

# Using BLAST for identifying gene and protein names in journal articles

Michael Krauthammer<sup>a,\*</sup>, Andrey Rzhetsky<sup>a,b</sup>, Pavel Morozov<sup>b</sup>, Carol Friedman<sup>a,c</sup>

<sup>a</sup> Department of Medical Informatics, Columbia University, New York, NY, USA

<sup>b</sup> Columbia Genome Center, Columbia University, New York, NY, USA

<sup>c</sup> Department of Computer Science, Queens College CUNY, New York, NY, USA

Received 24 April 2000; received in revised form 18 August 2000; accepted 8 September 2000

Received by T. Gojobori

## Abstract

We describe a system which automatically identifies gene and protein names in journal articles, an important and non-trivial first step in knowledge extraction of protein and gene actions. Our system uses a database of gene and protein names and is based on BLAST [Altschul et al., *Nucleic Acids Res.* 25 (1997) 3389–3402], a popular tool for DNA and protein sequence comparison. We describe a method that consists of mapping sequences of text characters into sequences of nucleotides that can be processed by BLAST. We demonstrate that this approach is feasible: the system matches gene and protein names with a recall of 78.8% and a precision of 71.7%, which includes names that are not part of the system database. An analysis of the results suggests techniques that can be used to improve performance further. © 2000 Elsevier Science B.V. All rights reserved.

**Keywords:** Natural language processing; Regulatory pathways; Sequence comparison tools; String matching

## 1. Introduction

### 1.1. Overview

In order to aid hypothesis-driven experimental gene discovery, we are designing a computer application for the automatic retrieval of signal transduction data from electronic versions of scientific publications using natural language processing (NLP) techniques (Koike and Rzhetsky, 2000; Rzhetsky et al., 2000). Before a NLP system such as MedLEE (Friedman, 1997) can be applied to texts in English, the names of genes and proteins have to be identified and tagged accordingly.

Systems that identify biological terms in journal articles have been described in the literature. Rule-based systems look at the morphologic characteristics of names

and use part-of-speech information and keywords to discover and tag names. Dictionary-based systems identify gene or protein names by matching them to dictionary entries. Unlike rule-based systems, dictionary-based methods have the advantage that the dictionary can list references to GenBank (Benson et al., 1998) or other knowledge sources, which is important to our project. On the other hand, a dictionary may not contain recently introduced names and may not cover all spelling variations of gene and protein names.

PROPER is an example of a rule-based system that identifies protein names in articles with a recall of 98.8% and a precision of 94.7% (Fukuda et al., 1998). ARBITER is another system with a slightly different application that uses dictionaries and rules to identify binding terms in articles with a recall of 72% and precision of 79% (Rindfleisch et al., 1999).

Using approximate text string matching techniques, our system is a dictionary-based system that recognizes spelling variations in names while keeping the reference to the closest nearest match. For example, although *interleukin-3* may not be listed in the dictionary, the program does recognize this name as being closely related to *interleukin-2* which is included in the diction-

Abbreviations: Bad, bcl-2 associated death agonist; ErbB, epidermal growth factor related protein; ERK, extracellular signal-regulated kinase; MAP, mitogen activated protein; Mek-1/2, MAP kinase kinase 1 and 2; Mpl, myeloproliferative leukemia protein; PDGFR, platelet-derived growth factor receptor; PTK, protein tyrosine kinase; *zgap1*, gene for Gz-selective GTPase-activating protein.

\* Corresponding author. Tel.: +1-212-928-2612; fax: +1-212-305-3302.

E-mail address: mk651@columbia.edu (M. Krauthammer)

ary. Our system is based on BLAST (Altschul et al., 1990, 1997), a popular DNA and protein sequence comparison tool, and on a database of gene and protein names. We adapted BLAST for a system that tags gene and protein names in scientific journals.

