International Journal of Document Analysis (2006) 8(1): 1–14
DOI 10.1007/s10032-005-0146-7

REGULAR PAPER

Eduardo Torres Schumann · Klaus U. Schulz

Stable methods for recognizing acronym-expansion pairs: from rule sets to hidden Markov models

Received: 1 February 2005 / Published online: 10 May 2005
© Springer-Verlag 2005

Abstract The replacement of textual units by synonymous canonical forms is an important prerequisite for many variants of automated text analysis. In scientific texts, one common normalization step is the consistent replacement of acronyms by their definitions. For many acronyms, the definition is found at a certain position of the text where the acronym is introduced and "expanded" to a synonymous sequence of full words. A recent approach to detecting acronym-expansion pairs by Park and Byrd [19] describes possible graphical correspondences between acronyms and expansions by means of fine-grained rules. Here we show how rule sets as used in [19] can be translated into hidden Markov models that abstract from details of the graphical correspondence and improve recall in a significant way. Stability in terms of precision is ensured by exploiting simple properties of the expansion with an optional reinforcement of linguistic knowledge. With this extension of the original formalism, the introduction of large rule sets can be avoided and a fixed model can be applied to a large variety of texts without retraining, with good values both for recall and precision.

Keywords Acronym recognition · Biomedical texts · Automated text analysis · Terminological expressions · Hidden Markov models

1 Introduction

Terminological expressions [2, 4, 11, 27] play a crucial role in document indexing and information retrieval [6, 22], text classification [3, 13], machine-assisted translation [7, 8], and computational lexicology [20]. They yield a more precise picture of the contents of a given document and a more succinct description of a specific concept than general keywords. In domain-specific scientific and technical texts, terminological expressions are often replaced by acronyms. For automated text understanding, normalization techniques are important that expand acronyms and lead to a unique symbolic representation of a fixed concept or entity.

Fortunately, acronyms used in a text are often introduced at some position: the meaning of the acronyms is explained in a corresponding sequence of words, the expansion, or definition. In this paper we consider methods for automatically detecting acronym-expansion pairs in texts. These methods help to normalize the given text, and they may also be used for constructing acronym dictionaries for narrow domains where ambiguities can be controlled.

Automated text analysis, indexing, annotation, and text mining are particularly relevant for the field of biomedical texts [1, 9, 10, 24]. The current interest in genetics and biomedicine has recently led to an enormous number of publications. For example, more than 12 million publications are now available in the PubMed (Medline) database [15], and 460,000 references were added in 2002 [18]. Since the use of acronyms in biomedical texts is very popular, automated detection of acronyms and their definitions represents one important step toward improved methods for automated text analysis in this genre.

Techniques for finding acronyms and their expansions originated in [25] and have been studied by various authors [12, 17, 19, 21, 26, 29]. In our own work, we chose rule-based approaches [12, 19, 26] as a starting point. In these approaches, only a modest set of linguistic background resources is needed, which simplifies implementation. For our initial experiments, we implemented and evaluated the approach described in [19]. This method models the creation of an acronym in a declarative, flexible, and natural way as a kind of rewriting process, improving ideas from earlier work [12, 26].

When using a rule-based approach to acronym recognition, the central problem is the selection of a good rule set on the basis of a given training corpus. If the set of rules is too small, then many acronym definitions in the evaluation/application corpus are not captured appropriately. The manual collection of rule sets with sufficient coverage on

E. Torres Schumann (✉) · K. U. Schulz
CIS, University of Munich, Oettingenstr 67, D-80538 München, Germany
E-mail: {torres, schulz}@cis.uni-muenchen.de

the basis of training data is costly, in particular in situations where most rules have a low frequency. Large rule sets, on the other hand, may give rise to ambiguities that are difficult to resolve and lead to reduced precision. Hence the question arises as to whether the selection of rule sets is *convergent* in the sense that after a series of training experiments a *stable* rule set can be found that leads to good results on any (realistic) evaluation corpus.

In our experiments with the method from [19], we found it difficult to obtain a fully satisfactory recall on test corpora. An error analysis showed that most of the rules needed for training or test corpora are very specific. This explains why rules derived on training corpora often do not help for test corpora. We then developed a new method that can be considered as a modification and extension of the original formalism.

In this paper we show how rule sets such as are used in Park's and Byrd's approaches can be directly translated into hidden Markov models (HMMs) of a particular type. Roughly speaking, transition sequences in these HMMs represent "generalized rules" where we abstract from details of the graphical correspondence between acronyms and expansions. States and probability parameters for the HMMs are directly obtained from the corresponding rule sets, so no additional training is needed. In general, the derived HMMs allow for many transition paths that are "new" in the sense that they do not correspond to a rule already seen in the training corpora. This explains why translated HMMs are more flexible and able to find many acronym-expansion pairs where a new rule would be needed in the original approach.

The graphical correspondence between acronyms and expansion candidates, which in the hybrid approach was mainly expressed in the rules, is now more strongly controlled and evaluated in the preference scheme. Combining HMMs with these methods, a loss of precision can be avoided and we end up with a method that is stable in the above sense and leads to satisfactory values for precision and recall. Since trained HMMs can be directly obtained by translating rule sets as those used in [19], our formalism inherits most advantages of the latter method, such as declarativity, flexible user customization, adaptivity to new styles, and editorial conventions.

The paper is structured as follows. In Sect. 2 we briefly look at typical acronyms occurring in biomedical texts and introduce the terminology that is used in the remaining sections. In Sect. 3 we briefly summarize the algorithmic problem considered in this paper and formally define variants of precision and recall that are used later. Section 4 gives a compact description of the hybrid approach [19]. Evaluation results that measure the stability (in the aforementioned sense) of this approach are given in Sect. 5. We also analyze the errors that typically occur when applying the method to a new class of documents. In Sect. 6 we describe our technique for translating rule sets into HMMs. Evaluation results for our own method are given in Sect. 7. Section 8 comments on related work. Some points for future work are summarized in the conclusion.

2 Acronyms in biomedical texts

Using examples from biomedical texts we briefly review distinct ways in which acronyms and their explanations are introduced in texts and examine in more detail the possible graphical relationship between an acronym and its expansion.

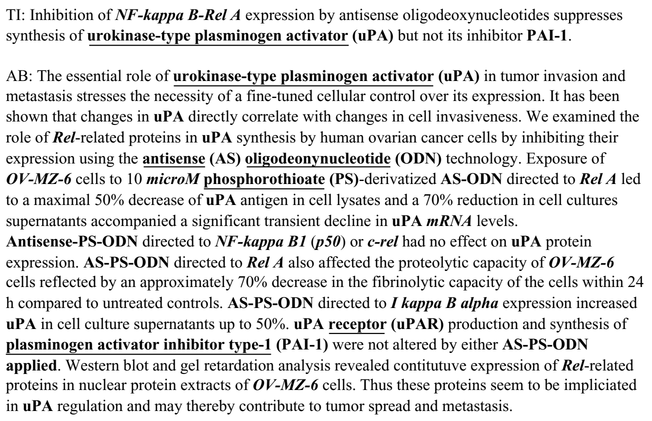
2.1 Terminology

Acronyms can be described as short sequences of symbols that represent abbreviations for compound technical expressions and concepts. The acronym and its expansion are treated as synonyms. We may distinguish between **proper acronyms**, where we have a purely graphical correspondence between the acronym and its expansion, and **pseudoacronyms**, where the relationship between the abbreviation and the underlying expression is more indirect. In this paper, by acronym we always mean a proper acronym unless noted otherwise.

Example 1 The sequence "uPA" represents a possible acronym for "urokinase-type plasminogen activator" since each letter of "uPA" occurs in the expansion. In contrast, "H_2O" is not a proper acronym for "water" since the association between both expressions is not graphical but rather based on special encoding conventions for chemical substances.

The acronyms in a given text can be partitioned into two classes. For some acronyms, the proper expansion is given in the text. Other "popular" acronyms are used without any explanation, assuming that the reader knows how to interpret the sequence.

Example 2 Figure 1 depicts an abstract from the Medline [15] database. Occurrences of acronyms and pseudoacronyms are written in bold. Acronyms where the expansion can be found somewhere in the abstract are written in Roman letters. "Popular" acronyms not defined in the text are written in italics.



TI: Inhibition of ***NF-kappa B-Rel A*** expression by antisense oligodeoxynucleotides suppresses synthesis of **urokinase-type plasminogen activator** **(uPA)** but not its inhibitor **PAI-1**.

