Title
Text Mining of Point Mutation Information from Biomedical Literature

Author
Lee, Lawrence Chet-Lun

Publication Date
2008

Peer reviewed|Thesis/dissertation

Text Mining of Point Mutation Information from Biomedical Literature

by

Lawrence Chet-Lun Lee

DISSERTATION

Submitted in partial satisfaction of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in

Program in Biological and Medical Informatics

in the

[redacted]

Text Mining of Point Mutation Information from Biomedical Literature

To Florence Horn, who inspired me with her hard-work and determination, and to whom I owe the fruits of my thesis.

Acknowledgments

Many friends and family were a part of my life during my time as a graduate student. Some have come and gone over the course of my studies, others have entered my life in the middle of grad school, and few have been constants in my life since before I can even remember. Each have impacted me in a way that cannot be measured individually, and I owe each and every one of you my thanks for being in my life.

First, to Wai Mun and Tracy Lee, my parents, thank you for teaching me the principles of hard work and dedication. Thank you for providing me with an education, and thank you for believing in my abilities to achieve. I could have done without the Kumon and Chinese school classes when I was young, though. To my sister, Pearl, and brother Leonard, thanks for providing me the unconditional support which comes with being siblings. While we aren't the young kids that we once were, you still have a dedicated place in my life.

To Nithin and Sheetal, who lovingly adopted me during the last year of my graduate career. I definitely couldn't have done it without you guys. Both of you are like an extended family for me, and I consider myself lucky every day to have such great friends.

To Clifford and Maria, who watched me grow as a person at the same time I watched them grow as a couple. Clifford has been one of my dearest friends throughout all the stages of my life, and to have Maria as a part of that makes it more special. I know Clifford's jealousy of my degree has caused the temporary insanity required to go back to school, but both of you will have my support and love in whatever endeavors you undertake.

To William, whom I have known to be a student even longer than I have been

one. I will miss our student privileges of partying and golfing on the weekdays. I know soon enough you will be in the ranks of the worker bees, and we will retire our weekday activities to just golfing on the weekends. Thanks for all your friendship and support while I've been in school.

To Satish, my original and continuing partner in crime. It's good to know that after all these years we are still young at heart. Your energy and friendship have helped me get through the good times and the bad.

To Yumi, my Hong Kong travel companion and long time party friend. We have become much more than just drinking buddies, although you might as well have put your signature on my damaged liver. I remember the days when we were young and energetic with great fondness, and would not trade those memories for anything... even graduating a year or two faster! Irish car bombs!

To Josh, my erstwhile roommate and permanent friend. I miss our 99 Walter shenanegans and late night Castellito runs. We may never join a bowling league together again, but I'll probably be a better person for it...

To my former labmates, Cedric, Anthony, Barney, Alex S., Alex B., Erik, Jay, and Jenni, thanks for injecting a little bit of spice in my otherwise bland grad student experience. It was a pleasure to share some space with you on your way to bigger and better things, and I miss you all.

To Denise, who has been a greater influence on my life than anybody else I know. The results of my work belong to you just as much as it does to me. Thank you for having been a part of my life.

Abstract

Text Mining of Point Mutation Information from Biomedical Literature

by

Lawrence Chet-Lun Lee

Doctor of Philosophy in Progam in Biological and Medical Informatics

University of California, San Francisco

Professor Fred E Cohen, Chair

Text mining is a powerful approach to efficiently identify and extract information from large amounts of text. Its application to biomedical literature promises to allow researchers the ability to process hundreds and thousands of articles in a way that was never possible before. This new technology, coupled with the increasing amount of published literature and open access literature sources may usher in a new era of meta-research in which computational algorithms can generate new scientific hypotheses from existing information.

Many obstacles, however, stand in the way of truly automated methods for text mining. The difficulty in obtaining full text literature, specialized jargon in scientific research, and the ambiguity of biological entity terms are but a few of the challenges that need to be overcome. That being said, semi-automated text mining methods can still greatly aid researchers by identifying topics of interest, reducing the number of articles to read, and targeting relevant information in articles. The judicious and dedicated use of semi-

automated methods has the possibility of having a great impact in efficiently distributing the task of manual reading and processing of scientific literature.

Our contribution to semi-automated methods for biomedical text mining center on the identification and extraction of point mutation information. Point mutations are an integral aspect of protein research, as they are the vehicle of diversity and the key to functional changes in proteins. They are also represented in a format that lends itself to text mining and can be referenced to the growing numbers of biological sequence databases. This dissertation focuses on the ability to parse literature for point mutations and extract their functional effects. We show that, using statistical, graph theoretical, and machine learning methods, we can efficiently transform information that was previously embedded in the text into information that is computationally stored and processable.

Contents

List of Figures

List of Tables

Chapter 1

Introduction and Background

1.1 Introduction

1.1.1 Importance of Point Mutations

The goal of bioinformatics and computational biology is to leverage quantitative methods to solve biological problems. This is an overarching description, and a wide selection of research can be described as bioinformatics. If we narrow the field of bioinformatics to that which attempts to cure human diseases, the breadth of research generally focuses on the function and dysfunction of proteins in the human body. Most of the major heritable diseases are a result of dysfunctional proteins in key metabolic pathways, which are a result of amino acid deletions, insertions, fusions, point mutations, or amplification of unstable sequences. A range of genetic disorders including blood-based, cancers, nervous system, and neonatal diseases can be linked to the altered functions of proteins. Consequently, the study of inherited genomic variations plays an important role in the understanding, treatment, and cure of these diseases.

Point mutations can refer to changes in the DNA sequence of an organism as well. From an evolutionary perspective, point mutations, or sequence variations, can be markers for the divergence of protein sequences. Often these DNA point mutations are synonymous, or silent, in nature, but their presence or absence in orthologous species can indicate common ancestry and reveal characteristics about the phylogeny of the species and the protein. The variations in the same protein across many different organisms can reveal how different adaptations affect the functional aspects of the protein.

While naturally occuring genetic variations can manifest themselves as disorders, investigator-generated mutations are used to explore the overall function of proteins. By

cloning and altering a gene and expressing a protein, researchers are able to vary the amino acid sequence of the protein by point mutations, insertions, and deletions, thereby changing the structure and possibly the ability of the protein to function properly. When used to create point mutations, this strategy is called site-directed mutagenesis and is an indispensible tool for studying structure-function relationships. A researcher can create a mutation at an amino acid that is thought to be important for protein function; if an assay reveals no change after the point mutation, it can be concluded that the particular residue is not critical for function. On the other hand, if the protein ceases to function normally, the mutated residue position is thought to have functional significance.

Point mutations are generally represented in text as single-letter or three-letter amino acid abbreviations. For example, the replacement of an alanine to a threonine at position 100 of the protein sequence could be represented as `A100T` or `Ala100Thr`. Sometimes, characters are added between the amino acid characters and the sequence position to generate terms such as `A(100)T`, `A-100-T`, or `A100→T`.

1.1.2 Point Mutation Databases

It is not surprising, given the relative importance of point mutations to biological research, that many electronic databases exist that catalog these mutations. Many disease specific databases exist, ranging from prion point mutations to haemophilia causing mutations. OMIM (Online Mendelian Inheritance in Man) [**?**], is arguably the most important and trusted database of human genes and genetic disorders, with over 18,000 gene/disease associations as of this writing. Its sister database, OMIA (Online Mendelian Inheritance in Animals) [Lenffer et al., 2006], dbSNP [Sherry et al., 2001] is a database of single-nucelotide-

polymophisms (SNPs) across many organisms. dbGaP[1] is a database that links genotypes to phenotypes based on experimental evidence. Much larger databases such as GenBank [Benson et al., 2004] and UniProt [Boutet et al., 2007] also contain mutation or sequence variation information within their entries.

While these databases are extremely helpful, they require manual curation to remain current and are dependent on the altruism and accuracy of contributing researchers. Depending on the level and frequency of curation, these databases will have varying degrees of coverage over their respective bodies of scientific literature. What is desired, then, is to have computational methods to expedite and aid the curation process. If dedicated curators could leverage high-efficiency text mining applications to identify articles of interest and/or extract relevant data, the speed and coverage of the curation efforts will increase accordingly. It is necessary for text mining researchers to determine which types of information and data are pertinent for each type of database, and then design tools accordingly.

1.2 General Text Mining

There exists many categories and subcategories for computational research in spoken and written natural language processing (NLP), commonly referred to as "computational linguistics" or "statistical natural language processing". The books by Manning and Schütze [1999] and Jurafsky and Martin [2000] provide a good foundation and solutions for basic problems. These include finding commonly co-located words, part-of-speech tagging, document classification, and information extraction. The proliferation of the internet

[1] http://www.ncbi.nlm.nih.gov/sites/entrez?db=gap

and electronic documents has expanded these problems to a whole new scope and level of sophistication. The phrase "text mining" most commonly refers to the use of advanced computational methods to identify and extract information from text. Topics such as artificial intelligence, information theory, information management, and operations management have all been applied to text mining in some form. Users want to be able to identify and extract relevant textual information from the sea of electronic textual information on the internet.

Internet search engines endeavor to serve end users by implementing sophisticated algorithms to retrieve web pages corresponding to specific queries. This is probably the most widespread and common "text mining" application implemented. On a much smaller scale, business intelligence companies seek to parse internal documents to identify new trends, needs, or problems without manual processing. As pervasive as textual information in the internet and personal computing era has become, applications that can efficiently and seamlessly allow users to search for information in text is becoming a need rather than a luxury.

1.2.1 Text Mining Terminology

While it is impossible to provide an overview of all text mining and statistical natural language processing concepts, it is important to provide the definitions of some terms that will be used frequently in the remainder of this manuscript. The concept of statistical inference in NLP is the ability to predict patterns or occurences in text based on previous known instances. One example is predicting the next word in a sentence given the previous words. This is a stochastic problem that can be represented by a probability

function P given in Manning and Schütze [1999]:

$$P(w_n|w_1, \ldots, w_{n-1}) \tag{1.1}$$

While one could chose an arbitrary number n for this equation, people usually use sequences of 2, 3, and 4 words for predictive models, which are commonly referred to as *bigrams*, *trigrams*, and *four-grams*. The general term for a predictive model based on the histories of previous words is an n-gram model. I extensively use *bigrams* in my text mining algorithms for point mutations. Another way of thinking about an n-gram is simply n consecutive words in a sentence; a bigram would be two consecutive words such as "White House" or "cordless phone".

Words can be loosely categorized by their *part-of-speech* (POS) such as *nouns*, *verbs*, *adverbs*, and *articles*. The method of POS-tagging, or simply *tagging*, is to assign words their POS based on prior information. Tagging, in fact, is very similar to sequence analysis of DNA or amino acids. For example, many tools exist to predict the secondary structure of proteins based on their sequence information. Each amino acid in the sequence is given a label for a particular secondary structure and consecutive amino acids with the same label are considered to have that structure. In tagging, each word of a sentence is given a POS label using probabilistic models. One such example is the sentence:

1. `R31C and R31L Are Internalized More Rapidly than the Wild-type CFTR.`

which, when tagged produces:

2. `R31C/NN and/CC R31L/NN Are/VBP Internalized/VBN More/RB Rapidly/RB than/IN the/DT Wild-type/NN CFTR/NN ./.`

Within this example, /NN represents a noun, /VBP a verb, /VBN a verb, past participle, /RB an adverb, /IN a preposition, /DT a singular determiner, and /CC a conjunction.

In probabilistic parsing of text, the standard terms *parse* and *chunk* have similar meanings but different implementations. We can first define chunking as "recognizing higher level units of structure that allow us to compress our description of a sentence" [Manning and Schütze, 1999]. A simple chunker can just place boundaries between words in a sentence to separate phrases. The result of passing a tagged sentence into a chunking algorithm may output something like this:

3. `[NP R31C/NN and/CC R31L/NN ] [VP Are/VBP Internalized/VBN ] [ADVP More/RB Rapidly/RB ] [PP than/IN ] [NP the/DT Wild-type/NN CFTR/NN ]`

The chunked phrases are marked by the [and] brackets, and the first word in each bracket denotes the phrase type (i.e. NP for noun phrase and VP for verb phrase). When a tagged sentence is parsed, the units of structure labeled can be more detailed and nested. Note that in the chunked sentence, each chunk is non-overlapping and there are no chunks within chunks. Parsing seeks to recursively break the sentence down into smaller structures until only the word remains. The standard output of a parser is a *parse tree* according to a predifined grammar. If the tagged sentence (2) above is parsed, the parse tree may look like:

4.
```
(TOP
        (S
            (NP
                (NN R31C)
                (CC and)
                (NN R31L))
            (VP
```

```
(VBP Are)
(PP
    (VBN Internalized)
        (ADVP
            (RB More)
            (RB Rapidly)))
        (PP
            (IN than)
                (NP
                    (DT the)
                        (NN Wild-type)
                        (NN CFTR))))))
```

The very top of the tree is labeled as `TOP` and the next level down is labeled `S` for sentence. Two branches extend from `S`, the `NP` and `VP`. The main difference between a parse and a chunk of a sentence is the nested structure of the parse tree. One can think of a chunked sentence as a parse tree that is only one level high.

Text mining also has different metrics for performance evaluation. While scientific tests are usually evaluated by sensitivity and specificity, information identification and extraction methods are evaluated by *precision*, *recall*, and *F-measure*. Precision is the percentage of true classifications that are actually true and recall is the percentage of all actual true instances that were classified as true. F-measure is the harmonic mean of the precision and recall. These values can be calculated with these equations:

$$\begin{aligned} \text{Precision} &= TP/(TP+FP) \\ \text{Recall} &= TP/(TP+FN) \\ \text{F-measure} &= 2(\text{precision} \cdot \text{recall})/(\text{precision} + \text{recall}) \end{aligned} \tag{1.2}$$

These terms by no means encompass the majority of text mining concepts as a whole, but they will be useful to understand when reading the later chapters of this thesis.

1.3 Text mining in Biomedical Literature

When text mining is applied to biomedical literature, many interesting problems and opportunities present themselves. Primarily, biological research has many defined standards and goals. Hypotheses are created and empirically tested in a thorough manner, and the interesting results are published as journal literature. If it is possible to computationally process biological entities, experimental results, and concepts, the process of generating new hypotheses may be aided by prior textual information.

The earliest text mining applications for biomedical literature focused on the identification of biological terms. Since different fields in biology have different idiosyncratic nomenclatures, sometimes two terms will reference separate entities. This is especially true for gene and protein names. More often, however, are instances where a single gene or protein will have many synonyms that possess very little in common. One such example are the synonyms for the Scn4a protein (Swiss-Prot AC: P15390), whose synonyms are "Mu-1", "microI", "Voltage-gated sodium channel alpha subunit Nav1.4", "Nav1.4", "Sodium channel protein type IV alpha subunit", "NCHVS", and "Sodium channel protein, skeletal muscle alpha-subunit", as given by UniProt and Entrez Gene[2]. Approaches to identifying biological terms include dictionary and ruled-based approaches, machine learning approaches, and mixtures of both.

Once biological terms could be identified with reasonable accuracy, researchers tackled the task of extracting relationships between terms. The most common relationship extracted are protein-protein interactions, but variations of gene-protein, drug-protein,

[2]http://www.ncbi.nlm.nih.gov/sites/entrez?db=gene

protein-disease, and drug-disease relationships have also been extracted. All these relationships are derived from a common "entity-fact" association; in this case, two biological entities are identified and searched for a relationship fact. The benefit of creating these relationship networks is to allow somebody to have a broad overview of interactions between genes, proteins, drugs, and diseases. If we can think of genes giving rise to multiple proteins, proteins being the cause of multiple diseases, and diseases being treated by multiple drugs, then the power of automatically extracting these interactons is obvious.

Blaschke et al. [1999] authored one of the first papers describing a method for extracting protein-protein interactions. The authors used a template based approach that identified protein terms in abstracts from PubMed. Once protein terms were identified, they looked for certain "action" terms such as *activate*, *bind*, *destabilize*, *inhibit*, or *interact* between two protein terms. The pattern searched for is essentially `<protein A> <action> <protein B>`, where the `action` term describes the type of interaction between `protein A` and `protein B`. Using this approach, they were able to somewhat successfully recreate certain *Drosophilia* protein networks by abstract analysis. Daraselia et al. [2004] extended this approach by using a probabilistic context-free grammar parser to create multiple semantic variations of a sentence. Then an ontology filter is used to validate whether a particular protein-protein interaction is possible by analyzing the types of entities and interactions involved.

The BioCreAtIvE Challenge [Hirschman et al., 2005b] was created to evaluate text mining and information extraction systems as applied to the biological domain. The challenge is divided into three tasks: (1) gene mention tagging, (2) gene normalizaton, and

(3) protein-protein interaction extraction. The first component entails identifying mentions of gene and protein terms from scientific literature and the second task involves assigning each gene or protein mention a unique Entrez Gene identifier. The challenge was very successful in providing researchers with a standardized and pre-annotated set of text for training and testing, and it also generated many new approaches to solving the three tasks.

The Text Retrieval Conference (TREC) Genomics Track [Hersh et al., 2004] is another biological natural language processing and text-mining challenge. The goal of the TREC Genomics Track has varied from year to year, and in the last incarnation, researchers were given the task of assigning a topic to a passage of text. The goal was to use contextual evidence in the passage to classify it. Some of the "topics" included "antibodies", "drugs", "genes", and "pathways".

The KDD Cup [Yeh et al., 2003] offered two tasks for its competition: information extraction from biomedical articles and yeast gene regulation prediction. For the information extraction task, the goal was for researchers to determine whether a journal article contained relevant gene-expression information for *Drosophila* and is a candidate for Flybase database curation. The yeast gene regulation prediction task required finding the "activity class" of knocked out genes in *S. cerevisiae*.

Chapter 2

Point Mutation Text Mining Methods

2.1 Introduction

In the previous section, we discussed the importance of point mutations to biological research, general text mining goals and approaches, and text mining methods as applied to biomedical literature. In this section, we will specifically address prior methods of text mining as applied to the identificaton and extraction of point mutation information. Notably, three prior methods have laid the foundation for and framed the problem of point mutation extraction.

2.2 MEMA

MEMA, developed by Rebholz-Schuhmann et al. [2004], was the first application to identify point mutations in biomedical literature. A more detailed comparison of the performance differences between MEMA and Mutation GraB is discussed in Chapter 3.5.1 on page 59, but I will describe the general methods MEMA uses for its text mining application. MEMA automatically extracts point mutations from Medline abstracts and consists of separate modules, or components for:

1. Identification of gene names.
2. Identification of point mutation terms.
3. Disambiguation of gene to mutation associations.

