Unpublished Technical Note, March 2002.

Extracting Gene and Protein Names from Biomedical Abstracts

Razvan Bunescu, Ruifang Ge, Raymond J. Mooney
Department of Computer Sciences
University of Texas
Austin, TX, USA 78712

Edward Marcotte, Arun Kumar Ramani
Institute for Cellular & Molecular Biology
University of Texas
Austin, TX, USA 78712

Paper ID: Pxxxx

Keywords: information extraction, protein/gene interaction

Contact Author: Raymond Mooney, mooney@cs.utexas.edu

Abstract

Automatically extracting information from biomedical text holds the promise of easily consolidating large amounts of biological knowledge in computer accessible form. We are investigating the use of information extraction techniques for processing biomedical text. Currently, we have focused on the initial stage of identifying information on interacting proteins, specifically the problem of recognizing protein and gene names with high precision. We present preliminary results on extracting protein names from Medline abstracts.

Extracting Gene and Protein Names from Biomedical Abstracts

Paper ID: Pxxxx

Abstract

Automatically extracting information from biomedical text holds the promise of easily consolidating large amounts of biological knowledge in computer accessible form. We are investigating the use of information extraction techniques for processing biomedical text. Currently, we have focused on the initial stage of identifying information on interacting proteins, specifically the problem of recognizing protein and gene names with high precision. We present preliminary results on extracting protein names from Medline abstracts.

1 Introduction

An incredible wealth of biological information generated using biochemical and genetic approaches is stored in published articles in scientific journals. However, retrieving and processing this information is very difficult due to the lack of formal structure in the natural-language narrative in these documents. Automatically extracting information from biomedical text holds the promise of easily consolidating large amounts of biological knowledge in computer accessible form. Information extraction (IE) systems could potentially gather information on global gene relationships, gene functions, gene-gene interactions, and other important information on biological processes.

In fact, all projects focused on extracting information about genes or proteins must face the initial challenge of recognizing the gene/protein names in the text. This process is exacerbated by the lack of standardized gene naming conventions in biology. Gene/protein name styles vary considerably from organism to organism; even for genes from the same organism, few rules exist. For example, human genes/proteins may be named with standard English words, such as "light", "map", "complement", and "Sonic hedgehog". Names may be alphanumeric, may include greek or latin letters, may be case sensitive, and may be composed of multiple words. Names are frequently substrings of each other, such as "epidermal growth factor" and "epidermal growth factor receptor", which refer to two different proteins. It is therefore necessary that an information extraction algorithm be specifically trained to extract gene and protein names accurately.

A number of recent projects have focused on the manual development of IE systems for extracting information from biomedical literature (Fukuda et al., 1998; Humphreys et al., 2000; Blaschke and Valencia, 2001; Proux et al., 2000; Rindflesch et al., 2000; Thomas et al., 2000; Hahn et al., 2002). We are in the beginning stages of a project focused on using machine-learning methods to develop IE systems for extracting information on gene/protein function and gene/protein interactions from Medline abstracts. For our purposes, genes and proteins are interchangeable since, typically, there is a one-to-one correspondence between proteins and the genes that code for them. In this paper, we present initial results on extracting gene and protein names, the first step in the process of extracting protein interactions or other gene specific data.

2 Biological Data

2.1 Tagging of Medline Abstracts

In order to assemble training and test data for extracting proteins, 5,000 Medline abstracts with the MeSH term 'human' were downloaded from Medline. This set of 5,000 was filtered using a tagging tool to isolate 218 abstracts that spoke about genes, and from these the final