AB: The essential role of **urokinase-type plasminogen activator** **(uPA)** in tumor invasion and metastasis stresses the necessity of a fine-tuned cellular control over its expression. It has been shown that changes in **uPA** directly correlate with changes in cell invasiveness. We examined the role of ***Rel***-related proteins in **uPA** synthesis by human ovarian cancer cells by inhibiting their expression using the **antisense** **(AS)** **oligodeonynucleotide** **(ODN)** technology. Exposure of ***OV-MZ-6*** cells to 10 ***microM*** **phosphorothioate** **(PS)**-derivatized **AS-ODN** directed to ***Rel A*** led to a maximal 50% decrease of **uPA** antigen in cell lysates and a 70% reduction in cell cultures supernatants accompanied a significant transient decline in **uPA** ***mRNA*** levels. **Antisense-PS-ODN** directed to ***NF-kappa B1*** (***p50***) or ***c-rel*** had no effect on **uPA** protein expression. **AS-PS-ODN** directed to ***Rel A*** also affected the proteolytic capacity of ***OV-MZ-6*** cells reflected by an approximately 70% decrease in the fibrinolytic capacity of the cells within 24 h compared to untreated controls. **AS-PS-ODN** directed to ***I kappa B alpha*** expression increased **uPA** in cell culture supernatants up to 50%. **uPA** **receptor** **(uPAR)** production and synthesis of **plasminogen activator inhibitor type-1** **(PAI-1)** were not altered by either **AS-PS-ODN** **applied**. Western blot and gel retardation analysis revealed contitutuve expression of ***Rel***-related proteins in nuclear protein extracts of ***OV-MZ-6*** cells. Thus these proteins seem to be impliciated in **uPA** regulation and may thereby contribute to tumor spread and metastasis.

Fig. 1 Acronyms in a Medline abstract

Remark 1 Biomedical texts, as well as other texts, use a considerable number of acronyms without explicitly explaining their meaning in the text. Even with special background dictionaries, the correct interpretation of these "alien" acronyms is difficult due to a large number of ambiguities. Acronyms are introduced for an enormous number of distinct entities and concepts. Often a given entity can be referred to using different terminological expressions. A given terminological expression may be encoded using distinct acronyms. Since short sequences are preferred, most acronyms can be associated with a substantial number of concepts or entities. When interpreting a given text, the context and domain may be used to obtain a partial disambiguation. However, often a complete disambiguation is not possible.

The problem of how to correctly interprete acronyms that are not defined in the text is ignored for the rest of this paper. In order to analyze text passages where the meaning of an acronym is explained, we distinguish between the following:

1. The *acronym*;
2. The *expansion* of the acronym;
3. The *introduction context*, i.e., the minimal window of the text where we find both the expansion and the acronym, possibly additional intermediate text;
4. The *graphical relationship* between acronym and expansion. This relation ship can typically be described as a mapping from the symbols of the acronym to occurrences of symbols in the expansion.

Example 3 Figure 2 illustrates these concepts using two introductions for the acronym "PAI-1" from the Medline database. The acronym is written in bold. Its expansion "plasminogen activator inhibitor type-1" is highlighted using a bright window. The introduction context is highlighted using a dark window.

2.2 Introduction contexts

The recognition of introduction contexts for acronyms is one important subtask in the automatic collection of acronyms and their expansions. In texts, the following techniques for introducing acronyms may be distinguished.

Implicit synonymy. In many cases the acronym is simply written in brackets. The situation where the acronym is written immediately after the expansion can be seen as a standard case. Since deviations are possible, difficulties may arise:

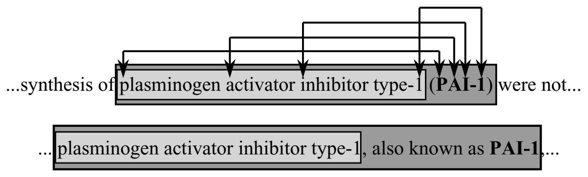

...synthesis of
plasminogen activator inhibitor type-1
(PAI-1)
were not...
...plasminogen activator inhibitor type-1, also known as PAI-1,...

Fig. 2 Acronym, expansion, graphical relationship, and introduction context

...plasminogen activator inhibitor type-1 (PAI-1)...
...plasminogen activator inhibitor (PAI-1) type-1...

Explicit synonymy. In a few cases, the synonymy between the acronym and its expansion is explicitly stated using natural language terms such as in the following examples:

...plasminogen activator inhibitor type-1, *also known as* PAI-1,...
...PAI-1 *stands for* plasminogen activator inhibitor type-1,...
...PAI-1, or plasminogen activator inhibitor type-1,...
Plasminogen activator inhibitor type-1, *also denoted* PAI-1,...

It is straightforward to collect a list of "typical" natural language expressions that are used to express synonymy. Still, such a list will necessarily be incomplete. Furthermore, problems may be caused by multifunctional expressions like "or."

In a way, these two classes only cover the simplest examples. The automated mapping from acronyms to expansions is complicated by a number of further phenomena.

Distributive introduction. In Fig. 1 we find the introduction "...antisense (AS) oligodeoxynucleotide (ODN) technology...." Later the combined acronym "AS-ODN" is used. In order to find its expansion, we have to combine the distributed expansions for "AS" and "ODN."

Use of acronyms within expansions. Often a given acronym is further extended in a new introduction. In Fig. 1 we find "...phosphorothioate (PS)-derivatized AS-ODN...," which explains the expressions "Antisense-PS-ODN" and "AS-PS-ODN" used below. Similarly, "uPA" is extended to "uPA receptor (uPAR)."

Enumerative contexts. Sometimes several related acronyms are explained in an enumerative context. To find the correct expansion, special mechanisms for these constructions have to be taken into account: "...total body (TBW) and extracellular (ECW) water...."

2.3 Graphical correspondence

The most obvious graphical relationship is given if the acronym is composed of initial letters of words of the expansion and if the order of initial letters is preserved in the acronym. However, we often find a more complex situation.

One phenomenon is the distinct number of letters in a word of the expansion that are used in the acronym. Furthermore, symbols from distinct words may be separated or written in a directly consecutive way. For example, in the Medline corpus the acronyms "AS-ODN," "AsOdn," and "AS ODN" are used for "antisense oligodeoxynucleotide." Generally, the use of uppercase and lowercase letters in acronyms does not follow any systematic scheme, which may be seen from introduction contexts such as "Recombinant interferon (rIFN)". Another phenomenon is deviations concerning the expected order of symbols: “horseradish peroxidase conjugated cholera toxin B (CB-HRP).” The latter two examples show that often inner symbols from tokens are selected for creating acronyms. Other acronyms copy prefixes.

Acronym composition may interact with morphology. For example, “ODNs” may indicate a plural, and a derivation “anti-ODN” may result in “aODN.” Variants of acronyms are used for pragmatic reasons to add emphasis or to suppress parts of the expansion that are clear from the given context (e.g., “AS-PS-ODN” as “PS-AS-ODN,” “Antisense-PS-ODN,” or “PS-ODN”).

3 Algorithmic problem and evaluation parameters

The **algorithmic problem** studied in this paper is the following: given a text T, find the acronyms defined in the text and their expansions. The quality of an algorithm $\mathcal{A}$ for solving the above problem may be measured in terms of standard notions of precision and recall. To formalize these notions we assume that the output set of $\mathcal{A}$ contains pairs of the form $\langle A, E\rangle$, where A represents a sequence of symbols treated as an acronym by the algorithm and E represents portions of text treated as the expansion of A. The pair $\langle A, E\rangle$ is *correct* if in fact A is an acronym and E is its expansion. The **precision** of $\mathcal{A}$ (w.r.t. T) is defined as the percentage of all correct answer pairs among the number of all answer pairs, and **recall** is defined as the percentage of all correct answer pairs w.r.t. the total number of all acronym-expansion pairs in T.

In our experiments, the large majority of all acronyms were introduced using implicit synonymy. To search texts for acronyms, we used a regular expression, Γ (see below for details). Matches for Γ in a given text that are surrounded by brackets are treated as **acronym candidates**. We first search for occurrences of acronym candidates and then calculate the most plausible expansion in the neighborhood of a given occurrence. By **Γ-restricted recall** we mean the modified recall value where we only refer to those acronym-expansion pairs in the given text where the acronym is parenthesized and matches Γ.

