

Creating an Online Dictionary of Abbreviations from MEDLINE

Jeffrey T. Chang
Stanford Biomedical Informatics
jchang@smi.stanford.edu

Hinrich Schütze, Ph.D.
Novation Biosciences
hinrich@novationbio.com

Russ B. Altman, M.D., Ph.D. ¹
Stanford Biomedical Informatics
Stanford School of Medicine
Medical School Office Building, X-215
251 Campus Dr.
Stanford, CA 94305
(650) 725-3394
russ.altman@stanford.edu

¹To whom correspondence should be addressed

Abstract

The immense volume and rapid growth of biomedical literature present special challenges for humans as well as computer programs analyzing it. One such challenge comes from the common use of abbreviations that effectively augments the size of the vocabulary for the field. To cope with this, we have developed an algorithm to identify abbreviations in text. It uses a statistical learning algorithm logistic regression to score abbreviations based on their resemblance to previously identified ones, achieving up to 84% recall at 81% precision. We then scanned all of MEDLINE and found 781,632 high-scoring abbreviation definitions. We are making these available as a public abbreviation server at <http://abbreviation.stanford.edu/>.

1 Introduction

The amount of literature in biomedicine is exploding as MEDLINE grows by 400,000 citations each year. With biomedical knowledge expanding so quickly, professionals must acquire new strategies to cope with it. To alleviate this, the biomedical informatics community is investigating methods to organize ([1]), summarize ([2]), and mine ([3]) the literature.

Understanding biomedical literature is particularly challenging due to its expanding vocabulary, including the unfettered introduction of new abbreviations. An automatic method to define abbreviations would help researchers by providing a self-updating abbreviation dictionary and also facilitate computer analysis of text.

In this paper, we define abbreviation broadly to include all strings that are shortened forms of sequences of words (its long form). Although the term acronym appears more commonly in literature, it is typically defined more strictly as a conjunction of the initial letter of words; some authors also require them to be pronounceable.

Using such a strict definition excludes many types of abbreviations that appear in biomedical literature. Authors create abbreviations in many different ways as summarized here:

Abb.	Definition	Description
VDR	⇒ vitamin D receptor	The letters align to the beginnings of the words.
PTU	⇒ propylthiouracil	The letters align to a subset of syllable boundaries.
JNK	⇒ c-Jun N-terminal kinase	The letters align to punctuation boundaries.
IFN	⇒ interferon	The letters align to some other place.
SULT	⇒ sulfotransferase	The abbreviation contains contiguous characters from a word.
ATL	⇒ adult T-cell leukemia	The long form contains words not in the abbreviation.
CREB-1	⇒ CRE binding protein	The abbreviation contains letters not in the long form.
beta-EP	⇒ beta-endorphin	The abbreviation contains complete words.

Nevertheless, the numerous lists of abbreviations covering many domains attest to broad interest in identifying them. Opau, a web portal for abbreviations, contains links to 152 lists

alone ([4]). Some are compiled by individuals or groups ([5], [6]). Others accept submissions from users over the internet ([7], [8]). For the medical domain, a manually-collected published dictionary contains over 10,000 entries ([9]).

Because of the biomedical literature's size and growth, manual compilations of abbreviations suffer from problems of completeness and timeliness. Automated methods for finding abbreviations are therefore of great potential value. In general, these methods scan text for candidate abbreviations and then apply an algorithm to match them with the surrounding text. Most abbreviation finders fall into one of three types.

The simplest type of algorithm matches an abbreviation's letters to the initial letters of the words around it. The algorithm for recognizing this is relatively straightforward, although it must perform some special processing to ignore common words. Taghva gives an example `Office of Nuclear Waste Isolation (ONWR)` where the `O` can be matched with the initial letter of either `Office` or `of` ([10]).

More complex methods relax the first letter requirement and allow matches to other characters. These typically use heuristics to favor matches on the first letter or syllable boundaries, upper case letters, length of acronym, etc. ([11]) However, Yeates notes the challenge in finding optimal weights for each heuristic and further posits that machine learning approaches may help ([12]).

Another approach recognizes that the alignment between an abbreviation and its long form often follows a set of patterns ([13], [14]). Thus, a set of carefully and manually crafted rules governing allowed patterns can recognize abbreviations. Furthermore, one can control the performance of the system by adjusting the set of rules, trading off between the leniency in which a rule allows matches and the number of errors that it introduces.

In their rule-based system, Pustejovsky et al. introduced an interesting innovation by

including lexical information ([14]). Their insight is that abbreviations are often composed from noun phrases, and that constraining the search to definitions in the noun phrases closest to the abbreviation will improve precision. With the search constrained, they found that they could further tune their rules to also improve recall.

Finally, there is one completely different approach to abbreviation search based on compression ([15]). The idea here is that a correct abbreviation gives better clues to the best compression model for the surrounding text than an incorrect one. Thus, a normalized compression ratio built from the abbreviation gives a score capable of distinguishing abbreviations.

In this paper, we present two contributions: a novel algorithm for identifying abbreviations, and a publically-accessible abbreviation server containing all abbreviation definitions found in MEDLINE.

2 Methods

We decompose the abbreviation-finding problem into four components: 1) scanning text for occurrences of possible abbreviations, 2) aligning the candidates to the preceding text, 3) converting the abbreviations and alignments into a feature vector, and 4) scoring the feature vector using a statistical machine learning algorithm (Figure 1).