2.2.1 Methods

The identification modules for gene and point mutation terms are based on regular expressions. For gene names, variations on upper and lower case combinations of characters are generated. An example given is the gene name "COL1A1", which is parsed to generate the patterns "COL1A1" and "[Cc]ol1a1". In the second pattern, the first character is allowed to be either upper or lower case, and the remaining letters are specified to be lower case. MEMA used the gene names from HUGO to build its dictionary used for searching.

For the point mutation terms, MEMA utilized the regular expression "[AC-IK-NP-TVWYZ][0-9]+[AC-IK-NP-TVWYZ]", which can capture words such as "C282Y" and "K300W". These words represent the common abbreviation for point mutations where the characters on either end of the word are single letter forms of amino acids. This pattern also matches DNA mutations, and unfortunately, there is no simple way to disambiguate between DNA and protein point mutations in text from analyzing only the point mutation term.

The disambiguation component determines which gene to associate with a point mutation if multiple gene names are identified within an abstract. If only one gene term is identified within an abstract, then all point mutations identified in the same abstract are automatically associated with that gene. If multiple genes are found in an abstract, then sentence co-location is used as a metric for gene-mutation association. Finally, if multiple genes are found in a sentence, then syntactical rules and proximity parameters were used to disambiguate.

Table 2.1: Results of MEMA on 100 PubMed Abstracts.

	Recall	**Precision**
Contained	0.352	0.986
Cited	0.747	0.934

2.2.2 Results

MEMA was evaluated by two categories of point mutation identification: "contained" and "cited". "Contained" refers to whether an abstract reports a mutation-gene pair, whereas "cited" refers to instances of mutations alone. Since an abstract may contain multiple mutation-gene pairs, the "contained" measurement is more stringent.

Table 2.1 shows results for MEMA when applied to 100 PubMed abstracts. Both "contained" and "cited" results show high precision but low recall, indicating that many point mutations are missed. In fact, the requirement for finding a mutation-gene pair in the abstract lowered recall from 0.747 to 0.352. While this is a decent result for an initial attempt at solving the problem of point mutation extraction, MEMA still contains many shortcomings which is further addressed by other methods.

2.3 MuteXt

The foundation for my thesis work was the MuteXt application by Horn et al. [2004]. The goal for MuteXt was to extract point mutations from literature for protein family databases using a set of expert generated regular expression patterns. Specifically, MuteXt extracted point mutations from G protein-coupled receptors and nuclear hormone

receptors for the GPCRDB[1] and NuclearDB[2] databases. MuteXt was a landmark application for point mutation extraction for a number of reasons. First, instead of examining only abstracts, MuteXt extracted point mutations from full-text HTML and PDF sources. This allowed MuteXt access to a far greater amount of point mutations than examining abstracts alone. Second, MuteXt validated its extracted point mutations by comparing the amino acid at the mutation position to a standardized sequence database, in this case Swiss-Prot [Boeckmann et al., 2003]. In doing so, a built-in filter for false positive mutations was created, and it allowed for a direct association between a point mutation and a unique entry in a protein database.

2.3.1 Methods

Figure 2.1 shows the process flow for MuteXt. First, a PubMed query identifies the relevant abstracts for extraction. If a full-text article is available, an attempt is made to retrieve the PDF or HTML source. Then, a number of post-processing steps are done to either extract the raw text from the PDF or to remove the tags from the HTML source. Mutation regular expressions are used to identify point mutation terms in the text, and dictionary terms from Swiss-Prot are used to identify protein and organism terms. The regular expression used to identify point mutation terms was:

[1] http://www.gpcr.org/7tm/mutation/
[2] http://www.receptors.org/NR/mutation/

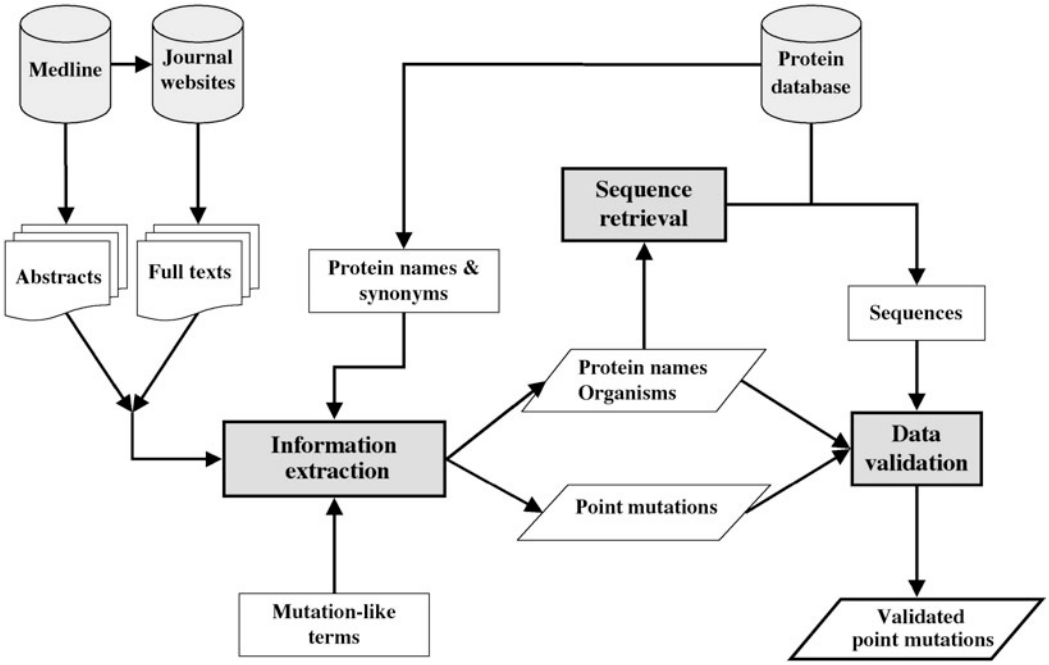


Figure 2.1: The process flow for MuteXt. Full text articles are downloaded from journal websites and processed to identify point mutation, protein, and organism name terms. The protein and organism names are referenced in Swiss-Prot to retrieve a unique protein entry, and the sequence for the protein is compared to the wild-type amino acid of the point mutation. If this comparision matches, then point mutation is validated.

$$([A-Z][1-9][0-9]+\$)|([A-Z][1-9][0-9]*[A-Z]\$)|$$
$$([A-Z][a-z][a-z][1-9][0-9]*\$)|$$
$$([A-Z][a-z][a-z][1-9][0-9]*[A-Z][a-z][a-z]\$) \quad (2.1)$$

In order to pinpoint the correct Swiss-Prot entry, MuteXt needed to identify the correct gene/protein and organism name. Because many orthologous proteins have the same identifier, the organism name is necessary for differentiation. The "triangulation" of point mutation, protein, and organism name terms results in having each point mutation associated with a unique Swiss-Prot identifier. If no Swiss-Prot protein can be found, the point mutation is labeled a true negative. If multiple Swiss-Prot identifiers match the sequence validation step a *distance filter* is employed to find the correct identifier. This distance filter calculates the shortest number of words that lie between the pairs of {point mutation, protein name}, {point mutation, organism name}, and {protein name, organism name} terms. The smallest sum of these distances that results in a unique Swiss-Prot identifier for the protein and organism names is assigned to the point mutation.

2.3.2 Results

The results of applying MuteXt to the GPCR and NR literature is shown in Table 2.2 and 2.3, respectively. MuteXt achieves a high F-measure of 0.875 for the GPCR literature set and 0.872 for the NR literature set. In addition to the high accuracy of extraction, Horn et al. [2004] was able to show that the automated extraction from full text literature retrieved a larger set of mutations than was present in the manually curated

Table 2.2: Results of MuteXt extraction on GPCR literature sources.

	Abstracts	**HTML full text**	**PDF full text**	**Total**
Articles	360	368	337	732
Point mutations	922	2448	2829	3996
Precision	0.884	0.908	0.873	0.879
Recall	0.829	0.872	0.871	0.871
F-measure	0.855	0.890	0.872	0.875

Table 2.3: Results of MuteXt extraction on NR literature sources.

	Abstracts	**HTML full text**	**PDF full text**	**Total**
Articles	73	225	258	343
Point mutations	167	947	1108	1338
Precision	0.904	0.857	0.860	0.858
Recall	0.825	0.896	0.894	0.887
F-measure	0.863	0.876	0.876	0.872

tinyGRAP database for GPCR mutatons. It can be conclusively stated that the utilization of MuteXt for these protein families generates a high quality set of point mutations that would have taken exponentially longer to identify manually.

From an end-user standpoint, MuteXt was a great application to extract point mutations for deposit into curated databases. It allowed researchers to quickly scan through hundreds of articles in a fraction of the time as manual curation would have taken. However, MuteXt was not without certain drawbacks. Since the programmer for MuteXt was also a curator of the point mutation databases, high precision and recall were prioritized at the cost of program generality. Many hand-coded rules were used in identifying the proper protein names and point mutation terms that were specific to the GPCR and NR family nomenclature. While this is helpful for the application towards those protein families, those hand-coded rules are less helpful when processing other protein families. Also, the distance

rules for determining association between a point mutation and a unique protein identifier is suboptimal in certain situations. Some hand coded rules were implemented to bypass the distance rules for some frequently seen point mutations, and evaluation of the rules is difficult as a result.

The overall benefits of MuteXt, however, cannot be discounted by these areas of desired improvement. Besides being the first paper of its kind to extract point mutations in a high-throughput fashion, MuteXt generated quantifiable data that confirmed our suspicion about textual information. By offering a direct comparison to manually curated databases, MuteXt proved that text mining alone cannot duplicate the coverage offered by manual curation. Partially due to the inaccessibility of full text articles, MuteXt was only able to identify 2254 out of 5451 mutations in the GPCR and NR databases. MuteXt was also the first application that extracted information from full text articles instead of Abstracts alone, and the knowledge gained from retrieving and processing full text articles was indispensable for my subsequent research.

2.4 Mutation Miner

The Mutation Miner application described in Baker and Witte [2004], Witte and Baker [2005], Baker and Witte [2006] is also one of the first applications of point mutation extraction from biomedical literature. However, instead of a stand-alone text mining application, Mutation Miner also maps the extracted point mutations to the 3D structure of the protein. This multifaceted approach provides an end user with a vehicle for identifying point mutation terms in text and viewing the mutations in its protein of origin, which is

extremely useful and powerful for examining the structural effects of point mutations. A more detailed analysis of Mutation Miner versus Mutation GraB is provided in Chapter 3.5.2 on page 61. In this section, I will focus mainly on the text mining component of Mutation Miner. This text mining component consists of multiple subcomponents:

1. Text preprocessing;
2. Gene name and polymorphism identification;
3. Noun phrase chunking; and
4. Relation detection and disambiguation.

2.4.1 Methods

The preprocessing component tokenizes the text and provides a major and minor type label for each token. These type assignments can be chemicals, drugs, protein names, person names, or measurements. The biomedical dictionary list was built from MeSH and Swiss-Prot terms. Next, the text is split into sentences and then part-of-speech tagged using the Hepple taggerHepple [2000].

A finite state transducer based on regular expressions is used to combine individual tokens into named entities (NEs). These NEs consist of persons, protein expressions, database accession identifiers, and mutation expressions. It is not clear the difference between this step and the aforementioned major and minor labels. Another finite state transducer is applied to group tokens into noun phrases, or multiple consecutive and non-overlapping words which represent a larger noun entitiy. For example, the three tokens

"xylanase", "II", and "proteins" would be grouped into one noun phrase "xylanase II proteins".

The final and most important component identifies relations between mutations and proteins and between proteins and organisms. This relation detection algorithm is simply a sentence co-location metric, and an implicit assumption is made that co-occurence of point mutation and protein terms denotes a biological relationship between the two. Again, like MuteXt, Mutation Miner finds the protein of origin by using a protein and organism name as identifiers. However, unlike MuteXt, when Mutation Miner does not identify a organism name, the point mutation is discarded.

Similar to MuteXt, Mutation Miner queries PubMed for articles describing point mutations in a protein family of interest. Baker and Witte [2006] chose to study point mutations from the xylanase, haloalkane dehalogenase, and biphenyl dioxygenase proteins. Unfortunately, Baker and Witte [2006] did not evaluate their mutation extraction component specifically, but since their method is not as thorough as the one described in Horn et al. [2004], it is not expected to perform better in terms of precsion and recall.

2.5 MutationFinder

The MutationFinder application described in Caporaso et al. [2007a,b] attempts the less ambitious task of point mutation term identification without association to a gene/protein term. The authors state that a high precision and recall point mutation term identifier is more important than an application that both identifies and associates with gene/protein terms poorly. Their point is well taken, and MutationFinder has been

shown to achieve a high precision and recall measure over a diverse data set.

2.5.1 Methods

In their first publication [Caporaso et al., 2007a], MutationFinder was compared to the MuteXt Horn et al. [2004] as a baseline performance level. Like the previously described applications, MutationFinder uses regular expression patterns to identify not only one- and three- letter abbreviation of point mutations, but point mutations described grammatically as well. MutationFinder builds upon MuteXt in six ways:

1. wNm format mentions with one-letter abbreviations must have N>9.
2. wNm format mentions with one-letter abbreviations must appear in upper-case letters.
3. Wild-type and mutant residue/base identities must not be the same.
4. MutationFinder specifies patterns incorporating non-alphanumeric characters, whereas MuteXt removes non-alphanumeric.
5. MutationFinder identifies mutations described in natural language with specific patterns.
6. MutationFinder splits text on sentences and applies its regular expressions to each sentence, whereas MuteXt splits both on words and sentences and applies different regular expressions to each.

MutationFinder was augmented in Caporaso et al. [2007b] by adding a pattern refinement and addition component. This component is given text which contains point

Table 2.4: Contents of Gold Standard and Testing Corpus for MutationFinder

	DEV	**TEST**	**TOTAL**
Documents	305	508	813
Documents containing mutation mentions	111	212	323
Documents not containing mutation mentions	194	296	490
Point mutation annotations (partial)	605 (56)	910 (150)	1515 (206)
Insert annotations (partial)	0 (0)	0 (3)	0 (3)
Deletion annotations (partial)	4 (0)	10 (5)	14 (5)

mutation descriptions and "learns" a regular expression pattern that can identify that block of text. Patterns are iteratively added and refined so that a minimal number of patterns can identify an entire set of given point mutation descriptions. The steps of pattern refinement is shown in Figure 2.2. While the initial patterns were automatically generated, the final patterns used for testing were refined manually, modified, and re-tested to generate the final performance figures.

2.5.2 Results

The first iteration of MutationFinder was compared to MuteXt in the extraction of 910 point mutations from 508 abstracts. The corpus composition is detailed in Table 2.4. Partial point mutations are ones in which a wild-type or mutant amino acid is absent from the description, but the sequence position remains.

The results of MutationFinder on the corpus are shown in Table 2.5. In the table *Extracted Mentions* refers to the identification of all point mutation terms in the text, including duplicate mentions. *Normalized Mentions* refers to the identification of at least one instance of point mutations in the text, regardless of duplicate instances, and *Document Retrieval* represents the ability of the application to denote whether a point mutation

	Step 1.1 result: text containing mentions identified (Sec. 3.1.1)	**Step 1.2 result:** raw patterns generated (Sec. 3.1.2)	**Step 2 result:** raw patterns refined (Sec. 3.2)	**Step 3 result:** mutation patterns generated (Sec. 3.3)
(a)	*"the **Ala64Gly** mutation"*	`RESPOSRES`	`RESPOSRES` *(no change)*	`WRESSPOSMRES`
(b)	*"the **Ala64→Gly**"*	`RESPOS-->RES`	`RESPOS-->RES` *(no change)*	`WRESSPOS-->MRES`
(c)	*"we mutated **Ala64** to **glycine**"*	`RESPOS to RES`	`RESPOS to RES` *(no change)*	`WRESSPOS to MRES`
(d)	*"we mutated **Ala64** to **glycine**, **leucine**, and **valine**"*	`RES-POS to RES, RES, and RES`	`RES-POS to RES, RES, and RES` *(no change)*	`WRES-SPOS to MRES` `WRES-SPOS to RES, MRES` `WRES-SPOS to RES, RES, and MRES`
(e)	*"**alanine** was mutated to **glycine** at positions **64** and **72**"*	`RES was mutated to RES at positions POS and POS`	`RES was mutated to RES at positions POS and POS` *(no change)*	`WRES was mutated to MRES at positions SPOS` `WRES was mutated to MRES at positions POS and SPOS`
(f)	*"the catalytic residues (**Asp325**, **Glu354**, and **Asp421**) are necessary"*	`RESPOS, RESPOS, and RESPOS`	False positive, manually eliminated (Sec. 3.2.2)	*Not applicable*
(g)	*"A **pro**-memoria on the occasion of the **200**th anniversary of **his** death"*	`RES-memoria on the occasion of the POSth anniversary of RES`	False positive, automatically eliminated (one Medline occurrence) (Sec. 3.2.1)	*Not applicable*
(h)	*"the **Ala64Gly** and **Ala72Thr** mutations"*	`RESPOSRES and RESPOSRES`	Manually eliminated (two occurrences of row (a) pattern) (Sec. 3.2.3)	*Not applicable*
	Step 4 result: regular expressions generated *(row (a) example only)* (Sec. 3.4)	`(^|[\s\(\[\'"/,\-])(?P<wt_res>(RES_PATTERN))` `(?P<pos>[1-9][0-9]*)` `(?P<mut_res>(RES_PATTERN))(?=([\.,\s)\]\'":;\-?!\]|\$))`		

Figure 2.2: The steps taken by MutationFinder to identify and refine regular expression patterns for point mutation term identification..

Table 2.5: Results of MuteXt and MutationFinder on 508 Abstracts.

	MuteXt			**MutationFinder**		
	P	**R**	**F-measure**	**P**	**R**	**F-measure**
Extracted Mentions	0.940	0.644	0.764	0.984	0.819	0.894
Normalized Mentions	0.918	0.685	0.785	0.975	0.807	0.883
Document Retrieval	0.946	0.769	0.849	0.994	0.890	0.939

can be found in the abstract or not. The additional steps built into MutationFinder on top of MuteXt provided an increase in F-measure of approximately 0.1 for *Extracted* and *Normalized* mentions as well as *Document Retrieval* performance.

While the mutation term identification capabilities of MutationFinder is comprehensive and performs well on a multitude of corpuses, the lack of ability to associate the point mutations with protein/gene terms limits the functionality of the application. As most, if not all, uses of point mutation extraction necessitates the examination of the point mutations at a protein level, the absence of protein information negates the utility of the point mutation terms. An ideal situation would be to pair the MutationFinder process with another tool for protein association, such as Mutation GraB. The authors have hinted that this is a future addition to the MutationFinder application.

