#

Research on Recognition Methods for Chinese Sensitive Word Variants (Postprint)

Authors: Fu Cong, Yu Dunhui, Zhang Lingli

Date: 2018-05-02T00:00:00+00:00

Abstract

To purify the network environment, it is necessary to review network information. The identification of sensitive words contained in network information, especially Chinese sensitive word variants, has become an urgent problem that needs to be solved. By analyzing the structural and phonetic features of Chinese characters, a recognition method for Chinese sensitive word variants is proposed. This method designs a Sensitive Word Recognition algorithm based on Confusable Pinyin Grouping (SPGR) for pinyin-based variants, a String Abbreviation Recognition algorithm (SNR) for abbreviated variants, and a Word Splitting-KMP algorithm (WS-KMP) for split-character variants, effectively improving the accuracy and efficiency of sensitive word review. Experimental results show that this method has high recall and precision rates in identifying Chinese sensitive word variants.

Full Text

Study on Identification Methods for Chinese Sensitive Word Variants

Authors: Fu Cong[1], Yu Dunhui[1,2]†, Zhang Lingli[1]
[1]School of Computer Science & Information Engineering, Hubei University, Wuhan 430062, China
[2]Hubei Provincial Center for Education Information Technology Studies, Wuhan 430062, China

Abstract: To purify the online environment, network information must be reviewed. Recognizing sensitive words in network information, particularly variants of Chinese sensitive words, has become an urgent problem. This paper proposes a recognition method for Chinese sensitive word variants by analyzing the structural and phonetic characteristics of Chinese characters. The method designs three specialized algorithms for different variant forms: the Sensitive Pinyin Grouping Recognition (SPGR) algorithm for phonetic variants, the String Name Recognition (SNR) algorithm for abbreviated variants, and

the Word Split KMP (WS-KMP) algorithm for character-split variants based on the KMP algorithm. These algorithms effectively improve the accuracy and efficiency of sensitive word review. Experimental results demonstrate that the proposed method achieves high recall and precision in recognizing Chinese sensitive word variants.

Keywords: variants; sensitive word recognition; edit distance; KMP algorithm

0 Introduction

With the rapid development of the Internet, various information resources have grown exponentially. Illegal discourse (such as pornography, gambling, terrorism, and violent content) frequently appears online [1-2]. This harmful information typically contains sensitive vocabulary that can trigger adverse chain reactions, posing serious threats to national security, social stability, and the healthy formation of the online environment, while causing significant negative impacts. Therefore, sensitive word identification has become an urgent research topic.

Current research on sensitive word recognition is relatively mature, generally based on sensitive word lexicons. The fundamental approach involves searching the text to be examined; if it contains sensitive words, the system flags the text for review. While simple to implement, this method suffers from low search efficiency. To address this, reference [3] proposed the ST-DFA algorithm, which constructs a decision tree for sensitive information based on the first letters of sensitive words' pinyin. Its advantage lies in not relying on a sensitive information corpus and improving detection efficiency, but it cannot handle sensitive word variants.

Research on identifying Chinese sensitive word variants is still in its infancy, with relatively few relevant 研究成果. Reference [4] proposed a method for variant sensitive words by converting special characters into visually similar letters before detection. For example, converting "@" to "a" transforms "@rse" into "arse" for processing. Reference [5] employed machine learning methods, using bigrams and word stems as features to classify text information and detect variants. Reference [6] utilized Bayesian filtering technology to detect malicious content, employing approximate string matching techniques to improve detection effectiveness. Reference [7] proposed a phonetic string matching algorithm to address matches between similarly pronounced strings. These methods work well for English characters but do not consider Chinese sensitive word variants.

Thus, finding more effective algorithms for Chinese sensitive word variant identification is imperative. This paper proposes a Chinese sensitive word variant identification method that combines English string variant recognition techniques with Chinese character phonetic and structural features. By incorporating three variant forms—pinyin, abbreviation, and character splitting—during recognition, the method can effectively identify Chinese sensitive words and their related variants.

1 Related Algorithms

Existing string recognition algorithms include similarity-based matching methods and exact pattern matching algorithms. Similarity-based matching using edit distance effectively represents the "cost" of transforming one string into another; smaller cost indicates higher similarity. The commonly used exact matching algorithm is KMP, which reduces string matching attempts compared to naive pattern matching.