2.1 Finding Abbreviation Candidates

We searched for possible abbreviations inside parentheses, assuming that they followed the pattern:

long form (abbreviation)

For every pair of parentheses, we retrieved the words up to a comma or semicolon. We rejected candidates longer than two words, candidates without any letters, and candidates that exactly matched the words in the preceding text.

For each abbreviation candidate, we saved the words before the open parenthesis (the prefix) so that we could search them for the abbreviation's long form. Although we could have included every word from the beginning of the sentence, as a computational optimization, we only used $3N$ words, where N was the number of letters in the abbreviation. We chose this limit conservatively based on an informal observation that we always found long forms well within $3N$ words.

2.2 Aligning Abbreviations with their Prefixes

For each pair of abbreviation candidate and prefix, we found the alignment of the letters in the abbreviation with those in the prefix. This is a case of the Longest Common Substring (LCS) problem studied in computer science and adapted for biological sequence alignment in bioinformatics ([16]).

We found the optimal alignment between two strings X and Y using dynamic programming in $O(NM)$ time, where N and M were the lengths of the strings. This algorithm is expressed as a recurrence relation:

$$M[i, j] = \begin{cases} 0 & : i = 0 \text{ or } j = 0 \\ M[i - 1, j - 1] + 1 & : i, j > 0 \text{ and } X_i = Y_j \\ \max(M[i, j - 1], M[i - 1, j]) & : i, j > 0 \text{ and } X_i \neq Y_j \end{cases} \quad (1)$$

M is a score matrix, and $M[i, j]$ contains the total number of characters aligned between the

substrings $X_{1\dots i}$ and $Y_{1\dots j}$. To recover the aligned characters, we created a traceback parallel to the score matrix. This matrix stored pointers to the indexes preceding $M[i, j]$. After generating these two matrices, we recovered the alignment by following the pointers in the traceback matrix.

2.3 Computing Features from Alignments

Next, we calculated feature vectors that quantitatively described each candidate abbreviation and the alignment to its prefix. For the abbreviation recognition task, we used 9 features described in Table 1. Each feature constituted one dimension of a 9-dimension feature vector.

2.4 Scoring Alignments with Logistic Regression

Finally, we used a supervised machine learning algorithm to recognize abbreviations. To train this algorithm, we created a training set of 1000 randomly-chosen candidates identified from a set of MEDLINE abstracts pertaining to human genes, which we had compiled for another purpose. For the 93 real abbreviations, we hand-annotated the alignment between the abbreviation and prefix.

Next, we generated all possible alignments between the abbreviations and prefixes in our set of 1000. This yielded our complete training set, which consisted of 1) alignments of incorrect abbreviations, 2) correct alignments of correct abbreviations, and 3) incorrect alignments of correct abbreviations. We converted these alignments into feature vectors.

Using these feature vectors, we trained a binary logistic regression classifier ([17]). We chose this classifier based on its lack of assumptions on the data model, ability to handle continuous data, speed in classification, and probabilistically interpretable scores.

Binary logistic regression fits the feature vectors into a log odds (logit) function:

$$\log \frac{p}{1-p} = \beta X \quad (2)$$

with some manipulation:

$$p = \frac{e^{\beta X}}{1 + e^{\beta X}} \quad (3)$$

where p is the probability of seeing an abbreviation, X is the feature vector, and β is the vector of weights. Thus, training this model consists of finding the β vector that maximizes the difference between the positive and negative examples.

We found the optimal β by maximizing the likelihood over all the training examples using:

$$\ell(\beta) = \sum_{i=1}^n (y_i \log p_i + (1 - y_i) \log (1 - p_i)) \quad (4)$$

where y_i is 1 if training example i was a real abbreviation and 0 otherwise.

Since there is no known closed form solution to this equation, we used Newton's method to optimize this equation to a global maximum. We initialize β to zeros, and although this is not guaranteed to converge, it usually does in practice ([17]). To alleviate singularity problems, we removed all the duplicate vectors from the training set.

Finally, an alignment's score is the probability calculated from Equation 3 using the optimal β vector. An abbreviation's score is the maximum score of all the alignments to its prefix.

2.5 Evaluation

We evaluated our algorithm against the Medstract acronym gold standard ([14]). This contains MEDLINE abstracts with expert-annotated abbreviations and forms the basis of the evaluation of

Acromed. The gold standard is publically available as an XML file at <http://www.medstract.org/gold-standards.html>.

We ran our algorithm against the Medstract gold standard (after correcting 6 typographical errors in the XML file) and generated a list of the predicted abbreviations, definitions, and their scores. With these predictions, we calculated the recall and precision at every possible score cutoff generating a recall/precision curve. Recall

$$\frac{\text{\# correct abbreviations}}{\text{all correct abbreviations}} \quad (5)$$

measures how thoroughly the method finds all the abbreviations. Precision

$$\frac{\text{\# correct abbreviations}}{\text{all predictions}} \quad (6)$$

indicates the number of errors produced.

We counted an abbreviation/long form pair correct if it matched the gold standard exactly, considering only the highest scoring pair for each abbreviation. To be consistent with Acromed's evaluation on Medstract, we allowed mismatches in 10 cases where the long form contained words not indicated in the abbreviation. For example, we accepted `protein kinase A` for PKA and did not require the full `cAMP-dependent protein kinase A` indicated in the gold standard.

