Identifying Abbreviation and Its Definition From Text String By Using LCS Algorithm

Khin Myo Han, Tin Tin Thein
Computer University (Mandalay)
khinmyohann@gmail.com

ABSTRACT

The understanding of abbreviations, widely used in writing is an important role for natural language processing (NLP) applications. Many abbreviations are followed a predictable pattern in which the first letter of each word in the definition corresponds to one letter in the abbreviation. Abbreviations detection is an important task of the dictionary building with their definitions. In this system, abbreviation and their definitions are implemented by using Longest Common Subsequence (LCS) algorithm. Additionally, this system is provided a method, pattern based rule together with the LCS algorithm. Thus, the simplicity of how abbreviations are formed from their definition is also described in this system. As a result, the extracted abbreviations are created into a large update database, a dictionary about computer literature. This system is implemented by C# programming language.

1. INTRODUCTION

Many organizations have a large number of on-line documents such as manual, technical reports, transcriptions of customer service call or telephone conferences and electronic mail which contain information of great potential value. To work on automatic glossary extraction, technical documents contain a lot of abbreviation terms, which carry important knowledge about the domains. Every content domain has its own acronyms and abbreviations. It is needed to be able to create common glossaries of domain-specific names and terms in order to utilize the knowledge on these data contain. Many acronym and abbreviation dictionaries are available, both in printed from and on the World Wide Web.

Today, rapid growth of the computer literature presents special challenges for both human and automatic algorithms. Understanding computer literature is particularly challenging due to its expanding vocabulary, including the unfettered introduction of new abbreviations. Abbreviations must be significantly shorter than their words and understanding of these abbreviations in a document is a difficult task for computer systems [6, 1].

Thus, the abbreviation problem has been shown to affect knowledge-base systems. A method to associate an abbreviation to its corresponding expansion is firstly needed as less well-known sense of abbreviation is not always defined in document. Consequently, an abbreviation database is needed to be built and updated with their sense periodically.

This paper is organized as follow. Section 2 presents related work of identifying abbreviation and section 3 also describes the methods and system detail design explanation. Section 4 explains the applying algorithm: Three Letter Acronym (TLA) and Acronym Finding Program (AFP). Finally section 5 concludes system implementation result.

2. RELATED WORK

Taghva and Gilbreth presented a solution for identifying abbreviation based on hand-built regular expression. Their paper introduced an automatic method for finding acronyms and definition in free text. They used method that based on an exact pattern matching algorithm applied to text surrounding the possible acronym. Their algorithm

use dynamic programming to find potential alignments between short and long form, and used the results of this to compute feature vector for correctly identified definitions [2].

Yeates examine acronyms in technical text. The author used a technique for extracting acronym is given with an analysis of the results. The author's technique was found to have a low number of false negative and high numbers of false positive. The author implemented the technique was called the Three Letter Acronym (TLA) [3].

Dana Dannells applied a rule-based method to solve the acronym recognition task and compares and evaluates the results of different machine learning algorithms on the same task. The author's method proposed was based on the approach that acronym definition pairs follow a set of patterns and other regularities that can be usefully applied for the acronym identification task. The rule-based algorithms utilizes predefined strong constraints to find and extract acronym-definition pairs with different patterns, it has the advantage of recognizing acronyms and definitions which are not indicated by parentheses [5].

Schwartz and Hearst presented the problem of identifying abbreviation's definitions can be solved with a much simpler algorithm than that proposed by other research efforts. They introduced a new algorithm for extracting abbreviation and their definitions from biomedical text [4].

3. THEORICAL BACKGROUND

First of all, Figure 1 illustrates the architecture of the proposed system introduce the whole processing of this system. Firstly, each user has a chance to enter a text string .After entering the text string; the tokenizer tokenized the input string into word by word. Abbreviation recognizer determines the letter within the parentheses whether it is a candidate abbreviation or not. Candidate acronyms were determined by matching the initial letter of each word in the context of a potential acronym against the appropriate letter. All non-alphabetic characters were converted to space and any multiple spaces replaced with a single space. Then the LCS algorithm identifies a common subsequence of the letters of abbreviation and a probable definition by using match pattern and extraction of abbreviation pattern from database. AFP provides the array creation and pattern matching is completed by TLA method. Finally, this system chooses the best definition of abbreviation for user.