Most approaches to acronym extraction use a similar notion of acronym candidates and the same kind of search strategy. “Recall” values in the literature very often represent values for some form of restricted recall. These “recall” values are not directly comparable; their significance strongly depends on the generality of the pattern used for finding acronym candidates. Note that in order to measure absolute recall we have to find all acronym-expansion pairs of the evaluation corpus, which is difficult if the corpus is large. In the conclusion, we briefly comment on absolute recall values for the methods discussed below.

It should also be noted that for measuring Γ-restricted recall we only consider the top-ranked expansion candidate for a given acronym candidate. Related recall values based on top-n expansion candidate sets would make sense for interactive systems but are ignored here.

One natural strategy for improving (recall and) precision values relies on improved methods for correctly recognizing introduction contexts. To analyze distinct sources of error and to measure the relevance of improved methods for context recognition, we sometimes look at “idealized” precision values where the answer set is restricted in the sense that only proper acronyms in real introduction contexts are taken into account. The idealized precision value then says in which percentage of all these cases the proper expansion was found. The value can be seen as an upper limit for the improvements that can be expected from better methods for recognizing real introduction contexts.

As indicated in the introduction, the *algorithmic metaproblem* faced here is the following: find a stable method for extracting acronym-expansion pairs, i.e., a method that is applicable to a large variety of texts without retraining and leads to good values for precision and recall.

4 The hybrid approach

“Hybrid text mining” [19] represents a sophisticated rule-based method for extracting acronyms and building up on and refining techniques from earlier approaches [12, 26]. The graphical relationship between acronym and expansion is described by a special set of matching operators. For searching expansion candidates and for selecting a preferred candidate, morphological and syntactic properties of words are taken into account. The approach is called “hybrid” since it uses a variety of linguistic background knowledge bases (prefix list, replacement table, dictionary, see below) and since each rule comes with an application probability that is used for selecting the preferred expansion candidate. In the remainder of this section, we introduce the variant of hybrid text mining that we used in our experiments with biomedical texts. Our own HMM-based method uses this variant as an ingredient for parameter estimation at the end of the initial training phase. Some places where we deviate from the original approach were motivated by our experiments. All major modifications are mentioned below.

In our implementation, a given input text is first segmented into sentences using some simple heuristics similar to those described in [16]. Each sentence is then split into tokens. Token delimiters are tabulators, white spaces, occurrences of the symbols “;”, “:”, “,” followed by white space, and periods marking a sentence end.

4.1 Acronym candidates and skeletons

Acronym candidates are expressions A that occur in a parenthesized form in the text and match the following regular expression,[1] Γ.

$([a-zA-Z]\backslash -?[a-zA-Z])\ |$
$([a-zA-Z](\backslash -?[a-zA-Z])\{2\})\ |$

[1] Here $\backslash -?$ denotes an optional occurrence of “$-$”, and $(\alpha)\{n\}$ means exactly n consecutive occurrences of matches for a, “$+$” means at least one occurrence.

$([a-zA-Z](\backslash-?[a-zA-Z])\{3\})\ |$
$([a-zA-Z](\backslash-?[a-zA-Z])\{4\})\ |$
$([a-zA-Z](\backslash-?[a-zA-Z])\{2\}\backslash-?[1-9]+)$

Γ captures tokens that may contain hyphens and have 2–5 (uppercase or lowercase) letters, or three letters followed by a number. Letters of the acronym candidate are said to have type c, and maximal connected sequences of digits are said to have type n. In this way, each candidate is split into a sequence of *acronym units* and associated with a unique sequence over the alphabet $\{c, n\}$, which is called the *skeleton* of the acronym candidate. The "-" symbol is ignored in the structural description.

Example 4 The units of SN-1999 are "S," "N," and '1999," which means that the skeleton is *ccn*.

Remark 2 In [19] acronym candidates are all tokens of length l, $2 \leq l \leq 10$, that start with a letter or a digit and contain at least one uppercase letter. Excluded are the initial words of sentences, uppercase words from a general background dictionary, and words in a predefined list of stop words and false positives.

4.2 Search windows, morphemes, expansion candidates

The search for expansion candidates takes place within a window in front of a given acronym candidate A.[2] The window length (number of tokens) is $L+10$, where $L = 2\cdot \mid A \mid$ if $\mid A \mid < 5$ and $L = \mid A \mid +5$ otherwise. Here $\mid A \mid$ denotes the number of symbols of A. Text analysis in the search window is based on a simplified morphological analysis where tokens are split into units called *morphemes*[3] using the background dictionary. For splitting tokens into morphemes we use the following borders: (1) transitions from letters to digits and vice versa and (2) the symbols "/", "\", ".", "(",")", "[","]", and "-". From the resulting units words with prefixes from the list

anti, bi, electro, inter, pre, sub, trans, un, hex, ex, cross, mono, di, tri, tetra, penta, hexa

are split into two parts, the prefix and the rest, if the rest is found in an English general dictionary used for this purpose. Both parts are then treated as independent morphemes. Eventually five *types of morphemes* are distinguished:

s: stop words (of, the, and, in, to, a, with, by),
p: prefixes from the aforementioned predefined list,
h: rest of a word obtained after deleting a prefix (the rest must be in the dictionary),
n: sequences of digits,
w: other words.

[2] In [19] two windows surrounding A are used.
[3] In general, these "morphemes" are not morphemes in a proper linguistic sense.

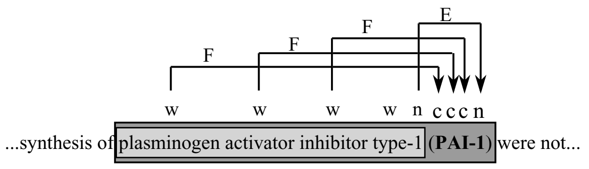

E
F
F
F
w
w
w
w
n
c c c n
...synthesis of plasminogen activator inhibitor type-1 (PAI-1) were not...

Fig. 3 Rewriting expansion "plasminogen activator inhibitor type-1" into the acronym PAI-1 using the rule $wwwwn \rightarrow [F(1){:}\ c][F(2){:}\ c]$ $[F(3){:}\ c][E(5){:}\ n]$

Each subsequence of consecutive morphemes in the search window represents an *expansion candidate*. As with acronym candidates, the sequence of morpheme types of an expansion candidate is used as a simplified *representation*.

Example 5 The morphemes of "10% transplantation rate" are "10" (type n), "trans" (type p), "plantation" (type h), "rate" (type w), which means that the representation is $nphw$.

4.3 Rule-based correspondence between acronym and expansion

Acronym construction is described as a process whereby certain rewrite operators are applied to the morphemes of the expansion. The resulting string, after an optional additional permutation, yields the acronym (cf. Fig. 3). The possible ways of deriving acronyms from expansions are described by means of a finite collection of *rules*. The left-hand side of a rule specifies the sequence of morpheme types of the expansion. The right-hand side describes the form of the acronym and encodes the graphical correspondence.

Example 6 The rule

$$wwwsw \rightarrow [F(1){:}\ c][F(2){:}\ c][F(3){:}\ c][F(5){:}\ c]$$

is applicable to any expansion E consisting of a sequence of morphemes respectively of type w, w, w, s, w (left-hand side). The first unit of the acronym, which has type c, is obtained from applying the operator F to the first morpheme of E (right-hand side, $[F(1){:}\ c]$). Operator F selects the first letter of a given morpheme. The remaining units of the acronym are the letters obtained by applying operator F respectively to the second, third, and fifth morphemes of the expansion.

Right-hand sides of rules may have subsequences of the form $[Op(j){:}\ x][Op(i){:}\ y]$, where $i < j$, which means that in the acronym the original order of symbols in the text is permuted.[4] The operators used in the original approach [19] are the following.

F: (First match) Defined as above.
I: (Inner match) This operator nondeterministically selects an inner letter of the given morpheme of type c of the expansion.

[4] In [19] the first atom of the right-hand side of a rule must be of the form $[Op(1){:}\ x]$.

Table 1 Some acronym-building rules with examples from Medline

$wwwn$	$\rightarrow$	$[F(1)\colon c][F(2)\colon c][I(2)\colon c][E(3)\colon c][E(4)\colon n]$
secretory phospholipase A2	$\rightarrow$	sPLA2
$nwwww$	$\rightarrow$	$[E(1)\colon n][E(2)\colon c][F(3)\colon c][F(4)\colon c][F(5)\colon c]$
20K human growth hormone	$\rightarrow$	20K-hGH
$wwwwww$	$\rightarrow$	$[F(4)\colon c][E(6)\colon c][F(1)\colon c][I(1)\colon c][F(2)\colon c]$
horseradish peroxidase conjugated cholera toxin B	$\rightarrow$	CB-HRP
$wwww$	$\rightarrow$	$[E(1)\colon c][F(2)\colon c][E(3)\colon c][F(4)\colon c]$
N-methyl-D-asparate	$\rightarrow$	NMDA

L: (Last match) This operator selects the last letter of a morpheme of type c.