2.6 Conclusion

While a number of researchers have attempted to tackle the problem of point mutation identification and extraction, none of their solutions have proven to be comprehensive. The main problems encountered by all previous methods include point mutation term identification, gene/protein name identification, and gene/protein to mutation association. While

MutationFinder has proven to be highly successful at identifying point mutation terms, the lack of a component to associate a gene or protein renders the application almost useless to most biological researchers. Mutation Miner and MEMA, on the other hand, are able to associate point mutations with genes or proteins, but their text mining capabilities are not as strong as MutationFinder, and their association function is not as comprehensive as that of MuteXt. In the remainder of this thesis I will describe the MutationGraB application, which seeks to integrate an improved point mutation term identifier with a sophistcated graph approach to associate point mutations to their proteins of origin.

Chapter 3

Identification of Point Mutation Terms - MutationGraB

3.1 Abstract

The principal problem of point mutation extraction is to link the point mutation with its associated protein and organism of origin. Our algorithm uses a graph-based bigram traversal to identify these relevant associations and exploits the Swiss-Prot protein database to verify this information. The graph bigram method is different from other models for point mutation extraction in that it incorporates frequency and positional data of all terms in an article to drive the point mutation-protein association. Our method was tested on 589 articles describing point mutations from the G protein-coupled receptor (GPCR), tyrosine kinase, and ion channel protein families. We evaluated our graph bigram metric against a word-proximity metric for term association on datasets of full-text literature in these three different protein families. Our testing shows that the graph bigram metric achieves a higher F-measure for the GPCRs (0.79 versus 0.76), protein tyrosine kinases (0.72 versus 0.69), and ion channel transporters (0.76 versus 0.74). Importantly, in situations where more than one protein can be assigned to a point mutation and disambiguation is required, the graph bigram metric achieves a precision of 0.84 compared with the word distance metric precision of 0.73. We believe the graph bigram search metric to be a significant improvement over previous search metrics for point mutation extraction and to be applicable to text-mining application requiring the association of words. This chapter was published as "Automatic Extraction of Protein Point Mutations Using a Graph Bigram Association" [Lee et al., 2007].

3.2 Introduction

With the advent of ultra high-throughput screening and high-density array technology, the biological community has come to appreciate the value of unbiased surveys of complex biological systems. Bioinformatics tools have become an integral part of the analysis of these extensive datasets. When complex data is collected centrally, the analysis can be straightforward. When data is collected in a distributed fashion, investigators must agree on a centralized data-deposition strategy or we must develop tools to interrogate the published literature and extract relevant information. Manually curated online databases have developed to meet this need, but they are difficult to maintain and scale. Accordingly, the biological text-mining field has evolved to identify and extract information from the literature for database storage and access. Two types of tasks predominate in biological text mining: the extraction of gene and protein names [Mitsumori et al., 2005, Koike and Takagi, 2004, Tanabe and Wilbur, 2002, Zhou et al., 2004] and the extraction of interactions between proteins [Marcotte et al., 2001, Blaschke et al., 1999, Chang et al., 2004]. The BioCreAtIvE challenge [Hirschman et al., 2005a] was focused on name extraction [Yeh et al., 2005] with the additional task of functional annotation [Hirschman et al., 2005b]. Other text-mining applications focus on hypothesis generation [Srinivasan, 2004], probing protein subcellular localization [Stapley et al., 2002], and pathway discovery [Friedman et al., 2001].

Recent work has also focused on the extraction of protein point mutations from biomedical literature Rebholz-Schuhmann et al. [2004], Horn et al. [2004], Baker and Witte [2004, 2006], Witte and Baker [2005]. Protein point mutations, the substitution of a wild-type amino acid with an alternate one, can be important to our understanding of protein

function, evolutionary relationships, and genetic disorders. From a functional perspective, researchers introduce point mutations into proteins to assay the importance of a particular residue to protein function. Evolution relies upon mutations or polymorphisms in DNA, a mechanism for creating diversity in protein sequences. While the term "mutation" is used to imply deleterious changes, and "polymorphism" means a difference within species, for text-mining purposes we refer to a "point mutation" as a substitution of a different amino acid for the reference amino acid. dbSNP [Sherry et al., 2001] and the Human Gene Mutation Database [Cotton and Horaitis, 2002] are two of many databases that catalog point mutations and their downstream effects. These databases are manually curated, which limits the speed of input into the database and the breadth of information represented, but does aid in the incorporation of complex information that is difficult for text-mining tools to parse.

The task of point mutation extraction can be decomposed into two subtasks. First, it is necessary to identify the protein and mutation terms discussed within an article. After these entities are identified, an association must be made between the point mutation and its correct protein of origin. This problem is trivial when a paper discusses a single protein but increasingly complex when multiple proteins are present. In our evaluation of Mutation Graph Bigram (Mutation GraB), we downloaded 589 full-text PDF articles related to the GPCR, tyrosine kinase, and ion channel protein families from PubMed-provided links. Using our dictionary-based protein term identification method, we counted 350 articles out of the total 589 that contained a point mutation that could have belonged to multiple proteins. A few methods for point mutation extraction have been developed. [Rebholz-Schuhmann

et al., 2004] describe a method called MEMA that scans Medline abstracts for mutations. Baker and Witte [2004, 2006], Witte and Baker [2005] describe a method called Mutation Miner that integrates point mutation extraction into a protein structure visualization application. Our own group has presented MuteXt [Horn et al., 2004], a point mutation extraction method applied to G protein-coupled receptor (GPCR) and nuclear hormone receptor literature. MEMA and MuteXt use a straightforward dictionary search to identify protein/gene names and a word proximity distance measurement to disambiguate between multiple protein terms. Both methods, while providing a simple and successful method for point mutation extraction, were limited in two areas. First, the word distance measurement is not always correct in disambiguating between protein terms. Second, MEMA was evaluated on a set of abstracts, which are intrinsically more limited than the full-text article. In our literature set, the abstracts contained only 15% of the point mutations found in the full text. The point mutations were also validated against OMIM [Hamosh et al., 2002], which only contains disease-related point mutations. MuteXt was trained and evaluated on GPCR and intranuclear hormone receptor literature and contained customizations in the algorithm for dealing with problematic protein naming and amino acid numbering cases.

Mutation Miner approaches the problem differently. This method identifies and relates proteins, organisms, and point mutations using NLP analysis at a sentence level. An entity pair is assigned if both entities match noun phrase patterns. This method would work well if all point mutations were described in conjunction with associated proteins and organisms at the sentence level, which we have observed is not always the case. Mutation Miner also incorporates protein sequence information, but for use in annotating protein 3-

D structures with mutation information instead of point mutation validation. Our method improves on MEMA, MuteXt, and Mutation Miner by using a novel graph bigram metric that incorporates frequency and location of terms to disambiguate between proteins and searches full-text information. Like MuteXt, Mutation GraB utilizes the Swiss-Prot protein database [Boeckmann et al., 2003] for sequence validation, which intrinsically contains more sequence variation than OMIM. We addressed the utility of our application by standardizing the algorithm for all protein families and by evaluating our method on three different protein family literature sets covering 589 articles. More detailed comparisons with MEMA and Mutation Miner are described in the Discussion section.

3.2.1 Protein Term Identification

For our task of associating point mutations to protein terms, it is not sufficient to minimally tag a protein name in the literature; we must also find its correct gene identifier in a corresponding database. The BioCreAtIvE challenge addressed this problem with the 1B subtask of identifying a protein/gene mentioned in the text and annotating it with its correct gene identifier. Solutions for this challenge ranged from rule-based methods [Hamosh et al., 2002] to machine-learning approaches [Crim et al., 2005] to a combination of both. Unfortunately, some of these methods may not be applicable to our point mutation extraction task. The participants in the BioCreAtIve challenge were provided a large set of annotated sentences categorized under three different organisms; human, yeast, and fly. Some solutions for the subtask 1B consisted of learning the training data for each organism, then applying the learned functions to a test set also divided by organism. This approach is suboptimal for our task for two reasons. First, because point mutations are frequently

analyzed at a protein family and superfamily level, methods trained on protein names from organism-specific lexicons would not be well-suited for analysis across many species. Second, our goal is to create a broadly applicable methodology for point mutation extraction that can be utilized on any categorization of proteins (i.e., family, class, fold, etc.). Machine-learning approaches benefit from large detailed annotated training sets. In our experience, the manual labor involved in annotating the amount of text necessary to learn protein family-specific nomenclature on the scale presented by BioCreAtIve is likely to undermine the benefits of automated point mutation extraction.

Methods relying solely on rule-based features for protein-name identification generally perform at a lower precision and recall than methods incorporating machine learning. However, since rule-based methods do not necessarily require annotated training data, they are advantageous when such data is unavailable or difficult to acquire. Our approach to protein term identification is similar to other rule-based approaches Hanisch et al. [2005], Fundel et al. [2005], Koike and Takagi [2004]. We first create a dictionary using the names and synonyms of proteins in a protein family; the protein names are retrieved from their respective Swiss-Prot and Entrez Gene entries. The terms in the dictionary are then searched for in the journal literature. Depending on the character length and composition of these terms, we search by different regular expressions with varying levels of specificity. A further description of this is detailed in the Methods section.

3.2.2 Point Mutation Identification

Point mutations are represented in a variety of ways in the literature, but all consist of three distinct parts: a wild-type amino acid, a sequence position, and a mutant

amino acid. A typical representation of a point mutation is A123T, denoting a change from alanine to threonine at position 123 of a protein using the single letter abbreviation for the amino acids. Variations on this shorthand form include A123 → T, A(123)T, and A-123-T, and the three-letter amino acid abbreviations Ala123Thr, Ala123 → Thr, Ala(123)Thr, and Ala-123-Thr. Aside from those frequent representations, point mutations are also represented grammatically such as "position 123 was mutated from an alanine to a threonine" or "positions 100-110 were mutated to proline." In our literature sets, however, <1% of all true positive point mutations were grammatical, so we chose to focus on the single-letter and three-letter abbreviation variants of point mutations instead.

3.2.3 Point Mutation-Protein Association

The task of associating point mutations to proteins is unique and has no true corollaries from other text-mining applications. Protein-protein interactions are explicit binary relationships that usually occur locally within a sentence or two. A point mutation usually has a one-to-one relationship with a protein; however, this relationship is often implicit over the length of the whole text. For example, a journal article may present a protein term in the abstract and the introduction sections while describing point mutations to that protein in the methods and discussion sections. It is implied that the point mutations discussed in the latter sections refer to the protein term in the former sections. When more than one protein is discussed in the text, a method is required to choose the correct protein or species.

Previous methods have used a simple and effective word distance metric for protein term disambiguation; a point mutation is assigned to its nearest occurring protein term.

Our graph bigram method improves on this approach by accounting for all occurrences of the point mutation and protein terms throughout the length of the text instead of measuring one local relationship. This method uses the t test to measure the significance of bigrams in the text, then employs a graph shortest-distance search to traverse significant bigrams to associate a point mutation with its correct protein term. An example of a graph generated from an ion channel transporter article (PMID 11553787) is shown in Figure 3.1. While this graph is too complex to provide any algorithmic examples, we can see that nodes found closer to the center grouping in the graph are involved in more bigrams than the peripheral nodes. In general, paths that traverse the central grouping of nodes will be shorter and more significant than paths taken around peripheral nodes.

3.2.4 Mutation GraB Approach

Our approach to point mutation extraction consists of the following steps. 1) Target a protein family of interest and retrieve full-text articles discussing point mutations within the protein family. 2) Identify protein and organism terms within the articles using a dictionary generated from protein databases (creating an implicit link between the protein term and database identifier). 3) Identify point mutation terms using a set of regular expressions. 4) For each point mutation, generate a set of possible associated proteins by comparing the wild-type amino acid with that contained in the protein sequence. If this set contains several possible proteins, use the graph bigram method to disambiguate and to find the correct association.

The process flow of Mutation GraB is shown in Figure 3.2. In our execution and evaluation of Mutation GraB, we wanted to focus on three different aspects of point

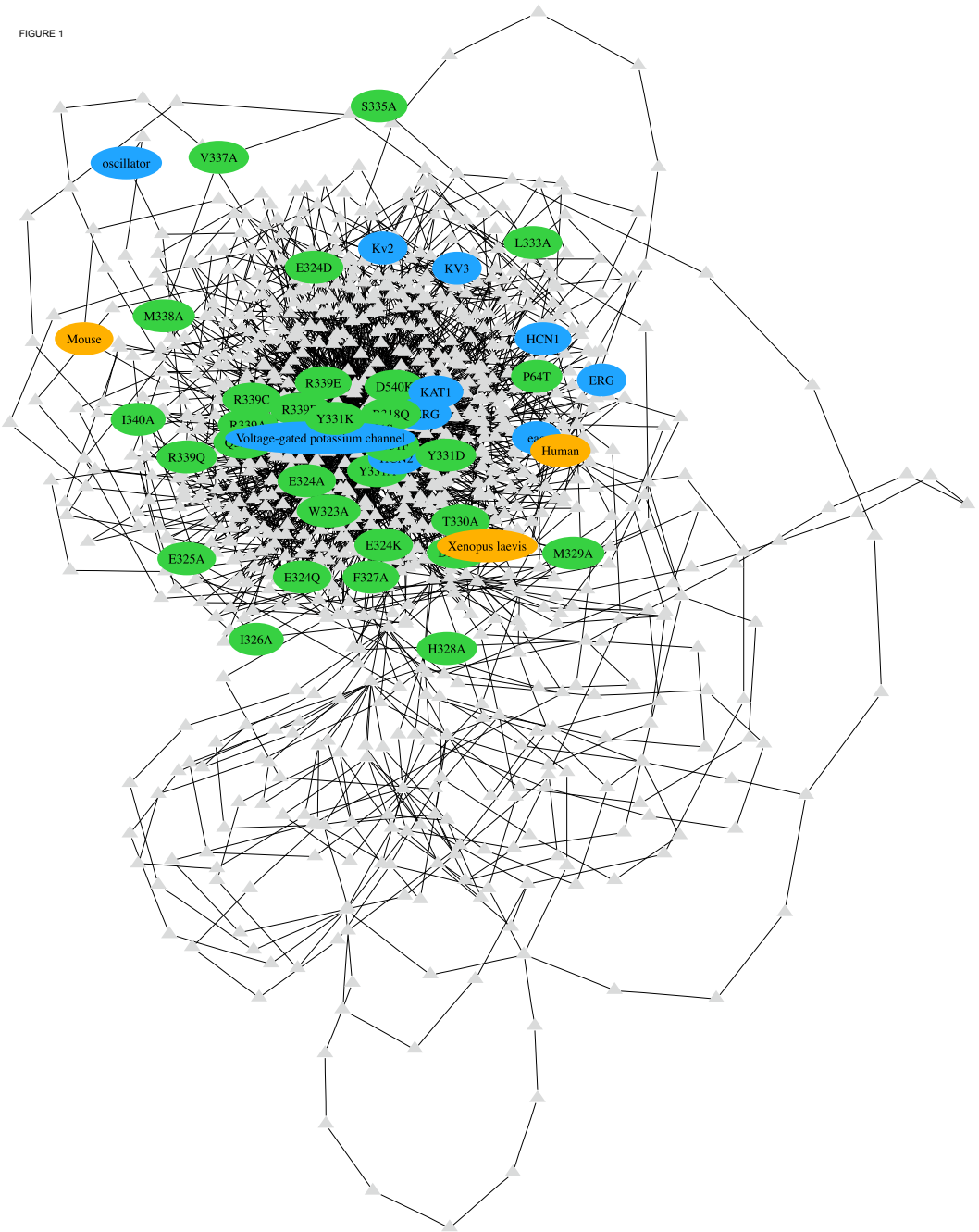


Figure 3.1: The blue ellipses represent protein term nodes, green ellipses represent point mutation nodes, and orange ellipses represent organism nodes. The gray triangles represent regular words. The connecting edges show terms or words represented by the nodes that are present as a bigram in the text. For this article, a total of 1,052 terms are contained in 2,287 bigrams.

mutation extraction. First, we wanted to gauge the feasibility of systematically extracting point mutations from the literature in a fully automated fashion. Our initial testing and previous methods showed that automated point mutation extraction is a wholly viable endeavor, with the main challenges of identifying protein terms and associating them to the proper point mutation. Second, and most important, we wanted to assess different search methodologies available to extract point mutations and to devise a superior search metric for extraction. We hypothesize that a search metric that integrates the relative position of the words and frequency data into its heuristic will outperform a metric that solely relies on positional information. Last, we wanted to create an application that could be used extensively on all protein family literature to create a database of point mutations. Therein, Mutation GraB is a self-contained application where most, if not all, the information used is gathered from database sources and not expert-user opinion.

3.3 Methods

The overview of Mutation GraB is shown in Figure 3.2. The following sections describe how the protein family literature sets were generated, how Swiss-Prot entries were chosen, and how each article was processed to extract the point mutations.