In addition, we evaluated the coverage of the database against a list of abbreviations from the China Medical Tribune, a weekly Chinese language newspaper covering medical news from Chinese journals [18]. The web site includes a dictionary of 452 commonly used English medical abbreviations with their long forms. We searched the database for these abbreviations (after correcting 21 spelling errors) and calculated the recall as

$$\frac{\# \text{ long forms identified}}{\# \text{ abbreviations (= 452)}} \quad (7)$$

2.6 Scanning MEDLINE

We searched MEDLINE abstracts up to the end of the year 2001 for abbreviations and kept all predictions that scored at least 0.001. We then put those predictions into a relational database and built an abbreviation server, a web server that, given queries by abbreviation or word, returns abbreviations and their definitions. The server can also search for abbreviations in text provided by the user (Figure 2).

We implemented the code in Python 2.2 ([19]) and C with the Biopython 1.00a4 and mxTextTools 2.0.3 libraries. The website was built with RedHat Linux 7.2, MySQL 3.23.46, and Zope 2.5.0 on a Dell workstation with a 1.5GHz Pentium IV and 512Mb of RAM.

3 Results

We ran our algorithm against the Medstract gold standard and calculated the recall and precision at various score cutoffs (Figure 3). Identifying 140 out of 168 correctly, it obtained a maximum recall of 83% at 80% precision (Table 2). The recall/precision curve plateaued at two levels of precision, 97% at 22% recall (score=0.88) and 95% at 75% recall (score=0.14).

At a score cutoff of 0.14, the algorithm made 8 errors. 7 of those errors are abbreviations missing from the gold standard: primary ethylene response element (PERE), basic helix-loop-helix (bHLH), intermediate neuroblasts defective (ind), Ca²⁺-sensing receptor (CaSR), GABA(B) receptor (GABA(B)R1), polymerase II (Pol II), and GABAB receptor (GABA(B)R2). The final error

occurred where an unfortunate sequence of words in the prefix yielded a higher scoring alignment than the long form: Fas and Fas ligand (FasL).

Then, we scanned all MEDLINE abstracts until the end of 2001 for abbreviations. This required 70 hours of computation using 5 processors on a Sun Enterprise E3500 running Solaris 2.6. In all, we processed 6,426,981 MEDLINE abstracts (only about half the 11,447,996 citations had abstracts) at an average rate of 25.5 abstracts/second.

From this scan, we identified a total of 1,948,246 abbreviations from MEDLINE, and 20.7% of them were defined in more than one abstract. Only 2.7% were found in 5 or more abstracts. 2,748,848 (42.8%) of the abstracts defined at least 1 abbreviation and 23.7% of them defined 2 or more.

Out of the nearly two million abbreviation/definition pairs, there were only 719,813 distinct abbreviations because many of them had different definitions e.g. AR can be autosomal recessive, androgen receptor, amphiregulin, aortic regurgitation, aldose reductase, etc. 156,202 (21.7%) abbreviations had more than one definition.

781,632 of the abbreviations have a score of at least 0.14. Of those, 328,874 (42.1%) are acronyms, i.e. they are composed of the first letters of words.

The growth rate of both abstracts in MEDLINE and new abbreviation definitions is increasing (Figure 4). 64,262 new abbreviations were introduced last year, and there is an average of 1 new abbreviation in every 5-10 abstracts.

We searched for abbreviations from the China Medical Tribune against our database of all MEDLINE abbreviations. Allowing differences in capitalization and punctuation, we matched 399 of the 452 abbreviations to their correct long forms for a maximum recall of 88% (Figure 5). Using a score cutoff of 0.14 yields a recall of $\frac{395}{452} = 87\%$.

Out of the 53 abbreviations missed, 11 of them were in the database as a close variation,

such as Elective Repeat Caesarean-Section instead of Elective Repeat C-Section. Also, the algorithm could identify all but 8 of the 53 with a score cutoff of 0.14.

4 Discussion

With the enormous number of abbreviations currently in MEDLINE and the rate at which prolific authors define new ones, maintaining a current dictionary of abbreviation definitions clearly requires automated methods. Since nearly half of MEDLINE abstracts contains abbreviations, computer programs analyzing this text will frequently encounter them and can benefit from their identification. Since less than half of all abbreviations are formed from the initial letters of words, automated methods must handle more sophisticated and non-standard constructs.

Thus, we used machine learning to create a method robust to varied abbreviating patterns. We evaluated it against the Medstract gold standard because it was easily available, it eliminated the need to develop an alternate standard, and it provided a reference point to compare methods.

The majority of the errors on this data set (see Table 2) occurred because the gold standard included synonyms, words and phrases with identical meanings. In these cases, the algorithm could not find the correspondences between letters, indicating a fundamental limitation of letter matching techniques.

The largest remaining source of error was from our strong assumption that the abbreviation must be inside parentheses and the long form outside. The algorithm missed 7 abbreviations where the abbreviation immediately preceded the long form, which was inside parentheses. To handle this case, the candidate finder should also allow this pattern. However, it is unclear how adding more candidates may impact the precision.