E: (Exact match) This operator completely selects a morpheme of type n or w. In [19] the operator E may only be applied to morphemes of type n. Our extension is motivated by acronyms occurring in expansions that are copied as subacronyms into the defined acronym.

R: (Replacement) This operator replaces a morpheme by an abbreviation or placeholder, using the following list of possible replacements:

x	hex, ex, trans, cross
1	one, first, 1st
2	two, second, 2nd
...	...
9	nine, ninth, 9th
0	zero, null

Example 7 Figure 3 illustrates the rewriting operations encoded in the rule $wwwn \rightarrow [F(1)\colon c][F(2)\colon c][F(3)\colon c][E(5)\colon n]$, which is applied to the expansion "plasminogen activator inhibitor type-1" from Fig. 2 and produces the acronym PAI-1 with skeleton $cccn$.

Motivated by various acronyms that we found in the training corpus, we add a new operator:

C: (Contiguous match) This operator may only be used immediately after applying one of the operators F, I, or C. Assume that the latter application selects a symbol σ from a morpheme m of the expansion, copying it to a fixed position of the acronym. The operator may be applied to the symbol σ' following σ in m. The operator just copies σ' to the next position of the acronym. In a rule, permutations that would split the sequence $\sigma\sigma'$ in the acronym are excluded.[5]

Operator C takes into account that often a sequence of consecutive letters of an expansion morpheme is directly copied into the acronym.

Example 8 The acronym Chr-22, which is split into units C, h, r, 22, has the skeleton $cccn$. It is used for "chromosome 22," which is of the form wn (morphemes "chromosome" and "22"). Our rule describing the relationship has the form

$wn \rightarrow [F(1)\colon c][C(1)\colon c][C(1)\colon c][E(2)\colon n]$.

After F selects c, the two applications of the C-operator respectively select the letters h and r.

Table 1 gives some additional examples for rules as used within the hybrid approach for describing the derivation of acronyms from expansions.

Remark 3 In some of our experiments we also looked at another operator. The operator Ins (Insertion) inserts a unit of type n or c into the acronym that does not have a correspondence in the expansion. It enables a correspondence between an acronym and an underspecified expansion. For example, the acronym ODN, which has the skeleton ccc, is used in one particular Medline abstract as short for "oligonucleotides," which is of the form w. Our rule describing the graphical relationship for this case is as follows:

$w \rightarrow [F(1) : c][Ins : c][I(1) : c]$.

Many other Medline abstracts provide evidence of ODN usually standing for "oligodeoxynucleotides," and it is natural to assume that the letter D comes from the missing "deoxy" rather than corresponding to the letter "d" at the end of the expansion. The use of the operator Ins is controversial. On the one hand, we found a nonnegligible number of examples in the Medline corpus where Ins is in fact needed to obtain the correct graphical correspondence between acronym and expansion. On the other hand, Ins is too powerful since it can "explain" any symbol in any acronym. Hence an uncontrolled use leads to many ambiguities and decreased precision. In practice, the use of Ins has to be controlled in some suitable way. For simplicity, we describe all our experiments in a variant where the use of Ins is completely excluded. For the rule-based approach, this generally leads to better results. For the HMM-based method, an interesting variant tolerates one single application of Ins, adding linguistic conditions on expansion candidates in the preference scheme as a kind of additional control. Details are described below.

4.4 Selection of preferred expansion candidate

Given an acronym candidate A, the skeleton of A determines a unique subset M of rules where the right-hand side has the

[5] Clearly, with the original definition of I there is overlap in the uses of C and I. Thus we restrict the use of I in the rules to the cases where the use of C is not possible.

Table 2 Corpora statistics

	Candidates	Acronyms	Introduction contexts	Rules
Training (year 1999)	500	464	449	107
Test (year 2000)	500	465	436	118

appropriate form. For each rule in M, for each consecutive sequence of morphemes in the search window we check if the morphemes have the appropriate type as specified in the left-hand side of the rule. Some conditions must be considered when selecting expansion candidates in this way. All morphemes of an expansion candidate E have to belong to the same sentence of the text. The first and the last morpheme of E must not be a preposition, a form of the verb *be*, a conjunction, or a determinant. This can be regarded as a simplified recognition of noun phrase borders.[6]

Distinct expansion candidates are first sorted by distance to the acronym candidate. The candidate that is nearest to the acronym and was obtained with the most frequent rule in the training corpus is chosen as the expansion for A in the output. The preference scheme used in [19] is slightly distinct. We also tested it and other preference schemes, without improving results.

5 Evaluating convergence and stability

In the original paper [19], the hybrid approach is used in an interactive environment where the user may add new rules suggested by the system during a session. In such a context, new rules are added by demand, and recall can be controlled by the user. We are interested in methods that, after initial training, work in a fully automated way, which seems more useful when analyzing large corpora with many acronyms. A formalism is needed where training is convergent and leads to a stable procedure in the sense explained in the introduction.

In order to study convergence and stability of the hybrid approach in the area of biomedical texts, we conducted a series of experiments on the Medline corpus. As mentioned above, we focused on experiments that disregarded the operator Ins. The precision and recall measurements were performed on two corpora with sentences from the Medline database.

The *training corpus* contained sentences occurring in texts from 1999. A subcorpus with 500 acronym candidates was singled out using the regular expression Γ in parenthesized form. After inspecting the acronym candidates, we found that 464 candidates represented proper acronyms. Among those, 449 were defined in the same sentence before the acronym. Hence we had a total of 449 acronym-expansion pairs. We manually derived the rules

Table 3 Number of rules vs. rule frequency

Training		Test	
# Rules	Freq. of use	# Rules	Freq. of use
1	111	1	94
1	50	1	49
1	30	1	31
1	24	1	26
1	21	1	21
2	12	1	14
1	9	1	8
1	8	1	7
1	7	2	6
5	5	1	5
3	4	6	4
9	3	10	3
8	2	13	2
72	1	78	1

that correctly describe the graphical correspondence between acronyms and expansions and annotated all acronym-expansion pairs as well as pseudoacronyms for evaluation purposes. Table 3 shows the frequency distribution for the 107 rules that were obtained. For the training corpus, we achieved a Γ-restricted recall of 96% and a precision value of 99%.

For the *test corpus* we used a similar collection of sentences using Medline texts from 2000. We again added sentences until we obtained 500 acronym candidates. In this case the number of proper acronyms (acronym-expansion pairs) was 465 (436). In order to support an automated evaluation of precision and recall, we again annotated all acronym-expansion pairs as well as the pseudoacronyms. Using the rule set $\mathcal{R}$ derived from the training corpus, we then automatically retrieved acronym-expansion pairs from the test corpus. A Γ-restricted recall (precision) of 78% (96%) was obtained.

The loss of recall when applying the rules from the training corpus to the test corpus is considerable and raises two obvious questions:

1. Can we expect a better recall on the test corpus when deriving a larger rule set from a larger training corpus? In the negative case, what are the reasons?
2. How is precision affected when we add more rules during the training phase?

As a first step, we manually derived all rules needed for the test corpus. We found 118 rules, only 35 of which were from the rule set $\mathcal{R}$! Besides, these rules in the intersection were accompanied by a large set of rules with just one application. The high number of rules with very low frequency gave a first hint that the derivation of a sufficiently large rule set during training might be difficult.

To see in more detail how recall and precision values reached in the test corpus depend on the number of rules that are derived in the training corpus, we imitated an "idealized" training process where the most frequent rules are found first. Starting with the subset $\mathcal{R}_1$ of $\mathcal{R}$ that contains the 10 most frequent rules found on the training corpus, an

[6] In [19], the first and last morphemes of a candidate must not be prepositions, *be* verbs, modal verbs, conjunctions or pronouns. A candidate is also excluded if it contains symbols such as "(", ")", "[", "]", "{", "}", "=", "!" "?". Besides, the condition in footnote 4 is effective, and, like here, all expansion words have to belong to the same sentence.

