International Journal of Computer Applications Technology and Research
Volume 14–Issue 10, 21 – 24, 2025, ISSN:-2319–8656
DOI:10.7753/IJCATR1410.1005

Performance Evaluation of Multi-Module Recognition: From Dictionary Matching to Hybrid Collaborative Methods

Zehua Lv
Chengdu University of Information Technology
College of Communication Engineering
ChengDu, China

Chao Tang
Chengdu University of Information Technology
College of Communication Engineering
ChengDu, China

Ximing Yuan*
Chengdu University of Information Technology
College of Communication Engineering
ChengDu, China

Abstract:With the development of artificial intelligence, symptom recognition in the medical field has come into public view; to address issues such as the insufficient robustness of pure dictionary matching and the weak generalization ability of single rule-based systems, this paper designs and implements a multi-module symptom recognition system. By comparing the performance of pure dictionary matching, regular expressions, rule-based methods, and their combined approach, the paper verifies the effectiveness of a hierarchical collaborative strategy in symptom recognition tasks. Specifically, this method enhances the precise matching ability of pure dictionary matching through regular expressions, and improves the matching ability of pure dictionary matching for fuzzy problems through rule-based methods. The experiment adopts a comparative approach, conducting quantitative evaluations from four dimensions—accuracy, precision, recall and processing speed—and concludes with the advantages of the hybrid model.

Keywords: rule-based methods;pure dictionary matching; regular expressions;recognition;hybrid model

1. INTRODUCTION

This paper designs a hybrid module query system, which constructs a recognition framework based on a dictionary, supplemented by regularization methods and rule-based methods. This integrated solution addresses the recognition bias caused by users' non-standard expressions and provides accurate and comprehensive symptom data for subsequent disease inference. The three components together form a "basic matching-precise optimization-flexible expansion" logic, serving as the core technical support for efficient diagnosis.

The dictionary, a preset structured set of standard symptoms (e.g., "headache", "fever"), acts as the primary reference and screening tool[1]. It first extracts symptoms through direct matching with user input and provides a unified benchmark—all results from regularization and rule-based methods are ultimately mapped to its standard terms, ensuring consistent data for disease inference.

Regularization optimizes dictionary matching by addressing "Same symptoms, different words" through pre-compiled character-set patterns. It identifies variant expressions (e.g., colloquial abbreviations) and converts them to standard dictionary terms, enhancing precision without expanding the dictionary, thus retaining efficiency while overcoming rigid exact-match limitations[2].

Rule-based methods further extend recognition by focusing on semantic associations for "same symptom, distinct expressions". Using logic-based patterns, they identify semantically related but linguistically different descriptions (e.g., "stomach upset" for "abdominal pain") and map them to standard symptoms, handling complex non-standard inputs to broaden coverage[3].

These three components collaborate hierarchically: the dictionary extracts basic symptoms via direct matching; regularization supplements with variant recognition; rule-based methods expand through semantic associations. Their unified output of standardized symptoms ensures efficiency, accuracy, coverage, and flexibility, laying a robust foundation for reliable disease inference and diagnosis suggestions.

2. RELATED WORK

A domain adaptive Chinese sentiment dictionary automatic construction algorithm based on semantic rules was proposed for Qinglan et al. Constructed a Chinese fixed emotion dictionary, effectively eliminating emotional ambiguity[4].Wang et al. established attribute feature dictionary and emotion dictionary[5].Li Jiachuan and others proposed using dictionary annotation resources to enhance the ancient Chinese translation system. The author collected the definitions of common Chinese characters in ancient Chinese and designed a two-stage machine translation framework of dictionary definition selection fusion to obtain effective information from dictionary definitions and remove invalid information[6].Chen Jun et al. proposed a method for constructing a fine-grained emotional dictionary in the field of education based on feature fusion[7].Lin Jing et al. proposed a regular expression matching technique based on a combination of generalized suffix trees and filtering factors[8].Yang Jiajia et al. proposed a high-performance regular expression matching algorithm based on untrusted character comparison, called α FA, to address the problem of low performance in current regular expression matching[9].When Zhu Jun et al. used regular expressions to construct DFAs, the state explosion caused the matching algorithm to require more storage space and running time, resulting in low algorithm efficiency. After adopting rule grouping, the state explosion problem can be suppressed to a certain extent Grouping regular expressions based on historical records in the cache can not only reduce the total number of states and suppress state explosion by using rule

grouping, but also reduce the overhead caused by rebuilding the DFA each time, improve matching efficiency, and enhance the real-time, accurate, and efficient performance of intrusion detection[10].Zhang Guilin et al. proposed a Turkish morphological disambiguation method based on morphological analysis dictionary and context constraint rules, and constructed a practical Turkish morphological disambiguation system through six modules: text preprocessing, named entity recognition, fixed collocation recognition, unknown word processing, morphological analysis and morphological disambiguation[11].