Our precision in this evaluation was hurt by abbreviations missing from the gold standard.

Our algorithm identified 8 of these, and 7 had scores higher than 0.14. Disregarding these cases yields a precision of 99% at 75% recall, which is comparable to Acromed at 98% and 72%.

Believing the algorithm to have sufficient performance, we ran it against all of MEDLINE and put the results in a database as an abbreviation server. During validation, we found that the server contained 88% of the abbreviations from the dictionary in the China Medical Tribune. Since this list was created independently of MEDLINE, the results suggest that this server contains nearly all biomedical abbreviations.

We note that using the abbreviation server to look up definitions must be done carefully. Since about a fifth of all abbreviations were degenerate, the correct one must be disambiguated using the abbreviation's context. Pustejovsky has done some work showing the suitability of the vector-space model for this task [14].

We are making the abbreviation server available at <http://abbreviation.stanford.edu/>. This server contains all the abbreviations in MEDLINE and also includes an interface that will identify abbreviations from user-specified text. We will be using the algorithm in support of text mining on the biomedical literature by identifying abbreviations defined in text and by looking up abbreviations against the abbreviation server. We hope that this server will also be useful for the general biomedical community.

5 Acknowledgements

This work was supported by NIH LM06244 and GM61374, NSF DBI-9600637, and a grant from the Burroughs-Wellcome Foundation. JTC was partially supported by a Stanford Graduate Fellowship. We thank Zhen Lin for invaluable help with translating the China Medical Tribune web site and Elmer Bernstam for thoughtful comments on earlier drafts of this paper. Finally, we thank James Pustejovsky and José Castaño for helpful discussions about Acromed and the Medstract gold standard.

References

- [1] Iliopoulos I, Enright A, Ouzounis C. Textquest: document clustering of medline abstracts for concept discovery in molecular biology. *Pac Symp Biocomput* 2001;:384–395.
- [2] Andrade M, Valencia A. Automatic annotation for biological sequences by extraction of keywords from medline abstracts. development of a prototype system. *Proc Int Conf Intell Syst Mol Biol* 1997;5:25–32.
- [3] Jenssen T, Laegreid A, Komorowski J, Hovig E. A literature network of human genes for high-throughput analysis of gene expression. *Nat Genet* 2001;28 (1):21–28.
- [4] Opau guide to lists of acronyms, abbreviations, and initialisms on the world wide web. <http://www.opau.com/acro.html>.
- [5] Acronyms and initialisms for health information resources. <http://www.geocities.com/~mlshams/acronym/acr.htm>.
- [6] Human genome acronym list. <http://www.ornl.gov/hgmis/acronym.html>.
- [7] Acronym finder. <http://www.acronymfinder.com/>.
- [8] The great three-letter abbreviation hunt. <http://www.atomiser.demon.co.uk/abbrev/>.
- [9] Jablonski S, editor. *Dictionary of Medical Acronyms & Abbreviations*. Hanley & Belfus, 1998.
- [10] Taghva K, Gilbreth J. Recognizing acronyms and their definitions. Technical report, ISRI (Information Science Research Institute) UNLV, 1995.

- [11] Yoshida M, Fukuda K, Takagi T. Pnad-css: a workbench for constructing a protein name abbreviation dictionary. *Bioinformatics* 2000;16:169–75.
- [12] Yeates S. Automatic extraction of acronyms from text. In: *New Zealand Computer Science Research Students' Conference*. 1999; pp. 117–124, pp. 117–124.
- [13] Larkey LS, Ogilvie P, Price MA, Tamilio B. Acrophile: an automated acronym extractor and server. In: *ACM DL*. 2000; pp. 205–214, pp. 205–214.
- [14] Pustejovsky J, Castaño J, Cochran B, Kotecki M, Morrell M. Automatic extraction of acronym–meaning pairs from medline databases. *Medinfo* 2001;10 (Pt 1):371–375.
- [15] Yeates S, Bainbridge D, Witten IH. Using compression to identify acronyms in text. In: *Data Compression Conference*. 2000; p. 582, p. 582.
- [16] Needleman S, Wunsch C. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J Mol Biol* 1970;48 (3):443–453.
- [17] Hastie T, Tibshirani R, Friedman J. *The Elements of Statistical Learning*. New York: Springer-Verlag, 2001.
- [18] China medical tribune. <http://www.cmt.com.cn/>.
- [19] Lutz M, Ascher D, Willison F. *Learning Python*. Sebastopol, CA: O'Reilly, 1999.
- [20] Knuth D. *The TeXbook*. Reading, Massachusetts: Addison-Wesley, 1986.