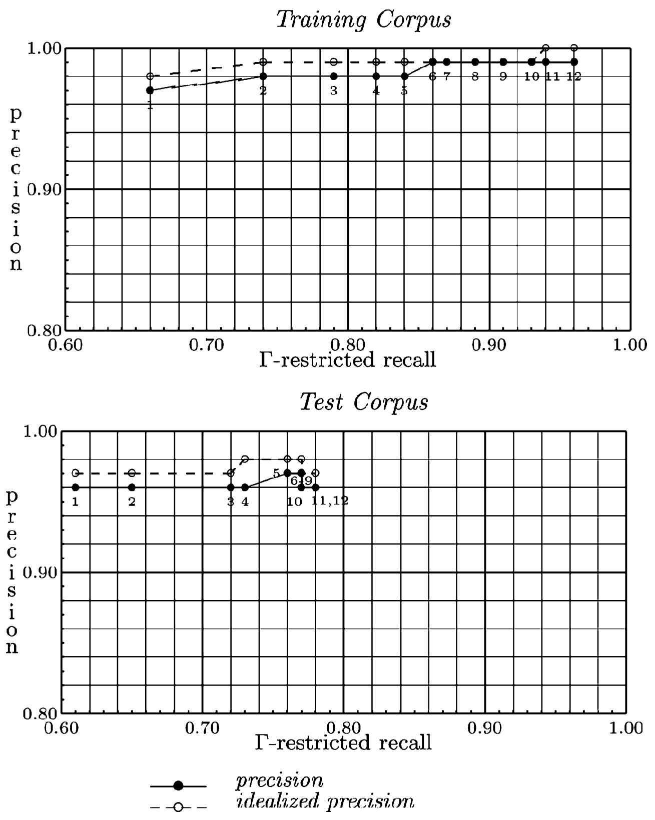


Fig. 4 Hybrid approach. Precision (*solid lines*) and Γ-restricted recall, values for rounds 1–12 on training and test corpora. The "idealized" precision (*dashed lines*) represents a hypothetical value that could be reached with a perfect recognition of introduction contexts

experiment with 12 rounds was designed. The rule set $\mathcal{R}_{i+i}$ for round $i+1$ contained $\mathcal{R}_i$ and in addition the 10 most frequent rules from $\mathcal{R} \setminus \mathcal{R}_i$. In the final round 12, the last 9 rules were added. In each round, we measured the Γ-restricted recall and precision values that were achieved for the (training and) test corpus using the rule set $\mathcal{R}_i$. The results are shown in Fig. 4 (solid lines).

For the training corpus, the addition of rules during the 12 rounds had the expected effect: we gradually improved recall, the precision was not affected. On the test corpus, however, recall – after a significant initial improvement – did not exceed a limit of 78%. The numbers in rounds 6–12 suggest that no significant gain in recall can be expected when adding new rules from a larger training corpus. Furthermore, the addition of new rules is likely to reduce precision values initially raised. Obviously, the use of the system as a tool for automated extraction becomes suboptimal with such a behavior.

For a more thorough analysis, four kinds of errors should be distinguished:[7]

1. *False positives.* A pseudoacronym is erroneously treated as an acronym and some sequence of letters from the search window is treated as its expansion. This kind of error only affects precision.
2. *Wrong rule.* For a proper acronym, the wrong rule was selected and an incorrect expansion was derived. The correct rule was available. This kind of error affects both precision and recall.
3. *Missing rule 1.* For a proper acronym, the correct rule was not in the rule set. Another rule was selected and the wrong expansion derived. This kind of error affects both precision and recall.
4. *Missing rule 2.* For a proper acronym, the correct rule was not in the rule set. No other rule was applied and no expansion derived. This kind of error only affects recall.

We investigated which improvements could be expected from a better recognition of introduction contexts. Since candidates not occurring in an introduction context were marked in the corpus, we could easily measure the "idealized" precision values where only candidates occurring in proper introduction contexts are taken into account and errors of type 1 (false positives) are thus excluded (cf. Sect. 3). The corresponding curves are represented in Fig. 4 with dashed lines. The results show that even with a perfect recognition of introduction contexts, precision values on the test corpus are only lifted by about 1% but still decrease with the number of rules.

Figure 5 shows the number of errors of types 2 and 3 and helps to explain the reasons for the difference in precision

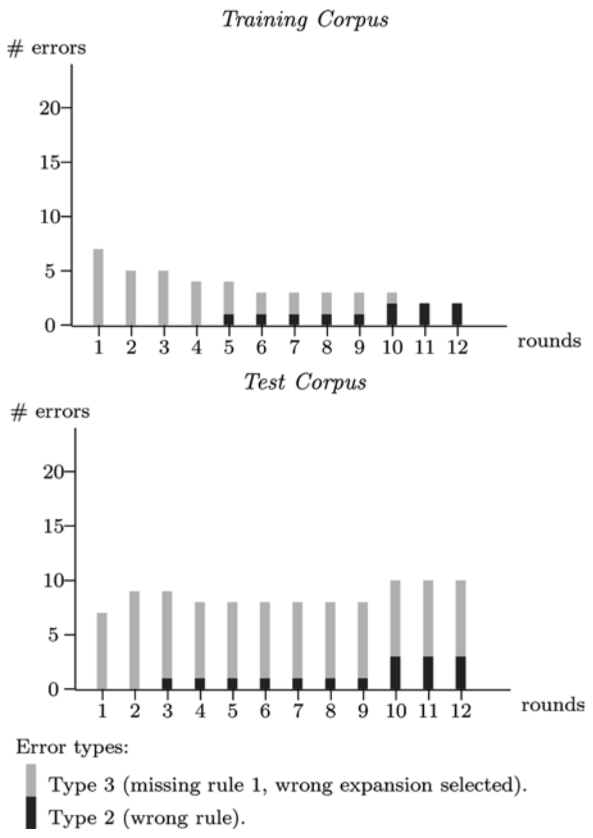


Fig. 5 Error types of extracted incorrect expansions

[7] In our experiments, another case could be neglected in which the window is too small to contain the expansion.

between training and test. The total height of each bar gives the number of *false* expansions extracted for *proper* acronyms in a round and for a corpus. When rules are added during the rounds, more proper acronyms are retrieved and found in the answer set, which explains why the number of errors of both kinds may grow. The lower part (in darker gray) corresponds to type 2 errors (proper rule available, wrong rule selected), the upper (in lighter gray) to type 3 errors (proper rule not available, wrong expansion selected). In the *training corpus*, where adequate rules are derived, errors of type 3 are eventually eliminated in round 12, where all needed rules are available. Some errors of type 2 persist where the preference schema selects the wrong rule and expansion. In the *test corpus*, in contrast, the majority of errors are of type 3. Note that most of the 107 rules that are at our disposal are in fact inappropriate since they do not yield a correct explanation for *any* example of the test corpus. The presence of these useless rules increases the probability of selecting the wrong rule, and during the 12 rounds we obtain an almost constant number of false expansions. In this sense the system is overtrained for the test corpus.

We conclude that extending the recall for a specific corpus by adding new rules can only be done at the risk of a loss of precision when moving to another corpus. Most rules only cover a few very specific cases. When adding these rules, the danger of errors of type 2 increases.

Remark 4 A parallel series of experiments were conducted allowing the use of Ins (cf. Remark 4) in the rules. Twelve additional rules could then be written for the training corpus, covering 3% of the introduction contexts. The results look very similar to those in Fig. 4. Recall is improved by 3% in the training corpus and by one point in the test corpus. The major difference is the negative evolution of precision in the test corpus, which decreases after round three (98%) and reaches a final value of 94% after round 12.

6 Replacing rule sets by Hidden Markov Models

The numbers seen in the previous section show that the high specificity of rules represents the main obstacle to obtain a fully satisfactory recall in the application phase. We now introduce the hidden Markov models (HMMs) used in our approach and show how parameters are directly estimated from an existing rule set.[8] Afterwards we show how the HMM is actually used given an acronym candidate and a search window. In a third step we show how the probabilistic information obtained from the HMM and additional parameters from the expansion candidates are combined in the preference scheme to select a preferred expansion candidate. In our experiments the remaining setup (tokenization routine, morpheme determination, definition of acronym candidates, search window, etc.) was as before.

[8] Readers not familiar with HMMs are referred to [5, 23].