3. SYSTEM ALGORITHM

3.1 Dictionary Matching

Dictionary Matching(DM) is a fundamental technical method in the field of information processing, particularly in natural language processing, data retrieval, and content recognition. Its core lies in relying on a "preset structured dictionary database" to perform "direct matching verification and extraction" of target information from input data. In essence, it is a "precision query logic based on a collection of known entries". Owing to its simple principle and strong interpretability, it is widely applied in scenarios requiring standardized information screening[12].

The core composition of Dictionary Matching includes two parts: first, the "dictionary database", which is a pre-organized collection of target information, usually stored in a structured format (such as key-value pairs, lists), with clear entries (e.g., "standard symptom names", "valid keywords", "entity terms")[13]. For instance, in medical scenarios, it may include standard symptoms like "headache", "fever", and "hypertension"; in text filtering scenarios, it may contain "sensitive words" and "domain-specific keywords". Second, the "matching logic": when processing input data (such as user text, form content, document fragments), the system scans the input content according to preset rules (e.g., full-character matching, case-insensitive matching) to determine whether there is information that is "completely consistent" with the entries in the dictionary database (or meets preset matching conditions). If such information exists, it is determined as a "successful match", and the corresponding entry and additional information (such as symptom labels, classification attributes) can be extracted; if no such information exists, it is determined as "no match", and no additional processing is performed.

Due to its characteristics of "simple implementation and deterministic results", Dictionary Matching has significant advantages in scenarios where high requirements are placed on "accuracy and interpretability" and there are few variants of target information. Common applications include: standardized information extraction, data verification and filtering, and simple retrieval in specific fields. However, pure dictionary lookup has many limitations: weak robustness and coverage that depends on the completeness of the dictionary.

3.2 Regular Expressions

A regular expression (abbreviated as Regex/RegExp) is a standardized string logic used to match, search for, and replace specific character patterns in text[14]. In essence, it is a "character pattern description language". By means of predefined syntax rules, it abstracts complex text matching requirements into concise expressions, thereby efficiently handling tasks such as string filtering, verification, and extraction. It is widely applied in scenarios including programming (e.g., text processing, data cleaning), office work (e.g., document content retrieval), and development (e.g., form validation, log analysis).

The syntax of regular expressions is composed of ordinary characters and metacharacters (characters with special meanings). Mastering core metacharacters is the key to understanding regular expressions. The most commonly used basic syntax is shown in Table 1:

Table 1. Regular Expression Core Syntax Table

Category	Syntax Symbol	Meaning & Explanation	Example
Ordinary Characters	Letters/numbers (e.g., a, 1)	Matches the character itself (no special meaning).	cat matches "cat" (case-sensitive).
Any Character Match	Dot (.)	Matches any single character except \n.	c.t matches "cat", "cot", "c1t".
Quantity Match	Asterisk (*)	Matches preceding character 0+ times.	ab* matches "a", "ab", "abb".
Quantity Match	Plus Sign (+)	Matches preceding character 1+ times.	ab+ matches "ab", "abb" (not "a").
Character Set	Square Brackets ([])	Matches any character inside; supports ranges.	[a-z0-9] matches lowercase letters/ digits.
Predefined Set	\d	Equivalent to [0-9] (matches any digit).	\d{4} matches "2024" (4-digit number).
Boundary Match	Caret (^) / Dollar ($)	Matches start/end of a string.	^Hello matches strings starting with "Hello"; World$ matches those ending with "World".
Group Match	Parentheses (())	Treats content as a group; extracts sub-matches.	(ab)+ matches "ab", "abab".