PMID - 10206993
TI - Differential mechanisms of recognition and activation of <prot> interleukin-8 receptor </prot> subtypes.
AB - We have probed an epitope sequence (His18-Pro19-Lys20-Phe21) in <prot> interleukin-8 </prot> (<prot> IL-8 </prot>) by site-directed mutagenesis. This work shows that single and double Ala substitutions of His18 and Phe21 in <prot> IL-8 </prot> reduced up to 77-fold the binding affinity to <prot> IL-8 receptor subtypes A </prot> (<prot> CXCR1 </prot>) and B (<prot> CXCR2 </prot>) and to the <prot> Duffy antigen </prot>. These Ala mutants triggered neutrophil degranulation and induced calcium responses mediated by <prot> CXCR1 </prot> and <prot> CXCR2 </prot>. Single Asp or Ser substitutions, H18D, F21D, F21S, and double substitutions, H18A/F21D, H18A/F21S, and H18D/F21D, reduced up to 431-fold the binding affinity to <prot> CXCR1 </prot>,<prot> CXCR2 </prot>, and the <prot> Duffy antigen </prot>. Interestingly, double mutants with charged residue substitutions failed to trigger degranulation or to induce wild-type calcium responses mediated by <prot> CXCR1 </prot>. Except for the H18A and F21A mutants, all other<prot> IL-8 </prot> mutants failed to induce superoxide production in neutrophils. This study demonstrates that <prot> IL-8 </prot> recognizes and activates <prot> CXCR1 </prot>, <prot> CXCR2 </prot>, and the <prot> Duffy antigen </prot> by distinct mechanisms.
AD - Department of Physiology and Biophysics and Sealy Center for Molecular

Figure 1: Sample Tagged Medline Abstract

50 abstracts were obtained. Tagging was carried out using the texttagger.pl software downloaded from `http://www-2.cs.cmu.edu/~kseymore/general_tagger.pl`. This program accepts a directory of files to be tagged and allows a user to tag them using a graphical user interface based on a file of possible labels and writes the tagged files into an output directory. An example of tagged abstract is shown in Figure 1.

2.2 Gene/Protein Dictionary

In order to identify gene/protein names, two fairly comprehensive dictionaries of human gene/protein names were assembled.

1. The `human.seq` file was downloaded from the EXPASY website (`http://us.expasy.org/sprot/hpi/hpi_ftp.html`) and processed to create one of the dictionaries.

2. The file `feb2002-tables.tar.gz` was downloaded from the Gene Ontology Database (`http://www.godatabase.org/dev/database/archive/latest`) and processed to create another dictionary of protein names.

Altogether, these dictionaries contain 42172 gene/protein names (synonyms included). These dictionaries were used to create a protein/gene name tagger as described in Section 3.

3 Identifying Protein Names

3.1 Protein Tagger

The task of recognizing protein names in biomedical corpora continues to represent a challenge, mainly because of the following factors:

- Many instances of new protein names do not follow exactly the standard nomenclature.

- Authors often refer to proteins already included in protein databases using variations which do not exist in those databases.

The success of a protein tagger depends on how well it captures the regularities of protein naming and name variations. Our approach was to start with an extensive set of protein names collected from two online dictionaries, henceforth referred to as the original dictionary, and then extend it using a carefully designed procedure. We centered our effort around this initial set of proteins, our aim being to develop a highly accurate tagger. The major task was to extend the coverage of the original set, while at the same time trying to minimize any decrease in accuracy.

The tagger was divided in two main algorithms:

1. An algorithm for the extension of the original dictionary (OD). The result would be a pair of dictionaries - a generic dictionary (GD) and a canonical dictionary (CD). The algorithm is described in Table 1.

2. An algorithm for traversing a given text and identifying protein names, based on the extended dictionaries produced by the

previous algorithm. The algorithm is described in Table 2.

$ID \equiv$ intermediate dictionary
let $ID \leftarrow \emptyset$
for each entry $e \in OD$ do
let $gen \leftarrow generalForm(e)$
let $ID \leftarrow ID \cup \{gen\}$
let $GD \leftarrow ID$
for each short name $sn \in ID$
let $CV \leftarrow caseVariations(sn)$
let $GD \leftarrow GD \cup CV$
let $CD \leftarrow ID$
for each entry $e \in ID$
let $cf \leftarrow canonicalForm(e)$
let $CD \leftarrow CD \cup \{cf\}$

Table 1: Extension Algorithm

$g \equiv$ textual n-gram
let $gf \leftarrow generalForm(g)$
if $gf \in GD$
tag g as protein name
else
let $cf \leftarrow canonicalForm(gf)$
if $cf \in CD$
tag g as protein name

Table 2: Recognition Algorithm

Two important procedures, namely the name generalization and the canonic form finder, were employed in both algorithms.

Generalizing a name means to identify those parts susceptible of being changed in new protein names, and replace them with generic placeholders. Thus, we isolate and replace numbers with $<n>$, latin letters with $<l>$ and greek letters with $<g>$. Table 3 shows some examples of name generalizations.