6.1 Converting rule sets into HMMs

The basic idea behind the use of HMMs can be described as follows. Given an acronym candidate A, each letter of the skeleton is treated as an emission of a matching operator. Matching operators are treated as states. Our HMMs are "hidden" in the sense that the sequence of all emissions (the skeleton) is visible, but we cannot see the underlying sequence of states (matching operators). From the HMM, we obtain different suggestions for plausible operator sequences, which in a second step are compared with the morphemes found in the search window.[9] Details are given below; see also Fig. 7.

The *set of states* S of the HMM contains the operators used in the hybrid approach and two additional states, b and 0:

$$S = \{b, C, F, I, L, E, R, 0\}\,.^{10}$$

The operator b is only used for technical reasons: state b is the first state of any transition through the HMM; it is not accessible from the other states. State b always emits the null symbol ϵ. State 0 is needed because the application of operator E can result in more than one symbol in the acronym skeleton, but in our HMM there is a one-to-one correspondence between operators (states) and emitted symbols. We modified E, allowing it to produce just the first symbol. All the following symbols formerly produced by E are now produced by state 0, which is accessible only from E or from 0.

The *set of emission symbols* is

$$K := \{\epsilon, c, n\}\,.$$

The vector of *initial state probabilities* $\Pi = \{\pi_i\}, i \in S$ is defined as $\pi_b := 1$ and $\pi_i = 0$ for all states $i \in S$ distinct from b.

We estimated the *transition probabilities* $a_{i,j} (i, j \in S)$ and the *emission probabilities* b_{ik} $(i \in S, k \in K$ directly from a set of rules obtained from the hybrid approach using a maximum-likelihood estimator (MLE). As a preparation, the rules were automatically converted into a different format in a first step, in which the states b and 0 are used in the aforementioned way.

Example 9 Rule $wphww \rightarrow [F(2): c][F(4): c][E(5): ccc]$ is converted into the sequence $\epsilon b, cF, cF, cE, c0, c0$.

Each pair $k_i s_j$ represents the emission of symbol k_i from state s_j. With the MLE the emission probability for such an event is estimated from the relative frequency of events where s_j is emitted when the HMM is in state s_j:

$$P_{MLE}(o_t = k_i \mid X_t = s_j) = C(k_i s_j)/C(\_s_j)\,.$$

Here $C(k_i s_j)$ represents the absolute frequency of the event $k_i s_j$, and $\_$ indicates that the symbol is not specified for the count.

[9] Note that we do *not* calculate the most likely path using the Viterbi algorithm.

[10] Of course, Ins is included in S in the experiments that use this operator.

Emission probabilities

	ϵ	c	n
b	1.000	0.000	0.000
C	0.000	1.000	0.000
F	0.000	1.000	0.000
I	0.000	1.000	0.000
L	0.000	1.000	0.000
E	0.000	0.833	0.167
0	0.000	1.000	0.000
R	0.000	0.000	0.000

Transition probabilities, rows (columns) representing source (target) states

	F	I	C	L	E	0	R
b	1.000	0.000	0.000	0.000	0.000	0.000	0.000
F	0.753	0.166	0.040	0.020	0.020	0.000	0.000
I	0.647	0.294	0.000	0.029	0.029	0.000	0.000
C	0.556	0.111	0.333	0.000	0.000	0.000	0.000
L	0.000	0.000	0.000	0.000	0.000	0.000	0.000
E	0.000	0.000	0.000	0.000	0.000	1.000	0.000
0	1.000	0.000	0.000	0.000	0.000	0.000	0.000
R	0.000	0.000	0.000	0.000	0.000	0.000	0.000

Fig. 6 HMM obtained from the maximal rule set with 107 rules

The transition probabilities are estimated from the relative frequency of a transition with respect to all transitions starting from the same state:

$$P_{MLE}(X_{t+1} = s_k \mid X_t = s_j) = C(\_s_j, \_s_k)/C(\_s_j, \_\_)\,.$$

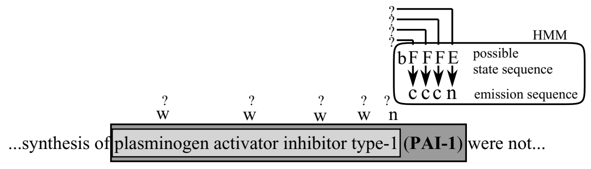
No kind of smoothing was performed that tried to reserve probability space for unseen events.

Figure 6 shows the HMM that was obtained when using the final rule set for round 12 in the experiment from the previous section for the hybrid approach. Accidentally, the operator R was not used in the rules derived for the training corpus, so all probabilities for transitions to this state are null. Nevertheless, from other experiments we know that this operator is useful. In general, in experiments with larger rule sets the probability distribution also became more complex.

Remark 5 Our HMM only formalizes the sequence of operations that are used to built an acronym and the letter type c or n of the acronym skeleton that results from the application of the operator. The HMM does *not* specify to which morpheme in the search window a given operator is applied. This point, which is illustrated in Fig. 7, will become clearer below, when we explain how the HMM is used for search.

Remark 6 An alternative, more fine-grained working model results from creating a state in the HMM for each combination of an operator with an expansion morpheme type. The training costs, however, would be much higher since the number of transitions is given by the square of the number of states.

Remark 7 The fact that we do not consider the rule use frequency in our estimations for the different parameters can be seen as a way of factoring out noise introduced by the expansion skeleton that is used in the rule. Rules can only have a high frequency if they use a frequent expansion skeleton. However, when applying the HMM we do not want to make any assumption on the expansion.



?
?
?
?
HMM
bFFFE
possible state sequence
cccn
emission sequence
? ? ? ? ?
w w w w n
...synthesis of plasminogen activator inhibitor type-1 (PAI-1) were not...

Fig. 7 Recovery of expansion "plasminogen activator inhibitor type-1" from acronym PAI-1. Given the skeleton *cccn*, the HMM suggests operator sequences such as *bFFFE* that correspond to possible paths. For each candidate path, the correspondence between the operator sequence and morphemes of expansion candidates is checked in another step (*question marks*)

6.2 Applying the HMM

The HMM acts as a network with an infinite set of possible transition paths. For each observed acronym candidate A, there are different paths through states of the HMM that can produce the skeleton of A. When looking for an expansion for A, the HMM produces alternative paths (operator sequences) based on the form of the skeleton, which we can check against the search window, selecting suitable morphemes for applying the operators. The correspondence between acronym and expansion is thus not driven by a finite set of rules but emerges dynamically.

In practice, operator sequences produced by the HMM are deterministically grouped into subsequences, each subsequence corresponding to a set of operators that are all applied to the same expansion morpheme. We use the fact that operator F marks the beginning of a new morpheme and that each of the operators E and R[11] "stands for" a whole morpheme. In an operator sequence we can draw a border in front of these operators. Sequences between borders are applied to the same morpheme.

Example 10 *EFCCRFI* is divided into the subsequences $E \mid FCC \mid R \mid FI$.

The border partition can be used to discard operator sequences that are not compatible with some structural properties of the acronym not visible in the skeleton. Dashes appearing in the acronym usually indicate morpheme divisions in the expansion. We can thus ignore operator sequences where the number of borders is smaller than the number of dashes found in the acronym.

Example 11 The operator sequence $FII \mid FI$ cannot generate the acronym P-CH-R.

Other characteristics of the acronym like transitions from lowercase to uppercase turned out to be unreliable as division markers and were not used in our experiments.

Given an acronym candidate A and a search window, operator subsequences are matched against the morphemes in the window. There is a match for the whole operator sequence if each subsequence matches at least one morpheme in the window such that all matched morphemes are different. Each possible combination of expansion morphemes

[11] As well as Ins.

that matches the operator sequence is treated as an expansion candidate. Each candidate encloses the words between the two outermost matched morphemes.