As a key text processing tool, regular expressions play a core role in breaking through the limitations of basic matching methods and providing more flexible adaptability for symptom recognition. Through predefined character pattern

logic, they can identify diverse expressions that cannot be covered by pure dictionary matching — whether it is colloquial variants, common typos, or non-standard cases such as synonym substitutions in user input, all of which can be accurately captured by corresponding regular patterns. These expressions are then mapped to unified standard symptom labels. This processing method not only expands the coverage of symptom recognition and reduces missed recognition caused by expression differences, but also maintains high processing efficiency by virtue of the characteristics of precompiled patterns. It strikes a balance between "accurate recognition" and "scenario adaptability", serving as an important intermediate link connecting basic dictionary matching and complex rule-based processing, and helping to improve the practicality and reliability of the overall symptom recognition system.

3.3 Rule-based Methods

Rule based methods are a classic algorithmic framework in the fields of artificial intelligence, machine learning, and data processing, characterized by relying on clear, pre-defined rules to complete decisions, data classification, or specific task solving. These rules are usually derived from domain expertise, expert experience, or logical reasoning, essentially transforming human understandable "judgment criteria" into structured, executable instructions that allow the system to gradually process problems according to fixed logic.From an execution perspective, this type of method first constructs a "rule base" that stores rules based on "condition-action" logic, which can include single conditions, multi-condition combinations, and priority settings. Next, an "inference engine" processes input data: it first preprocesses the data, then scans the rule base to identify "triggered rules" that meet the conditions. If multiple rules are triggered simultaneously, a predefined mechanism (such as priority ranking) is used to resolve conflicts. Finally, the actions defined by the selected rules are executed to complete tasks such as classification and recommendation generation. The entire process has clear and traceable bases, requiring no model training. Maintenance only involves adjusting the rule base. In scenarios with clear domain knowledge and stable logic, it offers advantages such as low development costs, rapid implementation, and strong interpretability[15].

The processing procedure of this advanced rule-based system can be summarized as a complete closed-loop process consisting of "Rule System Initialization → Text Input Receipt → Rule Matching → Conflict Resolution → Action Execution → Result Output", as shown in Figure 1, which unfolds specifically through these six stages:

Figure 1 Flowchart of the Rule-based Methods

1) For Rule System Initialization: When the system starts up, it constructs a structured rule base that includes priorities, multi-type conditions (regular expression matching, context judgment, co-occurrence analysis, negation detection) and corresponding actions (classification, exclusion, risk early warning). Meanwhile, it initializes an inference engine integrated with a conflict resolution module and creates an enhanced test set covering basic, composite, edge and conflict scenarios, laying the foundation of rules and tools for subsequent processing.
2) For Text Input Receipt: The system acquires a segment of text to be processed (e.g., "Cough has persisted for 3 days, accompanied by feelings of chest tightness and palpitations"), which serves as the original data source for rule matching.
3) For Rule Matching: The inference engine processes the input text, scans the preconfigured rule base, verifies one by one whether the text meets the trigger conditions of each rule, and filters out all triggered rules that satisfy the conditions. For example, the above-mentioned sample text will trigger the composite rule R3 for "Cough + lasting N days" and the rule R4 for "Co-occurrence of chest tightness and palpitations".
4) For Conflict Resolution: The conflict resolution module processes the filtered triggered rules. First, it sorts the rules in descending order of priority (rules with higher priority are executed first). If there are rules with the same priority, the rule with more specific conditions (i.e., a greater number of conditions) is selected to avoid result confusion caused by overlapping triggering of rules.
5) For Action Execution: The system executes the rule actions determined after conflict resolution. If the action type is "classification", it extracts the labels corresponding to the rules (e.g., R3 corresponds to "cough", R4 corresponds to "chest tightness" and "palpitations") and records the confidence level. If the action type is "exclusion" (e.g., rules R5 and R6 triggered when the text contains negation or test expressions such as "pretend" and "no headache"), it clears the current results and records the reason for

exclusion. If a rule includes a risk early warning (e.g., R4 specifies "Priority assessment of cardiovascular risks is required"), the early warning information is recorded simultaneously.

