BIOINFORMATICS ORIGINAL PAPER Vol. 22 no. 24 2006, pages 3089–3095
doi:10.1093/bioinformatics/btl534

Data and text mining

Building an abbreviation dictionary using a term recognition approach

Naoaki Okazaki[1,2,*] and Sophia Ananiadou[3,4]

[1]Graduate School of Information Science and Technology, The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8651, Japan, [2]Japan Society for the Promotion of Science (JSPS), 8 Ichiban-cho, Chiyoda-ku, Tokyo, Japan, [3]School of Computer Science, The University of Manchester and [4]National Centre for Text Mining (NaCTeM), Manchester Interdisciplinary Biocentre, Oxford Road, Manchester, M13 9PL, UK

Received on July 1, 2006; revised on October 10, 2006; accepted on October 12, 2006
Advance Access publication October 18, 2006
Associate Editor: Golan Yona

ABSTRACT

Motivation: Acronyms result from a highly productive type of term variation and trigger the need for an acronym dictionary to establish associations between acronyms and their expanded forms.

Results: We propose a novel method for recognizing acronym definitions in a text collection. Assuming a word sequence co-occurring frequently with a parenthetical expression to be a potential expanded form, our method identifies acronym definitions in a similar manner to the statistical term recognition task. Applied to the whole MEDLINE (7 811 582 abstracts), the implemented system extracted 886 755 acronym candidates and recognized 300 954 expanded forms in reasonable time. Our method outperformed base-line systems, achieving 99% precision and 82–95% recall on our evaluation corpus that roughly emulates the whole MEDLINE.

Availability and Supplementary information: The implementations and supplementary information are available at our web site: http://www.chokkan.org/research/acromine/

Contact: okazaki@mi.ci.i.u-tokyo.ac.jp

1 INTRODUCTION

Acronyms result from a highly productive type of term variation, which substitutes fully expanded terms (e.g. *retinoic acid receptor alpha*) with shortened term-forms (e.g. *RARA*). Chang *et al*. (2006) reported that 64 242 new acronyms were introduced in 2004 in MEDLINE abstracts. Terminological resources and scientific databases, (such as UMLS[1], Swiss-Prot[2], SGD[3], FlyBase[4] and UniProt[5]) cannot keep up-to-date with the growth of neologisms (Pustejovsky *et al*., 2001). In practice, no generic rules or exact patterns have been established for dealing with acronym creation.

Gaudan *et al*. (2005) distinguished *global* acronyms from *local* acronyms based on the presence of their definitions in texts. Global acronyms appear in documents without the expanded form explicitly stated, while local acronyms accompany their expanded forms in the document. Global acronyms hinder text-mining tasks, such as information retrieval and information extraction. Wren *et al*. (2005) reported that PubMed could retrieve 5477 documents for *JNK* but only 3773 documents for its full term, *c-jun N-terminal kinase*.

Thus, an acronym dictionary is necessary for advanced text-mining tasks to establish associations between acronyms and their expanded forms. Adar (2004) noted that previous work had mostly found acronym definitions within a text in a similar manner to information extraction. He saw the need for additional tasks for a practical acronym resource, such as merging similar definitions and providing disambiguation information. Although we find such components indispensable for text-mining applications, here we focus on the task of finding acronym definitions, as the first step in building an accurate acronym dictionary.

Another important aspect for building an acronym dictionary is the distinction between *dynamic* and *common* acronyms (Yu *et al*., 2002). Dynamic acronyms are one-time substitutions valid within a document and therefore always local. In contrast, common acronyms are used over two or more publications, and may appear in documents with or without their expanded forms. An acronym dictionary should focus on common acronyms since they are potential global acronyms, i.e. might be written without their definitions in some documents. In this paper we do not deal with the identification of dynamic acronyms, which can be recognized by letter matching techniques. We collect definitions of local and common acronyms in source documents.

One of the main challenges of text-mining is dealing with an enormous amount of documents in a scalable and efficient manner. At the same time, we can also utilize the amount of textual data to obtain accurate and comprehensive results. We present a methodology for building a good quality acronym dictionary of common acronyms and their expanded forms, making effective use of large amount of texts. The method proposed in this paper was applied to the whole MEDLINE (7 811 582 abstracts). It extracted 886 755 candidates of acronyms and recognized 300 954 expanded forms. Detailed evaluation results and their comparison with existing methods are also given in the paper.

2 RELATED WORK

Acronym recognition aims to extract pairs of short-forms (acronyms or abbreviations) and long-forms (their expanded forms or

[*]To whom correspondence should be addressed.
[1]http://www.nlm.nih.gov/research/umls/
[2]http://www.ebi.ac.uk/swissprot/
[3]http://www.yeastgenome.org/
[4]http://www.flybase.org/
[5]http://www.ebi.ac.uk/GOA/

definitions) occurring in text. Most studies share pattern (1) to locate a textual fragment with an acronym and its expanded form (Schwartz *et al*., 2003; Wren *et al*., 2002).

$$\text{long form '('short form')'} \qquad (1)$$

For example, the sentence, 'The exact *route was determined by magnetic resonance imaging* (*MRI*)', could yield the textual fragment marked with the italic letters[6]. The task is to identify the 'authentic' long-form in the textual fragment if any. Existing methods for solving this problem can be categorized into three groups: using heuristics and/or scoring rules (Adar, 2004; Ao *et al*., 2005; Schwartz *et al*., 2003; Taghva *et al*., 1999; Wren *et al*., 2002; Yu *et al*., 2002); machine learning (Chang *et al*., 2006; Pakhomov, 2002; Nadeau *et al*., 2005); and statistics (Hisamitsu *et al*., 2001; Liu *et al*., 2003).