From the HMM we obtain a first parameter for the preference scheme. Consider an expansion candidate associated with an operator sequence X that generates an observed acronym skeleton O. Given the model n, the probability of producing O on path X is

$$P(O, X \mid \mu) = P(O \mid X, \mu) \cdot P(X \mid \mu) \quad (1)$$

$$= \prod_{t=1}^{T} \frac{C(o_i X_t)}{C(\_X_t)} \cdot \prod_{t=1}^{T} \frac{C(\_X_t, \_X_{t+1})}{C(\_X_t)}. \quad (2)$$

In order to be able to directly compare the probabilities of all operator sequences X' that generate the same skeleton O, we associated to each sequence X the normalized probability N:

$$N(O, X \mid \mu) = \frac{P(O, X \mid \mu)}{\sum_{X'} P(O, X' \mid \mu)}.$$

6.3 Preference scheme and graphical correspondence

As the preference schema used for the hybrid approach worked well, we developed a similar one for the HMM version. Again, expansion candidates are first ordered by ascending distance to the acronym candidate. From among all candidates with minimal distance we select the one where a second parameter, $N_{M,F}$, is maximal. The parameter

$$N_{M,F} \colon = N(O, X \mid \mu) \cdot M \cdot P$$

modifies the normalized probability $N(O, X \mid \mu)$ of the operator sequence suggested by the HMM, introducing two weakening factors, M and P. M takes into account how many expansion morphemes appear as a letter or number in the acronym. It is given by

$$M := \begin{cases} 1 & \text{if } m = 0 \\ (1 - m/\mid E \mid) \cdot \frac{1}{4} & \text{if } m > 0 \end{cases},$$

where m is the number of candidate morphemes not represented in the acronym ("missing" morphemes) and $\mid E \mid$ the total number of morphemes of the candidate. Note that M decreases with the number of missing morphemes.

P reflects how drastically the order of the morphemes in the expansion candidate differs from the order of the corresponding letters/numbers in the acronym. We used the formula

$$P := \begin{cases} 1 & \text{if } lcs = \mid A \mid \\ (lcs/\mid A \mid) \cdot \frac{1}{8} & \text{if } lcs < \mid A \mid \end{cases},$$

where lcs denotes the longest subsequence of nonpermuted positions in the graphical correspondence between expansion morphemes and acronym symbols.

For example, for the graphical correspondence expressed by a rule with right-hand side $[F(2)\colon c][F(3)\colon c][F(4)\colon c]$ $[F(1)\colon c]$, we have $lcs = 3$ since in the subsequence $[F(2)\colon c][F(3) : c][F(4)\colon c]$ the order of expansion morphemes is completely copied to the corresponding order of acronym letters.

Motivated by linguistic studies on the typical form of terminological expressions [4, 11], we tried to refine the preference scheme with linguistic filters that help to recognize good expansion candidates. In the above variant of the HMM approach, these filters did not improve results. In the variant where HMMs use the operator Ins, the use of linguistic filters in the preference scheme helped to improve precision (see below).

7 Evaluation results for the HMM-based method

To evaluate the HMM-based method, we repeated the experiments designed for the hybrid approach using the same training and test corpora. The effect of training the model on different rule sets was observed, translating the rule sets obtained after the 12 rounds into 12 corresponding HMMs.

Results are given in Fig. 8. Already after the first round the recall of 82% reached with the translated HMM was better than the highest recall reached with the hybrid method

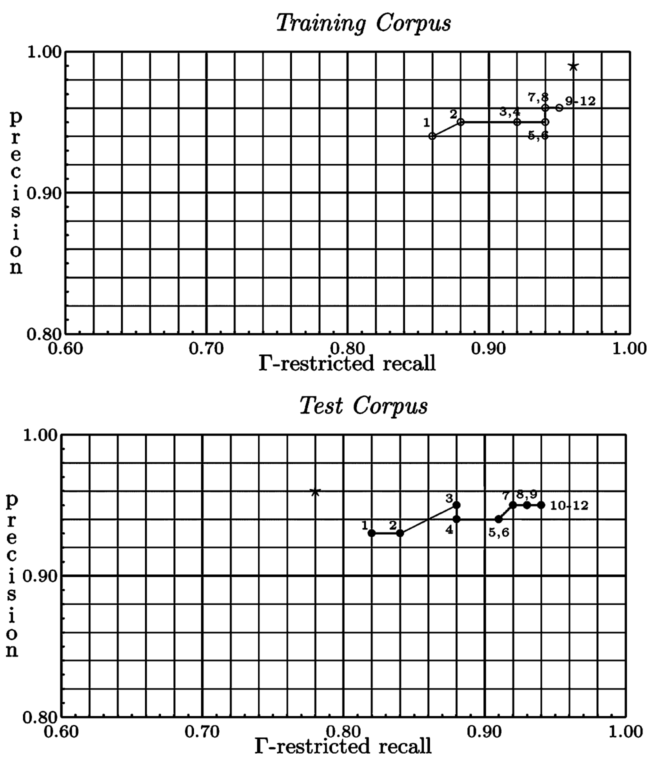


Fig. 8 Precision and Γ-restricted recall for the HMM-based method. The rule sets used for generating a HMM in each round correspond to the rule sets used for the rounds in the hybrid approach. The *star* marks the best result achieved with the hybrid method

after round 12. In the final round, the recall reached with the HMM-based method was 94%. In terms of precision, the HMM-based method achieves 95% and is comparable to the hybrid approach (96%). Summing up, the main advantage of the HMM-based method is the improved recall coupled with stability with respect to precision.

HMMs with Ins and linguistic filtering of expansion candidates. In the parallel experiment where we used HMMs with state Ins, we only considered paths with at most one occurrence of Ins. We obtained a better recall, and the improvement was 1%. At the same time, precision decreased by 4–5%. With a sophisticated linguistic preference scheme, this loss of precision could be reduced.

A final remark addresses efficiency. Our algorithm first computes for a given acronym skeleton O the normalized probability $N(O, X \mid \mu)$ for all paths X. In a second step, for each such operator sequence X we try to find admissible matches on the basis of the morphemes found in the search window. This matching step consumes most of the time. Hence a speedup can be obtained by excluding from the matching all paths X where the normalized probability is below a given threshold. We found that we could safely ignore all paths X with a normalized probability below $1/10 \cdot l$, where l is the number of acronym letters, without major changes for precision and recall.

8 Related work

An early and influential contribution to acronym-expansion detection is Taghva's *Acronym Finding Program* (AFP) [25, 26]. Acronym candidates are defined as "uppercased" words of length l, $3 \leq l \leq 10$, excluding sequences from a given list of false positives (e.g., FIGURE, TABLE). Any occurrence of a candidate A triggers a search for the associated expansion within a text window surrounding A with two parts, the pre- and the postwindow. Each subwindow contains $2 \mid A \mid$ consecutive words ($\mid A \mid$ denotes the number of symbols of A). For each subwindow W a confidence level $cf(W, A) := \frac{lcs(I(W), A)}{|A|} +$ error - rate is computed. Here $I(W)$ denotes the sequence of all initial letters of W, $lcs(A', A)$ denotes the length of a maximal common subsequence of two sequences A' and A, and *error rate* is a configurable parameter with default value 0.2. If $cf(W, A) < 1$, the search for an expansion of A in W stops. Otherwise, all longest common subsequences of A and $I(W)$ are computed. Each longest subsequence defines a unique subsequence of (not necessarily consecutive) words of W. Filling the holes between these words (i.e., adding skipped words) we obtain an expansion candidate. The preference scheme for selecting a *best* expansion candidate basically tries to maximize the number of content words in an expansion that contribute to the acronym.

The main advantage of the model is its simplicity and computational efficiency. Drawbacks are the simplicity of the search pattern and the fact that only matches corresponding to the operator Γ described above are formalized. Other possibilities are only taken into account by accepting suboptimal matches between acronym and expansion, which leads to a loss of precision.

Acrophile [12] is a dictionary of acronym definitions extracted from documents on the Web. For the construction of the dictionary, four acronym-finding algorithms have been designed and evaluated. While the base algorithm uses conventional search in arbitrary text windows surrounding acronym candidates, three refined versions take the structure of possible introduction contexts into account. In contrast to AFP, detailed descriptions of possible *acronym candidates* in terms of regular patterns and additional restrictions are given. Each of the four Acrophile algorithms comes with its own definition of expansion candidates. Generalizing the acronym-building principle of AFP, several letters of one word may contribute to a given acronym in the Acrophile approach. Matching basically means checking different combinations of F and I operator sequences on the search window. In this sense, the hybrid approach [19] refines both AFP and Acrophile with its richer set of matching operations. Acrophile's four algorithms use distinct preference schemes for selecting a best candidate. For the refined variants, occurrences in admissible introduction contexts are preferred.

A different approach relying on a deeper linguistic analysis of the text is reported in [21]. In the more improved version, introduction contexts are described as a nominal phrase standing for the expansion and an acronym candidate determined by a regular expression, which can appear in different configurations, e.g., *NP (acronym-candidate)* or *acronym-candidate (NP)*. For processing, the text is split into sentences. Each sentence is analyzed by a shallow parser, which marks chunks of words in the sentence as phrases of various types. A finite-state automaton scans the outputted sentence for the patterns defined for the introductory contexts. If one is found, a simple matching procedure is used to validate the NP and the acronym candidate. They are considered an expansion-acronym pair if the ratio of the matchable expansion words (not including stop words) to the number of acronym characters remains below a specified threshold. The method is very accurate (99% for some introduction context patterns) with an acceptable recall (60%–70%), but the resource overhead (parsing procedure, corpus-dependent lexica) is considerable.