3.3.1 Article Search and Retrieval

Articles were searched for using PubMed queries containing the protein family name, the MeSH term "point mutation", and the "full text" filter. The query string `<protein family> AND point mutation[mh] AND full text[sb]` was used to retrieve

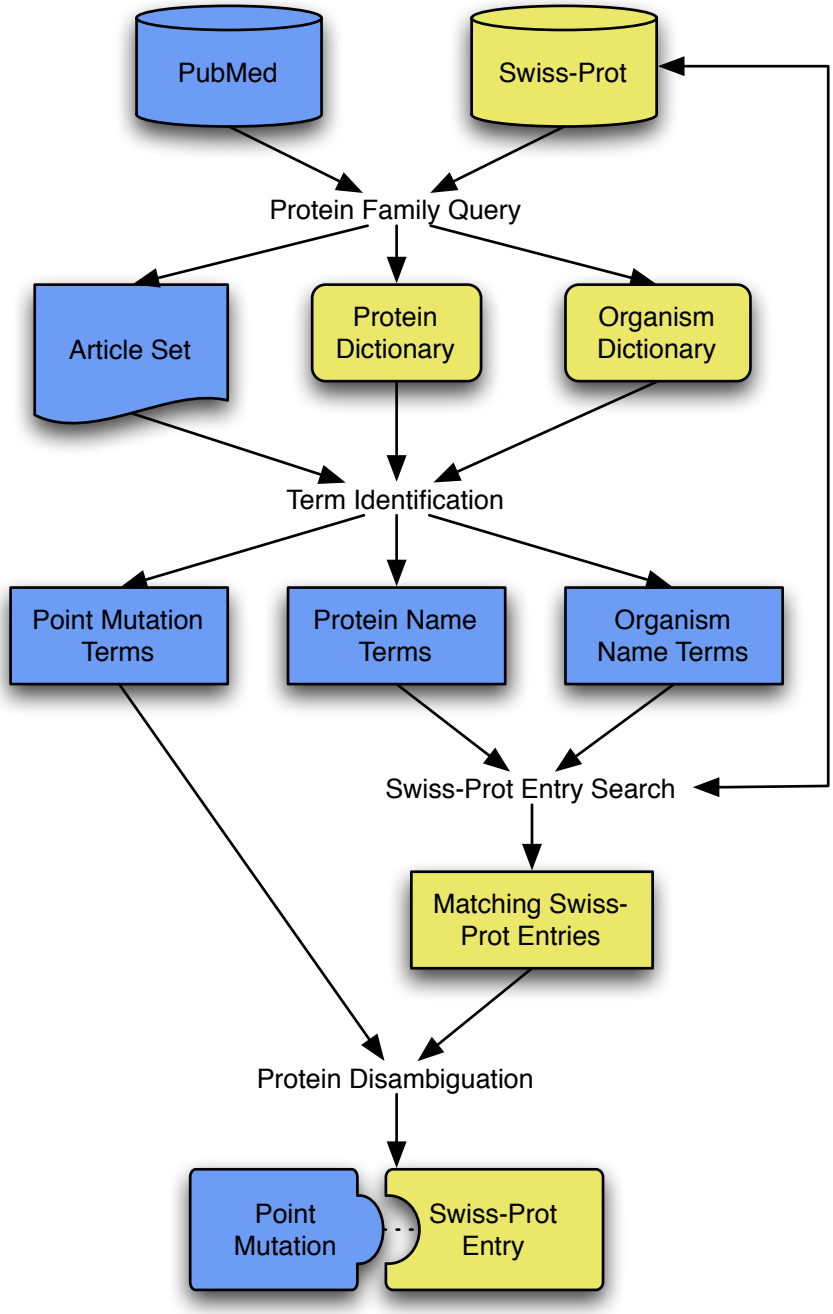


Figure 3.2: A General Overview of the Process Flow of Mutation GraB.

the relevant literature, where `<protein family>` is substituted by either "protein tyrosine kinase", or "ion channel transporter"; the GPCR PMID list was retrieved from the tGRAP database. The resulting lists of PMIDs were individually searched using the Entrez E-Utilities, retrieving the LinkOuts – external HTTP links – for the full-text articles. We followed the LinkOuts and parsed the returned HTML pages for links to PDF files, downloading the PDF file when possible. The downloaded PDF files are converted to Unicode text using the Unix `pdftotext` utility.

3.3.2 Text Preprocessing

After conversion from PDF, the text was preprocessed to create a more cohesive and manageable document. We removed the "References" and "Acknowledgements" sections from the text, as well as sections beginning with the text "this work was supported by", "to whom correspondence", and "the abbreviations used are". A stop list consisting of the words "the", "and", "for", "with", "were", "that", "was", "from", "this", "are", "which", "a", "an", "or", and "of" was used and those words subsequently removed.

3.3.3 Dictionary Creation

Two different dictionaries were created for each protein family to be searched, a protein name dictionary and an organism dictionary. The terms for both dictionaries were extracted from the Swiss-Prot and Entrez Gene databases. First, Swiss-Prot entries for the protein families of interest were chosen based on the contents of the "keyword" Swiss-Prot field. Protein tyrosine kinase entries contained the words "Protein Tyrosine Kinase" in the keyword field, GPCR entries contained "G Protein-Coupled Receptor", and ion channel

entries contained both “Ionic channel” and “Transmembrane”. To generate the protein name dictionary, the “protein name” and “gene name” fields from a protein family Swiss-Prot entry subset were compiled. If a Swiss-Prot entry had a related Entrez Gene database entry, the “gene name”, “official symbol”, “official fullname”, and “aliases” Entrez Gene fields were added to the dictionary if they were not duplicates of the SwissProt names. The organism dictionaries were generated by compiling the “organism” field from the respective Swiss-Prot entries. The word “murine” was added to the organism dictionaries as a synonym for “mouse”.

A second dictionary, which we called a permutation dictionary, was also created from protein full names three words or longer. The entries in this dictionary were permutations of the full names with the permutations also being at least three words long. Since a set of n elements will have $n!$ permutations, and the number of ways of obtaining an ordered k elements from a set of n elements is $n!(n-k)!$, a term that consisted of five words will spawn $5!/(5-3)! = 60$ permutated terms. For example, permutations of the full name “Parathyroid Cell calcium-sensing receptor” include “calcium sensing receptor”, “cell calcium sensing”, and “cell parathyroid sensing calcium receptor”. Fortunately, nonsensical word permutations are unlikely to appear in actual text. This dictionary is searched in the same manner as the protein full name terms in the standard dictionary.

3.3.4 Manual Annotation of Articles

For each article processed by Mutation GraB, we manually read and extracted the point mutations from the text. We counted a point mutation as a TP (true positive) point mutation if the associated protein belonged to the corresponding Swiss-Prot protein

family set and if the wild-type amino acids of the mutation and the protein matched. A point mutation is considered a TN (true negative) if its associated protein is not part of the Swiss-Prot set (i.e., belonging to a different protein family) or if the wild-type amino acids do not match. Point mutations that contain typographical errors are considered TN point mutations as well as are point mutations whose position numbering differs from the provided sequence in its corresponding Swiss-Prot entry. These annotated TP and TN point mutations were used as our "gold standard" sets to evaluate Mutation GraB performance.

3.3.5 Term Identification and Extraction

Regular expressions were constructed to identify point mutation, protein name, and organism name terms. A point mutation description usually consists of a wild-type amino acid name, followed by the amino acid position number, which is followed by the mutant amino acid name. The amino acids can be represented in the single-letter or three-letter format, and the regular expressions allow for some punctuation between the position and the amino acids. Some common examples of point mutation strings are "R123Y", "R(123)Y", "R-123-Y", "Arg123Tyr", "Arg(123)Tyr", and "Arg-123-Tyr". Point mutations with a different formatting or representation in the grammar of the text were ignored.

To search for organism names, we created different levels of case-sensitive and insensitive regular expressions. A tiered rule-based approach based on similar methods [2,23,25], however, was used to identify protein name terms. First, the protein name dictionary was split into two groups, symbols (i.e., EPHB1) and full names (i.e., Ephrin type-B receptor 1). Two types of regular expressions were created. One, a strict regular expression, is case-sensitive and does not allow for variation from the protein symbol. A second reg-

ular expression, which is more relaxed, is case-insensitive and allows for non-alphanumeric characters to be removed or substituted by spaces. For the protein symbols, both strict and relaxed regular expressions were used with the addition of organism modifier prefixes or suffixes. The prefixes "h", "m", and "r" were allowed for human, mouse, and rat modifiers, and the "p" suffix was allowed for *S. cerevisiae*. For protein full names, both types of regular expressions were used without any additional modifications. We searched with the protein symbols first followed by the protein full names, in each instance using the strict regular expression formation followed by the relaxed. We also allowed for Roman numeral replacement. If a protein name has a single digit as the last character, such as "XYN2", we also searched for the term "XYNII".

3.3.6 Point Mutation-Protein Association

After identifying all the point mutation, protein name, and organism name terms present in the text, we looked for Swiss-Prot entries that corresponded to the protein and organism names found. For example, the protein full name "Alpha 1-B Adrenergic Receptor" and the organism name "rat" correspond to the unique Swiss-Prot entry P15823. If an article contains multiple protein and organism names, multiple unique Swiss-Prot entries may be represented. The wild-type amino acid of each point mutation was then compared to the amino acid at the specified position of each Swiss-Prot protein found in the text. We also compared the amino acid sequence of any isoforms of the Swiss-Prot protein as well as removing the signal sequence of the protein if present. If the amino acids from the point mutation and the protein sequence match, that protein was categorized as a possible association for the point mutation. When a single Swiss-Prot protein was possible for a

point mutation, that protein was automatically associated with the point mutation. When multiple proteins are possible for a point mutation, the word distance or graph bigram methods were used to select the best match for association.

Word Distance Metric

Let M be the set of point mutations with multiple possible Swiss-Prot protein associations in an article. For each $m \in \mathbf{M}$, let $\mathbf{P}$ be the set of protein names and $\mathbf{O}$ be the set of organism names represented by the possible proteins for m. Thus, for $m \in \mathbf{M}$, $p \in P|m$, and $o \in \mathbf{O}|m$, we created a list $\mathbf{T} = \{t_1 = \langle m, p_1, o_1 \rangle, \ldots, t_i = \langle m, p_j, o_k \rangle\}$ that represents the possible protein associations (PPAs) for point mutation m. The word distance metric between terms w_i and w_j in the text, $h_{word}(w_i, w_j)$, is the shortest number of words that separate any two instances of w_i and w_j. Using this measurement, we associated point mutation m to the protein whose protein name p and organism name o resulted in the smallest $\delta_{m,word} = h_{word}(m, p) + h_{word}(m, o) + h_{word}(p, o)$. This is essentially the triangulation of distances between the point mutation term, protein name, and organism name, where the smallest sum of distances represents the assumed correct association between point mutation and Swiss-Prot protein.

Graph Bigram Metric

The graph bigram metric works in the same manner as the word distance metric in terms of triangulating the smallest distances between the relevant terms in the text. The difference lies in how the distances are calculated. A graph was constructed by assigning nodes to all of the words and terms in the text. An edge connected two nodes if the

represented words and/or terms were adjacent to each other in the text. The reciprocal t statistic provided a sensible way of measuring how likely it is that any two words will occur next to each other. The t test quantifies the likelihood that the adjacency of two words is significant. The larger the t statistic, the greater the significance of the relationship. We set the value of an edge between two nodes containing words w_i and w_j to be the reciprocal of the t statistic:

$$h_{graph}(w_i, w_j) = \left[\frac{\tilde{x} - \mu}{\sqrt{s^2/N}} \right]^{-1} \tag{3.1}$$

where $\tilde{x}$ = sample mean, s^2 = sample variance, N = sample size, and μ = mean of distribution. When the t statistic is applied to a text mining application, $\tilde{x} = (w_i \wedge w_j)/N$, where $(w_i \wedge w_j)$ equals the number of times w_i is adjacent to w_j, and N equals the number of words in the text. The mean of the distribution, or the null hypothesis, is $\mu = w_i/N \times w_j/N$, where w_i and w_j are the number of occurrences of word i and j, respectively. For large samples, the variance $s^2 \approx \tilde{x}$. Dijkstra's algorithm is used to calculate the shortest path between any two nodes in a graph, utilizing h_{graph} as the edge weight values. Since Dijkstra's algorithm does not work for negative distances, if any t statistic for a bigram is negative, all t statistics for bigrams in that article are normalized by addition of the negative value so that the smallest t statistic = 0. When calculating the δ value for two terms, if the terms are adjacent, we use the reciprocal t statistic value. If the terms are not adjacent to each other in the text, we find the shortest path between the two terms, and δ is equal to the sum of distances h_{graph} between nodes in the graph within the shortest path.

A more detailed example of the differences between the graph bigram and word

metrics is shown in Figure 3.3. Figure 3.3A shows text from a GPCR article (PMID 10889210) that was used to create the graph shown in Figure 3.3B. The text is a paragraph from a figure label from the article. Figure 3.3C shows the distances generated by the two search metrics between some selected words from the text; below the diagonal the numbers are generated by the word distance metric, and above the diagonal by the graph bigram metric. This example is not meant to show an instance of mutation extraction, but is only meant to highlight characteristics of text that are interpreted differently by each metric. Since most full-text articles have point mutations, protein names, and organism names scattered about the entirety of the text, an example detailing a point mutation extraction would be too complicated to illustrate in a figure. The path in Figure 3.3B highlighted in red shows a bigram traversal between the word "fig" and the word "bars". In the text in Figure 3.3A, we can see that the words are each found only once in the text and they are on opposite ends of the paragraph. Using the word distance metric, h_{word}(fig, bars) = 157, which is one of the largest distances measured for that text. The graph bigram metric measures h_{graph}(fig, bars) = 5.26, which, when compared with the other values for h_{graph}, is not the largest measured. This is because the bigram path traverses the word "receptor", which is found as a bigram with both "basal" and "bars". This example shows how two words can be far apart in word distance but still be measured more significantly using the graph bigram metric.

Conversely, we can examine the path in Figure 3.3B highlighted in blue. The words "alteration" and "scatchard", when measured by the word distance metric, yield h_{word}(alteration, scatchard) = 41, meaning there are only 40 words that separate the two.

Table 3.1: Protein Family Literature Sets.

Protein Family	**Total Articles**		**True Positive Mutations**	
	Development	**Validation**	**Development**	**Validation**
GPCR	95	99	962	902
Protein tyrosine kinase	100	98	334	153
Ion channel transporter	99	98	446	514

This measure is fairly significant when compared with the other h_{word} measurements. However, when using the graph bigram metric, we see that h_{graph}(alteration, scatchard) = 9.80, a larger and far less significant relationship when compared with other h_{graph} measurements. The path generated in the graph for these words is far longer than for "fig" and "bars", and, accordingly, the graph bigram distance is larger. This highlights a situation where two words close in word distance have a less significant graph bigram measurement.

The histograms in Figures 3.4 and 3.5 show the distribution of distance values for all pairs of words in Figure 3.3. The distribution of word distances exponentially decreases from 1, while the distribution of graph bigram distances follows a bell-shape. The blue bar in each figure represents the distance of the {alteration, scatchard} pair, while the red bar represents the distance of the {fig, bars} pair. It can be seen that while the {alteration, scatchard} pair has a much higher word distance than the {fig, bars} pair, the reverse is true for the graph bigram distance.

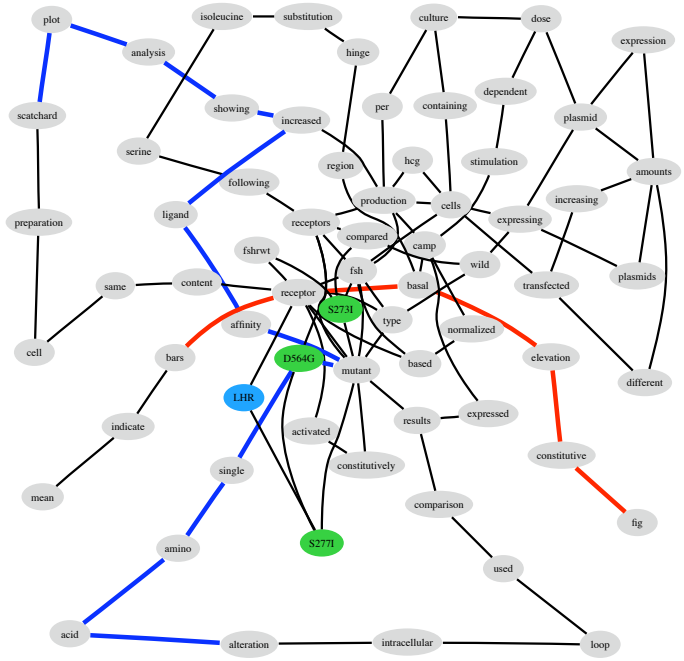
3.4 Results

We evaluated the effectiveness of Mutation GraB by using it to extract point mutations from literature describing three different protein families: tyrosine protein kinases,

A

FIG. 2. Constitutive elevation of basal cAMP production and increased ligand affinity of mutant LH and FSH receptors following serine to isoleucine substitution in the hinge region. A, basal cAMP production in cells transfected with increasing amounts of plasmids expressing wild-type LH receptor (LHR WT) or the S277I mutant. The constitutively activated LH receptor mutant (D564G) with a single amino acid alteration in the intracellular loop 3 is used for comparison. Results are expressed as cAMP production per culture containing cells transfected with different amounts of the expression plasmid. B, dose-dependent stimulation of cAMP production by hCG in cells expressing wild-type or mutant (S277I or D564G) LH receptors. Production of cAMP is normalized based on LH receptor content of the same cell preparation. C, Scatchard plot analysis showing increased ligand affinity of mutant LH receptors as compared with the wild-type receptor. D, basal cAMP production in cells transfected with increasing amounts of the plasmid expressing wild-type FSH receptor (FSHRWT) or the S273I mutant. Results are expressed as cAMP production per culture. E, dose-dependent stimulation of cAMP production by FSH in cells expressing wild-type (WT) or mutant (S273I) FSH receptors. Production of cAMP is normalized based on FSH receptor content of the same cell preparation. F, Scatchard plot analysis showing increased ligand affinity for the mutant FSH receptor S273I as compared with the wild-type FSH receptor. Bars indicate mean S.D.

B



C

	D564G	acid	affinity	alteration	amino	analysis	bars	basal	constitutive	elevation	fig	increased	ligand	plot	receptor	scratchard	showing	single
D564G		3.03	1.72	4.03	2.02	4.33	3.43	2.82	4.84	3.83	5.85	2.90	2.31	5.05	2.38	5.77	3.62	1.01
acid	3		4.75	1.01	1.01	7.36	6.46	5.84	7.87	6.86	8.87	5.92	5.33	8.08	5.40	8.79	6.64	2.01
affinity	17	32		5.75	3.74	2.61	3.07	3.19	5.22	4.21	6.22	1.18	0.59	3.33	2.01	4.05	1.90	4.18
alteration	4	1	33		2.01	8.37	7.46	6.85	8.87	7.87	9.88	6.93	6.34	9.08	6.41	9.80	7.65	2.73
amino	2	1	31	2		6.35	5.45	4.83	6.86	5.85	7.87	4.92	4.33	7.07	4.40	7.78	5.64	1.01
analysis	13	44	4	43	45		5.68	3.61	5.63	4.63	6.64	1.44	2.03	0.72	4.63	1.43	0.72	5.35
bars	86	117	10	116	118	14		2.23	4.26	3.25	5.26	4.24	3.66	5.38	1.05	4.66	4.97	4.44
basal	19	22	5	23	21	11	62		2.03	1.02	3.03	2.17	2.76	4.32	1.18	4.79	2.89	3.83
constitutive	36	39	7	40	38	83	156	2		1.01	1.01	4.20	4.78	6.35	3.20	6.81	4.92	5.85
elevation	35	38	6	39	37	82	155	1	1		2.01	3.19	3.78	5.34	2.20	5.81	3.91	4.85
fig	37	40	8	41	39	84	157	3	1	2		5.20	5.79	7.35	4.21	7.82	5.92	6.86
increased	15	34	2	35	33	2	12	3	5	4	6		0.59	2.15	3.19	2.87	0.72	3.91
ligand	16	33	1	34	32	3	11	4	6	5	7	1		2.74	2.60	3.46	1.31	3.32
plot	12	43	5	42	44	1	15	12	82	81	83	3	4		4.33	0.72	1.43	6.06
receptor	2	5	3	6	4	7	1	1	28	27	29	5	4	6		3.61	3.91	3.39
scratchard	11	42	6	41	43	2	16	13	81	80	82	4	5	1	5		2.15	6.78
showing	14	45	3	44	46	1	13	10	84	83	85	1	2	2	6	3		4.63
single	1	2	30	3	1	46	119	20	37	36	38	32	31	45	3	44	47	

Figure 3.3: (A)Text is taken from a figure label from the article PMID 10889210 (B)Graph generated by bigram traveral using the graph bigram method. The point mutation terms are in green, protein terms in blue, and regular words in gray. (C)Table shows the measurements between some selected words in the text using both the word distance and graph bigram metrics. The word-distance measurements are below the diagonal, and the graph bigram measurements are above the diagonal. Two different word pairs are examined, {fig, bars} and {alteration, scatchard}. The {fig, bars} words are shown in red in (A), the path is colored in red in (B), and the metric measurements are highlighted in red in (C). The {alteration, scatchard} items are highlighted in blue, correspondingly.