The first category uses predefined heuristic rules/algorithms to find a long-form in a textual fragment. For example, Schwartz *et al*. (2003) implemented a letter matching algorithm that maps all alpha-numerical letters in the short-form to the long-form, starting from the end of both the short and long-forms and moving right to left. Even though the core algorithm is very simple, the authors report 96% precision and 82% recall on the Medstract gold standard[7]. Adar (2004) proposes scoring rules to find the most likely long-form, accepting multiple long-form candidates, e.g. determined by magnetic resonance imaging and magnetic resonance imaging in the fragment, yielding 95% precision and 85% recall on the Medstract corpus.

The second category obtains such rules by using a machine learning technique. Chang *et al*. (2006) applied a logistic regression to calculate the likelihood of long-form candidates. They enumerate possible long-form candidates with longest common substring (LCS) formalization (Taghva *et al*., 1999). The likelihood of the candidates is estimated as the probability calculated from a logistic regression with nine features, such as the percentage of long-form letters aligned at the beginning of a word, the percentage of short-form letters aligned to the long-form, etc. Their method achieved 80% precision and 83% recall on the Medstract corpus.

The third category, where our proposed method belongs, utilizes statistical clues in the source documents, e.g. co-occurrence between short-forms and long-forms. Hisamitsu *et al*. (2001) proposed a method for extracting useful parenthetical expressions from Japanese newspaper articles. Their method measures the co-occurrence strength between the inner and outer phrases of a parenthetical expression via mutual information, χ^2-test with Yate's correction, Dice coefficient, log-likelihood ratio, etc. Unfortunately, their method deals with generic parenthetical expressions (i.e. abbreviation, non-abbreviation paraphrases, supplementary comments), not focusing exclusively on acronym recognition. Liu *et al*. (2003) based their method on collocations occurring before the parenthetical expressions. Enumerating long-form candidates as collocations appearing more than once in a text collection, their method eliminates unlikely candidates with rules, such as ''remove a set of candidates T_w formed by adding a prefix word to a candidate w if the number of such candidates T_w is greater than 3''. They report a precision of 96.3% and a recall of 88.5% for abbreviation recognition on their test corpus.

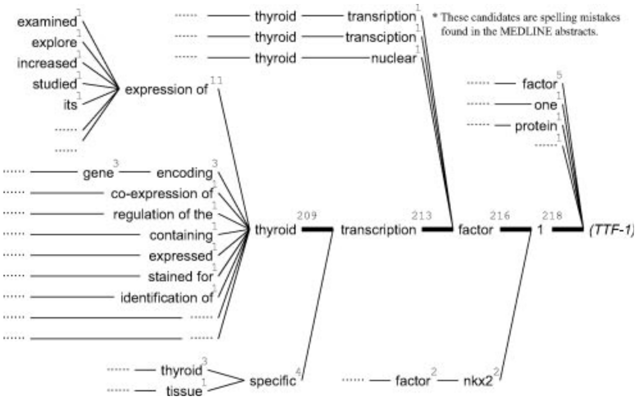

Fig. 1. Expressions appearing before the acronym TTF-1 in parentheses.

3 METHODOLOGY

3.1 Recognizing acronyms based on co-occurrence

We assume a word sequence is a possible long-form[8] if the word sequence co-occurs frequently with a specific acronym and not with other surrounding words. Figure 1 illustrates our assumption with the acronym *TTF-1*. The tree consists of expressions collected from all sentences with the acronym *TTF-1* in parentheses and appearing before the acronym. A node represents a word, and a path from any node to *TTF-1* represents a long-form candidate[9]. The figure above each node shows the co-occurrence frequency of the corresponding long-form candidate. For example, long-form candidates 1, *factor 1*, *transcription factor 1* and *thyroid transcription factor 1* co-occur 218, 216, 213 and 209 times, respectively with the acronym *TTF-1* in the text collection.

Even though long-form candidates 1, *factor 1* and *transcription factor 1* co-occur frequently with *TTF-1*, they also co-occur frequently with *thyroid*. Meanwhile, the candidate *thyroid transcription factor 1* is used in a number of contexts (e.g. *expression of thyroid transcription factor 1*, *expressed thyroid transcription factor 1*, etc.). Therefore, we observe the strongest relationship is between acronym *TTF-1* and its long-form candidate *thyroid transcription factor 1* in the tree. We apply a validation rule (described later) to the long-form candidate to make sure an acronym-definition relation does occur. In this example, the candidate pair is likely to be in an acronym-definition relation as the long-form *thyroid transcription factor 1* contains all the alphanumeric letters in the short-form *TTF-1*.

This approach detects the starting point of the long-form without using letter matching. A simple method based on letter matching may misrecognize the long-form *transcription factor 1* since it also contains the necessary elements to produce the acronym *TTF-1*. Whereas previous work dealt with this case by introducing, e.g. a set of complicated rules, scoring or machine learning techniques, our approach uses overlapping definitions of an acronym stated by a number of authors. This characteristic of our approach also contributes to finding a long-form whose short-form is arranged in a different word order, such as *beta 2 adrenergic receptor* (*ADRB2*) and *water activity* (*AW*).