### 1.2. Background

Approximate text string matching and sequence comparison are related problems concerned with pattern detection in text and DNA or protein sequences respectively. Both problems are traditionally solved with dynamic programming techniques, which conveniently record and score substitution, deletion or insertion of characters (or nucleotides and amino acids respectively) (Gusfield, 1997). BLAST is representative of a new class of sequence comparison tools that run considerably faster than the dynamic programming-based programs. BLAST achieves much of its speed by rapidly identifying database sequences that *may* align with the query sequence to form high scoring segment pair (HSPs). A HSP is a stretch of local similarity between two sequences encoded as two sets of coordinates within the sequences under comparison. Usually HSPs are selected in such a way that alignment scores cannot be improved by extension or trimming. This is achieved by singling out database entries that match with high scoring words of the query sequence, small chunks of the query sequence that act as ‘seeds’ for further processing. Consequently, BLAST usually considers relatively few sequences as possible match candidates, thus reducing overall processing time. BLAST output is a list of database entries (DNA or protein sequences) ordered according to their alignment scores and *E*-value (statistical measure) together with their position in the query sequence. The first version of BLAST (Altschul et al., 1990) features the described fast look-up mechanism for singling out database entries that may pair to form HSPs (finding ‘hits’) before performing a costly letter-by-letter comparison (extending ‘hits’). A new version of BLAST (Altschul et al., 1997) requires two hits before an extension is invoked and supports gapped alignment.

## 2. Methods

### 2.1. Overview

In our approach, an exhaustive list of gene and protein names extracted from GenBank is translated into an alphabet of DNA sequences by substituting each character in the name with a predetermined unique nucleotide combination. The encoded names are then imported into the BLAST database using the FASTA format. Using the same nucleotide combination, the

scientific articles are translated into a continuous string of nucleotides. A query is then applied to match the translated articles against the nucleotide representation of gene and protein names stored in the standard binary BLAST database. Significant alignments associated with gene and protein names are listed in the BLAST output file. In order to identify the location of the gene and protein names in the articles, the output file is subsequently processed with a series of scripts written in the Perl programming language (Christiansen and Torkington, 1998). The locations are marked by adding tags or meta-data to the journal article. Tagging is accomplished with XML (Extensible Markup Language), a language that provides the ability to augment a document with additional elements of information and makes documents machine-independent.

To adapt the problem to BLAST’s statistical foundation, special efforts described below were made to limit the output to the most relevant gene and protein names. In order to fine-tune the matching process, different BLAST parameters were adjusted.

### 2.2. Hardware and BLAST executables

BLAST Version 2.0.10 for SGI IRIX computers was downloaded from the NCBI ftp file server (see <ftp://ncbi.nlm.nih.gov/blast/executables/>). The BLAST package contains the ‘blastall’ and ‘formatdb’ programs that were used in this study. BLAST and the Perl scripts were run on a SGI IRIX computer (1 × 195 MHz IP32 Processor).

### 2.3. Extraction of gene names and encoding of names and text

Gene and protein names were extracted from GenBank’s gene symbol index file (see <ftp://ncbi.nlm.nih.gov/genbank/gbgen.idx.Z>). The original file was treated with a Perl script to separate gene symbols from the other types of information in the file (accession number and locus information). We referred to the extracted elements as gene and protein names, because gene symbols correspond either to the gene itself or its protein product. Names were further filtered excluding frequently occurring English words and entries consisting only of numbers.

The names were then converted into nucleotide sequences by substituting each name character with a predetermined unique nucleotide combination. Upper and lower case were treated equally. The conversion table (Table 1) lists the character–nucleotide pairs for letters, numbers and punctuation marks. Characters not listed in the conversion table are translated into remaining nucleotide combinations as discussed in Section 2.5. As an example, the gene name *zgap1* would be

Table 1  
Conversion table

|          |      |          |      |          |      |          |      |          |      |          |      |          |      |
|----------|------|----------|------|----------|------|----------|------|----------|------|----------|------|----------|------|
| <b>A</b> | AAAC | <b>E</b> | AACG | <b>I</b> | AAGT | <b>M</b> | ACAC | <b>Q</b> | ACCG | <b>U</b> | ACGT | <b>Y</b> | AGAG |
| <b>B</b> | AAAG | <b>F</b> | AACT | <b>J</b> | AATC | <b>N</b> | ACAG | <b>R</b> | ACCT | <b>V</b> | ACTC | <b>Z</b> | AGAT |
| <b>C</b> | AAAT | <b>G</b> | AAGC | <b>K</b> | AATG | <b>O</b> | ACAT | <b>S</b> | ACGC | <b>W</b> | ACTG |          |      |
| <b>D</b> | AACC | <b>H</b> | AAGG | <b>L</b> | AATT | <b>P</b> | ACCC | <b>T</b> | ACGG | <b>X</b> | ACTT |          |      |