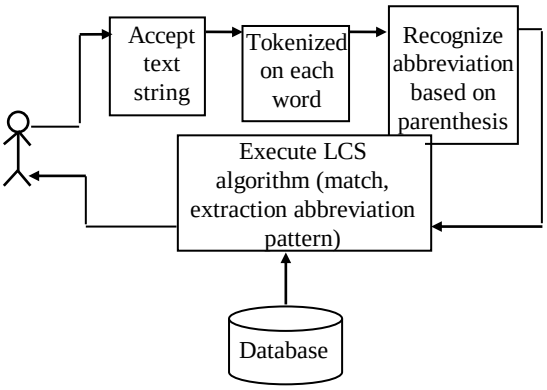

Figure 1. System Architecture

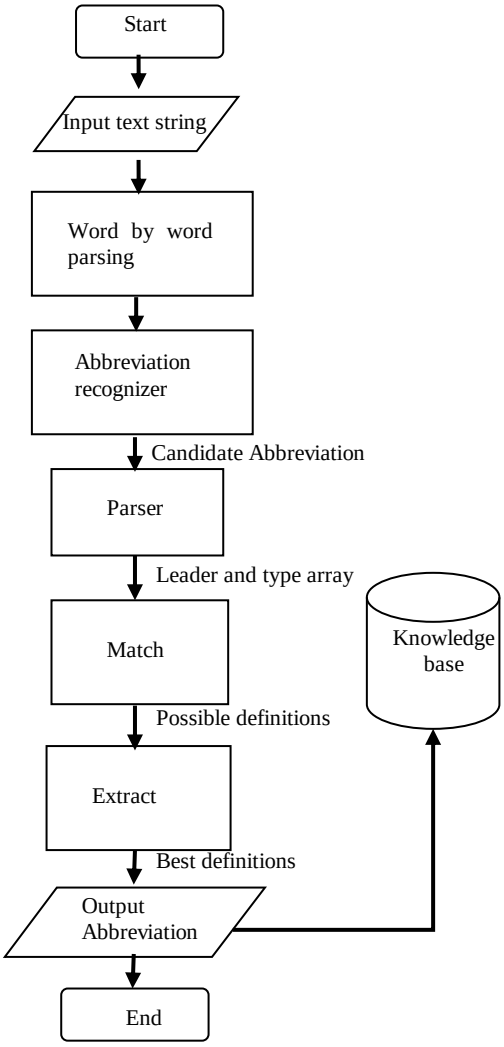

Figure 2. Overview of System Flow

The detailed flow of the system is shown in figure[2]. In longest Common Subsequence (LCS) algorithm component, two implemented task such as match and extract are responsible for matching on four patterns. In order to apply the proposed algorithm, users need two characters arrays. One is a leader array which contains the letters of abbreviation. The proposed system is developed by using these two arrays. The system stores the abbreviation and its corresponding definition in the Abbreviation Dictionary in order to meet the user's desire.

There are several types of abbreviation methods to identify abbreviation in the input string in order to understand the string. There are two methods and pattern based rule to extract abbreviations from the input string and define their definition.

3.1. Identifying Abbreviation and Definition Candidate

In this system, the process of extracting abbreviations and their definitions from input text string consists of three main tasks. They are scanning, extraction and identifying. In scanning, the input is scanned for occurrences of possible abbreviation. In extraction, the pair candidate (definition, abbreviation) are extracted from the text. In identifying, the correct definition is identified from the candidates in the sentence that surrounds the abbreviation.

Abbreviation candidates are deter-mined by adjacency to parentheses. The two cases are: definition (abbreviation) and abbreviation (definition) [5]. In this system, definition (abbreviation) type is used to support the candidate determination.

3.2. Finding Abbreviation-Definitions Candidate

The abbreviation can be defined within text in many possible ways. The definition of the abbreviation may be before the left parenthesis or after the left parenthesis. The definition candidate must appear in the same sentence as the abbreviation. The size of the text window is a function of a length of the abbreviation and maximally allowed distance between a definition and its abbreviation. The maximum length of a definition D of an abbreviation A is calculated as in Equation 1.

$$\max |D| = \min \{|A| + 5, |A| * 2\} \quad (1)$$

For short abbreviations (from two to four characters), the length of a definition in word should not be greater than twice the abbreviation length. For long abbreviation (five or more characters), the definition should not be longer than the abbreviation length plus 5.

The maximum length means the longest distance of a definition form an abbreviation. If a definition is in the left context, the distance is the number of word from the abbreviation to the first word of the definition [5, 7].