[6] Assuming we take $(l + 4)$ words appearing before the parenthetical expression (Adar, 2004), where l is the number of letters in the short-form.

[7] http://www.medstract.org/

[8] A sequence of words that co-occurs with an acronym does not always imply the acronym-definition relation: the acronym *5-HT* co-occurs frequently with the term *serotonin*, but their relation is interpreted as a synonymous relation. We deal with this issue with a validation rule later.

[9] The words with function words (e.g. *expression of*, *regulation of the*, etc.) are merged into a node. This is due to the requirement for a long-form candidate discussed later (Section 3.2).

3.2 Formalizing long-form recognition as a term extraction problem

Having collected all sentences with a specific acronym (hereafter *contextual sentences*), we deal with the problem of extracting long-form candidates from the contextual sentences in a similar manner to the term recognition task, which extracts terms from a given text. For this purpose, we modified the C-value method (Frantzi *et al*., 1999), a domain-independent method for automatic term recognition (ATR). The *C*-value approach is characterized by the extraction of nested terms that gives preference to terms appearing frequently in a given text but not as a part of specific longer terms.

Given a contextual sentence, we tokenize it by non-alphanumeric characters (e.g. space, hyphen, colon) and apply a stemming algorithm (Porter *et al*., 1980) to obtain a sequence of normalized words. Pattern (2)[10] extracts long-form candidates from the sequence:

$$[\mathrm{:WORD:}].*\$. \tag{2}$$

The extraction pattern accepts a word or word sequence if it begins with any non-function word[11], and ends with any word just before the corresponding short-form in the contextual sentence.

Consider the example of a contextual sentence, ''we studied the expression of thyroid transcription factor 1''. We extract the following substrings as long-form candidates (words are stemmed): *1*; *factor 1*; *transcript factor 1*; *thyroid transcript factor 1*; *expression of thyroid transcript factor 1* and *studi the expression of thyroid transcript factor 1*. The list of function words is not used for removing specific words in long-form candidates (e.g. *expression of thyroid transcript factor 1* contains a function word *of*), but for preventing invalid candidates beginning with a function word, such as *of thyroid transcript factor 1*.

The original C-value method assigns termhood (likelihood to be a term) to a candidate term,

$$\mathrm{CV}(c) = \log[\mathrm{len}(c)] \cdot \mathrm{freq}(c) - \frac{\sum_{t \in T_c} \mathrm{freq}(t)}{|\, T_c \,|}. \tag{3}$$

In formula 3, c is a candidate term; freq(c) denotes the frequency of occurrence of term c; len(c) denotes the length (number of words) of term c; T_c is a set of candidate terms which contain term c; $t \in T_c$ is a candidate term which contains term c and $|\, T_c \,|$ represents the number of such candidate terms T_c. Multiplying log[len(c)] with freq(c) is based on the consideration that a longer string appears less frequently than a shorter string (Frantzi *et al*., 1999). However, longer terms are not useful as long-forms, as the previous work excluded candidates longer than the maximum length estimated by the number of letters in a short-form (Park *et al*., 2001). In addition, formula 3 always yields zero for a one-word candidate.

Formula 4 amends the original formula of C-value (formula 3) to define the long-form likelihood LH(c) for a candidate c:

$$\mathrm{LH}(c) = \mathrm{freq}(c) - \sum_{t \in T_c} \mathrm{freq}(t) \times \frac{\mathrm{freq}(t)}{\sum_{t \in T_c} \mathrm{freq}(t)}. \tag{4}$$

In formula 4, c is a long-form candidate; freq(c) denotes the frequency of occurrence of a candidate c in the contextual sentences (i.e. co-occurrence frequency with a short-form); and T_c is a set of nested long-form candidates, each of which consists of a preceding word followed by the candidate c.

The first term of the formula is equivalent to the co-occurrence frequency of a long-form candidate with a short-form. The second term discounts the first term based on the frequency distribution of nested candidates. Given a long-form candidate $t \in T_c$, $[\mathrm{freq}(t)/\sum_{t \in T_c} \mathrm{freq}(t)]$ presents the occurrence probability of candidate t in the nested candidate set T_c[12]. Therefore, the second term of the formula calculates the weighted average of the frequency of occurrence of nested candidates accounting for the frequency of candidate c. The underlying idea of the subtraction is to disregard the candidate as a part of specific longer candidates. If a long-form candidate c often occurs selectively as a part of a nested candidate $t \in T_c$, $LH(c) \rightarrow 0$ as the second term of the formula becomes close to the first term. If a long-form candidate c does not occur as part of a nested candidate, $LH(c) \rightarrow \mathrm{freq}(c)$ as the second term becomes close to zero.

