A Term Recognition Approach to Acronym Recognition

Naoaki Okazaki *
Graduate School of Information
Science and Technology
The University of Tokyo
7-3-1 Hongo, Bunkyo-ku, Tokyo
113-8656 Japan
okazaki@mi.ci.i.u-tokyo.ac.jp

Sophia Ananiadou
National Centre for Text Mining
School of Informatics
Manchester University
PO Box 88, Sackville Street, Manchester
M60 1QD United Kingdom
Sophia.Ananiadou@manchester.ac.uk

Abstract

We present a term recognition approach to extract acronyms and their definitions from a large text collection. Parenthetical expressions appearing in a text collection are identified as potential acronyms. Assuming terms appearing frequently in the proximity of an acronym to be the expanded forms (definitions) of the acronyms, we apply a term recognition method to enumerate such candidates and to measure the likelihood scores of the expanded forms. Based on the list of the expanded forms and their likelihood scores, the proposed algorithm determines the final acronym-definition pairs. The proposed method combined with a letter matching algorithm achieved 78% precision and 85% recall on an evaluation corpus with 4,212 acronym-definition pairs.

1 Introduction

In the biomedical literature the amount of terms (names of genes, proteins, chemical compounds, drugs, organisms, etc) is increasing at an astounding rate. Existing terminological resources and scientific databases (such as Swiss-Prot[1], SGD[2], FlyBase[3], and UniProt[4]) cannot keep up-to-date with the growth of neologisms (Pustejovsky et al., 2001). Although curation teams maintain terminological resources, integrating neologisms is very difficult if not based on systematic extraction and collection of terminology from literature. Term identification in literature is one of the major bottlenecks in processing information in biology as it faces many challenges (Ananiadou and Nenadic, 2006; Friedman et al., 2001; Bodenreider, 2004). The major challenges are due to term variation, e.g. spelling, morphological, syntactic, semantic variations (one term having different termforms), term synonymy and homonymy, which are all central concerns of any term management system.

Acronyms are among the most productive type of term variation. Acronyms (e.g. RARA) are compressed forms of terms, and are used as substitutes of the fully expanded termforms (e.g., *retinoic acid receptor alpha*). Chang and Schütze (2006) reported that, in MEDLINE abstracts, 64,242 new acronyms were introduced in 2004 with the estimated number being 800,000. Wren et al. (2005) reported that 5,477 documents could be retrieved by using the acronym *JNK* while only 3,773 documents could be retrieved by using its full term, *c-jun N-terminal kinase*.

In practice, there are no rules or exact patterns for the creation of acronyms. Moreover, acronyms are ambiguous, i.e., the same acronym may refer to different concepts (*GR* abbreviates both *glucocorticoid receptor* and *glutathione reductase*). Acronyms also have variant forms (e.g. NF kappa B, NF kB, NF-KB, NF-kappaB, NFKB factor for nuclear factor-kappa B). Ambiguity and variation present a challenge for any text mining system, since acronyms have not only to be recognised, but their variants have to be linked to the same canonical form and be disambiguated.

Thus, discovering acronyms and relating them to their expanded forms is important for terminology management. In this paper, we present a term recognition approach to construct an acronym dic-

*Research Fellow of the Japan Society for the Promotion of Science (JSPS)

[1] http://www.ebi.ac.uk/swissprot/

[2] http://www.yeastgenome.org/

[3] http://www.flybase.org/

[4] http://www.ebi.ac.uk/GOA/

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 643–650,
Sydney, July 2006. ©2006 Association for Computational Linguistics

tionary from a large text collection. The proposed method focuses on terms appearing frequently in the proximity of an acronym and measures the likelihood scores of such terms to be the expanded forms of the acronyms. We also describe an algorithm to combine the proposed method with a conventional letter-based method for acronym recognition.