1.1 Edit Distance

In 1965, Russian scientist Vladimir Levenshtein proposed the concept of edit distance [8], also known as Levenshtein distance, which refers to the minimum number of editing operations required to transform one string into another. Permitted operations include substituting one character for another, inserting a character, or deleting a character. Edit distance is frequently used to calculate the difference between two strings; smaller edit distance indicates greater similarity [9-10]. This paper uses edit distance to calculate similarity and employs normalization to map it to [0,1].

Let $edit(s, a)$ denote the edit distance between strings s and a, $length(s)$ denote the length of string s, and $max(length(s), length(a))$ denote the larger length between the two strings. The similarity calculation formula is:

$$Similar(s, a) = 1 - \frac{edit(length(s), length(a))}{max(length(s), length(a))}$$

1.2 KMP Algorithm

In 1969, Knuth, Morris, and Pratt proposed the fast single-pattern matching KMP algorithm [11]. Its main idea is: whenever a character mismatch occurs during matching, the pointer does not need to backtrack. Instead, it utilizes the "partial match" information already obtained to determine the next matching position. Let the target string be $T = [t_1, t_2, ..., t_n]$ with length n, and the pattern string be $P = [p_1, p_2, ..., p_m]$ with length m, where $n > m$. If there exists i such that $T[i+1] = P[1], T[i+2] = P[2], ..., T[i+m] = P[m]$, then matching succeeds and pattern string P appears in target string T at position i. If $T[i+1] \neq P[j+1]$, then i remains unchanged, j becomes $next(j)$, and the next round of matching continues between $T[i+1]$ and $P[next(j)+1]$. The $next$ array value represents the length of the longest suffix in $P[1..j-1]$ that equals its prefix. The $next$ array is defined as:

$$next(j) = \begin{cases} 0, & j = 1 \\ \max\{k | 0 < k < j \text{ and } p_1 ... p_k = p_{j-k} ... p_{j-1}\}, & \text{when such } k \text{ exists} \\ 1, & \text{otherwise} \end{cases}$$

In the next matching attempt, only the position of j needs to be determined, thereby improving pattern matching efficiency.

2 Sensitive Word Variants

This paper studies three patterns of sensitive word variants:

1. **Phonetic Pattern**: In Chinese, the same pinyin can correspond to multiple different characters. Due to regional dialect differences, some pinyin sounds are easily confused (e.g., distinguishing "l" and "n" in some regions).
2. **Abbreviation Pattern**: In daily life, words with many characters are often replaced by abbreviations. Statistics show that approximately 20% of sentences in Chinese news articles may contain abbreviations [12].
3. **Splitting Pattern**: As online platforms increasingly scrutinize information, users have begun splitting characters to evade review. For example, "林" can be split into "木木" [13].

Taking the word "贩卖毒品" (drug trafficking) as an example, its variant structures are shown in Figure 1 [Figure 1: see original paper].

3 Identification Algorithms for Sensitive Word Variants

3.1 Phonetic Pattern

Currently, many cases involve replacing characters in sensitive words with homophones or near-homophones. For example, "去死" (qusi, "go die") is a sensitive word. To evade platform review, malicious users often substitute "去屎" (qushi, "go shit"), which expresses the same meaning without being flagged. To identify phonetic variants, this paper converts sensitive words and suspected variants into sound codes and calculates their similarity using edit distance to determine if the word is a sensitive word variant.

3.1.1 Sound Code Encoding (SC)

Sound code is an encoding method for Chinese pinyin that can represent the phonetic characteristics of Chinese characters. Based on sound code, Chinese pinyin can be converted into corresponding character sequences, as shown in Figure 2 [Figure 2: see original paper].

This paper adopts a three-digit sound code format, using letters to encode each digit. In the three-digit sound code, the first digit represents the initial consonant position, the second represents the vowel position, and the third represents the tone position. According to the "Chinese Pinyin Scheme" approved and promulgated by the First Session of the Fifth National People's Congress in 1958 [14], there are 23 initial consonants and 34 vowels. For encoding convenience, tones are divided into level tone, rising tone, falling-rising tone, falling tone, and neutral tone. Partial encodings for initial consonants, vowels, and tones are shown in Tables 1 , 2 , and 3 .

Through this encoding, Chinese characters can be converted into character sequences for subsequent calculation and comparison. Taking "海洛因" as an example, the sound code obtained through the above encoding is "KDCHXDPTA".