3.3. Abbreviation Patterns and Definition Patterns

Abbreviation recognizer first decides the prototype of abbreviation and then recognizes and stores it into a character array. Moreover, abbreviation patterns are generated for a candidate abbreviation.

An abbreviation patterns is a string of ‘c’ and ‘n’ character. An alphabetic character is replaced with a ‘c’ and a sequence of numeric characters is replaced with an ‘n’ regardless of its length [1, 7]. Some of candidate abbreviations and their pattern are shown in table 1. The string consists of non-alphabetic, numeric and special characters such as -, / are not considered in abbreviation patterns.

Each character (w, s) is used to denote the normal word and stopword in which the parsed words are included [1, 5, 7]. Some of definition and definition patterns are shown in table 2.

Table 1. Abbreviation Patterns

Abbreviations	Patterns
ICS	ccc
2GB	ncc
U5M	cnc
MP4	ccn
UCSM	cccc

Table 2. Definition patterns

Definitions	Patterns
Introduction to Computer System	wsww
Business to Business	wsw
Artificial Intelligence	ww
Operating System	ww
Natural Language Processing	www

4. IMPLEMENTATION OF THE SYSTEM

In this section, the implementation of Longest Common Subsequence (LCS) algorithm and the workflow of its components named Three Letter Acronym (TLA) extractor and Acronym Finding Program (AFP) is also demonstrated in simple.

4.1. Three Letter Acronym (TLA) Extractor

Candidate acronyms and their definitions were selected from a stream of words by using analyzer. All non-alphabetic characters were converted to space and any multiple spaces replaced with a single space. Candidate acronyms were determined by matching the initial letter of each word in the context of a potential acronym against the appropriate letter by using heuristic checker. In smoother component, each candidate is examined to define the method of abbreviation using LCS. In this system, it can be defined to determine what attribute and what combinations of attributes were important ones for making the decision by using TLA method [3].

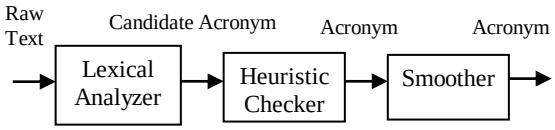

Figure 3. The Acronym Extractor TLA

4.2. Acronym Finder

Acronym Finder fined the acronym. Acronym Finding Program (AFP) which is designed specifically for finding acronyms and improving post-processing of text captured using Optical Character Recognition (OCR). Acronym candidate is typically defined upper-case words from three to ten characters in this system.

Acronym candidates are tested against a list of "reject words" that commonly appeared in upper-case, such as TABLE, FIGURE, and roman numerals. If each candidate passed this test, AFP constructed text window surrounding the acronym which was search for its definitions.

In both cases the numbers of words in the window are twice the number of characters in the acronym candidate. After dividing the sub window they were parsed and generated two symbolic arrays for the window. The leader array consisted of the first letter of each candidate word, and the type array consisted of the type of each word in the sub window. And the array of initial letters of these words was matched against the acronym itself using LCS [3].

In AFP, the first letter of each word is only considered for finding acronym. In TLA, each letter of word is considered for converting non-alphabetic character to space.

4.3. Longest-Common-Subsequence (LCS) Algorithm

Longest-Common Subsequence (LCS), a string matching algorithm, is the most useful algorithm to find abbreviations and acronyms. String matching algorithm (LCS) presented an algorithm that generated a set of paths through the window of the text adjacent to an abbreviation (string from the leftmost character) and scored these paths to find the most likely definition. The algorithm identifies a common subsequence of the letters of abbreviation and the leader array to find a probable definition. A subsequence of a given sequence is just the given sequence with some elements removed. The longest common subsequence LCS of any two strings X and Y is a common subsequence with the maximum length among all common subsequences [2].

Suppose that the system has achieved a sentence "A primary program, which include nature of a National Waste Terminal (NWT) program". The pre-window of the NWT is "nature of a National Waste Terminal ", which is generated by TLA and maximum length of definition and abbreviation. Then, the leader array [n o a N W T] and the relative type array [w s s w w w] are generated by a method called AFP. Consequently, the built-LCS-matrix is created over leader array and abbreviation in the sentence as in Figure 3.