|          |      |          |      |           |      |   |      |   |      |       |      |   |      |
|----------|------|----------|------|-----------|------|---|------|---|------|-------|------|---|------|
| <b>0</b> | AGCC | <b>4</b> | AGGG | <b>8</b>  | AGTT | / | ATCC | , | ATCC | ?     | ATCC | - | ATCC |
| <b>1</b> | AGCG | <b>5</b> | AGGT | <b>9</b>  | ATAT | \ | ATCC | ; | ATCC | "     | ATCC |   |      |
| <b>2</b> | AGCT | <b>6</b> | AGTC | <b>] </b> | ATCC | ( | ATCC | : | ATCC | .     | ATCC |   |      |
| <b>3</b> | AGGC | <b>7</b> | AGTG | <b>[</b>  | ATCC | ) | ATCC | ! | ATCC | space | ATCC |   |      |

translated into:

```
AGATAAGCAAACACCCAGCG
```

The encoded gene and protein names were then converted into the FASTA format. FASTA has two components: a definition line and the sequence information. For this project, the sequence information was *replaced* with the encoded gene names. The FASTA entry for the above example would look as follows:

```
>gi|3045 species,gp,zgap1
```

```
AGATAAGCAAACACCCAGCG
```

The definition line (marked with '>') contains information about the database entry. There are no restrictions on the content of this line. For the purpose of this study, the definition line was composed of a 'gi' (GenBank ID) field and a species, semantic class and name field. The 'gi' field is used for a key to the database and helps to trace GenBank's ID number. The species field contains known species information. The semantic class field specifies the biologic class the name stands for ('gp' for gene/protein). The last entry in the definition line is the name of the protein or gene, *zgap1* in this case. Using the same nucleotide combinations, the scientific articles are translated into a continuous string of nucleotides. For example, the sentence

For instance, ErbB, Ras, and Raf all lie on the ERK MAP kinase (MAPK) pathway

would be represented by

```
AACTACATACCTATCCAAGTACAGACGCACGGAAACACAGAAATAACGATCCATCCAAC
GACCTAAAGAAAAGATCCATCCACCTAAACACGCATCCATCCAAACACAGAACCATCCAC
CTAAACAACCTATCCAACAATTAATTATCCAATTAAGTAACGATCCACATACAGATCCA
CGGAAGGAACGATCCAACGACCTAATGATCCACACAAACACCCATCCAATGAAGTACAG
AAACACGCAACGATCCATCCACACAAACACCCAATGATCCATCCATCCACCCAAACACG
GAAGGACTGAAACAGAG
```

#### 2.4. Scanning of BLAST output and markup of scientific journals

The FASTA entries of gene names were compiled into a BLAST compatible format using the program 'formatdb' included in the BLAST package. A query is then applied to match the translated articles against the nucleotide representation of gene names in the BLAST database. The query is executed with the 'blastall' program. Significant alignments associated with gene names are listed in the BLAST output file:

```
>gi|73879 species,gp,men
Length = 12
Score = 24.4 bits (12), Expect = 0.003
Identities = 12/12 (100%)
Strand = Plus / Plus
```

```
Query: 55501 acacaacgacag 55512
          |||
Sbjct: 1 acacaacgacag 12
```

The first line denotes the database entry as described above. The second line denotes the database sequence length, followed by the alignment score and the

Table 2  
BLAST parameters

| Database <sup>a</sup> | <i>E</i> -value |         | Wordsize <sup>b,c</sup> |         | Alignments |         | Penalty   |         |
|-----------------------|-----------------|---------|-------------------------|---------|------------|---------|-----------|---------|
|                       | Optimized       | Default | Optimized               | Default | Optimized  | Default | Optimized | Default |
| 3                     | 10              | 10      | 12 (3)                  | 11 (<3) | 2000       | 250     | –6        | –3      |
| 4–5                   | 10              | 10      | 16 (4)                  | 11 (<3) | 2000       | 250     | –6        | –3      |
| 6–10                  | 10              | 10      | 20 (5)                  | 11 (<3) | 250        | 250     | –3        | –3      |
| 11–20                 | 10              | 10      | 40 (10)                 | 11 (<3) | 250        | 250     | –3        | –3      |
| >20                   | 10              | 10      | 80 (20)                 | 11 (<3) | 250        | 250     | –3        | –3      |

<sup>a</sup> Numbers correspond to word length(s) in each database.

<sup>b</sup> Parentheses show corresponding letters in text.