Table 1. Long-form candidates for ADM

Candidate	Len	Freq	Score	Valid
adriamycin	1	727	721.4	o
adrenomedullin	1	247	241.7	o
abductor digiti minimi	3	78	74.9	o
doxorubicin	1	56	54.6	x (missing letters)
effect of adriamycin	3	25	23.6	x (expansion)
adrenodemedullated	1	19	17.7	o
acellular dermal matrix	3	17	15.9	o
peptide adrenomedullin	2	17	15.1	x (expansion)
effects of adrenomedullin	3	15	13.2	x (expansion)
resistance to adriamycin	3	15	13.2	x (expansion)
amyopathic dermatomyositis	2	14	12.8	o
brevis and abductor digiti minimi	5	11	9.8	x (expansion)
minimi	1	83	5.8	x (nested)
digiti minimi	2	80	3.9	x (nested)

3.3 Extracting authentic long-forms for acronyms

Even if the long-form likelihood LH(c) assigns higher scores to a long-form candidate c occurring frequently with a specific acronym, this does not assert that the candidate c is the long-form for an acronym. Table 1 shows a list of long-form candidates for acronym *ADM* in descending order of their likelihood scores. Candidate *adriamycin* co-occurs the most frequently with acronym *ADM*. Since the long-form candidate *adriamycin* contains all letters in the same order as the acronym *ADM*, it is considered as an authentic long-form (marked as 'o'). This is also true for the second and third candidates (*adrenomedullin* and *abductor digiti minimi*).

The fourth candidate *doxorubicin* is interesting, i.e. its score is high although it lacks the necessary letters *a* and *m* for *ADM*. This is because *doxorubicin* is a synonym of *adriamcycin*, and many authors give *ADM* in parentheses following the word without the proper long-form (*adriamcycin*). In this case, although the strong co-occurrence between *doxorubicin* and *ADM* implies a meaningful relation, we do not extract such pairs, counting them as invalid (not a proper pair of short/long-form).

Most studies (e.g. Adar, 2004, Schwartz *et al*., 2003; Wren *et al*., 2002) introduce a rule to validate a long-form for a short-form: ''all (alphanumeric) letters in a short-form must appear in the corresponding long-form in the same order''. However, one advantage of our approach over the previous work based on letter matching is that it can suggest, based on statistics, a long-form whose short-form is arranged in a different word order, e.g. *water activity (AW)* and *beta 2 adrenergic receptor* (*ADRB2*). Hence, we accept a long-form candidate if the words in the long-form candidate can be rearranged so that all alphanumeric letters in the short-form appear in the

[10] [:WORD:] matches a non-function word; .* matches an empty string or any word(s) of any length; and $ matches a short-form of the target acronym.

[11] Twenty-nine function words are held in an external dictionary: three articles (*a*, *an*, *the*); two conjunctions (*and*, *or*); seventeen prepositions (*of*, *to*, *in*, etc.); seven forms of the verb *be*.

[12] Note that $\sum_{t \in T_c}{}^{\mathrm{freq}(t)}$ is not equal to ${}^{\mathrm{freq}(c)}$ only if any contextual sentence beginning with the long-form c exists.

rearranged long-form candidate in the same order. For example, the long-form candidate *ADRB2* is recognized as a valid expression since the words in the candidate are rearranged as *ADRB2*. In contrast, the long-form candidate *rate* for acronym *ER* is rejected because the letters ‘e’ and ‘r’ appear in the same word so that changing the word order cannot resolve the order discrepancy between the short-form and long-form.

We call the fifth candidate *effect of adriamycin* an expansion of a long-form since it consists of the authentic long-form *adriamycin* with some preceding words (i.e. *effect of*). As *adriamycin* has a higher score than this candidate, we can disregard the expansion candidates, such as *effect of adriamycin* and *resistance to adriamycin* (marked as ‘expansion’) because they contain unnecessary elements (i.e. *effect of* and *resistance to*) attached to the long-form. Similarly, we also disregard nested candidates, such as *minimi* and *digiti minimi* (marked as ‘nested’) since they lack the necessary elements (i.e. *abductor digiti* and *abductor*) to create the correct long-form *abductor digiti minimi*.

To summarize the long-form extraction algorithm described above, a long-form candidate is considered valid if the following conditions are met: (1) it has a likelihood score ≥ 2.0 (i.e. a long-form candidate must appear at least twice); (2) the words in the long-form can be rearranged so that all alphanumeric letters in the short-form appear in the same order and (3) it is not nested or an expansion of the previously chosen long-forms.

3.4 Implementation

The implemented system first enumerates all short-forms in a given text which are likely to be acronyms by focusing on parenthetical expressions [see Pattern (1)]. Following the heuristic rules (Schwartz *et al*., 2003), we regard parenthetical expressions as short-forms if they consist of at most two words; their length is between 2 to 10 characters; they contain at least an alphabetic letter; and the first character is alphanumeric. All sentences containing a short-form are associated with their short-forms in a database for efficient access by later processes. For each short-form in the database, the system retrieves all contextual sentences for that short-form and generates a list of long-form candidates and their likelihood scores. The algorithm described in Section 3.3 determines the authentic long-forms in the list. Iterating this process for all short-forms, the system yields the list of acronyms and their expanded forms.

Using a desktop computer running on an Intel Pentium 4 3.40 GHz processor with 2 GB main memory, we conducted a feasibility experiment, applying the system to the whole MEDLINE database, which contained 7 811 582 abstracts (out of 16 069 250 citations)[13]. It took about 12 h to recognize 886 755 unique short-forms in the abstracts and to insert 9 223 039 contextual sentences into the intermediate database. The short-form occurring the most frequently in the abstracts was *II* (50 923 times), followed by *CT* (32 507 times), *III* (30 184 times), $P < 0.05$ (27 284 times), *PCR* (26 486 times), etc. Some of the candidates, such as III and $P < 0.05$ are not real short-forms even though they often appear in parentheses in scientific articles. We do not provide any processing stage in short-form mining to exclude them since they are unlikely to be accompanied by specific long-form candidates and, therefore, to be qualified in the subsequent stages.