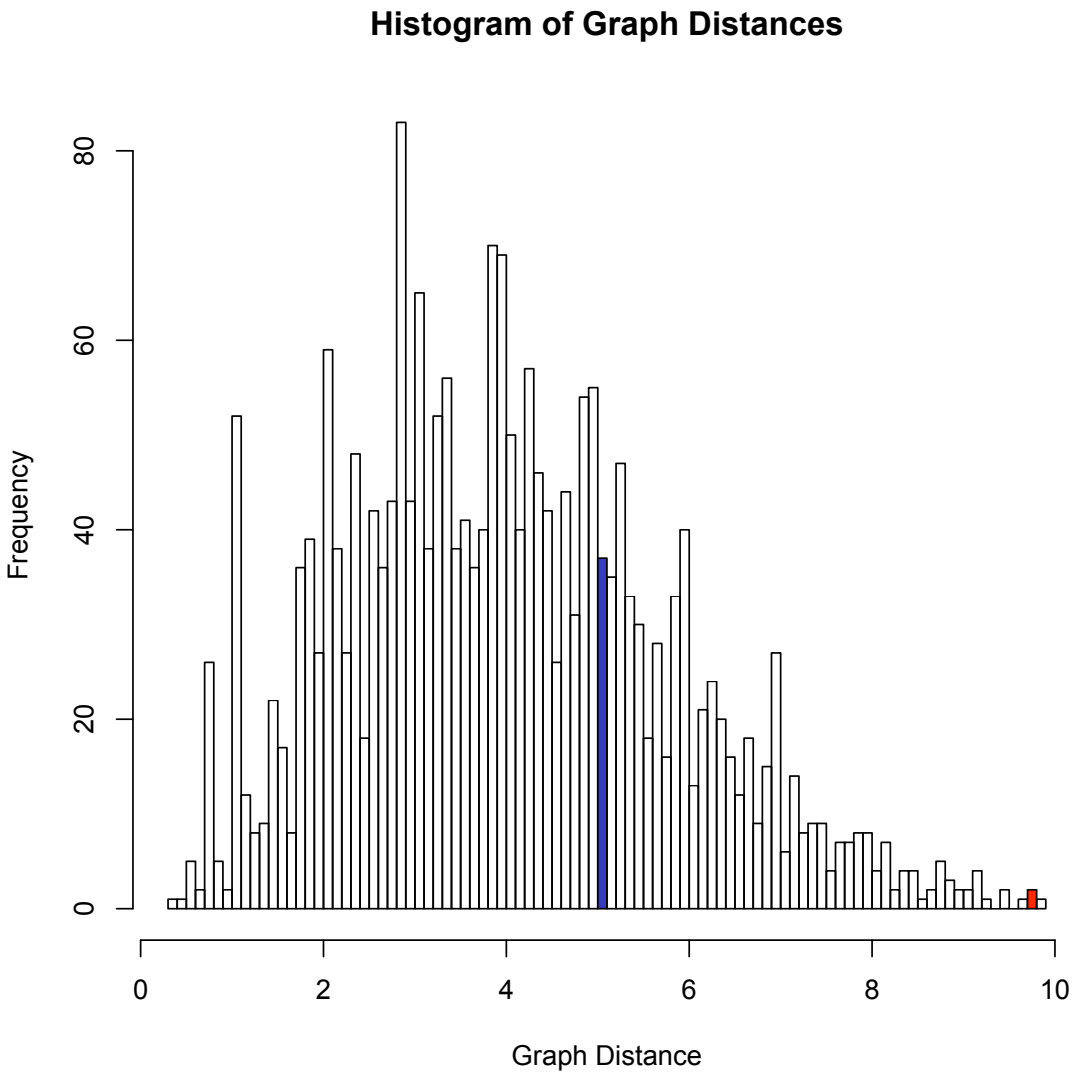


Figure 3.4: This shows a histogram of graph bigram distance values for all word pairs shown in Figure 3.3. The blue bar shows the location of the distance between {alteration, scatchard} and the red bar the loction of {fig, bars}.

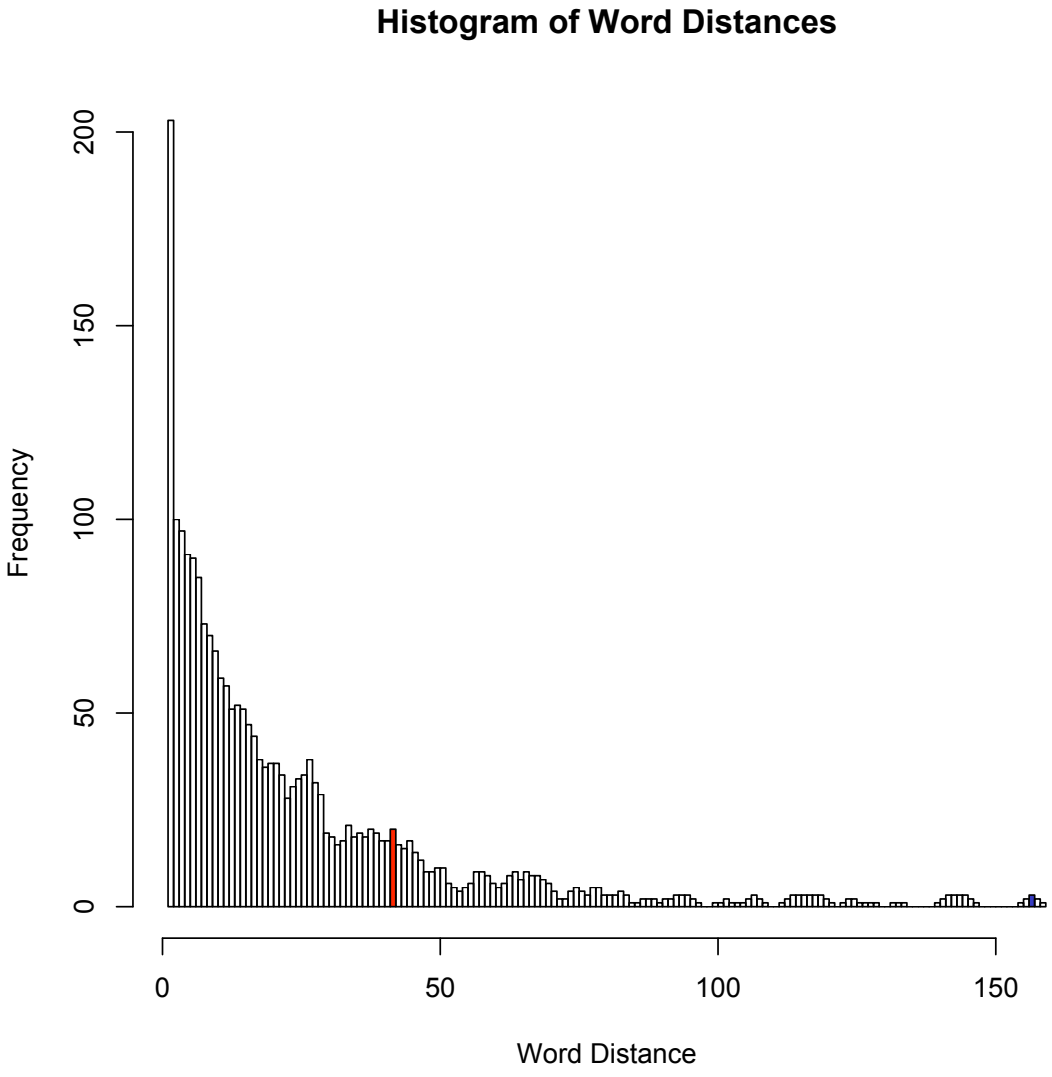


Figure 3.5: This shows a histogram of word distance values for all word pairs shown in Figure 3.3. The blue bar shows the location of the distance between {alteration, scatchard} and the red bar the loction of {fig, bars}.

GPCRs, and transmembrane ion channels. Since most studies of protein structure and function focus at the protein family level, and different protein families have specific and differentiating nomenclatures, this approach is representative of real-life usage and tests the flexibility of Mutation GraB for distinct protein families. Each of the protein family literature sets were split into two groups, a "development" set and a "validation" set of articles. We selected the articles for each set randomly from the all the articles downloaded for each protein family. The development set was used to optimize Mutation GraB performance, while the validation set was used to confirm the performance on the development set. The number of articles in each protein family literature set and the number of true positive mutations manually identified is listed in Table 3.1, and the Swiss-Prot entry, protein name dictionary, and organism dictionary sizes are listed in Table 3.2. Throughout our efforts, these datasets were kept entirely distinct so that the validation set represents a true measure of the generality of the algorithm optimized on the development set. Within each protein family literature set, we ran Mutation GraB twice, once using the graph bigram association metric and the second time using the word distance metric. We also tested the use of the permutatin dictionary in the training set. If the addition of the permutation dictionary did not increase performance, it was not used for the test set. Performance for each search metric can be compared within each protein family literature set.

3.4.1 Evaluation Methods

Mutation GraB was scored against manually annotated "gold standard" sets for each protein family. Point mutations that Mutation GraB and manual curation assigned to the same protein are considered true positive (TP) classifications. Point mutations that

Table 3.2: Protein Family and Dictionary Information.

Protein Family	**Swiss-Prot Entries**	**Protein Name Dictionary Terms**	**Organism Name Dictionary Terms**	**Permutation Dictionary Size**
GPCR	2,249	4,910	560	25,329
Protein tyrosine kinase	430	1,577	108	N/A
Ion channel transporter	1,095	108	143	N/A

Mutation GraB assigned to a protein but were manually classified discordantly are counted as false positive (FP) mutations. In addition, point mutations that were manually classified as TP, but assigned to the wrong protein by Mutation GraB are also ruled as FP mutations. Point mutations that Mutation GraB missed but manual curation assigned are labeled false negative (FN) mutations.

We chose to evaluate Mutation GraB in two different ways. First, we compared the traditional text-mining measurements of precision, recall, and balanced F-measure between the graph bigram and word distance metrics within the development, validation, and complete protein literature sets. Precision is calculated as $P = TP/(TP+FP)$, recall is calculated as $R = TP/(TP+FN)$, and the balanced F-measure is computed as $2PR/(P+R)$. Second, and more significantly, we examined the precision of each search metric on point mutations versus the number of PPA (possible protein associations) for each point mutation. Since the main purpose of the search metric is to disambiguate multiple proteins for each point mutation, the more robust metric will have a higher precision at higher PPA.

Table 3.3: Mutation GraB Performance on the GPCR Literature Sets.

Evaluation Metric	**Development Set**		**Validation Set**		**All Articles**	
	Word	**Graph**	**Word**	**Graph**	**Word**	**Graph**
TP Mutations	656	636	652	684	1,217	1,320
Precision	0.77	0.86	0.82	0.86	0.80	0.86
Recall	0.65	0.68	0.80	0.80	0.72	0.74
F-measure	0.70	0.76	0.81	0.83	0.76	0.79

3.4.2 G Protein-Coupled Receptors

For GPCRs, we took advantage of the manually curated tGRAP database [26] to identify journal literature that describes GPCR point mutations. The tGRAP database subset contains a total of 5,451 point mutations, 1,495 GPCRs, and 914 article citations. We retrieved 386 of these citations as PDF documents and annotated 95 articles as a development set and 100 articles as the validation set. The Swiss-Prot database [Boeckmann et al., 2003] contained 2,249 entries that correspond to GPCR proteins. From these Swiss-Prot entries and their existing corresponding Entrez Gene [Maglott et al., 2005] entries, a standard protein name dictionary of 4,910 terms, an organism dictionary of 560 terms, and a protein name permutation dictionary of 25,329 terms were generated. A permutation dictionary is generated by taking protein name terms of three words or greater and changing the order of the words. We observed that the use of both standard and permutation dictionaries was helpful in identifying a greater number of protein names than the use of the standard dictionary alone. We describe the generation of the permutation dictionary in detail within the Methods section.

The performance of the word distance and graph bigram metrics for the development and validation sets are shown in Table 3.3. In the development set (Dataset S1), the

graph bigram metric achieved an F-measure of 0.76, while the word distance metric achieved an F-measure of 0.70. Examining the point mutation counts between the two metrics, we saw that the graph bigram metric was able to identify 636 true positive mutations versus 565 for the word distance metric. In the validation set (Dataset S2), the graph bigram metric and word distance metric achieved F-measures of 0.83 and 0.81 with true positive mutation counts of 684 and 652, respectively. Combining the two sets, the graph bigram metric outperformed the word distance metric with an F-measure of 0.79 to 0.76.

Figures 3.6- 3.8 graphs the precision of both search metrics measured at different PPA levels for all three protein family literature sets. The precision is measured for the development and validation sets together. We cannot calculate the recall for this analysis because false negative point mutations belong to a protein not represented in the possible associations. The yellow bars represent the number of point mutations counted at each level of PPA. For the GPCR literature set shown in Figure 3.6, there was a large spread of PPA for point mutations. While 651 point mutations only had one PPA, 844 point mutations had multiple possibilities, and the average number of associations per point mutation was 2.19. Figure 3.6 shows that the graph bigram metric achieved a higher precision at all levels of PPA greater than one for the GPCR literature sets.

3.4.3 Protein Tyrosine Kinases

We were able to retrieve 554 PDF articles from the PubMed protein tyrosine kinase query "tyrosine kinase[mh] AND point mutation[mh] AND full text[sb]". We annotated 99 of these articles for use as our development set and 98 articles as our validation set. Searching the Swiss-Prot database for protein tyrosine kinases yielded 430 different entries.

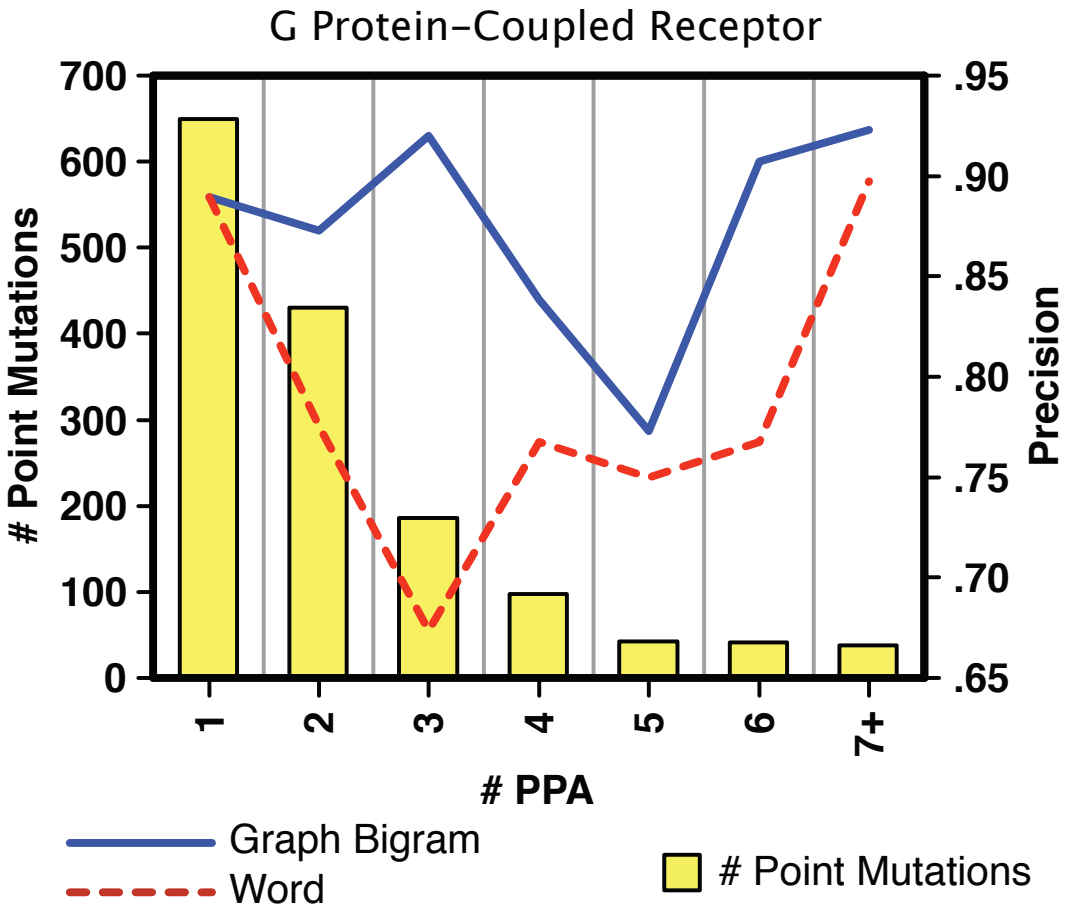


Figure 3.6: **GPCR PPA Results.** This data is for the cumulative development and validation sets combined. The yellow bars show the number of point mutations counted at each PPA. The solid blue line represents the precision measured for these point mutations using the graph bigram metric, and the dotted red line is measured using the word distance metric.

Table 3.4: Mutation GraB Performance on the Protein Tyrosine Kinase Literature Sets.

Evaluation Metric	**Development Set**		**Validation Set**		**All Articles**	
	Word	**Graph**	**Word**	**Graph**	**Word**	**Graph**
TP Mutations	254	279	130	133	384	412
Precision	0.58	0.64	0.54	0.55	0.57	0.61
Recall	0.87	0.88	0.88	0.88	0.87	0.88
F-measure	0.70	0.74	0.67	0.68	0.69	0.72

The protein tyrosine kinase standard dictionary contained 1,577 terms, and the organism dictionary contained 108 terms. Our initial tests indicated that the use of a permutation dictionary would hurt performance for this protein family as many protein names have several elements in common with other family members.

Table 3.4 summarizes our results for the protein tyrosine kinase literature sets. The articles as a whole contained fewer TP and more TN point mutations than the GPCR literature sets, affecting performance by decreasing precision for both graph bigram and word distance metrics. Performance on the development set for the graph bigram and word distance metric were closer together, with F-measures of 0.74 and 0.70, respectively. The validation set yielded even closer results with F-measures of 0.68 for the graph bigram metric and 0.67 for the word distance metric. This small difference is largely due to the few number of true positive mutations in the validation set, with the graph bigram metric identifying 133 to the 130 identified by the word distance metric. Overall, the graph bigram metric outperformed the word distance metric (F-measure 0.72 versus 0.69).