<sup>c</sup> Also optimized for speed.

Table 3  
Results

| Parameter settings | Names marked by evaluators and included in database ( <i>n</i> =753) |               |                | Names marked by evaluators and not included in database ( <i>n</i> =409) |               |                | Names not marked by evaluators |
|--------------------|--|---------------|----------------|--|---------------|----------------|--------------------------------|
|                    | True positive  |               | False negative | True positive  |               | False negative | False positive                 |
|                    | Full match   | Partial match | No match       | Full match   | Partial match | No match       |                                |
| Optimized          | 712 (94.6%)  | 23 (3.1%)     | 18 (2.4%)      | 18 (4.4%)  | 163 (39.9%)   | 228 (55.7%)    | 362                            |
| Default            | 424 (56.3%)  | 69 (9.2%)     | 260 (34.5%)    | 18 (4.4%)  | 104 (25.4%)   | 287 (70.2%)    | 214                            |

expectation value (*E*-value). The next line indicates paired matches, mismatches and gapped alignment (the latter two are not shown in this example). The lines ‘Query’ and ‘Sbjct’ show the actual alignment between the query text and database sequence. In the above example, the nucleotide representation for the name *men* matches the query text between positions 55 501 and 55 512. Note that these numbers are four times the actual numbers of the positions in the journal text, the result of substituting one character with four nucleotides. The output file, which is sometimes several megabytes long, is subsequently processed with Perl-scripts which compare the starting and end points of the match with the actual text, making sure that the matched name is an ‘autonomous’ entity in the query text. For example, while *per* in *activation of per gene* should be recognized as a gene name, *per* in *personal* should be ignored. The final result consists of the original journal article with XML tags (‘phr’) surrounding the gene and protein names:

```
For instance, <phr sem="gp">ErbB</phr>, <phr
sem="gp">Ras</phr>, and <phr sem="gp">Raf</phr> all lie on
the <phr sem="gp">ERK</phr> <phr sem="gp">MAP</phr> <phr
sem="gp">kinase</phr> (<phr sem="gp">MAPK</phr>
) pathway
```

## 2.5. Program adaptation and BLAST parameters

BLAST does not use a specific substitution matrix for the comparison of nucleotide sequences. Therefore BLAST may match database names to wrong parts of the journal article by allowing many incorrect letter substitutions. We tackled this problem by limiting the amount of letter substitutions in matches to 10%.

In order to adapt the system to BLAST’s statistical foundation, different measures were undertaken to limit BLAST’s output to the most relevant gene and protein names. Given a long query sequence and a large sequence database (large search space), individual matches have a lower statistical significance and may not be listed in the BLAST output file. In addition, matches with few letters generally have a lower statistical significance than matches with many letters.

To limit the search space, the query sequence was divided into 10 parts (approximately 30 paragraphs,

depending on the length of the journal article) and queries are run on each part separately. To further limit the search space and to improve the results of short database names, the sequence database (with the gene and protein names) was separated into five databases, each containing names of different length. In addition, gene and protein names with less than three letters in the database were ‘expanded’, i.e. space characters were added at the beginning and the end of the name. As can be seen in the conversion table (Table 1), the nucleotide representation for a space character is ATCC, which also stands for several other punctuation marks that act as delimiters for gene or protein names.

In order to fine-tune the matching process, different BLAST parameters had to be adjusted. Starting with the default values, Unix scripts incrementally changed parameters to above and below the default values. Results were then compared manually and the setting with the best performance was chosen (Table 2). The following parameters were examined.

- *Expectation value (E-value)*: sets the threshold for the significance value of matches in the output file.
- *Word size*: sets the size of the high scoring words, thus influencing the sensitivity of finding HSPs. The word size is four times the actual number of letters in the text (corresponding to the substitution of one letter with four nucleotides).
- *Numbers of alignments to show*: sets the number of alignments that show up in the BLAST output file. Large values help to find true matches of low significance that are sometimes listed at the very end of the output file.
- *Mismatch penalty*: sets the penalty for a nucleotide mismatch and influences the final score of a match. A high penalty results in exact string matching while a low penalty results in approximate string matching.