The canonical form of a protein name refers to that part of the name contained in all variations. We consider that all variations of a protein name create an equivalence class, and once that we have identified the equivalence class of a protein name, we take only a representant of this class and insert it in a dictionary of canonical forms. The representant of a class will be chosen to be the smallest element of the class i.e. the element contained in any variation from that class.

Protein Name	Generalized Name
interleukin-1 beta	interleukin $<n>$ $<g>$
interferon alpha-D	interferon $<g>$ $<l>$
NF-IL6-beta	NF IL $<n>$ $<g>$
TR2	TR $<n>$

Table 3: Generalizations

We have identified three types of protein names, each with its own class of variation rules:

1. Short names - this category contains all abbreviation-type names (ex: FGF1, hGITRL, CCT-beta, TNF-alpha);

2. Full names - this category contains detailed protein names (ex: tumor necrosis factor ligand superfamily member 14);

3. One word names - names whose head is a lower case word usually referring to a class of proteins (ex: Alpha E-catenin, interleukin 10);

Another independent reason for implementing slightly different versions of the canonical form procedure is given by the context in which this procedure is actually used. We distinguish between:

- The extension version, used for deriving the canonical dictionary, and

- The recognition version, used for creating canonical forms from textual n-grams.

Below we give the three main steps of deriving the canonical form for a short name (recognition version):

1. Strip the short name of all elements from the list A of affix words (Table 4).

2. Replace the acronym-type token from the reduced short name with $<x>$.

3. If the new token is in the class C of short name variations (Table 4), then its canonical form is the acronym-type token that was replaced with $<x>$ in the previous step.

A	*h, human, anti, precursor, receptor, R ligand, L, chain, kinase, protein, ...*
C	$<\mathbf{x}>$ $<x><l>$, $<x><g>$, $<g><x>$, $<x><n>$, $c<x>$, $s<x>$, $<x><l><n>$, $<x><l><g>$, $<x><g><n>$, $<x><g><l>$, $<g><x><n>$, $<x><n><l>$, $<x><n><g>$, $<x><g><l><g>$, $<l><x><n><l>$, $<l><x>$

Table 4: Variation Class for Short Names

For the extension version, the affix set A is considerably reduced. One element that is eliminated from this set is "receptor", because by removing "receptor" from a protein name we might end up with a non-protein name - for instance "vitamin A receptor" vs. "vitamin A".

Example:

- [Extension] NF-25 is a protein name in the original dictionary *OD*.
- [Extension] By generalization we get NF$<n>$. This general form, together with its case variation Nf$<n>$, is to be included in the general dictionary *GD*.
- [Extension] NF$<n>$ is a short name, hence we can use the previous procedure for deriving its canonical form. We substitute the acronym NF with $<x>$, and we get $<x><n>$, which is a member of the class C of variations, therefore its canonical form will be $<x>$ = NF. This form will be added to the canonical form dictionary *CD*.
- [Recognition] The textual n-gram NF-kappa B is not in the original dictionary, however it generalizes to NF$<g><l>$.
- [Recognition] Further, replacing the acronym NF with $<x>$, we get $<x><g><l>$, which is a member of the variations class C. Its canonical form, NF, is found in the canonical dictionary, therefore NF-kappa B will be tagged as a protein name.

We've paid great attention to case variations, and finely tuned the procedures for deriving canonical forms, distinguishing between the extension case and the recognition case, as well as among the 3 types of protein names. In this way, we were able to consider, for instance, PHE3 (Dihydrolipoamide dehydrogenase, mitochondrial precursor) as a gene/protein name and avoid overgeneralizing it to the form Phe$<n>$ which would have covered non-protein entities like the amino acid residues Phe93, or Phe205. Other examples, often encountered in the test abstracts, include THR1 (vs. Thr151), MET1 (vs. Met150), and many others. This may be seen as an improvement over other approaches that would tag them as protein names, and supports once more the general observation that tagging protein names is a difficult task.

3.2 Experimental Evaluation

The protein tagger was tested against a set of 50 abstracts containing 742 protein names which have been previously tagged. We used the standard measures of precision and recall. The results are summarized in Table 5, where the first line refers to a baseline tagger which simply tries to match n-grams from abstracts against entries from the original dictionary, while the second line refers to the actual tagger.