2 Related Work

The goal of acronym identification is to extract pairs of short forms (acronyms) and long forms (their expanded forms or definitions) occurring in text[5]. Currently, most methods are based on letter matching of the acronym-definition pair, e.g., *hidden markov model (HMM)*, to identify short/-long form candidates. Existing methods of short-/long form recognition are divided into pattern matching approaches, e.g., exploring an efficient set of heuristics/rules (Adar, 2004; Ao and Takagi, 2005; Schwartz and Hearst, 2003; Wren and Garner, 2002; Yu et al., 2002), and pattern mining approaches, e.g., Longest Common Substring (LCS) formalization (Chang and Schütze, 2006; Taghva and Gilbreth, 1999).

Schwartz and Hearst (2003) implemented an algorithm for identifying acronyms by using parenthetical expressions as a marker of a short form. A character matching technique was used, i.e. all letters and digits in a short form had to appear in the corresponding long form in the same order, to determine its long form. Even though the core algorithm was very simple, the authors report 99% precision and 84% recall on the Medstract gold standard[6].

However, the letter-matching approach is affected by the expressions in the source text and sometimes finds incorrect long forms such as *acquired syndrome* and *a patient with human immunodeficiency syndrome*[7] instead of the correct one, *acquired immune deficiency syndrome* for the acronym *AIDS*. This approach also encounters difficulties finding a long form whose short form is arranged in a different word order, e.g., *beta 2 adrenergic receptor (ADRB2)*. To improve the accuracy of long/short form recognition, some methods measure the appropriateness of these candidates based on a set of rules (Ao and Takagi, 2005), scoring functions (Adar, 2004), statistical analysis (Hisamitsu and Niwa, 2001; Liu and Friedman, 2003) and machine learning approaches (Chang and Schütze, 2006; Pakhomov, 2002; Nadeau and Turney, 2005).

Chang and Schütze (2006) present an algorithm for matching short/long forms with a statistical learning method. They discover a list of abbreviation candidates based on parentheses and enumerate possible short/long form candidates by a dynamic programming algorithm. The likelihood of the recognized 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. Their method achieved 80% precision and 83% recall on the Medstract corpus.

Hisamitsu and Niwa (2001) propose 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 by using statistical measures such as mutual information, χ^2 test with Yate's correction, Dice coefficient, log-likelihood ratio, etc. Their method deals with generic parenthetical expressions (e.g., abbreviation, non abbreviation paraphrase, supplementary comments), not focusing exclusively on acronym recognition.

Liu and Friedman (2003) proposed a method based on mining collocations occurring before the parenthetical expressions. Their method creates a list of potential long forms from collocations appearing more than once in a text collection and eliminates unlikely candidates with three rules, e.g., "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". Their approach cannot recognise expanded forms occurring only once in the corpus. They reported a precision of 96.3% and a recall of 88.5% for abbreviations recognition on their test corpus.

3 Methodology

3.1 Term-based long-form identification

We propose a method for identifying the long forms of an acronym based on a term extraction technique. We focus on terms appearing fre-

[5] This paper uses the terms "short form" and "long form" hereafter. "Long form" is what others call "definition", "meaning", "expansion", and "expanded form" of acronym.

[6] `http://www.medstract.org/`

[7] These examples are obtained from the actual MEDLINE abstracts submitted to Schwartz and Hearst's algorithm (2003). An author does not always write a proper definition with a parenthetic expression.

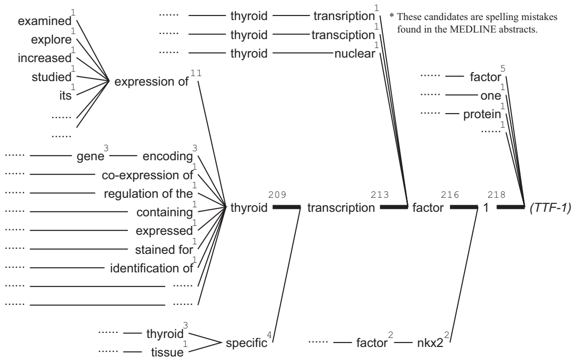

Figure 1: Long-form candidates for *TTF-1*.

quently in the proximity of an acronym in a text collection. More specifically, if a word sequence co-occurs frequently with a specific acronym and not with other surrounding words, we assume that there is a relationship[8] between the acronym and the word sequence.

