 1990.

[9] S. Gauch and J. B. Smith, "An expert system for searching in full-text," Inf. Process. Manag., vol. 25, no. 3, pp. 253–263, 1989.

[10] K. Taghva, J. Borsack, and A. Condit, "Expert system for automatically correcting OCR output," in Document Recognition, 1994, vol. 2181, pp. 270–278.

[11] C. Yao et al., "A Convolutional Neural Network Model for Online Medical Guidance," IEEE Access, vol. 4, pp. 4094–4103, 2016.

[12] M. E. Matheny et al., "Detection of blood culture bacterial contamination using natural language processing.," AMIA Annu. Symp. Proc., vol. 2009, pp. 411–415, 2009.

[13] A. Turchin, N. S. Kolatkar, R. W. Grant, E. C. Makhni, M. L. Pendergrass, and J. S. Einbinder, "Using Regular Expressions to Abstract Blood Pressure and Treatment Intensification Information from the Text of Physician Notes," J. Am. Med. Informatics Assoc., vol. 13, no. 6, pp. 691–695, 2006.

[14] A. Bartoli, G. Davanzo, A. De Lorenzo, M. Mauri, E. Medvet, and E. Sorio, "Automatic generation of regular expressions from examples with genetic programming," GECCO'12 - Proc. 14th Int. Conf. Genet. Evol. Comput. Companion, pp. 1477–1478, 2012.

[15] R. Bakker, M. Marx, and F. Jansen, "Evolving Regular Expression Features for Text Classification with Genetic Programming," no. December, 2018.

[16] A. Bartoli, A. De Lorenzo, E. Medvet, and F. Tarlao, "Inference of Regular Expressions for Text Extraction from Examples," IEEE Trans. Knowl. Data Eng., vol. 28, no. 5, pp. 1217–1230, 2016.

[17] F. Denis, "Learning regular languages from simple positive examples," Mach. Learn., vol. 44, no. 1–2, pp. 37–66, 2001.

[18] A. Cetinkaya, "Regular expression generation through grammatical evolution," Proc. GECCO 2007 Genet. Evol. Comput. Conf. Companion Mater., pp. 2643–2646, 2007.

[19] O. Cicchello and S. C. Kremer, "Inducing grammars from sparse data sets: A survey of algorithms and results," J. Mach. Learn. Res., vol. 4, no. 4, pp. 603–632, 2004.

[20] P. Prasse, C. Sawade, N. Landwehr, T. Scheffer, and I. Titov, "Learning to identify concise regular expressions that describe email campaigns," J. Mach. Learn. Res., vol. 16, pp. 3687–3720, 2015.

[21] E. Kinber, "Learning regular expressions from representative examples and membership queries," in International Colloquium on Grammatical Inference, 2010, pp. 94–108.

[22] D. Redd, J. Kuang, A. Mohanty, B. E. Bray, and Q. Zeng-Treitler, "Regular Expression-Based Learning for METs Value Extraction.," AMIA Jt. Summits Transl. Sci. proceedings. AMIA Jt. Summits Transl. Sci., vol. 2016, pp. 213–21320, 2016.

[23] Y. Huang and H. J. Lowe, "A Novel Hybrid Approach to Automated Negation Detection in Clinical Radiology Reports," J. Am. Med. Informatics Assoc., vol. 14, no. 3, pp. 304–311, 2007.

[24] F. Bian, T. Li, and P. Cong, "Genetic programming," Fenxi Huaxue, vol. 26, no. 6, pp. 783–785, 1998.

[25] J. Sun, "'Jieba'Chinese word segmentation tool," 2018-08-25]. https://github. com/fxsjy/jieba. 2012.

[26] T. Bäck, D. B. Fogel, and Z. Michalewicz, Evolutionary computation 1: Basic algorithms and operators. CRC press, 2018.