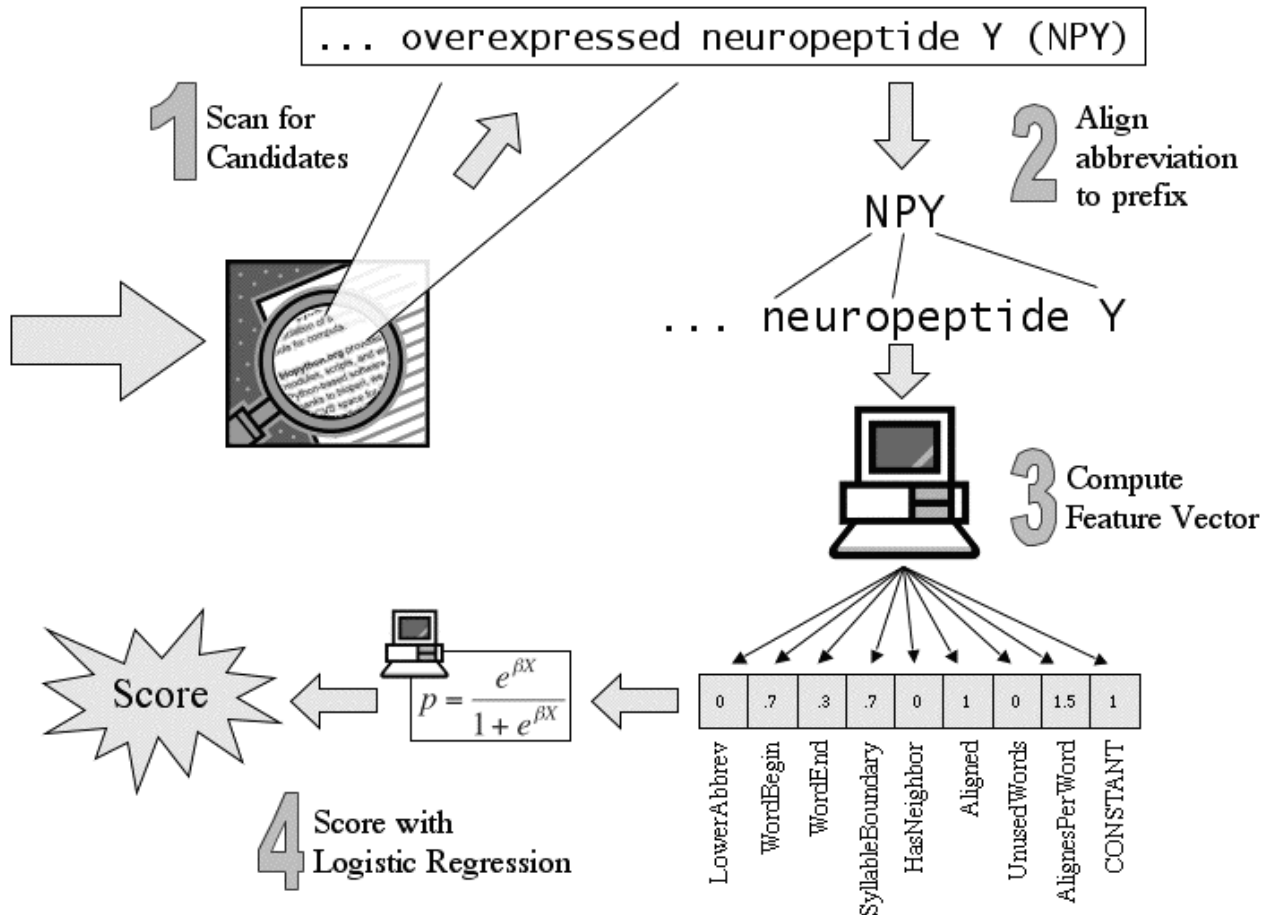


Figure 1: **System Architecture.** We use a machine learning approach to find and score abbreviations. First, we scan text to find possible abbreviations, align them with their prefix strings, and then collect a feature vector based on 8 characteristics of the abbreviation and alignment. Finally, we apply binary logistic regression to generate a score from the feature vector.