Finally, BLAST potentially matches database entries to different reading frames in the query sequence. The nucleotide combinations used in this study therefore have comma-free characteristics: given the nucleotide combination ATCC and its permutations CATC, CCAT and TCCA, only one of these permutations is selected for converting characters. This limits matches to a wrong reading frame. As an example, a sequence TCCA|TCAA (bars added for clarity) may match to the second reading frame of a sequence ATCC|ATCC|ATCC which consists of permutations of the first sequence. This produces a nonsense result that could compromise the significance of true matches, which are confined to the first reading frame. Because of this limitation on the nucleotide combinations, it was necessary to represent each character with four nucleotides. Otherwise, a triplet-representation would have been sufficient. Comma-free codes have been discussed extensively in the early days of DNA research (Crick et al., 1957) before the ‘true’ genetic code was identified.

### 3. Evaluation

To evaluate the program, two experts in the field of biology separately marked gene and protein names in a review article. The results were compared and the same experts discussed and resolved differences in their text markups producing a gold standard file containing the original article with tagged gene and protein names. Another program partitioned the marked up names in the gold standard into names listed and not listed in the database of gene and protein names. The gold standard was then compared with the same article tagged by the BLAST program.

In general, if the start and end tags of two names aligned, a match was counted as a full match (even if some letters or numbers were substituted, as in *interleukin-2* matching *interleukin-3*). A partial match was defined as one tag of one name matching *within* the start and end tag of another name.

### 4. Results

#### 4.1. Evaluation process

Evaluator 1 marked up 1294 gene and protein names, while evaluator 2 marked up 1143 names. Compared with evaluator 1, evaluator 2 fully matched 896 names (69.2%), partially matched 177 names (13.7%), and did not mark up 221 names (17.1%). Evaluator 2 marked up 70 names that were not marked up by evaluator 1. After comparison and discussion of the differences, the two evaluators settled on marking up 1162 gene and protein names in the review article (gold standard).

#### 4.2. BLAST program

The BLAST program marked up 1278 names in the review article. Results were divided into three main groups (Table 3): names marked by the evaluators and included in the BLAST database, names marked by the evaluators and not included in the BLAST database, as well as names marked by the BLAST program but not marked by the evaluators. In the group of names marked by the evaluators and included in the BLAST database (753 names), the program fully matched 712 names (true positive, 94.6%), partially matched 23 names (true positive, 3.1%), and did not match 18 names (false negative, 2.4%). In the group of names marked by the evaluators and not included in the BLAST database (409 names), the program fully matched 18 names (true positive, 4.4%), partially matched 163 names (true positive, 39.9%), and did not match 228 names (false negative, 55.7%). The program marked up 362 names that were not marked up by the evaluators (false positive). All together, 916 names marked by the system fully or

partially matched the 1162 names marked by the evaluators. This results in a recall of 78.8%. The system marked 916 out of 1278 names correctly, and precision is therefore 71.7%. Results were dependent on the parameters used. Table 3 compares two parameter-settings (default and optimized).

## 5. Discussion

### 5.1. Fully matched names

#### 5.1.1. Names represented in the database

Where names are represented in the database, BLAST fully matches 94.6% of names. This number is somewhat low, considering that a brute force exact string matching technique should recognize close to 100% of these names (Gusfield, 1997). The missing 5.4% of names are either only partially matched or missed. Looking at some partially mapped names, we see that the program failed to recognize the full name of gene *notch1* but rather matched to a shorter database entry *notch*. The reason therefore is that the five databases used in this program each represent names of different length. It turns out that *notch1* is represented in the database containing names with six to 10 letters, while *notch* is represented in the database with four to five letters. *Notch1* must ‘compete’ with longer sequences (up to 10 letters long) for statistical significance, with the result being that only the shorter database entry, *notch*, is listed in the BLAST output file. Normally, this situation is easily resolved by adjusting the parameters for *E*-value and for the number

of alignments to show. In rare cases, some names may nevertheless be missed.

#### 5.1.2. Names not represented in the database

Conventional dictionary-based matching methods do not fully match names that are not represented in their databases. In this study, 4.4% of names not included in the database were fully matched by our system. Table 4 lists some examples of fully matched words that were not represented in the database.