3.1.2 Confusing Pinyin Grouping

Reference [15] proposed a phonetic string matching algorithm for matching similarly pronounced English strings. Based on this algorithm, this paper proposes grouping Chinese confusing pinyin and assigning each group a similarity factor (value range [0,1]; lower values indicate higher string similarity).

Confusing pinyin 主要分为 three types: flat tongue vs. curled tongue, lateral vs. nasal sounds, and front nasal vs. back nasal sounds. Table 5 shows partial data for confusing pinyin grouping. The grouping and similarity factors reference literature [12]. Each group's similarity factor represents the "cost" of replacing one pinyin with another within the same group. If two pinyin are identical, the cost is 0; if different and not in the same group, the cost is 1. Through research on Chinese pinyin, partial grouping is shown in Table 4 .

Taking "海洛因" and "海诺因" as examples, let θ be 90%. Their pinyin are "hailuoyin" and "hainuoyin", which convert to sound codes "KDCHXDPTA" and "KDCGXDPTA". Before applying confusing pinyin grouping, the similarity is 88.89%; after grouping, it becomes 94.44%. The similarity significantly improves and exceeds 90%, thus "海诺因" is identified as a phonetic variant of "海洛因".

3.1.3 Sensitive Pinyin Grouping Recognition Algorithm (SPGR)

To better identify phonetic variants, this paper proposes the Sensitive Pinyin Grouping Recognition (SPGR) algorithm based on an existing sensitive word list. The algorithm's execution process is described in Algorithm 1.

Algorithm 1. SPGR Algorithm
Input: Sensitive word S, suspected variant A
Output: Whether A is a phonetic variant of S

1. If $S.equals(A)$ is true, then S and A are identical strings; end.
2. If $S.equals(A)$ is false, obtain sound codes s and a for S and A according to the encoding tables, where $i = length(s)$ and $j = length(a)$.
3. Calculate the edit distance between sound codes using the method $edit(i, j)$.
4. Obtain the larger length between the two sound codes using $max(i, j)$.
5. Calculate similarity $Similar(s, a)$. If $Similar(s, a) > \theta$ (where θ is the threshold), then string A is a phonetic variant of S; end.
6. If $Similar(s, a) \leq \theta$, then string A is not a phonetic variant of S; algorithm ends.

3.2 Abbreviation Pattern

3.2.1 Word Abbreviation

Abbreviation patterns include initialism and word abbreviation. Initialism examples include "法轮功" abbreviated as "flg". Word abbreviations generally fall into three forms: compression, omission, and generalization [16], with the combination of compression and omission being most common. Compression divides the full term into several words and extracts the most representative characters from each, e.g., "贩卖毒品" becomes "贩毒". Omission directly removes some words, leaving others as the abbreviation, e.g., "复旦大学" becomes "复旦". Both compression and omission select and recombine characters from the full term, generally preserving character order. Since abbreviation characters are all contained in the full term, finding a subset of the full term reveals its abbreviation. For a Chinese string $S = s_{1s}2...s_n$ composed of n characters, if another string $A = a_{1a}2...a_m$ $(1 \leq m < n)$ exists where for any character $a_i \in A$ there exists $s_j \in S$ and as i increases, j also increases, then A is called an abbreviation of S.

3.2.2 Sensitive Word Abbreviation Recognition Algorithm (SNR)

To accurately identify sensitive word abbreviations, this paper proposes the Sensitive Name Recognition (SNR) algorithm. For initialism recognition, Pinyin4j is first used to obtain the first letters of sensitive words and suspected variants, then the first letters are compared as strings. For abbreviation recognition, each character of the suspected variant is matched sequentially within the sensitive word. The algorithm's execution process is described in Algorithm 2.

Algorithm 2. SNR Algorithm
Input: Sensitive word S, suspected variant A
Output: Whether A is an abbreviation variant of S

1. Obtain the first letters S' of string S. If $S' = A$, then A is an abbreviation of S; end.
2. Based on the ordered sets of strings S and suspected variant A, set indices i, j for S, A respectively, both initialized to 0.
3. If $S[i] = A[j]$, then i and j both increment by 1 and continue to step 4; otherwise, i increments by 1 and continue to step 4.
4. If $j = length(A)$, then matching succeeds; end. Otherwise, continue step 3.
5. If $i = length(S)$ and $j \neq length(A)$, then matching fails; end.