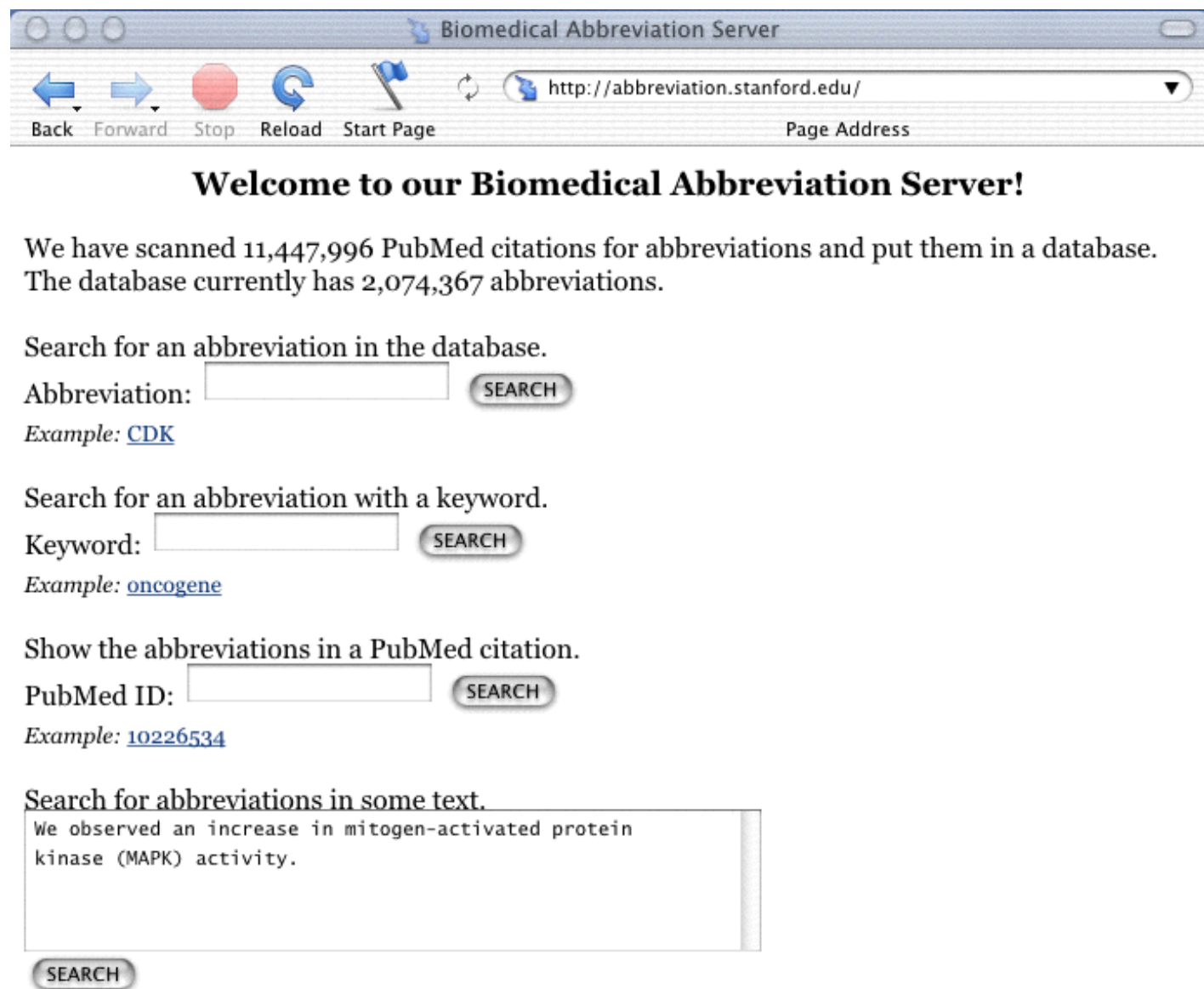


Figure 2: **Abbreviation Server.** Our abbreviation server supports queries by abbreviation or keyword.