We continued the subsequent steps of the feasibility experiment with 300 954 unique short-forms appearing in two or more contextual sentences. It took about 35 h to generate 182 585 unique pairs of short/long-forms. These experimental results reveal that it is feasible to construct an acronym dictionary from the whole MEDLINE abstracts with the proposed method. We now focus on quality aspects of the method.

4 EVALUATION

4.1 Base-line systems and evaluation task

We compare our method with three base-line systems and two variants of our method.

- Proposed method (AM): described in this paper.
- Schwartz and Hearst’s method (SH): Their implementation[14] was used as is.
- Adar’s method (SaRAD)[15]: We implemented the acronym recognition component in SaRAD described in the paper (Adar, 2004) and its Supplementary information[16]. The implementation is available on our web site.
- Liu and Friedman’s method (LF)[15]: We implemented an acronym recognition program described in the paper (Liu *et al*., 2003). The program receives the long-form candidates obtained from the method described in Section 3.2 and applies selecting, subsuming and separating to the long-form candidates. We did not use the SPECIALIST Lexicon (suggested in their paper) for normalizing term-forms, but Porter’s stemming algorithm. Having the same set of long-form candidates as a set of potential collocations, we compare the quality of collocation mining with the proposed method. This implementation is also available on our web site.
- Proposed method with C-value termhood (CV): This is a variant of the proposed method applying the C-value measure CV(c) described in Formula 3. A comparison between AM and CV will show the improvement of the likelihood measure.
- Proposed method with Frequency termhood (FREQ): This is a variant of the proposed method replacing the likelihood LH(c) with the frequency of occurrence of long-form candidate c.

Given a list of target short-forms and their contextual sentences, each system identifies the long-forms for the short-forms. Porter’s stemming algorithm was applied to the long-forms in order to match them to reference long-forms extracted by a bio-informatician. We emulate the process of building an acronym dictionary by screening long-forms that occur θ or more times in the text collection[17]. For example, setting θ to 2 implies removing short/long-form pairs occurring once in the text collection, i.e. definitions of dynamic acronyms. We drew a precision-recall curve for each system by changing the threshold θ from 2 to 20.

4.2 Evaluation corpus and results

Several evaluation corpora for acronym recognition are available. The Medstract Gold Standard Evaluation Corpus, which consists of 166 alias pairs annotated to 283 sentences in 201 MEDLINE abstracts, is widely used for evaluation (Chang *et al*., 2006; Schwartz *et al*., 2003). Although this corpus is suitable for

[13] The MEDLINE database was up-to-date on March 2006. The size of the input data amounted to 52GB (from medline06n0001.xml to medline05n0514.xml).

[14] http://biotext.berkeley.edu/software.html

[15] The evaluation results for SaRAD and LF are based on our implementations and might not reflect the actual performance.

[16] http://www.hpl.hp.com/research/idl/papers/srad/websup-070703.pdf

[17] In other words, statistical information (frequency of occurrence of long-forms) is incorporated even in the letter matching algorithms as a post-processing phase.

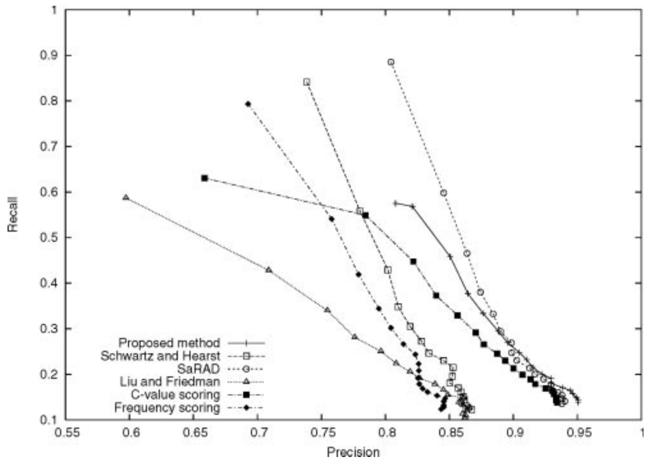

Fig. 2. Precision-recall calculated by the distinct numbers of long-forms.

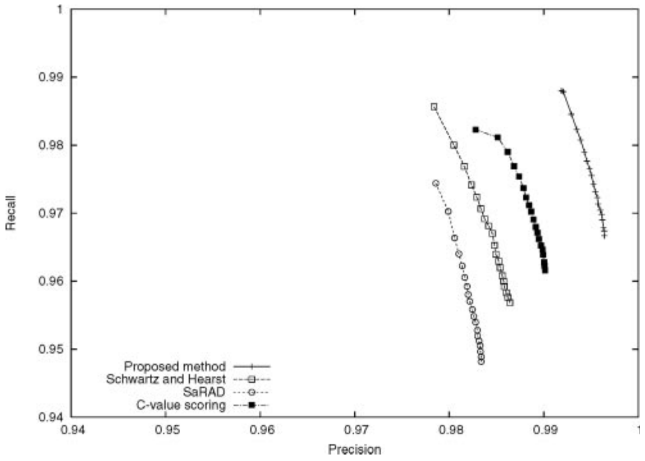

Fig. 3. Precision-recall calculated by the numbers of long-form occurrences.

evaluating a method that extracts aliases and their expanded forms defined locally within an abstract, the amount of text in the corpus is too small for the application of building an acronym dictionary.