Figure 1 illustrates our hypothesis taking the acronym *TTF-1* as an example. The tree consists of expressions collected from all sentences with the acronym 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 the acronym *TTF-1*, we note that they also co-occur frequently with the word *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*, *gene encoding thyroid transcription factor 1*, *etc.*). Therefore, we observe this to be the strongest relationship between acronym *TTF-1* and its long-form candidate *thyroid transcription factor 1* in the tree. We apply a number of validation rules (described later) to the candidate pair to make sure that it has an acronym-definition relation. In this example, the candidate pair is likely to be an acronym-definition relation because the long form *thyroid transcription factor 1* contains all alphanumeric letters in the short form *TTF-1*.

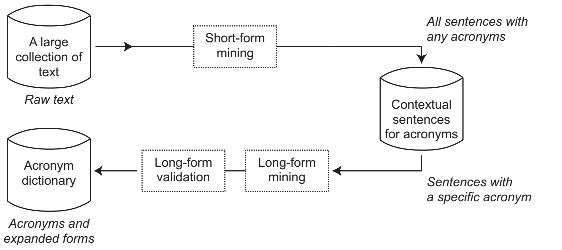
A large collection of text
Raw text
Short-form mining
All sentences with any acronyms
Contextual sentences for acronyms
Sentences with a specific acronym
Long-form mining
Long-form validation
Acronym dictionary
Acronyms and expanded forms

Figure 2: System diagram of acronym recognition

Figure 1 also shows another notable characteristic of long-form recognition. Assuming that the term *thyroid transcription factor 1* has an acronym *TTF-1*, we can disregard candidates such as *transcription factor 1*, *factor 1*, and *1* since they lack the necessary elements (e.g., *thyroid* for all candidates; *thyroid transcription* for candidates *factor 1* and *1*; etc.) to produce the acronym *TTF-1*. Similarly, we can disregard candidates such as *expression of thyroid transcription factor 1* and *encoding thyroid transcription factor 1* since they contain unnecessary elements (i.e., *expression of* and *encoding*) attached to the long-form. Hence, once *thyroid transcription factor 1* is chosen as the most likely long form of the acronym *TTF-1*, we prune the unlikely candidates: nested candidates (e.g., *transcription factor 1*); expansions (e.g., *expression of thyroid transcription factor 1*); and insertions (e.g., *thyroid specific transcription factor 1*).

3.2 Extracting acronyms and their contexts

Before describing in detail the formalization of long-form identification, we explain the whole process of acronym recognition. We divide the acronym extraction task into three steps (Figure 2):

1. **Short-form mining**: identifying and extracting short forms (i.e., acronyms) in a collection of documents

2. **Long-form mining**: generating a list of ranked long-form candidates for each short

[8] A sequence of words that co-occurs with an acronym does not always imply the acronym-definition relation. For example, the acronym *5-HT* co-occurs frequently with the term *serotonin*, but their relation is interpreted as a synonymous relation.

[9] The words with function words (e.g., *expression of*, *regulation of the*, etc.) are combined into a node. This is due to the requirement for a long-form candidate discussed later (Section 3.3).

Acronym	Contextual sentence
...	....
HML	Hard metal lung diseases *(HML)* are rare, and complex to diagnose.
HMM	Heavy meromyosin *(HMM)* from conditioned hearts had a higher Ca++-ATPase activity than from controls.
HMM	Heavy meromyosin *(HMM)* and myosin subfragment 1 (S1) were prepared from myosin by using low concentrations of alpha-chymotrypsin.
HMM	Hidden Markov model *(HMM)* techniques are used to model families of biological sequences.
HMM	Hexamethylmelamine *(HMM)* is a cytotoxic agent demonstrated to have broad antitumor activity.
HMN	Hereditary metabolic neuropathies *(HMN)* are marked by inherited enzyme or other metabolic defects.
...	... .