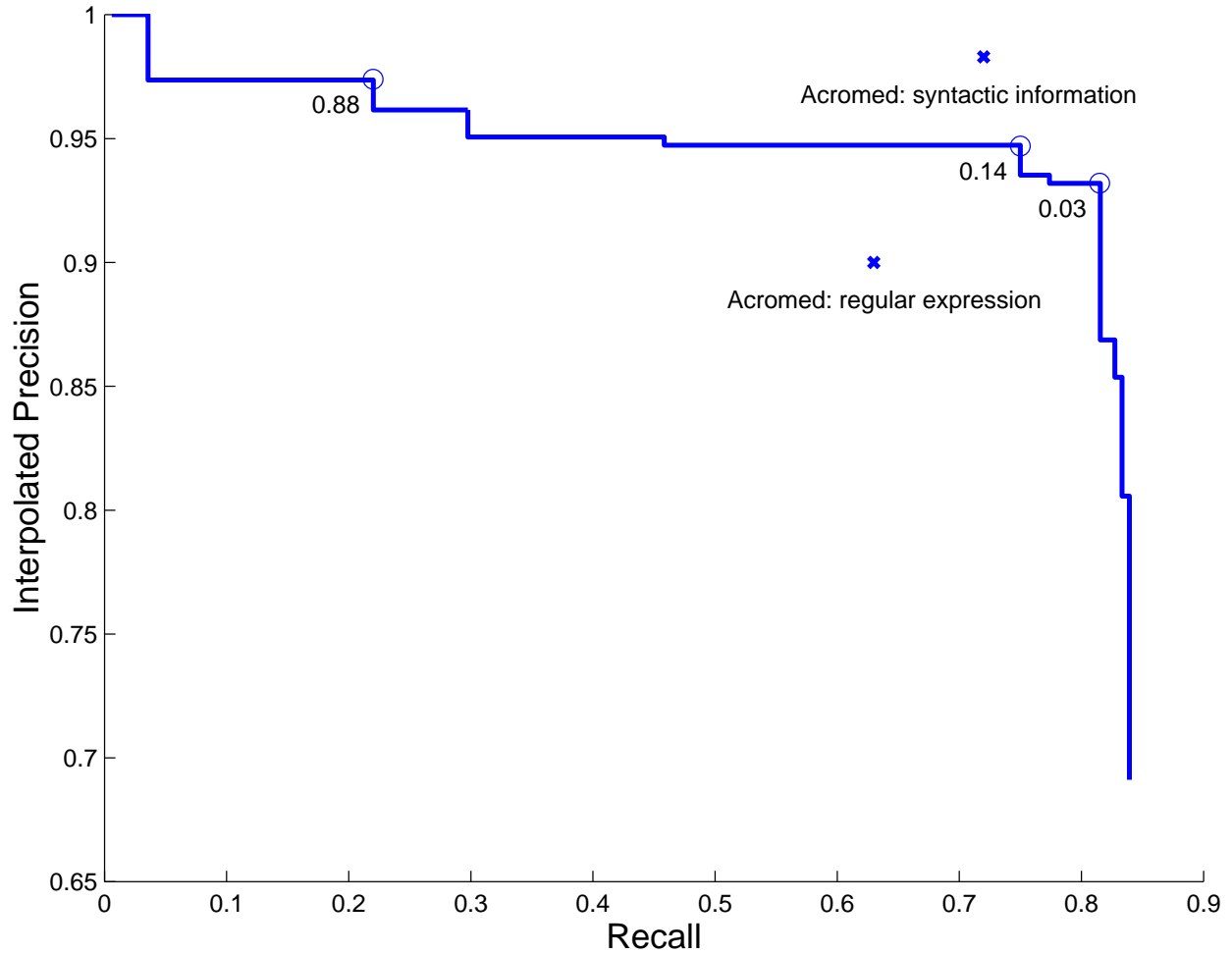


Figure 3: **Abbreviations Predicted in Medstract Gold Standard.** We calculated the recall and precision at every score cutoff and plotted the resulting curve. We marked the scores at various points on the curve. The performance of the Acromed system is shown for comparison.

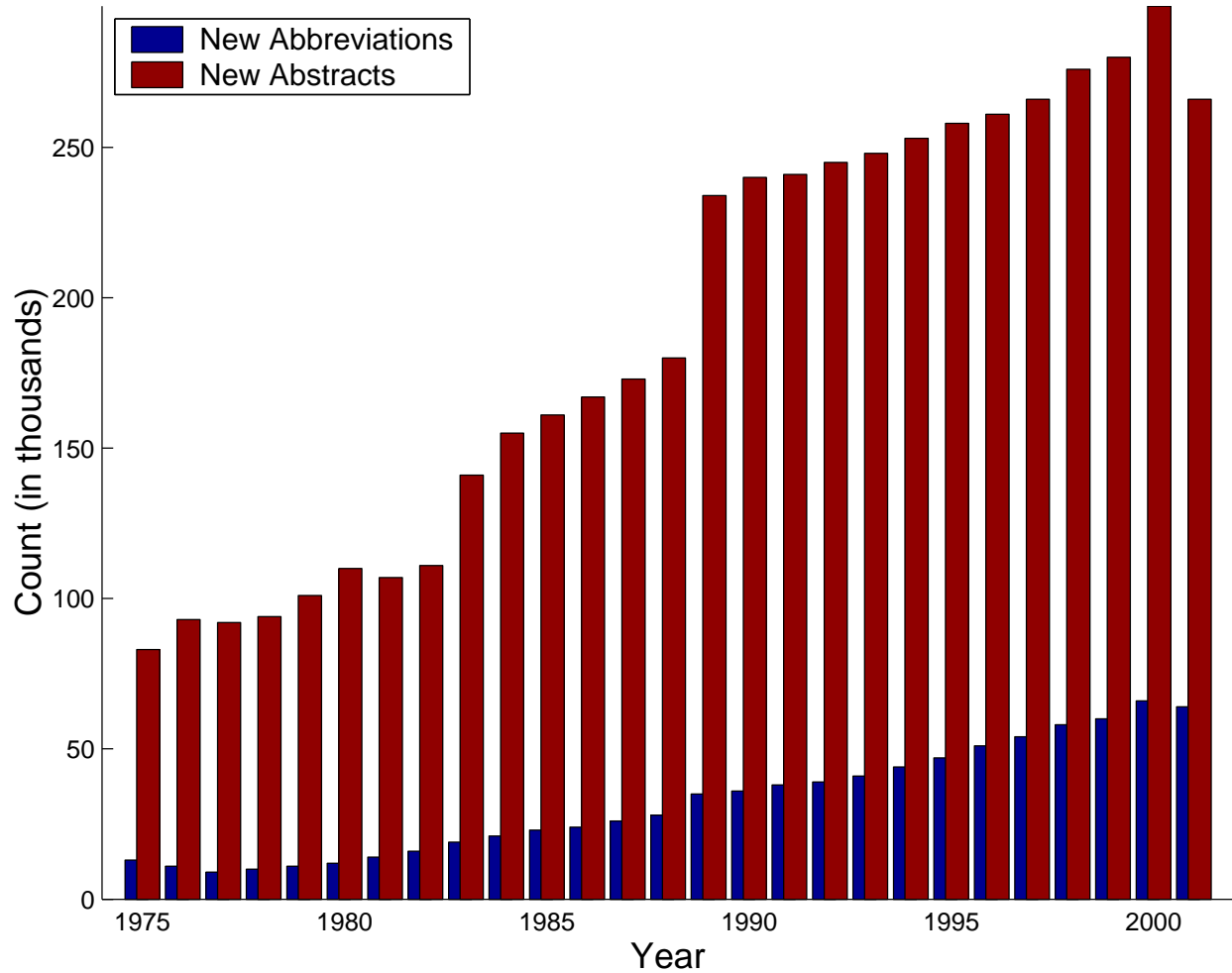


Figure 4: **Growth of Abstracts and Abbreviations.** The number of abstracts and abbreviations added to MEDLINE steadily increases.