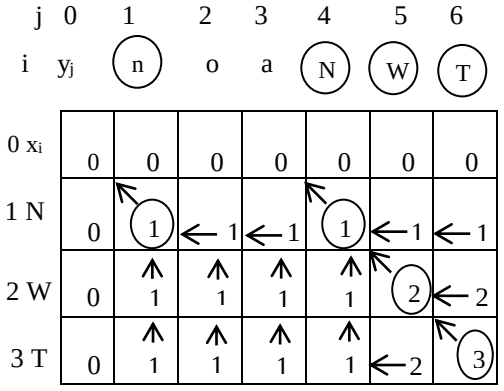

Figure 4. The c and b matrices computed by built and parse LCS-matrix on X=NWT and Y= noaNWT.

The algorithm calculates that when either X or Y are empty sequences, then the LCS is an empty string and *c [i,j]=0*. The matrix *c* shows the length of an LCS for string X and Y and store this value in *c [m,n]*. In the matrix *c, ' m'* represents the number of abbreviation and *' n '* also represents the number of the leader array.

A " ↖ " entry in *b[i,j]* assert that *X [i]= Y [j*], and *c[i-1, j-1] +1* is the selected according to equation 1. A"↑" or " ← " in *b[i, j]* assert that X *[i]≠ Y[j]* and *c[i-1, j]* or *c[i,j-1]* is also selected according to the equation 1,respectively.

In figure 3, if x[i]=y[j] then "↖", else "↑" or " ←".Then parse LCS matrix generate two vectors.

Two alternative vectors [100023] and [000123] are built by parse-LCS-matrix. These two vectors are derived from the resulting Parse-LCS-matrix: (1,1) (2,5) (3,6) and (1,4) (2,5) (3,6).

The best result defining abbreviation is chosen by comparing the vector over four elements as shown in Table.3. The following four elements for each vector are shown as follow:

misses: The number of zeros entries in the vector; disregarding leading zero, and those zero entries corresponding to word type of 's '.

stopcount: The number of stopword that will be used in the abbreviation definition if the vector is selected.

distance: The index of the last non-zero entry. This value measure the proximately of the definition to the actual definition.

size: The number of entries in the vector after removing leading and trailing zeros. The value represents the length of the definition in words.

Thus, the possibility of abbreviation is achieved by vector 2 because of the greater value of miss in vector 1[2].

Table 3. Comparison of the resulting vectors

	Vector 1	Vector 2
Misses	1	0
Stopcount	0	0
Distance	0	0
Size	5	3

As a result, the abbreviation NWT is recognized as a pair of abbreviation description with "National Waste Terminal".

5. CONCLUSION

Natural Language Processing is a broad application area in Artificial Intelligence. In this paper, the algorithm for extracting abbreviations and their definitions from input text is expressed. Moreover, pattern-matching rules are developed to map abbreviation to its full form and implement the rule into a software program. The rule is generated from pattern-based rules and Longest Common Subsequence (LCS). This paper shows that the developed system effective for pairing abbreviations with full forms when the abbreviations are defined in the text for building dictionary. Non-alphanumeric characters such as hyphen, slash, and ampersand are not considered in abbreviation patterns of this system.

REFERENCES

[1] D.Dannells"Automatic Acronym Recognition" Computational Linguistics, Department of Linguistics and Department of Swedish Language Goteborg University Getborg, Sweden.

[2] k. Taghva, and J. Gilbreth, "Recognizing Acronyms and their definition",pp.1-6 International Document Analysis and Recognition Springer-Verlag 1999 Information Science Research Institute, University of Nevada, Las Vegas, Received October 1, 1997 / Revised September 8, 1998.

[3] MS. L. Hongfang, Alan R. Aronson , Carol Friedman "A Study of Abbreviations in MEDLINE Abstracts", pp.1-3, Graduate School and University Center of CUNY National Library of Medicine Computer Science Department, Queens College of CUNY Department of Medical Informatics, Columbia University.

[4] N.L. Aye, "Pattern Based Extraction of Abbreviations and Their Definitions", May 2004, UCSY.

[5] S. Schwartz, and M. A. Hearst, "A simple algorithm for identifying abbreviation definitions in biomedical text", Proceedings of the Pacific Symposium on Biocomputing 2003.

[6] S. Yeates, "Automaic Extraction of acronym from text", in Proceedings of the Third NewZeland Computer Science Research Students Conference. Hamilton, New Zeland, April 1999, University of Waikato.

[7] X Wang, “Report for Abbreviation Recognition-on Project”, January 26, 2005.