Table 1: An example of extracted acronyms and their contextual sentences.

form by using a term extraction technique

3. **Long-form validation**: extracting short/long form pairs recognized as having an acronym-definition relation and eliminating unnecessary candidates.

The first step, *short-form mining*, enumerates all short forms in a target text which are likely to be acronyms. Most studies make use of the following pattern to find candidate acronyms (Wren and Garner, 2002; Schwartz and Hearst, 2003):

long form ’(’ *short form* ’)’

Just as the heuristic rules described in Schwartz and Hearst (Schwartz and Hearst, 2003), we consider short forms to be valid only if they consist of at most two words; their length is between two to ten characters; they contain at least an alphabetic letter; and the first character is alphanumeric. All sentences containing a short form in parenthesis are inserted into a database, which returns all contextual sentences for a short form to be processed in the next step. Table 1 shows an example of the database content.

3.3 Formalizing long-form mining as a term extraction problem

The second step, *long-form mining*, generates a list of long-form candidates and their likelihood scores for each short form. As mentioned previously, we focus on words or word sequences that co-occur frequently with a specific acronym and not with any other surrounding words. We deal with the problem of extracting long-form candidates from contextual sentences for an acronym in a similar manner as the term recognition task which extracts terms from the given text. For that purpose, we used a modified version of the C-value method (Frantzi and Ananiadou, 1999).

C-value is a domain-independent method for automatic term recognition (ATR) which combines linguistic and statistical information, emphasis being placed on the statistical part. The linguistic analysis enumerates all candidate terms in a given text by applying part-of-speech tagging, candidate extraction (e.g., extracting sequences of adjectives/nouns based on part-of-speech tags), and a stop-list. The statistical analysis assigns a termhood (likelihood to be a term) to a candidate term by using the following features: the frequency of occurrence of the candidate term; the frequency of the candidate term as part of other longer candidate terms; the number of these longer candidate terms; and the length of the candidate term.

The C-value approach is characterized by the extraction of *nested* terms which gives preference to terms appearing frequently in a given text but not as a part of specific longer terms. This is a desirable feature for acronym recognition to identify long-form candidates in contextual sentences. The rest of this subsection describes the method to extract long-form candidates and to assign scores to the candidates based on the C-value approach.

Given a contextual sentence as shown in Table 1, we tokenize a contextual sentence by non-alphanumeric characters (e.g., space, hyphen, colon, etc.) and apply Porter’s stemming algorithm (Porter, 1980) to obtain a sequence of normalized words. We use the following pattern to extract long-form candidates from the sequence:

$$\texttt{[:WORD:].*\$} \tag{1}$$

Therein: `[: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. The extraction pattern accepts a word or word sequence if the word or word sequence begins with any non-function word, and ends with any word just before the corresponding short form in the contextual sentence. We have defined 113 function words such as *a*, *the*, *of*, *we*, and *be* in an external dictionary so that long-form candidates cannot begin with these words.

Let us take the example of a contextual sentence, “we studied the expression of thyroid transcription factor-1 (TTF-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*

Candidate	Length	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	L
effect of adriamycin	3	25	23.6	E
adrenodemedullated	1	19	17.7	o
acellular dermal matrix	3	17	15.9	o
peptide adrenomedullin	2	17	15.1	E
effects of adrenomedullin	3	15	13.2	E
resistance to adriamycin	3	15	13.2	E
amyopathic dermatomyositis	2	14	12.8	o
vincristine (vcr) and adriamycin	4	11	10.0	E
drug adriamycin	2	14	10.0	E
brevis and abductor digiti minimi	5	11	9.8	E
minimi	1	83	5.8	N
digiti minimi	2	80	3.9	N
right abductor digiti minimi	4	4	2.5	E
automated digital microscopy	3	1	0.0	m
adrenomedullin concentration	2	1	0.0	N

Valid = { o: *valid*, m: *letter match*, L: *lacks necessary letters*, E: *expansion*, N: *nested*, B: *below the threshold* }

Table 2: Long-form candidates for *ADM*.

transcript factor 1. Substrings such as *of thyroid transcript factor 1* (which begins with a function word) and *thyroid transcript* (which ends prematurely before the short form) are not selected as long-form candidates.