These matches show the benefit of using a sequence alignment tool for matching names in journal articles. Substitutions of letters or punctuation marks do not lead to failed word recognition, but rather to a match with a somewhat lower alignment score. This score can be considered as a confidence measure showing the ‘distance’ between the original term and the matched term. As the goal of our system is the identification of the matched name, this distance can be used to put the match in the right hierarchical position in the knowledge model. Similar to the case of evolutionary distance (Nei, 1987) needed to group genes according to phylogenetic properties, such a ‘textual distance’ can be used to assign a matched term to a parent term in the knowledge model. For example, while *interleukin-3* closely matches *interleukin-2*, it would not be correct to say that these terms are the same entity. Rather, all that can be said is that *interleukin-2* most probably belongs to the same class of concepts as *interleukin-3*. It is therefore possible to assign *interleukin-3* to the class of *interleukins*, the parent concept of *interleukin-3*.

### 5.2. Partially matched names

This system partially recognizes names that are not in the database. The partial matches are grouped into three different types (Table 5) depending on the completeness of the match. Often, partial matches match a meaningful part of a name, as is the case with *Cbl* for *v-Cbl*. Other types of match are harder to decipher. For

Table 4  
Examples of fully matched names not listed in the database

| Database entry | Name in text (fully matched) |
|----------------|------------------------------|
| v-ErbB         | v-ErbA                       |
| Cyclin E       | Cyclin D                     |
| Mek-2          | Mek-1                        |

Table 5  
Types of partial match

| Types of partial match |   | Examples <sup>a</sup>   |   |
|------------------------|---|---|---|
|                        |   | Marked by evaluators  | Marked by BLAST   |
| 1                      | Name is single term,<br>BLAST matches part of term      | <v-Cbl><br><v-Mpl>  | v-<Cbl><br>v-<Mpl>  |
| 2                      | Name is a compound term,<br>BLAST matches all compounds | <PDGFR PTK> gene  | <PDGFR> <PTK> gene  |
| 3                      | Name is compound term,<br>BLAST matches some compounds  | <promyelocytic leukemia<br>Kruppel-like zinc finger> protein<br><Abl nonreceptor PTK> | Promyelocytic leukemia<br><Kruppel>-like zinc finger protein<br><Abl> nonreceptor <PTK> |

<sup>a</sup> Marked names are between tags < >.

example, *Abl* and *PTK* are matches for *Abl nonreceptor PTK*.

Partial matches were counted as true positive matches because concatenation of terms as described in the PROPER system (Fukuda et al., 1998) would likely be sufficient to determine the full and correct range of the match. Although it is not straightforward to give a database reference for partially matched names, simple heuristic methods for type 1 partial matches could construct a semantic relationship between the full name and its matched substring. For type 2 and 3 partial matches, the only solution to identify and reference names may be to include the names in the database.

### 5.3. BLAST parameters

The changing of parameter settings can influence recall and precision of the matching process. Using BLAST with optimized parameter settings, recall was 78.8% and precision was 71.7%. Using BLAST's default parameters, recall was 52.9% and precision was 74.2%. Different levels of alignments that show up in the BLAST output file make up most of the differences observed. Interestingly, the default settings consumed much more CPU time, although recall was considerably lower. By default, BLAST uses a word size of 11 letters (i.e. BLAST plants many small 'seeds'), which triggers many costly extensions in database sequences.

### 5.4. Recall and precision

The level of recall is dependent on the domain coverage of the database of gene and protein names. For this project we used a static list of gene and protein names, but an automated update of the database with new gene and protein names should help yield better coverage.

Both recall and precision are influenced by ambiguous database entries, where the entry may be both a common English word and a gene or protein name. For example, in this study we excluded the word *bad* from the system database. Some instances of the protein name *Bad* were therefore missed in the review article, which slightly lowered the system recall. On the other hand, precision was positively influenced by not accidentally marking the English word *bad* as protein name.

### 5.5. Evaluation process

The evaluation of the program revealed an important problem in identifying gene and protein names in journal articles: what exactly does represent a gene or protein name? An analysis of the inter-individual difference between the two evaluators involved in this study showed that they fully agreed in 69.2%, partially agreed

in 13.7%, and did not agree in 17.1% of names involved. These findings are consistent with previous studies involving experts determining gold standards. One large-scale study of natural language systems, the Fifth Message Understanding Conference (Will, 1994), found that experts differed in their previous interpretation of documents about 15% of the time, and also showed that experts differed from other experts about 30% of the time. Other studies have also shown that the judgments of experts differ a significant portion of the time (i.e. approximately 20 and 30%, respectively) (Yerushalmy, 1969; Hripcsak et al., 1995). Interestingly, the study discussed in Yerushalmy (1969) analyzed the variability in observer perception and description of chest X-ray films rather than interpretation of the X-ray reports. It was shown that intra-individual disagreement was 22% and inter-individual disagreement was 30%.