Therefore, we evaluated our method on 637 957 contextual sentences containing 4024 short/long-form pairs for 100 short-forms[18]: half of the short-forms were constituted by the top 50 short-forms[19] appearing most frequently in MEDLINE abstracts and the remaining 50 short-forms were chosen from those discussed in papers on acronym recognition. Note that the use of the frequent 50 short-forms for the evaluation does not favour our method, which is based on statistics. In fact, a great number of long-forms for the 50 short-forms were found rarely, e.g. as many as 1076 pairs occur only twice. Although the most frequent long-form for the short-form CT (32 507 times) is *computed tomography* (18 512 times), a great number of less frequent long-forms also exist in the corpus, e.g. *cavernous tissue* (2 times), *complex tone* (2 times), *cortical threshold* (2 times). It is difficult for the proposed method to recognize such rare short/long-form pairs. Refer to the Supplementary information for the details of the evaluation corpus.

Figure 2 shows the precision-recall curves when we count the number of distinct long-forms, i.e. count once even if a short/long-form pair *hidden markov model* (*HMM*) occurs multiple times in the text collection. In general, a system marks the highest recall and lowest precision (i.e. plotted at the left-top in a locus) when the threshold θ is 2. As the threshold θ increases, the recall and precision become lower and higher, respectively (i.e. a locus draws a downward-sloping curve). The proposed method (AM) achieved 80.8% precision and 57.5% recall at $\theta = 2$ and 95.0% precision and 13.9% recall at $\theta = 20$. When used with a higher threshold ($\theta \geq 9$), AM outperformed other methods, marking the highest precision. The simple approach using frequency of co-occurrence (FREQ) did not yield a good result. The comparison between AM and CV also revealed the great improvement of the proposed likelihood over the original C-value measure. These facts strongly suggest the importance of term recognition in statistical long-form recognition. SaRAD obtained the best result of all systems with a lower threshold, e.g. 80.4% precision and 88.5% recall at $\theta = 2$. This result reflects the advantage of the letter matching approach when statistical clues in the source text are unavailable.

Figure 3 shows the precision-recall curves when we count the number of positive/negative instances in the source text, e.g. count 188 true positives if a method identifies the acronym *hidden markov model (HMM)* defined 188 times in the source text. This evaluation metric assesses the appropriateness of dealing with frequent short/long-form pairs: a system loses precision/recall with this metric if it misrecognizes/missed a frequent pair in the text collection. We did not plot the precision-recall locuses for CV (ca. 86% precision and 85% recall) and LF (ca. 91% precision and 60% recall) to focus on the results of superior systems in the figure. Our method (AM) outperformed the other methods, obtaining the highest precision and recall with all thresholds ($2 \leq \theta \leq 20$). AM achieved 99.1% precision and 98.7% recall at $\theta = 2$ and 99.6% precision and 96.6% recall at $\theta = 20$. These figures revealed that the proposed method scarcely missed long-forms occurring frequently in the evaluation corpus.

4.3 Analysis on non-recognitions and misrecognitions

Table 2 reports the number of false short/long-form pairs unrecognized and misrecognized by the proposed method (AM) and three base-line systems (SH, SaRAD and LF) at the highest threshold ($\theta = 20$). The causes of non-recognitions and misrecognitions were manually examined. The amount of false cases of each system is represented by the number of distinct pairs (num) and the number of pair occurrences (freq). Note that the typical examples shown in the table are explanatory and not necessarily agreed among the systems. For example, AM could recognize *hepatitis c virus* (*HCV*) correctly while SH misrecognized *hepatitis c virus infection*.

[18] The 637 957 contextual sentences containing 100 short-forms were drawn from the intermediate database described in Section 3.4.

[19] We have excluded several parenthetical expressions, such as II, III, $P < 0.05$, etc. since they do not introduce acronyms. We have also excluded a few short-forms, such as RA (18 810 occurrences) and AD (17 240 occurrences) because there are too many variations of their expanded forms to handle in manual preparation of our evaluation corpus.

Table 2. The number of unrecognized and misrecognized short/long-form pairs ($\theta = 20$)

Cause	AM		SH		SaRAD		LF		Typical false example
	num	freq	num	freq	num	freq	num	freq	
Occurrence below than the threshold (20)	3430	16 589	3492	18 218	3425	16 514	3430	16 589	complex tone (CT)
Letters shuffled in the acronym	0	0	16	5135	17	5272	—	—	gamma interferon (IFN-GAMMA)
Algorithmic error	23	2818	11	1832	26	8610	166	220 701	(depending on algorithms)
Total number of unrecognized long-forms	3453	19 407	3519	25 185	3468	30 396	3596	237 290	—
Coincidental sharing of letters (ordered)	16	563	6	334	17	701	17	6511	systemic arterial pressure (MAP)
Coincidental sharing of letters (unordered)	6	350	—	—	—	—	4	7342	muscular atrophy (PMA)
Unnecessary word(s) attached to the head	3	535	37	2150	0	0	8	1263	anti-human immunodeficiency virus (HIV)
Unnecessary word(s) attached to the tail	0	0	22	2320	1	110	28	2443	hepatitis c virus infection (HCV)
Unnecessary word(s) inserted in the middle	0	0	2	535	3	654	0	0	major histocompatibility gene complex (MHC)
Necessary word(s) missing	3	242	7	1949	14	7811	7	12 373	protein kinase (PKA)
Total number of misrecognized long-forms	28	1690	74	7288	35	9276	64	29 932	—

About 3400–3500 distinct long-forms were unrecognized by the systems[20], occurring less than 20 times in the test corpus.