We define the likelihood $\mathrm{LF}(w)$ for candidate w to be the long form of an acronym:

$$\mathrm{LF}(w) = \mathrm{freq}(w) - \sum_{t \in T_w} \mathrm{freq}(t) \times \frac{\mathrm{freq}(t)}{\mathrm{freq}(T_w)}. \quad (2)$$

Therein: w is a long-form candidate; $\mathrm{freq}(x)$ denotes the frequency of occurrence of a candidate x in the contextual sentences (i.e., co-occurrence frequency with a short form); T_w is a set of nested candidates, long-form candidates each of which consists of a preceding word followed by the candidate w; and $\mathrm{freq}(T_w)$ represents the total frequency of such candidates T_w.

The first term is equivalent to the co-occurrence frequency of a long-form candidate with a short form. The second term discounts the co-occurrence frequency based on the frequency distribution of nested candidates. Given a long-form candidate $t \in T_w$, $\frac{\mathrm{freq}(t)}{\mathrm{freq}(T_w)}$ presents the occurrence probability of candidate t in the nested candidate set T_w. Therefore, the second term of the formula calculates the expectation of the frequency of occurrence of a nested candidate accounting for the frequency of candidate w.

Table 2 shows a list of long-form candidates for acronym *ADM* extracted from 7,306,153 MEDLINE abstracts[10]. The long-form mining step extracted 10,216 unique long-form candidates from 1,319 contextual sentences containing the acronym *ADM* in parentheses. Table 2 arranges long-form candidates with their scores in desending order. Long-form candidates *adriamycin* and *adrenomedullin* co-occur frequently with the acronym *ADM*.

Note the huge difference in scores between the candidates *abductor digiti minimi* and *minimi*. Even though the candidate *minimi* co-occurs more frequently (83 times) than *abductor digiti minimi* (78 times), the co-occurrence frequency is mostly derived from the longer candidate, i.e., *digiti minimi*. In this case, the second term of Formula 2, the occurrence-frequency expectation of expansions for *minimi* (e.g., *digiti minimi*), will have a high value and will therefore lower the score of candidate *minimi*. This is also true for the candidate *digiti minimi*, i.e., the score of candidate *digiti minimi* is lowered by the longer candidate *abductor digiti minimi*. In contrast, the candidate *abductor digiti minimi* preserves its co-occurrence frequency since the second term of the formula is low, which means that each expansion (e.g, *brevis and abductor digiti minimi*, *right abductor digiti minimi*, ...) is expected to have a low frequency of occurrence.

3.4 Validation rules for long-form candidates

The final step of Figure 2 validates the extracted long-form candidates to generate a final set of short/long form pairs. According to the score in Table 2, *adriamycin* is the most likely long-form for acronym *ADM*. Since the long-form candidate *adriamycin* contains all letters in the acronym *ADM*, it is considered as an authentic long-form (marked as 'o' in the Valid field). This is also true for the second and third candidate (*adrenomedullin* and *abductor digiti minimi*).

The fourth candidate *doxorubicin* looks interesting, i.e., the proposed method assigns a high score to the candidate even though it lacks the letters *a* and *m*, which are necessary to form the corresponding short form. This is because *doxorubicin* is a synonymous term for *adriamycin* and described directly with its acronym *ADM*. In this paper, we deal with the acronym-definition relation although the proposed method would be applicable to mining other types of relations marked by parenthetical expressions. Hence, we introduce a constraint that a long form must cover all alphanu-

[10] 52GB XML files (from `medline05n0001.xml` to `medline05n0500.xml`)

```
# [Variables]
#   sf: the target short-form.
#   candidates: long-form candidates.
#   result: the list of decisive long-forms.
#   threshold: the threshold of cut-off.

# Sort long-form candidates in descending order
candidates.sort(                    # of scores.
    key=lambda lf:lf.score, reverse=True)

# Initialize result list as empty.
result = []

# Pick up a long form one by one from candidates.
for lf in candidates:
    # Apply a cut-off based on termhood score.
    # Allow candidates with letter matching.....(a)
    if lf.score < threshold and not lf.match:
        continue
    # A long-form must contain all letters......(b)
    if letter_recall(sf, lf) < 1:
        continue
    # Apply pruning of redundant long form......(c)
    if redundant(result, lf):
        continue
    # Insert this long form to the result list.
    result.append(lf)

# Output the decisive long-forms.
print result
```

Figure 3: Pseudo-code for long-form validation.

meric letters in the short form.