The protein tyrosine kinase analysis in Figure 3.7 shows a smaller distribution of PPA than the GPCR literature set with 266 single PPA, 266 multiple PPA, and a 1.85 PPA average. However, as with the GPCR literature set, the graph bigram metric achieved a

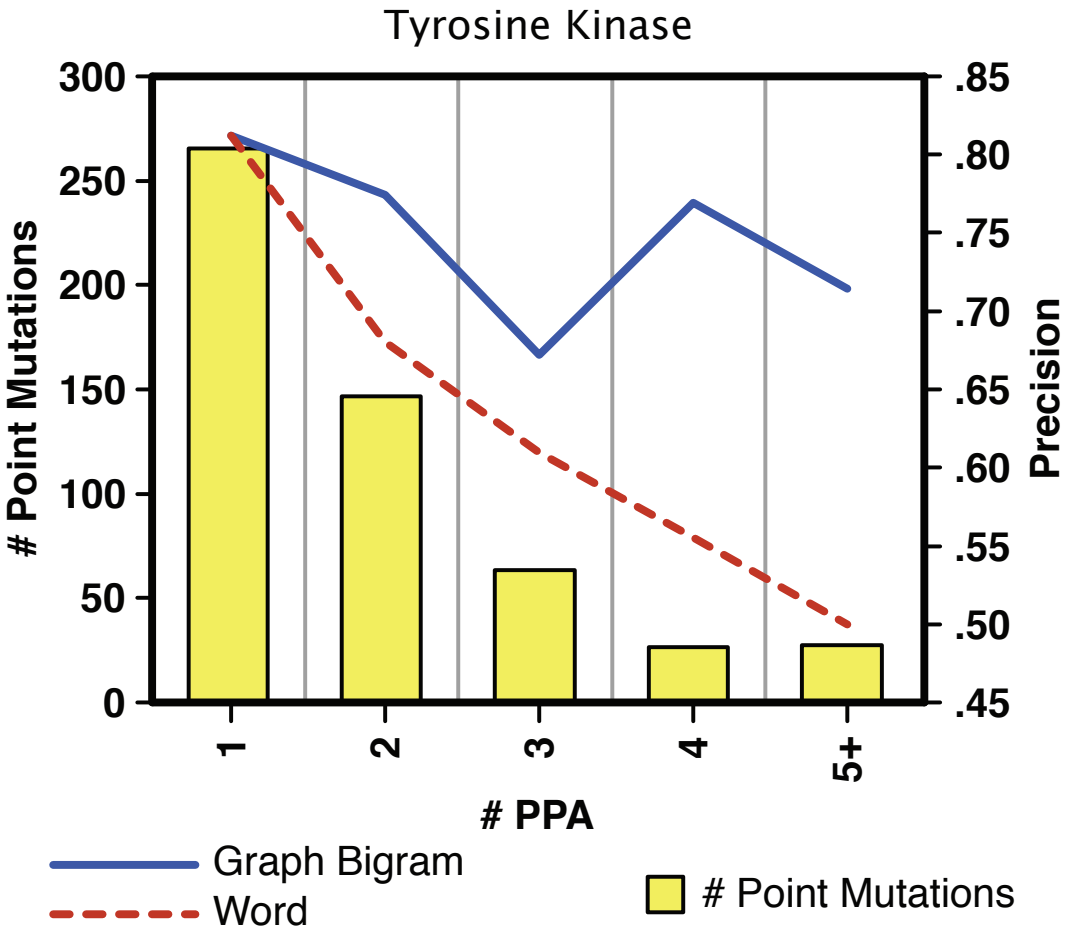


Figure 3.7: **Protein Tyrosine Kinase PPA Results.** This data is for the cumulative development and validation sets combined. The yellow bars show the number of point mutations counted at each PPA. The solid blue line represents the precision measured for these point mutations using the graph bigram metric, and the dotted red line is measured using the word distance metric.

higher precision at all PPA greater than one. The greatest difference in precision was for point mutations with four or more PPA where the graph bigram metric had more than a 0.21 increase than the word distance metric.

3.4.4 Ion Channel Transporters

The ion channel articles were identified with the PubMed query "ion channel[mh] AND point mutation[mh] AND full text[sb]", and 311 PDF articles were downloaded and converted to text. We used 100 of these articles as the development set and 98 articles as the validation set. A total of 1,095 ion channel proteins were identified using the Swiss-Prot "Ion Channel" and "Transporter" keyword identifiers, and 3,089 protein names were extracted from the associated Swiss-Prot and Entrez Gene entries. As with the protein tyrosine kinase literature set, the use of the permutation dictionary did not identify a greater number of protein names, so only the standard protein name dictionary was used. The organism dictionary contained 143 organism names.

Table 3.5 shows the results of the graph bigram and word distance metrics on the development and validation sets. Consistent with the GPCR and tyrosine kinase literature sets, the graph bigram metric had a greater performance gain in the development set (F-measure of 0.70 to 0.68) than in the validation set (F-measure of 0.80 versus 0.79). The graph bigram metric outperformed the word distance metric for both datasets, and overall the graph bigram metric achieved a higher F-measure of 0.76 to 0.74, extracting 624 TP mutations to 604 TP mutations for the word distance metric. The ion channel literature sets yielded the smallest performance difference between the two different search metrics.

Figure 3.8 shows that the ion channel literature set has the fewest PPA per point

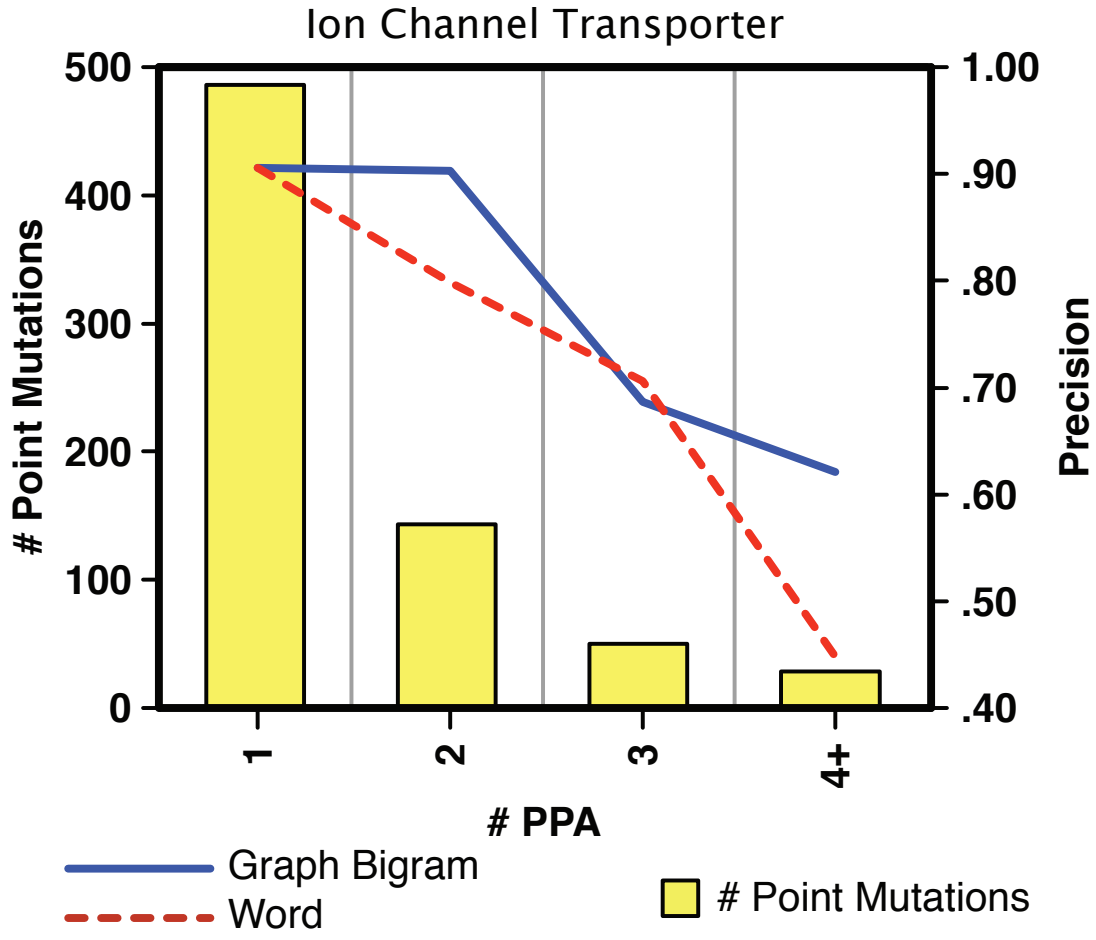


Figure 3.8: **Ion Channel Transporter PPA Results.** This data is for the cumulative development and validation sets combined. The yellow bars show the number of point mutations counted at each PPA. The solid blue line represents the precision measured for these point mutations using the graph bigram metric, and the dotted red line is measured using the word distance metric.

Table 3.5: Mutation GraB Performance on the Ion Channel Transporter Literature Sets.

Evaluation Metric	**Development Set**		**Validation Set**		**All Articles**	
	Word	**Graph**	**Word**	**Graph**	**Word**	**Graph**
TP Mutations	239	253	360	365	596	616
Precision	0.75	0.80	0.81	0.82	0.78	0.81
Recall	0.62	0.63	0.75	0.75	0.69	0.70
F-measure	0.68	0.70	0.78	0.79	0.73	0.75

mutation out of the three protein families. We counted 487 point mutations with only one PPA, while only 224 point mutations had multiple PPA. The average PPA per point mutation is also the smallest at 1.53. The precision difference measured across different PPA levels is less pronounced in the ion channel transporter literature set, with the word distance metric outperforming the graph bigram metric with a precision of 0.71 to 0.69 on point mutations with three PPA. At two PPA and four or more PPA, the graph bigram metric still achieves a higher precision than the word distance metric.

3.5 Discussion

We have introduced Mutation GraB, an application for identifying and extracting point mutations from biomedical literature. Our goal with Mutation GraB was to create a general-purpose application that could have consistent performance without relying on protein family customization. Across these representative protein families, customization was not required to achieve consistently accurate performance in both development and validation sets. This suggests that Mutation GraB should be useful for identifying point mutations in most if not all protein families. Mutation GraB performance is contingent on a number of factors, including the identification of protein names, organism names, and point mutation terms in the text, and the disambiguation of multiple proteins when present. We chose a rule-based approach for protein name identification that has been shown to be successful in other tests, and the search for organism names is accomplished with straightforward pattern matching. Our main performance goal, however, was to devise a disambiguation metric that outperforms current methods at choosing the correct protein

Table 3.6: Mutation GraB versus MEMA Performance.

Mutation	**MEMA**				**Mutation GraB**			
Extraction	**Recall**		**Precision**		**Recall**		**Precision**	
Types	**%**	**Total**	**%**	**Total**	**%**	**Total**	**%**	**Total**
Cited Mutation	74.7	204/273	98.6	204/207	77.3	130/168	97.7	130/133
Contained mutation-gene pairs	35.2	57/162	93.4	57/61	69.3	52/75	85.2	52/61

from a selection of several possible choices.

3.5.1 Comparison with MEMA

MEMA and Mutation GraB were created and tested in a different fashion that makes a direct comparison troublesome. MEMA was tested on 16,728 abstracts across many protein families with the precision and recall estimated from a random set of 100, while we have chosen to use full-text articles from selected families and provide the precision and recall for all 589 articles. The point mutations extracted by MEMA were associated with proteins contained in HUGO and validated with mutations in OMIM, while Mutation GraB utilizes the Swiss-Prot and Entrez Gene databases for protein identification and sequence validation. MEMA also extracted DNA mutations from their set of abstracts, and while the current version Mutation GraB can identify both DNA and protein point mutations, we only validate protein point mutations. Additionally, MEMA identifies a wider set of mutation types, including some that are described grammatically. Table 3.6 shows the performance of Mutation GraB against that of MEMA on the set of 100 abstracts with these caveats.

The row "Cited mutation" refers to the identification of the point mutation terms

in the text, while the row "Contained mutation-gene pairs" refers to the identification and association of the mutation to its protein of origin. The different counts are because of the absence of DNA mutations in the Mutation GraB analysis. Also, our manual analysis of the abstracts found a few mentions of mutated amino acid positions without specifying a mutant amino acid. These mentions were not included in our counts. The precision and recall for identifying "cited mutation" are essentially the same for both MEMA and Mutation GraB. Considering that Mutation GraB does not identify any grammar mutations while MEMA does, this is somewhat surprising. In comparing the identification of "contained mutation-gene pairs," however, Mutation GraB achieves a much higher recall (77.3% versus 35.2%) but a lower precision (85.2% versus 93.4%) than published results for MEMA. As Mutation GraB validates the mutation-protein pairs by comparing with Swiss-Prot sequences, these associations are more significant and may contribute to a lower number of total mutation-gene pairs found in the text. Mutation GraB's disambiguation metric and sequence validation steps help decrease the number of incorrect associations, thereby increasing the recall significantly.

In addition to the increase in recall, we believe Mutation GraB to be an improvement over MEMA for other reasons. One, while abstracts are more readily available than full-text articles, full-text articles are far more informative with regard to point mutations. MEMA extracted 24,351 point mutations mentions from 16,728 abstracts for an average of 1.45 point mutations per abstract. The 589 articles that Mutation GraB was evaluated against contained 3,216 unique point mutations, resulting in an average of 5.45 point mutations per article. Because Mutation GraB only counts unique point mutations per article,

the total number of point mutation terms identified is actually significantly higher. We found that a full-text article contains approximately seven times more point mutation mentions than the abstract alone. With this in mind, our processing of 589 full-text articles would be equivalent to a larger quantity of abstracts.

Second, validation against OMIM will only compare point mutations that are disease-related, while sequence validation against Swiss-Prot compares all point mutations that differ from the wild-type amino acid. Since Swiss-Prot is updated more frequently and contains more genes/proteins than OMIM, Mutation GraB can validate a greater number of point mutations than MEMA. Finally, out of the 100 abstracts MEMA analyzed, only 35 contained multiple gene/protein mentions. From our set of full-text articles, 81 contained point mutations belonging to multiple proteins, and we counted multiple gene/protein mentions in 562 out of 589 articles (95%). As a result, the protein disambiguation capability of Mutation GraB was tested more rigorously than MEMA.

3.5.2 Comparison with Mutation Miner

Mutation Miner differs in many respects to Mutation GraB. Foremost, their use of NLP on a sentence level differs from the statistical approach of Mutation GraB. When searching for a protein of origin, Mutation Miner uses the protein and organism terms to query Entrez Gene for a unique protein entry to retrieve sequence information. They show this solution to be suboptimal, because multiple proteins may be retrieved from the query and sometimes the target protein is not the first protein returned. Also, Mutation Miner uses the protein sequence information not to validate extracted point mutations, but instead to produce multiple sequence alignments of targeted proteins to provide mutation

annotations to 3-D structure displays.

The authors of Mutation Miner tested their methods on 19 abstract and full-text articles on the xylanase protein family. We sought to run Mutation GraB on the same 19 full-text articles to generate a performance comparison, but ran into some obstacles. Instead of measuring the precision and recall for the correct association of a protein-organism pair with each point mutation, Baker and Witte [2006] compute precision and recall for the identification of protein-organism pairs and the identification of point mutations separately. Since Mutation GraB identifies protein-organism pairs based on sequence validation against the point mutation, we cannot produce a comparable evaluation. Additionally, we were only able to retrieve 16 full-text articles from the list. One PDF was copy-protected, while two others did not have the full text available from PubMed. In those instances, the abstract was used. Also, the authors of Mutation Miner have counted a total of 54 point mutations in their 19 articles, while we have manually identified 111 point mutations. This discrepancy may affect the precision and recall of Mutation GraB since we are extracting twice as many point mutations.

Table 3.7 shows the PMID of the articles tested, format of the text, proteins described, point mutations identified, and numbers of point mutations counted by us (Number PM) and Baker and Witte [2006] (MM Number PM). For a majority of the articles, we manually identified more point mutations in the text. Table 3.8 shows the precison, recall, and F-measure achieved by Mutation Miner in extracting protein-organism pairs and identifying point mutations for these articles. It also shows performance for Mutation GraB on the same article set, save for the three abstracts used. We can see that Mutation GraB

is better at identifying point mutations than Mutation Miner, with an F-measure of 0.94 versus 0.90, even though we tested on a larger set of point mutations. Mutation GraB also extracted point mutation-protein-organism triplets at a higher accuracy than Mutation Miner extracted protein-organism pairs alone, with an F-measure of 0.87 versus 0.61. Judging by the low recall of Mutation Miner in extracting protein-organism pairs, analysis at the sentence level misses a majority of the protein-organism associations.

3.5.3 Protein Name Identification

A critical component of point mutation extraction is identifying the protein names for association with the point mutation terms. Since we do not have the luxury of large annotated training sets for our protein families, which are commonly used in more sophisticated methods for protein name recognition and normalization, we relied on a rule-based method. Our rule-based method was patterned after other quantified methods [Hanisch et al., 2005, Fundel et al., 2005] and should provide similar performance characteristics. In our set of 589 journal articles, true positive point mutations were represented by 519 proteins and we were able to identify 446 of these proteins for a recall of 0.86. Reasons for missing some of the protein names can be broadly grouped into two categories: (1) difference in name representation from Swiss-Prot or Entrez Gene and (2) formatting changes as a result of PDF-to-text conversion.

An example of the first category is the identification of the Scn4a protein (Swiss-Prot AC: P15390), whose synonyms are “Mu-1”, “microI”, “Voltage-gated sodium channel alpha subunit Nav1.4”, “Nav1.4”, “Sodium channel protein type IV alpha subunit”, “NCHVS”, and “Sodium channel protein, skeletal muscle alpha-subunit”, as given by Swiss-

Table 3.7: Xylanase Literature Set and Proteins and Point Mutations within.