The proposed method could not extract *conduct* (*CD*) since the extraction algorithm described in Section 3.3 forces to choose either *conduct disorder* (257 occurrences) or *conduct* (28 occurrences). The candidate *conduct* was eliminated from the list by comparing the long-form likelihoods. This is the typical example of unrecognized long-forms due to an algorithmic error. The word *disorder* (44 occurrences) was misrecognized as the long-form for the acronym *SD*. This phenomenon was due to: diverse expressions appear before *disorder* in the text collection, e.g. *somatic disorder* (26 occurrences), *sleep disorder* (7 occurrences), *schizophrene disorder* (1 occurrence), etc.; no diverse expansions could surpass *disorder*, i.e. Formula 4 assigned a higher score to *disorder* than to the diverse expressions; *disorder* contains letters ‘s’ and ‘d’ in the same order as the acronym and long-form extraction chose disorder and removed all the expansions from the candidate list. The proposed method could recognize 17 long-forms (5272 occurrences) whose letters are shuffled in the short-form, such as *gamma interferon* (*IFN-GAMMA*). Six shuffled long-forms (350 occurrences), such as *muscular atrophy* (*PMA*) were misrecognized due to coincidental satisfaction of the extraction algorithm. The comparison of these figures suggests that the proposed method does contribute to recognizing shuffled acronyms with little side effect.

Schwartz and Hearst’s method could not withdraw long-form candidates with unnecessary word(s) attached to the head or tail, e.g. *anti-human immunodeficiency virus* (*HIV*) (168 occurrences), *non-major histocompatibility complex* (*MHC*) (85 occurrences), *hepatitis c virus infection* (*HCV*) (33 occurrences) and *radiotherapy alone* (*RT*) (29 occurrences). These cases illustrate the major drawback of their algorithm. Such cases are likely to increase when we accept rare long-forms by lowering the threshold θ. In contrast, the proposed method eliminated these false candidates readily by the comparison of long-form likelihoods, e.g. *radiotherapy alone* [$\mathrm{LH}(c) = 27.2$] versus *radiotherapy* [$\mathrm{LH}(c) = 1602$].

SaRAD could not extract the acronym 12-*o-tetradecanoylphorbol-13-acetate* (*TPA*) (3656 occurrences) but the shorter long-form *tetradecanoylphorbol-13-acetate* (3917 occurrences), tokenizing expressions appearing before parentheses by non-alphanumeric characters (i.e. hyphens were replaced with spaces). Even though 93% of the occurrences of the latter candidate are derived from the former, the scoring function of SaRAD favoured the incorrect latter one. Besides, the scoring function sometimes assigns the same score to multiple candidates. For instance, both *systemic arterial pressure* (*MAP*) (44 occurrences) and *mean systemic arterial pressure* (42 occurrences) scored 2, according to the scoring function[21]. The proposed method also handled these non-trivial cases correctly.

Liu and Friedman’s method had difficulty in dealing with diverse expressions in a text collection. For instance, LF has the following rule to withdraw some long-form candidates: ‘‘remove a set of candidates T_c formed by adding a prefix word to a candidate w if the number of such candidates T_c is greater than a parameter t_0’’. This rule with $t_0 = 3$ applied to the collocation myocardial infarction removed the proper long-form acute *myocardial infarction* (5314 occurrences), for the acronym *AMI*, because *myocardial infarction* had 15 possible expansions in the MEDLINE abstracts, e.g. *anterior myocardial infarction* (34 occurrences), *phase myocardial infarction* (13 occurrences), *wall myocardial infarction* (5 occurrences), *inferior myocardial infarction* (4 occurrences), etc. Due to the rule, a number of frequent long-forms were missed, such as *epidermal growth factor* (*EGF*) (10 209 occurrences), *acquired immunodeficiency syndrome* (*AIDS*) (6111 occurrences), etc. This flaw might be improved by tweaking the parameters, but we would like to emphasize that our method achieved the result without a parameter.

4.4 Evaluation results estimated for the whole MEDLINE

Although we chose the 100 short-forms objectively, some might argue that our method lacks reproducibility of the actual distribution of acronyms appearing in the whole MEDLINE. Thus, we sampled 1/300 of the short-forms appearing more than 8 times in the whole MEDLINE and constructed another evaluation corpus containing 863 short/long-form pairs corresponding to the 248 short-forms.

[20] The small difference in the figures derives from the different methods of tokenization (e.g. handling of non-alpha-numerical letters, such as ‘-’ and ‘,’).

[21] Although the author did not describe a strategy to deal with tied score, our implementation prioritizes a shorter candidate over a longer based on the experimental results.

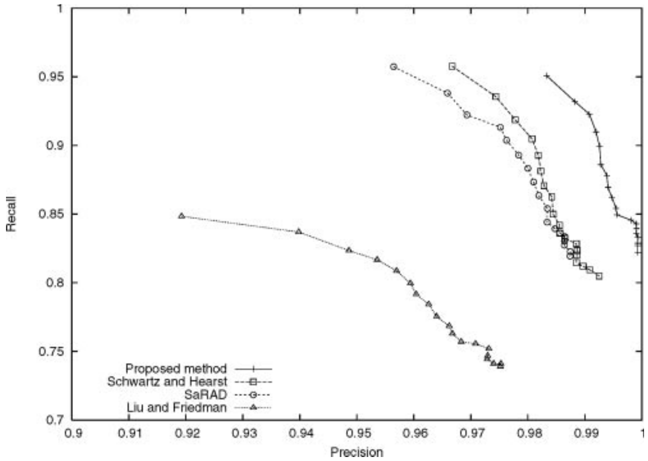

Fig. 4. Precision-recall curve on the evaluation corpus/metrics roughly emulating the whole MEDLINE.