Taking "上海交通大学" and "交大" as examples: $S =$ (上海交通大学), $A =$ (交大). The S index moves right one position each time for comparison until $S[3] = A[0]$, then both S and A indices move right simultaneously. After $S[4] = A[1]$, the A index equals $length(A)$, the algorithm ends, and "交大" is identified as an abbreviation of "上海交通大学".

3.3 Splitting Pattern

3.3.1 Chinese Character Splitting

Chinese characters can be divided into single-component and compound characters based on their composition. Single-component characters (e.g., 日, 月)

consist of strokes, while compound characters (e.g., 休, 取) consist of radicals. Modern Chinese character splitting must recognize composition patterns and follow Chinese writing conventions. Spatial relationships include intersection, separation, and connection [17]. Positional relationships include top-bottom, left-right, inside-outside, frame, and single-component structures.

To provide a unified national code for each Chinese character, China promulgated the national standard "Coded Chinese Character Set for Information Interchange" [18]. The zone-position code is a four-digit decimal number where each code corresponds to a unique Chinese character or symbol [19]. Based on these character features, characters in the sensitive word list are manually split and encoded using zone-position codes to create a Chinese character splitting table, as shown in Table 5.

3.3.2 KMP-Based Sensitive Word Splitting Recognition Algorithm (WS-KMP)

To identify split variants of sensitive words, this paper proposes the Word Split KMP (WS-KMP) algorithm. First, sensitive word S and suspected variant A are split according to the character splitting table and converted to corresponding zone-position codes, then pattern matching is performed using the KMP algorithm. The algorithm's execution process is described in Algorithm 3.

Algorithm 3. WS-KMP Algorithm
Input: Sensitive word S, suspected variant A
Output: Whether A is a split variant contained in S

1. Split strings S and A according to the character splitting table to obtain their zone-position code sequences.
2. Set S as the target string and A as the pattern string, then perform KMP algorithm matching.
3. If matching succeeds, A is a split variant contained in S; end.
4. If matching fails, A is not a split variant contained in S; end.

Taking sensitive word "海洛因吗啡" and suspected variant "口马口非" as examples: after splitting, we get "氵 每氵 各口大口马口非" and "口马口非". The corresponding zone-position codes are S = (676335316763248731582083315834773158233 9) and A = (3158347731582339). Using the KMP algorithm for pattern matching, A is successfully matched as contained in S, confirming it as a split variant.

4 Experiments

Experiments were conducted on a machine with a 2.4GHz Intel(R) Core(TM) i5 processor and 8GB RAM, running Windows 10, with Java as the programming language.

4.1 Dataset

To evaluate the effectiveness of the proposed Chinese sensitive word variant identification method, 800 news texts containing suspected sensitive words were randomly extracted from the SogouCA (version: 2012) full-web news database

of Sogou Labs (http://www.sogou.com/labs/). The dataset covers technology, sports, finance, society, entertainment, and other topics. Sensitive words and their variants in the dataset were manually identified and classified, yielding 67 sensitive words and 400 variants covering phonetic, abbreviation, and splitting patterns. These identified sensitive words were stored in a sensitive word lexicon. In the experiments, the 67 sensitive words were first manually split, then the variant dataset was randomly divided into five groups for testing: 80, 160, 240, 320, and 400 variants. Partial examples of sensitive words and their variants are shown in Table 6 .

4.2 Experimental Analysis

The effectiveness of the variant identification algorithm was verified through three metrics: precision, recall, and runtime. For the phonetic pattern, θ was set to 91%. Recognition results were categorized into four cases: True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN). Recall = TP/(TP+FN), Precision = TP/(TP+FP). Experimental results are shown in Table 7 .

Table 7. Experimental Results

Dataset	TP	TN	FP	FN	Recall	Precision
Group 1	71	5	3	1	88.75%	92.22%
Group 2	135	11	8	6	84.38%	92.47%
Group 3	212	14	11	3	87.91%	91.73%
Group 4	285	20	10	5	89.06%	93.13%
Group 5	357	24	12	7	89.25%	93.70%

The experimental results show that the proposed algorithm achieves average recall and precision of 87.87% and 92.65%, respectively. For recall, the lowest rate was 84.38% in Group 2, likely because some sensitive words and their variants were complex, requiring further improvement of the identification rules. As data volume increased, recall stabilized around 89%. For precision, all five groups exceeded 91%, with the highest approaching 94%. Runtime increased proportionally with data volume. Overall, the proposed algorithm demonstrates satisfactory performance.