### 5.6. Future directions

Our system is able to fully match some names in journal texts that are not listed in the system database. Further research is needed to optimize this capability, for example by studying parameter settings for gapped alignment. Gapped alignment can also be used to detect patterns in the text. For example, a generic database entry like *X protein* could potentially retrieve all Xs (protein names) that match the pattern. Some researchers have reported on improved sensitivities of BLAST DNA searches when alternative scoring systems are used (States et al., 1991). Unfortunately, this cannot be achieved without some changes to the BLAST source code. Expanding BLAST's amino acid alphabet to include more letters would be helpful for our system. Special adapted scoring matrices could then be used, where substitution scores would reflect common lexical changes of gene and protein names.

We are also considering combining the strength of rule-based systems and dictionary-based systems. For example, our system may profit from the incorporation of rules to disambiguate between English words and gene or protein names. For example, a capital B in *Bad* in the middle of a sentence is a strong indicator that a word is a protein name, rather than an English word. Once a word has been marked, the BLAST system could then be used to find the most closely related name in the database and give a reference to an outside data source, like GenBank.

### Acknowledgements

This study was partially supported by grants from the Center for Advanced Technology, New York State, and from Hitachi Software Engineering Co., Ltd.

## References

- Altschul, S.F., Gish, W., Miller, W., Myers, E.W., Lipman, D.J., 1990. Basic local alignment search tool. *J. Mol. Biol.* 215, 403–410.
- Altschul, S.F., Madden, T.L., Schaffer, A.A., Zhang, J., Zhang, Z., Miller, W., Lipman, D.J., 1997. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res.* 25, 3389–3402.
- Benson, D.A., Boguski, M.S., Lipman, D.J., Ostell, J., Ouellette, B.F., 1998. GenBank. *Nucleic Acids Res.* 26, 1–7.
- Christiansen, T., Torkington, N., 1998. *Perl Cookbook*. O'Reilly and Associates, Sebastopol, CA.
- Crick, F.H., Griffith, J.S., Orgel, L.E., 1957. Codes without commas. In: *Proceedings of the National Academy of Science of the USA*, 416–421.
- Friedman, C., 1997. Towards a comprehensive medical language processing system: methods and issues. *Proc. AMIA Ann. Fall Symp.*, 595–599.
- Fukuda, K., Tamura, A., Tsunoda, T., Takagi, T., 1998. Toward information extraction: identifying protein names from biological papers. *Pac. Symp. Biocomput.*, 707–718.
- Gusfield, D., 1997. *Algorithms on strings, trees and sequences: computer science and computational biology*. first ed. The Press Syndicate of the University of Cambridge, Cambridge.
- Hripcsak, G., Friedman, C., Alderson, P.O., DuMouchel, W., Johnson, S.B., Clayton, P.D., 1995. Unlocking clinical data from narrative reports: a study of natural language processing (see comments). *Ann. Intern. Med.* 122, 681–688.
- Koike, T., Rzhetsky, A., 2000. A graphic editor for analyzing signal transduction pathways. *Gene*. this volume.
- Nei, M., 1987. *Molecular Evolutionary Genetics*. Columbia University Press, New York.
- Rindflesch, T.C., Hunter, L., Aronson, A.R., 1999. Mining molecular binding terminology from biomedical text. *Proc. AMIA Symp.*, 127–131.
- Rzhetsky, A., Koike, T., Kalachikov, S., Gomez, S.M., Krauthammer, M., Kaplan, S.H., Kra, P., Russo, J.J., Friedman, C., 2000. A knowledge model for analysis and simulation of regulatory networks. *Bioinformatics*. in press.
- States, D.J., Gish, W., Altschul, S.F., 1991. Improved sensitivity of nucleic acid database searches using application-specific scoring matrices. *Methods Enzymol.* 3, 66–77.
- Will, C., 1994. Comparing human and machine performance for natural language information evaluation. In: *Proceedings of the Fifth Message Understanding Conference (MUC-5)*. Morgan Kaufmann, San Mateo, CA, pp. 53–68.
- Yerushalmy, J., 1969. The statistical assessment of the variability in observer perception and description of roentgenographic pulmonary shadows. *Radiol. Clin. North Am.* 7, 381–392.