The sampling procedure was designed to reproduce the distribution of acronyms in the whole MEDLINE. We chose every 300 entries in the list of short-forms arranged in descending order of their occurrences, i.e. *CT* (the most frequent short-form; 32 507 occurrences), *PCP* (the 301st frequent one; 3606 occurrences), *CFTR* (601st; 2079 occurrences), ..., 0.5 *MUG* (74 101st; 8 occurrences). Figure 4 shows the precision-recall curve, with interpolation (refer to the Supplementary material), when we count the occurrence number of positive/negative instances in the whole MEDLINE. The proposed method again outperformed other systems, achieving about 99% precision and 82–95% recall.

5 CONCLUSION

In this paper we described a term recognition approach to extract acronyms and their definitions from a large text collection. The main contribution of this study has been to show the usefulness of statistical information for building an acronym dictionary of good quality. The proposed method outperformed the base-line systems, achieving 99% precision and 82–95% recall on our evaluation corpus that roughly emulates the whole MEDLINE. Figures 2–4 confirmed the superiority of the proposed method in building a precise and comprehensive acronym dictionary. A future direction of this study would be to combine a letter matching algorithm to improve the recall of recognizing rare short/long-form pairs (if rare pairs are necessary) and to incorporate other types of relations expressed with parenthesis, such as synonym, paraphrase, etc.

ACKNOWLEDGEMENTS

The authors would like to thank Phillip Cotter, John McNaught and Jun'ichi Tsujii of the National Centre for Text-Mining for useful comments and suggestions. The UK National Centre for Text-Mining is funded by the Joint Information Systems Committee, the Biotechnology and Biological Sciences Research Council and the Engineering and Physical Sciences Research Council.

Conflict of Interest: none declared.

REFERENCES

Adar,E. (2004) SaRAD: a simple and robust abbreviation dictionary. *Bioinformatics*, **20**, 527–533.

Ao,H. and Takagi,T. (2005) ALICE: An algorithm to extract abbreviations from MEDLINE. *J. Am. Med. Inform. Assoc.*, **12**, 576–586.

Chang,J.T. and Schütze,H. (2006) Abbreviations in biomedical text. In Ananiadou,S. and McNaught,J. (eds), *Text Mining for Biology and Biomedicine*. Artech House Inc, London, pp. 99–119.

Frantzi,K.T. and Ananiadou,S. (1999) The C-value/NC-value domain independent method for multi-word term extraction. *J. Natural Lang. Proc.*, **6**, 145–179.

Gaudan,S. *et al.* (2005) Resolving abbreviations to their senses in Medline. *Bioinformatics*, **21**, 3658–3664.

Hisamitsu,T. and Niwa,Y. (2001) Extracting useful terms from parenthetical expression by combining simple rules and statistical measures: a comparative evaluation of bigram statistics. In Bourigault,D., Jacquemin,C. and L'Homme,M.-C. (eds), *Recent Advances in Computational Terminology*. John Benjamins, Amsterdam, pp. 209–224.

Liu,H. and Friedman,C. (2003) Mining terminological knowledge in large biomedical corpora. In *8th Pacific Symposium on Biocomputing (PSB 2003),* pp. 415–426.

Nadeau,D. and Turney,P.D. (2005) A supervised learning approach to acronym identification. In *8th Canadian Conference on Artificial Intelligence (AI'2005) (LNAI 3501),* pp. 319–329.

Pakhomov,S. (2002) Semi-supervised maximum entropy based approach to acronym and abbreviation normalization in medical texts. In *40th Annual Meeting of the Association for Computational Linguistics (ACL),* pp. 160–167.

Park,Y. and Byrd,R.J. (2001) Hybrid text mining for finding abbreviations and their definitions. In *2001 Conference on Empirical Methods in Natural Language Processing (EMNLP),* pp. 126–133.

Porter,M.F. (1980) An algorithm for suffix stripping. *Program*, **14**, 130–137.

Pustejovsky,J. *et al.* (2001) Automatic extraction of acronym meaning pairs from MEDLINE databases. *Med. info.*, 371–375.

Schwartz,A.S. and Hearst,M.A. (2003) A simple algorithm for identifying abbreviation definitions in biomedical text. In *Pacific Symposium on Biocomputing (PSB 2003).* **8**, pp. 451–462.

Taghva,K. and Gilbreth,J. (1999) Recognizing acronyms and their definitions. *Int. J. Doc. Anal. Recog.*, **1**, 191–198.

Wren,J.D. *et al.* (2005) Biomedical term mapping databases. *Nucleic Acids Res.*, **33**, D289–D293.

Wren,J.D. and Garner,H.R. (2002) Heuristics for identification of acronym-definition patterns within text: towards an automated construction of comprehensive acronym-definition dictionaries. *Meth. Inform. Med.*, **41**, 426–434.

Yu,H. *et al.* (2002) Mapping abbreviations to full forms in biomedical articles. *J. Am. Med. Inform. Assoc.*, **9**, 262–272.