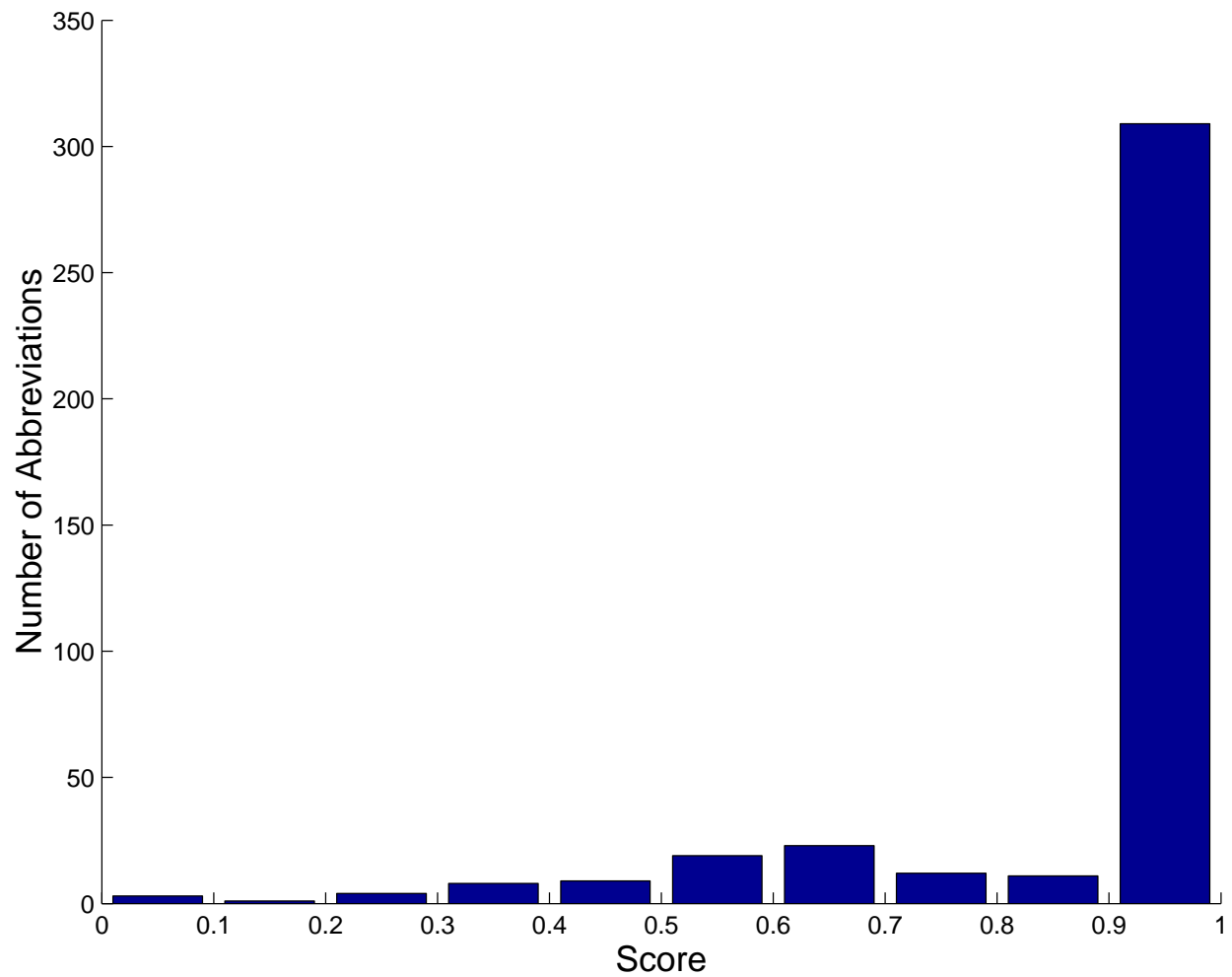


Figure 5: **Scores of Correct Abbreviations from the China Medical Tribune.** Using a score cutoff of 0.90 yields a recall of 68%, 0.14 87%, and 0.03 88%.

Feature	Description	β
Describes the abbreviation		
LowerAbbrev	Percent of letters in abbreviation in lower case.	-1.21
Describes where the letters are aligned		
WordBegin	Percent of letters aligned at the beginning of a word.	5.54
WordEnd	Percent of letters aligned at the end of a word.	-1.40
SyllableBoundary	Percent of letters aligned on a syllable boundary.	2.08
HasNeighbor	Percent of letters aligned immediately after another letter.	1.50
Describes the alignment		
Aligned	Percent of letters in the abbreviation that are aligned.	3.67
UnusedWords	Number of words in the prefix not aligned to the abbreviation.	-5.82
AlignsPerWord	Average number of aligned characters per word.	0.70
Miscellaneous		
CONSTANT	Normalization constant for logistic regression.	-9.70

Table 1: **Features Used for Scoring Abbreviations.** These features are used to calculate the score of an alignment using Equation 3. We identified syllable boundaries using the algorithm used in \TeX ([20]). The β column indicates the weight given to each feature. The sign of the weight indicates whether that feature is favorably associated with real abbreviations.

#	Description	Example
12	Abbreviation and long form are synonyms.	apoptosis⇒programmed cell death
7	Abbreviation is outside parentheses.	
3	Best alignment score yields incorrect long form.	FasL⇒Fas and Fas ligand
3	Letters in abbreviation are out of order.	ATN⇒anterior thalamus
25	TOTAL	

Table 2: **Abbreviations in Medstract Gold Standard Missed.** Our algorithm failed to find 25 total abbreviations in the Medstract gold standard. This table categorizes types of abbreviations and the number of each type missed.