PMID	Format	Proteins Described	Point Mutations	MG PM	MM PM
885954	PDF Full Text	P18429	E106D	5	3
		P07986	D164A, E168A, E274D, E274A		
1359880	PDF Full Text	P00694	D48E, D48S, E120S, E120D, E209D, E209S, E209C	7	3
8019418	PDF Full Text	P09850	D39N, Y97F, E106Q, E106D, Y108F, R140K, R140N, Y194F, E200D, E200Q, E200C	11	2
10220321	PDF Full Text	P09850	Y97F, R140K, R140N	3	1
10860737	PDF Full Text	P09850	N63D, E106Q, E200Q	3	1
11601976	PDF Full Text	P10478	S172A	1	1
10752608	PDF Full Text	P09850	N176C, S128C	2	5
9930661	Abstract	P33557	D64N	1	1
8376336	PDF Full Text	P36917	D537N, D541Q, H572N, E600Q, D602N, D645N	6	3
11377763	PDF Full Text	P36217	N42H, N43D, Y59M, N61L, N70E, N76D, S142C, Q157A, I161E, N186C, Q194Y, Q194L, Q194H, Q194K	14	3
11917150	PDF Full Text	-	-	0	11
15129722	Abstract	P36217	T34C, T60C	2	2
15260499	PDF Full Text	P36217	T34C, N43D, Y59F, T60C, N70E, K90R, S142C, N186C, Q194H	9	3
15278768	PDF Full Text	P36217	T33C, T39C, N43D, S47C, Y58F, T59C, N70E, K90R, L105C, V139C, S142C, N186C, A189C, Q194H, Q194C	15	3
7764794	Abstract	P26514	F196Y, R197E, R197K, N214D	4	3
9201919	PDF Full Text	P07986	E274D	7	2
		P26514	H122R, H122S, H122Y, H248K, H248E, H248R		
9681873	PDF Full Text	P26514	H127E, H127Q, H127F, H127A, H127K, H127W	6	1
10235626	PDF Full Text	P26514	W126F, W126A, W126H, Y213F, Y213A, Y213S, W307H, W307A, W307F, W315F, W315A, W315H	12	4
9731776	PDF Full Text	P07986	E168A, H246A, H246N	3	2
			Total	**111**	**54**

Table 3.8: Mutation GraB versus Mutation Miner Performance.

Evaluation Metric	**Mutation Miner**		**Mutation GraB**	
	Protein-Organism	**Mutations**	**Mutations-Protein-Organism**	**Mutations**
Precision	0.91	0.84	0.84	0.91
Recall	0.46	0.97	0.90	0.97
F-Measure	0.61	0.90	0.87	0.94

Prot and Entrez Gene. In the ion channel article PMID 10653790, this same protein is represented by the term "NaCh", presumably as an abbreviation for "sodium channel". However, the term "NaCh" is not remotely close to any of the provided synonyms given for the Scn4a protein. Another example is the identification of the protein Q98146 in the GPCR article PMID 10842179. The only synonym given for this protein is "G-protein coupled receptor homolog 74", and the representation used in the article is ORF74. In both of these instances, our dictionary-based search could not have possibly identified the protein terms in the article with the synonyms at hand.

The PDF-to-text conversion of journal articles also often generates unintended changes with regard to protein names. One such consequence is the modification of superscript and subscript formatting present in some PDF files.

Another effect of the PDF-to-text conversion is the mishandling of Greek characters. The pdftotext utility replaces Greek characters with their Unicode representation, and unless the characters are represented in Unicode within the PDF, the conversion removes them. Many protein names, especially in the GPCR family, rely on these designations for differentiation from other similar proteins. While the Unicode representation is found in some PDF files, frequently other font or image representations are used to denote Greek

characters. When the non-Unicode Greek characters are removed from these names, they are either skipped or misidentified for other terms. The ion channel article PMID 10097182 describes the α, β, and γ ENaC proteins (Swiss-Prot ACs P37089, P37090, and P3791). During the PDF-to-text conversion, these characters were stripped, making it impossible to identify which ENaC proteins are being discussed.

A number of overlooked protein names in the GPCR literature set could be recovered using the permutation dictionary, however. For historical reasons, GPCRs were originally named by physiologists studying features, then by pharmacologists focused on tissue specificity, and finally by the genomics community based on sequence homology. Some GPCRs have been renamed on more than one occasion, and the order of naming elements is often permutated. These factors are less relevant to the ion channel and tyrosine kinase literature; thus, the use of a permutation dictionary increased the recall by identifying some full-length protein terms, but this benefit was limited to the GPCR family. One example where the permutation dictionary was useful is the "Parathyroid Cell calcium-sensing receptor" (Swiss-Prot AC P41180). Protein symbols for this term include "CaSR", "Gprc2a", "Pcar1", and "FHH"; a wide variety of legacy naming. Unfortunately, authors frequently use the term "calcium sensing receptor" to describe this protein. While that term is less specific than the original full name, it is specific enough to identify that single Swiss-Prot entry from the set of GPCR entries. A permutation dictionary helped recover this term while other protein entity recognition methods would probably have not. Owing to the proliferation of GPCRs in the olfactory tissues, the permutation dictionary also contained a large number of nonsensical permutation terms such as "receptor 31 17"

generated from the "Olfactory receptor 17-31" term (Swiss-Prot P58170). However, these nonsensical terms are unlikely to be found in the text and the additional cost of precomputing the permutation dictionary and additional searching is modest. The protein tyrosine kinase and ion channel transporter literature, in contrast, did not benefit from the use of the permutation dictionary for identifying additional terms. The literature for these protein families contained a more standardized nomenclature, and the use of the permutation dictionary only increased the number of spurious terms identified.

Since protein term identification is independent from the rest of Mutation GraB, a switch from one method to another is transparent to the other parts. While protein name identification is a necessary component of Mutation GraB, it is not the full focus of our efforts and is more thoughtfully addressed in the recent BioCreAtIve challenge.

3.5.4 Protein Disambiguation

The main task of the search metrics was to select the correct protein to associate with a point mutation when several proteins are found in the text. When only one protein is found whose sequence matches the point mutation wild-type amino acid, no disambiguation is necessary; the graph bigram and word distance metrics are not utilized. In instances where more than one protein can be assigned to a point mutation, the search metrics are used to disambiguate. Therefore, the main performance difference between the two search metrics is not the overall F-measure, but the precision measured in instances of multiple protein disambiguation.

Figures 3.6-3.8 provides evidence that the graph bigram metric performs better than the word distance metric in these instances. Figures 3.6-3.7 shows that the graph

bigram metric achieves a higher P at all levels of PPA greater than one, while Figure 3.8 shows the graph bigram metric ahead in all cases except for three PPA. The GPCR point mutations were ideal for this analysis because of the wide range of PPA values; a large number of mutations had two to six PPA. The protein tyrosine kinase and ion channel transporter literature sets contained fewer point mutations in general and a smaller spread of PPA. The ion channel transporter set, especially, contained more than twice as many point mutations with one PPA>1. This fact can explain why the overall F-measures between the two metrics for the ion channel transporter literature sets are quite similar. For the set of point mutations with PPA≥1, the precision P = 0.84 using the graph bigram metric and P = 0.73 using the word distance metric. This highlights the value of the graph bigram metric over the word distance metric in disambiguation situations.

The basic assumption in using a word distance metric for point mutation extraction was that the relative positioning between entities in text is the best barometer of associability and significance. We do know, however, that authors describing point mutations often reference nonassociated proteins in close proximity to point mutations, having referenced the associated protein in a different part of the text. This led us to conjecture that frequency as well as positional data, codified in the graph bigram search metric, would be a better method for associating entities for point mutation extraction. Data from Figures 3.6-3.8 with PPA>1 supports this conjecture.

3.5.5 Manual Point Mutation Annotation

To approximate the performance of Mutation GraB on a large scale of articles with the breadth of PubMed, it is important to develop and test it on a smaller set of

representative articles. The size of our algorithm development and validation sets reflects a compromise between what is possible and what is practical. The manual processing time for one article ranges from 10-60 min, depending on the number of point mutations in the article and the difficulty in validating them against the Swiss-Prot database. The definition of these "gold standard" annotations may change with each updated release of Swiss-Prot, as some protein accession numbers, protein names, protein keyword classifications, and organism names change with each release. The generation and updating of the gold standard annotations can take as long as 100 h per 100 article set. This, coupled with the current difficulty in retrieving full-text PDF articles from journal sources, makes it prohibitive to work with literature sets larger than 100 articles. Fortunately, the trends in electronic publishing and the more open dissemination of scientific literature favor the availability of an increasingly large set of full-text articles.

3.5.6 Point Mutations in Images

When identifying point mutations in an article, we counted mutations that occurred within images as true positive mutations. These point mutations were represented commonly as text that occurs in a graphical diagram or chart. Because the information encapsulated within the image is not accessible to text-mining methods, Mutation GraB cannot extract those mutations if they occur exclusively within images in an article. Since a human reader can still identify those mutations, we felt it necessary to include their presence in our gold standard sets. However, removing them from the gold standard sets can more accurately reflect Mutation GraB's performance on solely textual information. Table 3.9 shows the precision, recall, and F-measure of the three protein family literature sets, with

Table 3.9: Mutation GraB Performance on All Protein Family Literature Sets with and without Image Mutations Using the Graph Bigram Metric.

	GPCR		**Tyrosine Kinase**		**Ion Channel**	
	Image	**No Image**	**Image**	**No Image**	**Image**	**No Image**
Image Mutations	295	-	12	-	74	-
Precision	0.86	0.86	0.61	0.61	0.82	0.82
Recall	0.74	0.88	0.88	0.90	0.70	0.76
F-Measure	0.79	0.87	0.72	0.73	0.76	0.79

and without the image mutations, processed by Mutation GraB using the graph bigram metric. As expected, the presence or absence of image mutations only affects the recall because they are classified as either TP or TN by Mutation GraB. The GPCR literature set contained the most image mutations, and removing those mutations from comparison would increase the F-measure from 0.79 to 0.87. The tyrosine kinase and ion channel literature sets contained fewer image mutations, and, accordingly, have smaller gains in F-measure with their removal.

The GPCR literature set may contain a higher percentage of image mutations because the articles were taken from the tGRAP database and are expected to be more specific on point mutations than its tyrosine kinase and ion channel literature set counterparts. The tyrosine kinase and ion channel literature sets were randomly selected from a resulting PubMed search and have a lower point mutation density due to a lower specificity of subject matter. Nonetheless, when viewing the performance of Mutation GraB on the literature sets, it is important to consider the effect of the image mutations on the recall and overall F-measure.

3.5.7 Utility

Mutation GraB, if used on a large set of literature, has many potential downstream applications. The immediate benefit would be to generate a database of point mutations found in the literature that could be linked to both its literature and its protein database sources. The result of this database is the ability to examine the effect of point mutations on the structure and function of proteins within the framework of protein families, subgroups, and superfamilies. It is difficult to judge the amount of time saved by using Mutation GraB versus hand annotation, but we estimate this difference as significant. It took upward of 100 h to manually annotate 100 articles, whereas Mutation GraB processed the same volume of articles in about 3 h. Even taking into account hand correction of precision and recall errors, which took anywhere from 10-15 hours per 100 articles, Mutation GraB should still reduce the time required by 80% when compared with exclusively manual annotation. As with most text-mining applications, errors in precision are more tolerable than recall errors; we believe it is more important to identify and label the point mutation, even though the protein association may be incorrect, than to miss point mutations completely. At a F-measure estimate of 0.7, using Mutation GraB and correcting the precision and recall errors is still far more efficient than manual annotation alone. The utility and efficiency of Mutation GraB also relies upon the specificity of the literature given. As one can imagine, examining a very large set of nonspecific articles for a narrow set of protein point mutations will yield low performance.

3.6 Conclusion

From our development and validation of Mutation GraB, we can draw a few conclusions regarding the extraction of point mutations from biomedical literature. Foremost, it is entirely possible to utilize text-mining tools to extract point mutations at a level that warrants its usage. We can process 100 articles in anywhere from 1 h to 3 h, depending on the number of point mutations found within the articles. Also, we know that most articles discuss mutations originating from a single protein. In these instances, no further processing is required to correctly associate mutations and the proteins of origin. For articles that discuss more than one protein, however, a metric for choosing the right protein is necessary. One idea for finding the correct association between proteins and point mutations is to use the word distance between two entities as a metric for association significance. We thought that this metric was insufficient in many regards, and sought to improve it by incorporating frequency data with positional data to generate a heuristic for entity association. The result of this was a metric that combines bigram analysis with graph-theoretic searching that outperforms the simple word distance measure. The graph bigram metric for entity association could have many other applications in the biotext-mining field, and could increase the amount of information that can be automatically extracted from the biomedical literature.

Chapter 4

Identification and Extraction of Functional Point Mutation Effects

4.1 Abstract

Point mutations, both naturally occurring and researcher generated, provide scientists with a vast and diverse amount of information regarding protein evolution and function. This information, however, is communicated via publications and therefore embedded within the text. In order to leverage the large body of scientific literature regarding the effects of point mutations in a time-efficient manner, a computational approach is required to mine these effects from the literature. We explore the use of dependency trees and support vector machine (SVM) classifiers to identify and extract functional point mutation effects from biomedical literature. The dependency tree structure represents dependencies between non-local words in a sentence as edges in a directed acyclic graph. When utilized in semantic analysis, the dependency tree is an extremely powerful tool for examining the relationships between words and phrases. As used in this application, dependency trees can relate a point mutation to another biological entity through an effect relationship. We use a standard bag-of-words feature set, a hybrid features set, and a tree kernel with the SVM to identify and extract functional point mutation effects. The bag-of-words kernel generated an F-measure of 0.803 for identifying sentences that contain functional effects, while the hybrid kernel gave an F-measure of 0.472 for extracting the subtree-structure of a sentence that describes the functional effect.

4.2 Introduction

The introduction of point mutations into the coding sequencing in genomes provides a common route for evolutionary adaptation. Experimentalists have adopted mu-

tational analysis as a tool to probe biological structure and function. The result is the availability of a seemingly endless list of publications where mutations are introduced and the impact on protein structure is studied. While text-mining tools have been developed to catalog the list of mutations that have been studied in a given protein [Baker and Witte, 2006, Caporaso et al., 2007a, Horn et al., 2004, Lee et al., 2007], it has proven more difficult to extract the functional implications of a mutation in an automated way. In part, this is because there is a relatively limited lexicon for describing point mutations, but a much more varied syntactical approach to describing the functional impact of a mutation. Some examples of sentences describing functional point mutation effects are:

1. "In contrast, the cellular activity of GR remained almost unaffected by the W300A mutation but was dramatically sensitive to S485Y and T525I exchanges."

2. "Y214C is the most active mutation (11-fold increase in k(cat)/K(0.5)(h)) and exhibits the most severe clinical effects of hypoglycemia."

3. "Furthermore, a reciprocal mutation at position 515 (I515M), when introduced into the M626I background, acts as revertant mutation by allowing accommodation of the isoleucine sidechain at position 626 and fully restoring the constitutive activity to the level of wild-type TSHR."

These sentences show just a few of the possible described effects, ranging from modest changes in the activity of a protein, like in sentence (1) to functional alterations that alters enzymatic activity, in sentence (2), which in turn causes common heritable human diseases. In fact, the range of functional point mutation effects (FPMEs) is much

larger than of protein functions themselves, as point mutations can modify more than just the canonical function of its protein. The concept is further complicated by the ability of a second mutation to ameliorate or enhance the impact of the first mutation, as shown in sentence (3).

Text-mining has proven to be an effective automated approach to understanding biological interrogation networks. For example, it has been possible to create networks that identify the relationship or interaction between pairs of known biological entities such as proteins, genes, drugs, or diseases. Obviously, connecting a protein or gene to a disease has functional implications. Prior work in biological entity relationship extraction includes hand-coded and machine learning methods that have been applied to identifying protein-protein [Blaschke et al., 1999, Marcotte et al., 2001], gene-gene [Schafer and Strimmer, 2005], and gene-disease [Chun et al., 2006, Watkinson et al., 2008] interactions. Also, a task of the BioCreAtIvE challenge [Blaschke et al., 2005] is to assign a gene ontology term to a protein based on evidence in surrounding text, which is essentially identifying protein-function relationships. A unifying theme to identifying these interactions is populating a template with the desired information [Novichkova et al., 2003]. For example, a template such as `<protein><relationship><disease>` would be used to extract protein-disease interactions. These template approaches, while providing a good foundation for entity relationship extraction, are generally limiting because they rely heavily on *a priori* knowledge of the entities for extraction; knowledge of biological entity names are required for term identification. While biological ontologies are increasing in size and sophistication [Friedman et al., 2006], they are not detailed or specific enough to capture the effects of

point mutations. Furthermore, because the entities involved in describing FPMEs are not cataloged, a template approach cannot be applied since the target terms are unknown.

4.3 Approach

In this paper, we explore identifying and extracting functional point mutation effects using dependency trees, which have been used extensively in general natural language processing [Culotta and Sorensen, 2004, Moschitti, 2006, Pyysalo et al., 2006] and biomedical text-mining [Erkan et al., 2007, Fundel et al., 2007, Katrenko and Adriaans, 2006, Kim et al., 2008] tasks. Tree kernels have also been used in other biological classification tasks, such as glycan classification [Yamanishi et al., 2007]. A dependency tree can draw relationships between non-adjacent words in a sentence, which is a distinct advantage over other statistical parsers or chunkers. Figure 4.1 shows a dependency tree parsed from sentence (1) above. In the figure, the rectangles represent mutation nodes, the elliptical shapes represent other word nodes, and the edges connecting the nodes represent the dependencies between nodes. In the sentence, the noun phrase "cellular activity of GR" is related to three mutations, W300A, S485Y, and T525I. One described effect is that the cellular activity of GR is unaffected by W300A, which is highlighted in gray in Figure 4.2. The effect was extracted from a subtree-structure of the complete dependency tree for the sentence.

Support vector machines (SVM) are used to classify the trees and subtree-structures using standard kernels and a kernel designed for tree-based comparisons. SVMs have shown high performance levels in biomedical text-mining applications [Mitsumori et al., 2005] and are designed to handle the high dimensionality of data involved.

While the tasks are not mutually exclusive - successful extraction implies successful identification - we decided to approach the identification and extraction tasks as two separate problems. By implementing the identification of FPME containing sentences as an individual task, we overcome the problem of lacking *a priori* knowledge of the terms indicative of functional effects. Instinctively, the extraction task is far more difficult than identification. It is possible for sentences that do not contain a FPME to be parsed into a subtree that describes a false FPME. We tested our methods on a manually annotated set of sentences taken from PubMed abstracts that contain point mutation terms. For the identification task, we found that the SVM using the bag-of-words (BOW) features performed the best with an F-measure of 0.803. In the extraction task, we found that the SVM using the hybrid dependency tree features gave the best results with an F-measure of 0.472. To our surprise, the SVM using the dependency tree kernel gave the worse performance with F-measures of 0.660 and 0.152 for the identification and extraction tasks, respectively.

4.4 Methods

While we have divided the identification and extraction of FPMEs into separate tasks, we utilize many of the same supervised machine learning classifiers to solve both problems. We will describe the goals and process for each task, then describe each classification scheme as they relate to each task. Table 4.1 lists the classifiers, their mode of usage, and the tasks for which were tested.