The fifth candidate *effect of adriamycin* is an expansion of a long form *adriamycin*, which has a higher score than *effect of adriamycin*. As we discussed previously, the candidate *effect of adriamycin* is skipped since it contains unnecessary word(s) to form an acronym. Similarly, we prune the candidate *minimi* because it forms a part of another long form *abductor digiti minimi*, which has a higher score than the candidate *minimi*. The likelihood score $LF(w)$ determines the most appropriate long-form among similar candidates sharing the same words or lacking some words.

We do not include candidates with scores below a given threshold. Therefore, the proposed method cannot extract candidates appearing rarely in the text collection. It depends on the application and considerations of the trade-off between precision and recall, whether or not an acronym recognition system should extract such rare long forms. When integrating the proposed method with e.g., Schwartz and Hearst's algorithm, we treat candidates recognized by the external method as if they pass the score cut-off. In Table 2, for example, candidate *automated digital microscopy* is inserted into the result set whereas candidate *adrenomedullin concentration* is skipped since it is nested by candidate *adrenomedullin*.

Figure 3 is a pseudo-code for the long-form validation algorithm described above. A long-form candidate is considered valid if the following conditions are met: *(a)* it has a score greater than a threshold or is nominated by a letter-matching algorithm; *(b)* it contains all letters in the corresponding short form; and *(c)* it is not nested, expansion, or insertion of the previously chosen long forms.

Rank	Parenthetic phrase	# contextual sentence	# unique long-forms
1	CT	30,982	171
2	PCR	25,387	39
3	HIV	19,566	13
4	LPS	18,071	51
5	MRI	16,966	18
6	ELISA	16,527	25
7	SD	15,760	165
8	BP	14,860	145
9	DA	14,518	129
10	CSF	14,035	34
11	CNS	13,573	47
12	IL	13,423	60
13	PKC	13,414	11
14	TNF-ALPHA	12,228	14
15	HPLC	12,211	16
16	ER	12,155	140
17	RT-PCR	12,153	21
18	TNF	12,145	13
19	LDL	11,960	24
20	5-HT	11,836	20
..	....	...	..
—	(overall 50 acronyms)	600,375	4,212

Table 3: Statistics on our evaluation corpus.

4 Evaluation

Several evaluation corpora for acronym recognition are available. The Medstract Gold Standard Evaluation Corpus, which consists of 166 alias pairs annotated to 201 MEDLINE abstracts, is widely used for evaluation (Chang and Schütze, 2006; Schwartz and Hearst, 2003). However, the amount of the text in the corpus is insufficient for the proposed method, which makes use of statistical features in a text collection. Therefore, we prepared an evaluation corpus with a large text collection and examined how the proposed algorithm extracts short/long forms precisely and comprehensively.

We applied the short-form mining described in Section 3 to 7,306,153 MEDLINE abstracts[10]. Out of 921,349 unique short-forms recognized by the short-form mining, top 50 acronyms[11] appearing frequently in the abstracts were chosen for our

[11] We have excluded several parenthetical expressions such as *II* (99,378 occurrences), *OH* (37,452 occurrences), and *P<0.05* (23,678 occurrences). Even though they are enclosed within parentheses, they do not introduce acronyms. We have also excluded a few acronyms such as *RA* (18,655 occurrences) and *AD* (15,540 occurrences) because they have many variations of their expanded forms to prepare the evaluation corpus manually.

evaluation corpus. We asked an expert in bioinformatics to extract long forms from 600,375 contextual sentences with the following criteria: a long form with minimum necessary elements (words) to produce its acronym is accepted; a long form with unnecessary elements, e.g., *magnetic resonance imaging unit (MRI)* or *computed x-ray tomography (CT)*, is not accepted; a misspelled long-form, e.g., *hidden markvov model (HMM)*, is accepted (to separate the acronym-recognition task from a spelling-correction task). Table 3 shows the top 20 acronyms in our evaluation corpus, the number of their contextual sentences, and the number of unique long-forms extracted.