	Precision	Recall
OD only	95.2%	44.5%
OD+GD+CD	93.4%	82.2%

Table 5: Tagger Evaluation

The recall was significantly improved, while the precision decreased with only 1.8%. The reason we got a smaller than 100% precision with the baseline tagger resides in the inherent ambiguity of some of the protein names, and this problem was propagated in the final tagger too. One example found in the test abstracts is CSF (colony stimulating factor in OD), used there as an abbreviation for cerebrospinal fluid. Cell types present another case of conflicting names (NK1 receptor in OD vs. NK cell type in text). We intend to make use of the context surrounding protein names in order to disambiguate them and thus increase the precision.

We have also computed the precision and recall for a modified version of our tagger in which all protein names from the original dictionary were converted to lower case names, in order to asses the importance of discriminating between lower case and upper case names when performing recognition. The recall in this case was almost the same as that of the unmodified tagger, while the precision dropped to 74%. This result shows that case is important for protein name recognition.

4 Related Work

Automated information extraction from biomedical text is a growing area of research. Most of the work in the area has concerned the manual development of IE systems for extracting specific pieces of information (Fukuda et al., 1998; Humphreys et al., 2000; Blaschke and Valencia, 2001; Proux et al., 2000; Rindflesch et al., 2000; Thomas et al., 2000; Hahn et al., 2002; Ono et al., 2001). There has also been some prior work in the area of automatically learning IE systems for biomedical tasks (Craven and Kumlien, 1999; Ray and Craven, 2001; Eliassi-Rad and Shavlik, 2001). This work has addressed tasks such as identifying the subcellular structures in which proteins are located. To our knowledge, our work is the first attempt to identify gene/protein names starting from a dictionary, then extending it to contain generalized patterns of gene and protein names.

5 Future Work

Our research on extracting information from biomedical literature is in its beginning stages. This section outlines some of our plans to extend the current work.

First, we hope to improve our current manually-developed protein tagger using machine learning. In particular, we plan to explore "bootstrapping" techniques such as those described in (Riloff and Shepherd, 1997; Riloff and Jones, 1999) to enlarge the current dictionary. First, we will use the current tagger to locate protein names in large numbers of Medline abstracts. Next, this tagged text will be used as training examples to learn surrounding contexts that are indicative of protein names using existing IE-learners such as BWI. These learned patterns can be used to increase the recall of protein tagging and also to add new protein names to the dictionary (possibly after filtering by a human biologist). Finally, this overall process can be iterated to construct increasingly accurate taggers.

Next, we plan to use the protein tagger as an initial step in indentifying protein interactions from Medline abstracts. We intend to base our approach on *boosted wrapper induction* (BWI) (Freitag and Kushmerick, 2000), which uses *boosting* (Freund and Schapire, 1996) to improve the performance of a pattern-learning system initially developed fro extracting information from structured web pages (Kushmerick et al., 1997). We also intend to explore additional IE-learning techniques such as Rapier (Califf and Mooney, 1999), and hidden Markov models (Bikel et al., 1999; Freitag and McCallum, 2000) and their recent improvements (McCallum et al., 2000; Lafferty et al., 2001). We also intend to explore using additional syntactic information, such as that provided by a full parser, to improve extraction performance (Craven and Kumlien, 1999; Ray and Craven, 2001). We hope to greatly increase the amount of training data we can provide to our system by using information in the Database of Interacting Proteins to produce *weakly-labeled* training data (Craven and Kumlien, 1999). First, we need to find all sentences in a large collection of Medline abstracts that reference both elements of a pair of known interacting proteins. By assuming that such sentences actually assert that these proteins interact, large numbers of (potentially noisy) extraction training examples can be automatically generated. As with protein tagging, this process can be iterated in order to "bootstrap" an existing database of interacting proteins into a much larger collection by exploiting information in the biomedical literature.

We also hope to apply similar techniques to extract other important biological information such as gene functions and global gene relationships. In addition, as observed in (Blaschke and Valencia, 2001), the information to be extracted is frequently not contained in the abstract and is only available in the full text of the article. Consequently, we also hope to apply our methods to extract from complete articles in cases where full text is also available in electronic form.

6 Conclusions

This paper has presented initial results on extracting protein and gene names from Medline abstracts, which is the first step for a system aimed at extracting gene/protein interactions. Although the current results are promising, extensive additional research is required in order to eventually be able to transform the rich biological knowledge present in the wealth of biomedical literature into structured and accessible electronic form.