Other methods are designed as classification tasks. Yeates [29] uses machine learning techniques to develop a naive Bayes classifier that distinguishes acronyms from nonacronyms. In [28] acronyms are detected by comparing the performance of a special compression model developed for acronyms with the performance achieved with other standard compression models for natural language.

9 Conclusion

In this paper we developed an extension of a rule-based approach for extraction of acronym-expansion pairs, translating a given set of extraction rules into a hidden Markov model and using a preference schema based on properties of the graphical correspondence between acronym and expansion.

In our experiments, a maximal recall of 78% reached on the test corpus with the original approach [19] could be lifted to 94% with the HMM-based extension of the formalism, while the precision remained stable.

The HMM formalism is stable in the sense that a given HMM can be used for new corpora. The flexibility of the matching process and the richness of the used set of operators guarantee that special graphical relationships can be detected that have not been observed in the training corpus. As a matter of fact, it is possible to (re)translate a successful run of the HMM for some acronym-expansion pair into a rule. In this way, the given HMM can also be used for suggesting rules. The translation of enlarged rule sets into HMMs can then be considered as a form of additional training.

A hint is provided on the usefulness of adding linguistic techniques for recognizing introduction contexts when the graphical correspondence is only approximate and the operator Ins (cf. Remark 4.7) is used.

One actual weakness of our approach is efficiency. The process whereby an operator sequence produced by the HMM is matched with the morphemes in the search window is complex. Since the HMM produces several sequences, search times are longer than for the original approach. Improvements on the efficiency side represent one point for future research.

A final remark addresses the difference between Γ-restricted recall (as defined in Sect. 3 and measured above) on the one hand and absolute recall on the other. The search pattern "(Γ)" for detecting acronym candidates described in Sect. 4 simplified the above experiments where we compared the original method with our extension. When the new approach is used in practice, absolute recall can be further improved using a more general set of patterns. In an experiment with Pustejovsky's *Acronym/Alias Identification Corpus* [14] we found that 123 of 149 acronyms occurring in actual introduction contexts could be identified using the pattern "(Γ)." The method gives a detection rate of 82.5%. The 149 introduction contexts had the following form and distribution:

139	*Expansion* (*Acronym*)
4	*Acronym* (for *Expansion*)
2	*Expansion* [*Acronym*]
2	*Acronym* (*Expansion*)
1	*Expansion-prefix* (*Acronym-for-prefix*) *Exp.-rest* (*Acronym-for-whole-exp.*)
1	*Expansion* (*Acronym*) *Expansion-rest*

Hence, for 16 missing acronyms a more general regular expression Γ was needed. Most of the remaining 10 cases can be covered with a straightforward modification of the search pattern "(Γ)."

Acknowledgements The authors thank the anonymous referees of the *International Journal of Document Analysis and Recognition* for remarks that helped to improve the presentation of the paper.

References

1. Andrade, M.A., Valencia, A.: Automatic extraction of keywords from scientific text: application to the knowledge domain of protein families. Bioinformatics **14**(7), 600–607 (1998)
2. Boguraev, B., Kennedy, C.: Applications of term identification terminology: domain description and content characterization. Nat. Lang. Eng. **5**(1), 17–44 (1999)
3. Basili, R., Moschitti, A.: Intelligent NLP-driven text classification. Int. J. Artif. Intell. Tools **11**(3), 389–423 (2002)
4. Teresa, C. M.: Terminology: Theory, Methods and Applications. John Benjamins John Benjamins Publishing Company, Amsterdam (1998)
5. Charniak, E.: Statistical Language Learning. MIT Press, Cambridge, MA (1993)
6. Cohen, J.D.: Highlights: language and domain independent automatic indexing terms for abstracting. J. Am. Soc. Inf. Sci. **46**(3), 162–174 (1995)
7. Dagan, I., Church, K.W.: Termight: identifying and translating technical terminology. In: Proceedings of the 7th Conference of the European Chapter of the Association for Computational Linguistics (EACL'95), pp. 34–40 (1995)
8. Fung, P., McKeown, K.: A technical word and term translation aid using noisy parallel corpora across language groups. Mach. Transl. J. (Special Issue on New Tools for Human Translators) pp. 53–87 (1996)
9. Gaizauskas, R., Demetriou, G., Humphreys, K.: Term recognition in biological science journal articles. In: Proceedings of the Workshop on Computational Terminology for Medical and Biological Applications and 2nd International Conference on Natural Language Processing (NLP-2000), pp. 37–44 Patras, Greece, (2000)
10. Hirschman, L., Park, J.C., Tsuji, J., Wong, L., Wu, C.H.: Accomplishments and challenges in literature data mining for biology. Bioinformatics **18**(12), 1553–1561 (2002)
11. Justeson, J.S., Katz, S.M.: Technical terminology: some linguistic properties and an algorithm for identification in text. Nat. Lang. Eng. **1**(1), 9–27 (1995)
12. Larkey, L.S., Ogilvie, P., Price, M.A., Tamilio, B.: Acrophile: an automated acronym extractor and server. In: Proceedings of the 5th ACM International Conference on Digital Libraries (2000)
13. Lehnert, W., Soderland, S., Aronow, D., Feng, F.: Inductive text classification for medical applications. J. Exp. Theor. Artif. Intell. **7**(1), 49–80 (1995)
14. Acronym/alias identification corpus of Brandeis University. http://www.medstract.org/gold-standards.html/ (2003)
15. Medline—Searchable with PubMed. http://www. ncbi.nlm.nih.gov/PubMed/. Service by the U.S. National Library of Medicine (2003)
16. Mikheev, A.: Periods, capitalized word, etc. Comput. Linguist. **28**(3), 289–318 (2002)
17. Nenadić, G., Spasić, I., Ananiadou, S.: Automatic acronym acquisition and term variation management within domain-specific texts. In: Proceedings of the 3rd International Conference on Language Resources and Evaluation, vol. VI, pp. 2155–2162. European Language Resources Association (2002)
18. U.S. National Library of Medicine: Fact sheet Medline. http://www.nlm.nih.gov/pubs/factsheets/medline.html (2002)
19. Park, Y., Byrd, R.J.: Hybrid text mining for finding abbreviations and their definitions. In: Conference on Empirical Methods in Natural Language Processing (EMNLP). http://citeseer.nj.nec.com/444674.html (2001)
20. Park, Y., Byrd, R.J., Boguraev, B.K.: Automatic glossary extraction: beyond terminology identification. In: Proceedings of COLING'02 (2002)
21. Pustejovsky, J., Castaño, J., Cochran, B., Kotecki, M., Morrell, M., Rumshisky, A.: Linguistic knowledge extraction from medline: automatic construction of an acronym database. Updated version of a paper presented at Medinfo. http://medstract.org/publications.html (2001)

22. Paice, C.D., Jones, P.A.: The identification of important concepts in highly structured technical papers. In: Proceedings of the 16th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval, pp. 69–78 (1993)
23. Rabiner, L.R.: A tutorial on Hidden Markov Models and selected applications in speech recognition. Proc. IEEE **77**(2), 257–286 (1989)
24. Swanson, D.R.: Medical literature as a potential source of new knowledge. Bull. Med. Libr. Assoc. **78**(1), 29–37 (1990)
25. Taghva, K., Gilbreth, J.: Recognizing acronyms and their definitions. Technical Report 95-03, ISRI Information Science Research Institute. University of Nevada, Las Vegas. (1995)
26. Taghva, K., Gilbreth, J.: Recognizing acronyms and their definitions. Int. J. Doc. Anal. Recog. **1**(4), 191–198 (1999)
27. Wright, S.E., Budin, G. (eds.): Handbook of Terminology Management, vol. 1, Basic Concepts of Terminology Management. John Benjamins, Amsterdam (1997)
28. Yeates, S., Bainbridge, D., Witten, I.H.: Using compression to identify acronyms in text. In: Conference on Data Compression, pp. 582 (2000)
29. Yeates, S.: Automatic extraction of acronyms from text. In: New Zealand Computer Science Research Students' Conference, pp. 117–124 (1999)