Using this evaluation corpus as a gold standard, we examined precision, recall, and f-measure[12] of long forms recognized by the proposed algorithm and baseline systems. We compared five systems: the proposed algorithm with Schwartz and Hearst's algorithm integrated (PM+SH); the proposed algorithm without any letter-matching algorithm integrated (PM); the proposed algorithm but using the original C-value measure for long-form likelihood scores (CV+SH); the proposed algorithm but using co-occurrence frequency for long-form likelihood scores (FQ+SH); and Schwartz and Hearst's algorithm (SH). The threshold for the proposed algorithm was set to four.

Table 4 shows the evaluation result. The best-performing configuration of algorithms (PM+SH) achieved 78% precision and 85% recall. The Schwartz and Hearst's (SH) algorithm obtained a good recall (93%) but misrecognized a number of long-forms (56% precision), e.g., *the kinetics of serum tumour necrosis alpha (TNF-ALPHA)* and *infected mice lacking the gamma interferon (IFN-GAMMA)*. The SH algorithm cannot gather variations of long forms for an acronym, e.g., *ACE* as *angiotensin-converting enzyme level, angiotensin i-converting enzyme gene, angiotensin-1-converting enzyme, angiotensin-converting, angiotensin converting activity*, etc. The proposed method combined with the Schwartz and Hearst's algorithm remedied these misrecognitions based on the likelihood scores and the long-form validation algorithm. The PM+SH also outperformed other likelihood measures, CV+SH and FQ+SH.

Method	Precision	Recall	F-measure
PM+SH	0.783	0.849	0.809
CV+SH	0.722	0.838	0.765
FQ+SH	0.716	0.800	0.747
SH	0.555	0.933	0.681
PM	0.815	0.140	0.216

Table 4: Evaluation result of long-form recognition.

The proposed algorithm without Schwartz and Hearst's algorithm (PM) identified long forms the most precisely (81% precision) but misses a number of long forms in the text collection (14% recall). The result suggested that the proposed likelihood measure performed well to extract frequently used long-forms in a large text collection, but could not extract rare acronym-definition pairs. We also found the case where PM missed a set of long forms for acronym *ER* which end with *rate*, e.g., *eating rate*, *elimination rate*, *embolic rate*, etc. This was because the word *rate* was used with a variety of expansions (i.e., the likelihood score for *rate* was not reduced much) while it can be also interpreted as the long form of the acronym.

Even though the Medstract corpus is insufficient for evaluating the proposed method, we examined the number of long/short pairs extracted from 7,306,153 MEDLINE abstracts and also appearing in the Medstract corpus. We can neither calculate the precision from this experiment nor compare the recall directly with other acronym recognition methods since the size of the source texts is different. Out of 166 pairs in Medstract corpus, 123 (74%) pairs were exactly covered by the proposed method, and 15 (83% in total) pairs were partially covered[13]. The algorithm missed 28 pairs because: 17 (10%) pairs in the corpus were not acronyms but more generic aliases, e.g., *alpha tocopherol (Vitamin E)*; 4 (2%) pairs in the corpus were incorrectly annotated (e.g, long form in the corpus *embryo fibroblasts* lacks word *mouse* to form acronym *MEFS*); and 7 (4%) long forms are missed by the algorithm, e.g., the algorithm recognized pair *protein kinase (PKR)* while the correct pair in the corpus is *RNA-activated protein kinase (PKR)*.

[12] We count the number of unique long forms, i.e., count once even if short/long form pair ⟨*HMM, hidden markov model*⟩ occurs more than once in the text collection. The Porter's stemming algorithm was applied to long forms before comparing them with the gold standard.