References

Daniel M. Bikel, Richard Schwartz, and Ralph M. Weischedel. 1999. An algorithm that learns what's in a name. *Machine Learning*, 34:211–232.

C. Blaschke and A. Valencia. 2001. Can bibliographic pointers for known biological data be found automatically? protein interactions as a case study. *Comparative and Functional Genomics*, 2:196–206.

Mary Elaine Califf and Raymond J. Mooney. 1999. Relational learning of pattern-match rules for information extraction. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI-99)*, pages 328–334, Orlando, FL, July.

M. Craven and J. Kumlien. 1999. Constructing biological knowledge bases by extracting information from text sources. In *Proceedings of the 7th International Conference on Intelligent Systems for Molecular Biology*, pages 77–86, Heidelberg, Germany.

T. Eliassi-Rad and J. Shavlik. 2001. A theory-refinement approach to information extraction. In *Proc. of 18th International Conference on Machine Learning (ICML-2001)*.

Dayne Freitag and Nicholas Kushmerick. 2000. Boosted wrapper induction. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI-2000)*, pages 577–583, Austin, TX, July. AAAI Press / The MIT Press.

Dayne Freitag and Andrew McCallum. 2000. Information extraction with HMM structures learned by stochastic optimization. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI-2000)*, Austin, TX. AAAI Press / The MIT Press.

Yoav Freund and Robert E. Schapire. 1996. Experiments with a new boosting algorithm. In Lorenza Saitta, editor, *Proceedings of the Thirteenth International Conference on Machine Learning (ICML-96)*. Morgan Kaufmann, July.

K. Fukuda, T. Tsunoda, A. Tamura, and T. Takagi. 1998. Information extraction: Identifying protein names from biological papers. In *Proceedings of the 3rd Pacific Symposium on Biocomputing*, pages 707–718.

U. Hahn, M. Romacker, and S. Schulz. 2002. Creating knowledge repositories from biomedical reports: The medsyndikate text mining system. In *Proceedings of the 7th Pacific Symposium on Biocomputing*, pages 338–349.

K Humphreys, G. Demetriou, and R. Geizauskas. 2000. Two applications of information extraction to biological science journal articles: Enzyme interactions and protein structure. In *Proceedings of the 5th Pacific Symposium on Biocomputing*, pages 502–513.

Nicholas Kushmerick, Daniel S. Weld, and Robert B. Doorenbos. 1997. Wrapper induction for information extraction. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence (IJCAI-97)*, pages 729–735, Nagoya, Japan.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. of 18th International Conference on Machine Learning (ICML-2001)*.

Andrew McCallum, Dayne Freitag, and Fernando Pereira. 2000. Maximum entropy Markov models for information extraction and segmentation. In *Proceedings of the Seventeenth International Conference on Machine Learning (ICML-2000)*, Stanford, CA, June.

T. Ono, H. Hishigaki, A. Tanigami, and T. Takagi. 2001. Automated extraction of information on protein-protein interactions from the biological literature. *Bioinformatics*, 17(2):155–161.

D. Proux, F. Rechenmann, and L. Julliard. 2000. A pragmatic information extraction strategy for gathering data on genetic interactions. In *Proceedings of the 9th International Conference on Intelligent Systems for Molecular Biology*, pages 279–85.

S. Ray and M. Craven. 2001. Representing sentence structure in hidden Markov models for information extraction. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI-2001)*, pages 1273–1279, Seattle, WA.

Ellen Riloff and Rosie Jones. 1999. Learning dictionaries for information extraction by multi-level bootstrapping. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI-99)*, pages 1044–1049, Orlando, FL, July.

Ellen Riloff and Jessica Shepherd. 1997. A corpus-based approach for building semantic lexicons. In *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing (EMNLP-97)*, pages 117–124, Providence, RI.

T.C. Rindflesch, L. Tanabe, J.N. Weinstein, and L. Hunter. 2000. EDGAR: Extraction of drugs, genes, and relations from the biomedical literature. In *Proceedings of the 5th Pacific Symposium on Biocomputing*, pages 515–524.

J. Thomas, D. Milward, C. Ouzounis, S Pulman, and M. Carol. 2000. Automatic extraction of protein interactions from scientific abstracts. In *Proceedings of the 5th Pacific Symposium on Biocomputing*, pages 541–553.