6) For Result Output: The system integrates all information from action execution, performs deduplication on the extracted labels, and then outputs a complete result including the identified label list, IDs of the applied rules, average confidence level, and risk early warning (if any). Throughout the entire process, the transparency of the rule-based method is retained, and at the same time, through multi-condition combination, priority control and conflict resolution, accurate processing of complex text scenarios is achieved.

4. EXPERIMENTS AND ANALYSIS

To verify the system performance, this experiment adopted the open-source MedXN regular expressions and the CCKS 2018 Medication Entity Annotation Dataset for testing. The tests were conducted in a PyTorch environment under the Windows operating system. The experimental results were statistically analyzed, covering metrics such as accuracy, precision, recall, and processing speed. The experiment shows that compared to the pure dictionary matching method, the combination model's processing speed slows down, but other indicators improve. The specific results are shown in Table 2:

Table 2. Effect Comparison Chart

Model	accuracy	precision	recall	processing speed
DM	40.2%	34.6%	28.5%	0.8 ms/sentence
DM+Regex	64.8%	58.8%	52.3%	1.1 ms/sentence
DM+Rule-based	51.6%	50.2%	41.7%	1.6 ms/sentence
DM+Regex+Rule-based	74.4%	68.8%	65.2%	2.0 ms/sentence

REFERENCES

[1] Zhao Xueqing,&Fu Lu (2024). Research on the compilation of foreign Chinese dictionaries under the background of international Chinese education resource construction Journal of Shaanxi Normal University (Philosophy and Social Sciences Edition), 53 (3), 5-15.

[2] Wang Hao and Wu Junhua (2024). Generation of regular expressions based on natural language syntactic information Computer Science, 51 (S02), 92-97.

[3] Wei Qinglan, He Yu,&Song Jinbao (2025). An automatic algorithm for constructing an adaptive sentiment dictionary based on semantic rules Journal of Beihang University, 51 (7), 2450-2459.

[4] Wei Qinglan, He Yu,&Song Jinbao (2025). An automatic algorithm for constructing an adaptive sentiment dictionary based on semantic rules Journal of Beihang University, 51 (7), 2450-2459.

[5] Huidiaoyan, Wang Zhi, He Zhenhua,&Qin Chunxiu (2025). Fine grained sentiment analysis for online evaluation based on the dictionary TexCNN-Word2Vec combination model Intelligence Theory and Practice, 48 (2), 168-177.

[6] Li Jiahuan, Wu Ruochun, Huang Shujian, Hu Wenjing, Chen Jixuan, Xu Weilu,&Chen Jiajun (2025). Enhanced ancient text machine translation with dictionary definitions Chinese Journal of Information Science, 39 (4), 85-95.

[7] Chen Jun, Xi Ningli, Li Jiamin,&Wan Xiaorong (2023). Constructing an emotional dictionary in the field of education by integrating Skip gram and R-SOPMI Journal of Applied Science, 41 (5), 870-880.

[8] Lin Jing and He Zhenying (2022). A regular expression matching algorithm based on generalized suffix tree combined with filtering factors Computer Applications and Software, 39 (1), 266-270286..

[9] Yang Jiajia, Guan Jian, Yu Zengming, Zhang Lei,&Yao Wangjun (2024). α FA: A high-performance regular expression matching algorithm based on untrusted character comparison Application of Electronic Technology, 50 (6), 57-60.

[10] Zhu Jun (2021). A rule-based grouping algorithm for matching DFA regular expressions Journal of Hunan University of Engineering (Natural Science Edition), 31 (2), 49-53.

[11] Zhang Guilin, Yi Mianzhu, Li Hongxin, Li Jian,&Yi Xiaoyu (2021). Implementation of Turkish morphological disambiguation system based on dictionary and rules Modern Linguistics, 9 (4), 1008-1017.

[12] Wu Shufang,&Yin Kai (2023). Research on constructing a network sensitive dictionary based on sensitive semantics and compound co-occurrence Intelligence Science, 41 (10), 12-20, 39.

[13] Zhang Jun,&Hu Wenfei (2023). The design features and system construction of an introverted Chinese English learning dictionary Foreign Languages, 39 (6), 91-100.

[14] Wang Hao and Wu Junhua (2024). Generation of regular expressions based on natural language syntactic information Computer Science, 51 (S02), 92-97.

[15] Zhang Enyao (2023). Design and Implementation of a Column Control Data Verification System Based on Rule Engine.