4.3 Experimental Conclusion

Experiments on different datasets reveal stable performance across recall, precision, and runtime metrics, proving the feasibility and effectiveness of the proposed algorithms.

5 Conclusion

This paper proposes a method that can identify sensitive words and their related variants through three Chinese character deformation forms: phonetic, abbreviation, and splitting. The method effectively improves the accuracy and

efficiency of sensitive word review. Experiments demonstrate that the proposed method can accurately identify various deformation forms of Chinese strings, with results approaching human recognition levels. However, character splitting requires manual operation, creating substantial workload when dealing with numerous sensitive words. Therefore, automatic Chinese character splitting is crucial and represents the next step in this research.

References

[1] Li Yang, Pan Quan, Yang Tao. Sensitive information identification based on short text sentiment analysis [J]. Journal of Xi'an Jiaotong University, 2016, 50(9): 80-84.

[2] Ye Lei, Zou Guoqi, Xiao Jian. Application of fuzzy genetic algorithm in sensitive word classification optimization [J]. Application Research of Computers, 2012, 29(7): 2549-2551.

[3] Xue Pengqiang, Nuerbuli, Wushouer·Silamu. Sensitive information filtering algorithm based on network text information [J]. Computer Engineering and Design, 2016, 37(9): 2447-2452.

[4] Yoon Taijin, Park Sunyoung, Cho H G. A smart filtering system for newly coined profanities by using approximate string alignment [C]// Proc of IEEE International Conference on Computer & Information Technology. 2010: 1035-1040.

[5] Sood S O, Antin J, Churchill E F. Using crowdsourcing to improve profanity detection [J]// Proc of AAAI Spring Symposium Series. 2012: 69-74.

[6] Ghauth K I, Sukhur M S. Text censoring system for filtering malicious content using approximate string matching and Bayesian filtering [C]// Computational Intelligence in Information Systems. Springer International Publishing, 2015: 331: 149-158.

[7] Li Shaoqing, Wu Chengrong, Zeng Jianping, et al. Lexical string similarity calculation for identifying variant keywords in harmful text [J]. Computer Applications and Software, 2015(3): 151-157.

[8] Levenshtein V I. Binary codes capable of correcting deletions, insertions and reversals [J]. Soviet Physics Doklady, 1966, 10(1): 707-710.

[9] Liu Baoyan, Lin Hongfei, Zhao Jing. Chinese sentence similarity computing based on improved edit-distance and dependency grammar [J]. Computer Applications & Software, 2008.

[10] Li Shengwen, Ling Wei, Gong Junfang, et al. A text similarity calculation method based on entropy [J]. Application Research of Computers, 2016, 33(3): 665-668.

[11] Knuth D, Morris J, Pratt V. Fast pattern matching in strings [J]. SIAM Journal on Computing, 1977, 6(2): 323-350.

[12] Chang Jingshin, Teng Weilun. Mining atomic Chinese abbreviation pairs: a probabilistic model for single character word recovery [J]. Language Resources and Evaluation, 2007, 40(3//4): 367–374.

[13] Li Dun, Cao Yuanda, Wan Yueliang. Identification of variant keywords in information security [J]. Computer Engineering, 2007, 33(21): 55-156.

[14] Chinese Language Reform Committee. Chinese Pinyin Scheme [S]. China: Chinese Language Reform Committee, 1958.

[15] Zobel J, Dart P. Phonetic string matching: Lessons from information retrieval [C]// Proc of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. 1996: 166-172.

[16] Yin Zhiping. Methods and principles for constructing abbreviations [J]. Language Teaching and Linguistic Studies, 1999(2): 73-80.

[17] Zhu Wenxuan. Automatic extraction technology for sensitive information in blog text content [D]. Shanghai: Shanghai Jiao Tong University, 2008.

[18] China State Bureau of Standards. Coded Chinese Character Set for Information Interchange [S]. China: China State Bureau of Standards, 1980.

[19] Yang Chao, Xie Jiangang. Chinese annotation of CNC programs based on zone-position code dictionary [J]. China Science and Technology Information, 2015(17): 65-66.

Note: Figure translations are in progress. See original paper for figures.