[13] Medstract corpus leaves unnecessary elements attached to some long-forms such as *general transcription factor iib (TFIIB)*, whereas the proposed algorithm may drop the unnecessary elements (i.e. *general*) based on the frequency. We regard such cases as *partly* correct.

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 recognizing acronyms in large text collections. The proposed method combined with a letter matching algorithm achieved 78% precision and 85% recall on the evaluation corpus with 4,212 acronym-definition pairs.

A future direction of this study would be to incorporate other types of relations expressed with parenthesis such as synonym, paraphrase, etc. Although this study dealt with the acronym-definition relation only, modelling these relations will also contribute to the accuracy of the acronym recognition, establishing a methodology to distinguish the acronym-definition relation from other types of relations.

References

Eytan Adar. 2004. SaRAD: A simple and robust abbreviation dictionary. *Bioinformatics*, 20(4):527–533.

Sophia Ananiadou and Goran Nenadic. 2006. Automatic terminology management in biomedicine. In Sophia Ananiadou and John McNaught, editors, *Text Mining for Biology and Biomedicine*, pages 67–97. Artech House, Inc.

Hiroko Ao and Toshihisa Takagi. 2005. ALICE: An algorithm to extract abbreviations from MEDLINE. *Journal of the American Medical Informatics Association*, 12(5):576–586.

Olivier Bodenreider. 2004. The Unified Medical Language System (UMLS): Integrating biomedical terminology. *Nucleic Acids Research*, 32:267–270.

Jeffrey T. Chang and Hinrich Schütze. 2006. Abbreviations in biomedical text. In S. Ananiadou and J. McNaught, editors, *Text Mining for Biology and Biomedicine*, pages 99–119. Artech House, Inc.

Katerina T. Frantzi and Sophia Ananiadou. 1999. The C-value / NC-value domain independent method for multi-word term extraction. *Journal of Natural Language Processing*, 6(3):145–179.

Carol Friedman, Hongfang Liu, Lyuda Shagina, Stephen Johnson, and George Hripcsak. 2001. Evaluating the UMLS as a source of lexical knowledge for medical language processing. In *AMIA Symposium*, pages 189–193.

Toru Hisamitsu and Yoshiki Niwa. 2001. Extracting useful terms from parenthetical expression by combining simple rules and statistical measures: A comparative evaluation of bigram statistics. In Didier Bourigault, Christian Jacquemin, and Marie-C L'Homme, editors, *Recent Advances in Computational Terminology*, pages 209–224. John Benjamins.

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

David Nadeau and Peter D. Turney. 2005. A supervised learning approach to acronym identification. In *8th Canadian Conference on Artificial Intelligence (AI'2005) (LNAI 3501)*, page 10 pages.

Serguei Pakhomov. 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)*, pages 160–167.

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

Martin F. Porter. 1980. An algorithm for suffix stripping. *Program*, 14(3):130–137.

James Pustejovsky, José Castaño, Brent Cochran, Maciej Kotecki, and Michael Morrell. 2001. Automatic extraction of acronym meaning pairs from MEDLINE databases. *MEDINFO 2001*, pages 371–375.

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

Kazem Taghva and Jeff Gilbreth. 1999. Recognizing acronyms and their definitions. *International Journal on Document Analysis and Recognition (IJDAR)*, 1(4):191–198.

Jonathan D. Wren and Harold R. Garner. 2002. Heuristics for identification of acronym-definition patterns within text: towards an automated construction of comprehensive acronym-definition dictionaries. *Methods of Information in Medicine*, 41(5):426–434.

Jonathan D. Wren, Jeffrey T. Chang, James Pustejovsky, Eytan Adar, Harold R. Garner, and Russ B. Altman. 2005. Biomedical term mapping databases. *Database Issue*, 33:D289–D293.

Hong Yu, George Hripcsak, and Carol Friedman. 2002. Mapping abbreviations to full forms in biomedical articles. *Journal of the American Medical Informatics Association*, 9(3):262–272.