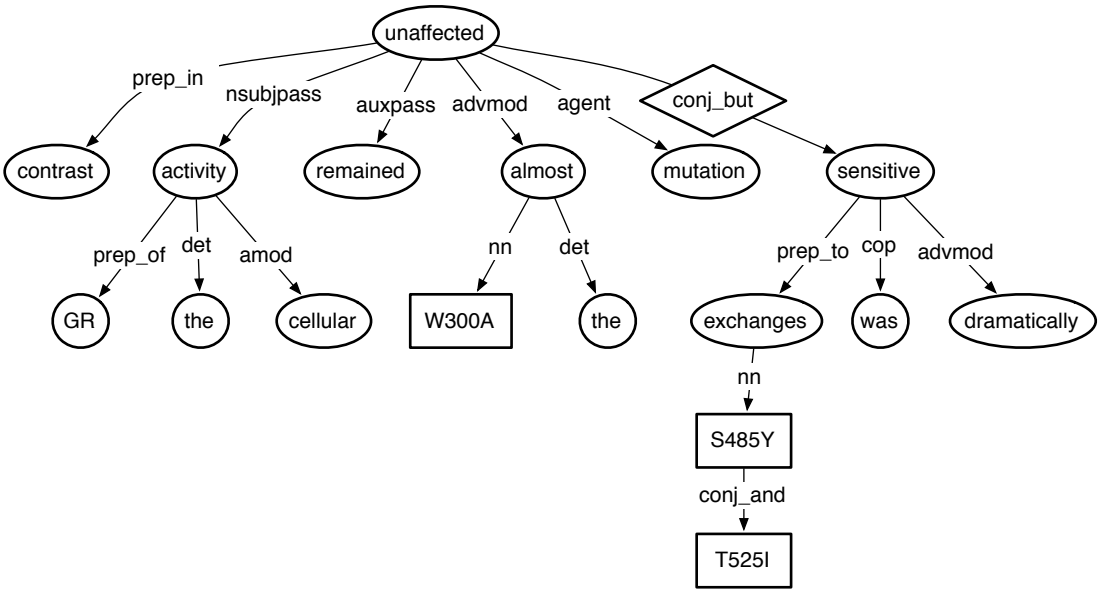


Figure 4.1: Structure of the typed dependency tree parsed from sentence (1) in the Introduction. The elliptical or rectangular shapes represent words in the sentence, and the directional edges represent the dependency relationship between words. Point mutation words are rectangle nodes, and the diamond-shaped edge label represents a conjunction dependency.

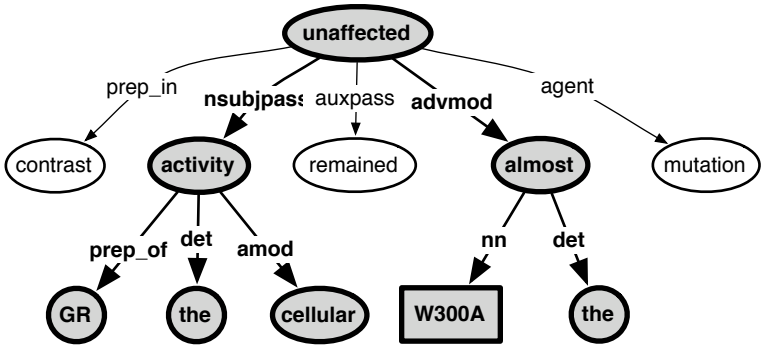


Figure 4.2: Subtree-structure containing an FPME from the original tree shown in Figure 4.1 and a detailed representation of subtree-structure I shown in Figure 4.3. The FPME-relevant nodes and dependencies are highlighted in bold.

Table 4.1: Classifiers, feature sets, and use in identification and extraction tasks. The feature sets are described in the Classifiers section.

Classifier	Feature Set	Identification	Extraction
SVM BOW	top	Yes	No
SVM BOW	all	Yes	Yes
SVM BOW	hybrid	Yes	Yes
SVM dependency tree kernel rule	N/A	Yes	Yes

4.4.1 General text retrieval and pre-processing

PubMed abstracts were retrieved that were labeled with the MeSH term "point mutation". We used regular expressions described in Mutation GraB Lee et al. [2007] to identify sentences that contain protein and DNA mutations. These sentences were part-of-speech (POS) tagged (e.g. NN for noun and VB for verb) using the MedPost tagger [Smith et al., 2004]. The sentences were then chunked using the OpenNLP English[1] treebank chunker into non-overlapping phrases such as NP for noun-phrase and VP for verb-phrase. The POS tagged sentences were fed into the Stanford NLP parser[2] to generate typed dependency trees.

4.4.2 Identifying FPME containing sentences

The goal of the identification task is to classify whether sentences contain FPMEs. In this task, we employed all three flavors of our SVM classifiers: BOW, dependency tree hybrid, and dependency tree kernel.

[1] http://opennlp.sourceforge.net/
[2] http://nlp.stanford.edu/software/lex-parser.shtml

Sentence annotation

We labeled sentences as containing FPMEs, or true positives, if they describe "molecular", "cellular", "interaction", or "disease" relationships. A molecular description is a qualitative or quantitative change in the biochemical, structural, enzymatic, and/or physical properties of a protein. Cellular descriptions are changes in the properties of the host cell, virus, tissue, and/or organ and associated processes. Interaction descriptions are changes in interactions of protein with other elements of its environment, and finally, disease descriptions specify the cause of change in disease phenotype due to the point mutation. Table 4.2 lists some sentences and their category of description and some non-FPME containing sentences. We did not consider mutation frequency or haplotype information without disease implications, as in sentence 7, to be functional. In sentence 8, there is an implied relationship between the point mutation and polycythemia vera pathogenesis, but not an explicit statement. Sentence 9, likewise, implies an interaction between `AChR` and `rapsyn`, but the actual assayed interaction is not presented in the sentence itself.

4.4.3 Extracting FPMEs from sentences

The goal of the extraction task is to classify whether subtree-structures contain FPMEs, which is appreciably different than classifying at the sentence level. To extract the FPME from its containing sentence, we divided the dependency tree structure of the sentence to identify the specific part that contains the functional effect. The SVM BOW and dependency tree kernel classifiers were used for this task, however, only unigram features were trained in the SVM BOW model.

Table 4.2: Types and examples of different FPME containing and non-containing sentences.

	Functional Effect	**Sentence**
1	molecular, cellular, interaction, disease	Here we compare the molecular phenotypes of two previously identified PPARgamma mutations: P467L, reported to be dominant negative; and F388L, reported to be devoid of dominant-negative activity.
2	molecular	There was no change in the stability of the AGT-Cys145S-(CH2)2Br intermediate formed in mutants Y158H and P140K.
3	cellular	We first transfected hippocampal neurons in culture with recombinant gamma2 constructs and showed that the gamma 2(R43Q) mutation prevented surface expression of the subunit, unlike gamma2(K289M) substitution.
4	disease	In contrast, patients with a 2713C>T (R905W) or a 2713C>G (R905G) mutation had more severe phenotypes.
5	interaction	Four mutations in the HCV protease (R155Q, A156T, D168A and D168V) have been identified in vitro in the HCV replicon system that confer resistance to BILN-2061 (a reference inhibitor).
6	molecular, cellular	The phenotype argues for dominant-negative activity for the P70T amelogenin, and for the robust nature of the process of amelogenesis.
7	None	Therefore, our results suggest that C1494T is a very rare event.
8	None	Also, continuing studies on the recently discovered JAK2V617F gene mutation may significantly improve our understanding of PV pathogenesis and facilitate its medical management.
9	None	We therefore studied the interaction of AChR containing the CHRND E381K mutation with rapsyn by evaluating expression and co-localization of rapsyn and mutated AChR subunits in co-transfected HEK 293 cells.

Subtree-structure generation

The process for generating subtree-structures is shown in Figure 4.3, which lists the subtree-structures generated from the dependency tree in Figure 4.1. First, mutation subtrees are generated by traversing up the tree starting from a mutation node. For each parent node traversed until the root node is reached, their child nodes are added to complete the subtree. Nodes 5, 7, 13, 16, and 1 are parents of point mutation nodes, so trees A-E are subtrees of tree A. To further diversify branches of the subtrees into structures that may more accurately describe a FPME, we create "bi-branch" and "conjunction branch" substructures from the original subtrees. First, branches of the tree that have a height greater than one, or non-terminal branches, are identified. Bi-branch substructures are generated by taking non-terminal branches and enumerating pairwise combinations of them as children of the same root node of the subtree. As shown in Figure 4.3, subtree A has three non-terminal branches at the nodes 3, 5, and 7. Enumerating different pairwise combinations of those nodes with the root node 1 yields the substructures F-H. Child nodes that are end leaves, or terminal nodes, such as 2, 4, and 6, are often neutral in FPME descriptions, are not enumerated in this process, and are kept in all resulting substructures. However, if all non-terminal branches contain a mutation term, as in substructure J with branches at nodes 5 and 13, then each branch is iteratively added to the root node with the terminal nodes as well. While this rule was not applicable for subtree A, when applied to substructure J forms substructures K and L.

Conjunction branch subtree-structures are generated when the conjunction words "and", "but", and "or" are encountered. When used in a dependency tree, the nodes

connected by a conjunction dependency can sometimes share the same child branches. In tree A, the dependency between nodes 1 and 7 are linked by a *conj_but* relationship shown by the blue diamond. To link the children of node 1 with node 7, we again first identify non-terminal branch children of node 1, then iterate through those branches and add them as children to node 7. For the case of subtree A, substructures I and J are generated, which are shown in Figures 4.2 and 4.6, respectively. Also, since the conjunction branch substructures are "new" trees, we recursively apply the bi-branch and conjunction branch rules to them. For substructure I, no new bi-branch or conjunction branch substructures can be made, while substructure J forms two new bi-branch substructures in K and L as described above.

Subtree-structure annotation

All sentences that are annotated as true negatives generate subtree-structures that are also true negatives. However, not all subtree-structures generated from true positive sentences will be true positives. We label a subtree-structure as true positive if it describes a single FPME while excluding nodes that describe other content. Nodes are allowed in the subtree-structure if they are neutral or complementary to the target FPME, but nodes that describe other content or add ambiguity to the FPME are not allowed in true positive subtree-structures. In Figure 4.3, the original tree A contains two FPMEs from the sentence and is therefore labeled a true negative subtree. The effects are more specifically found in substructures F and I shown in Figures 4.2 and 4.6, which are both labeled as true positive subtree-structures. Figures 4.4 and 4.5 also show other subtree-structures that describe FPMEs.

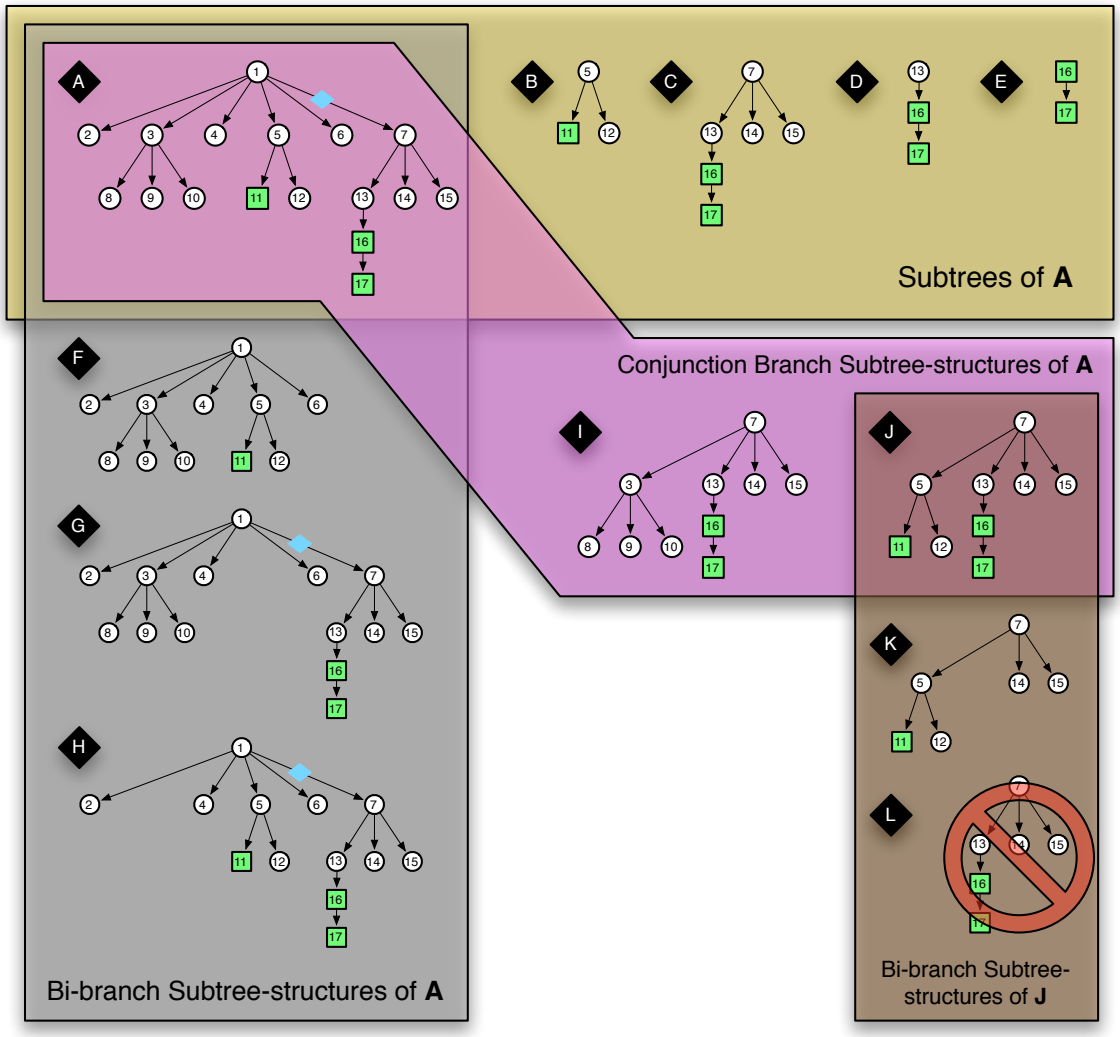


Figure 4.3: The subtree-structures generated from the tree shown in Figure 4.1. Subtree A is the same as the original tree in Figure 4.1. The words within the nodes are replaced by numbers in this figure, but the green square nodes still represent mutation nodes. The blue square on the edge between nodes 1 and 7 represent a conjunction dependency. First, subtrees B-E are generated from A by taking the parent nodes of the mutation nodes and including those parents children. Next, bi-branch subtree-structures F-H are generated by enumerating pairs of branches from node 1 that are not terminal nodes. Next conjunction subtree-structures I and J are made by placing node 7, the head of the conjunction branch, as the parent of branches with head nodes 3 and 5. Since I and J are new subtree-structures, we recursively apply the previous rules to them. Bi-branch subtree-structures K and L are generated from J, and L is discarded since it is the same as C.

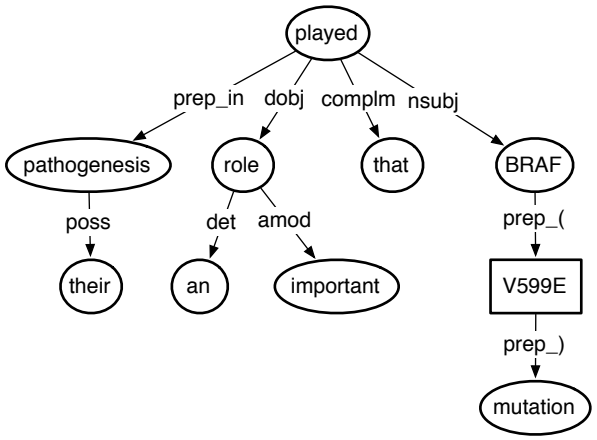


Figure 4.4: An FPME-containing subtree-structure generated from the sentence "BRAF(V599E) mutation rate was high in classic type PTC and tall cell type inferred that BRAF(V599E) mutation played an important role in their etiopathogenesis."

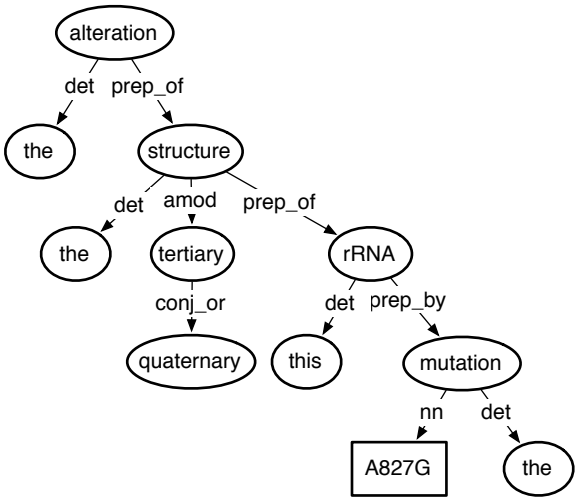


Figure 4.5: An FPME-containing subtree-structure generated from the sentence "It is possible that the alteration of the tertiary or quaternary structure of this rRNA by the A827G mutation may lead to mitochondrial dysfunction, thereby playing a role in the pathogenesis of hearing loss and aminoglycoside hypersensitivity."

4.4.4 Classifiers

Support vector machine with BOW features

Features used for the SVM BOW classifier were pooled from unigrams (single words such as activate) and bigrams (adjacent words such as activated by) found in the sentences. Frequently, the entire list of features, which we will call the "all" feature set, is pruned to create a smaller and more focused list of features to reduce computation time and overtraining. We used an information gain (IG) metric to select a subset of features from the all list for placement in the "top" feature set. The IG of a feature is the change in information entropy of the system if that feature is known and words with a higher IG value are thought to have a higher predictive value in the classification system. Words in unigram and bigram features were stemmed using the Porter stemming algorithm [Porter, 1980]. From a training set, an initial list of features with an IG>0 was created. Non-general features such as mutation terms ("G468T"), protein names ("EGFR"), and disease names ("cystic fibrosis") were removed from this feature list.

As customary with a BOW feature set, a sentence is represented by a binary vector representing the presence or absence of a particular feature. For the identification task, both "top" and "all" feature sets were used, but for the extraction task, which we based on the dependency structure of the sentence or subtree-structure, we only used the "all" features set without bigrams. The libsvm package (version 2.85) [Chang and Lin, 2001] was used to implement the SVM. We used the radial basis function (RBF) kernel with the values C=256 and gamma=0.001953125, which were discovered using the parameter optimization script provided by libsvm.

Support vector machine with dependency tree hybrid features

This method uses information generated by the dependency tree to supplement words in a BOW-fashion. Unlike the BOW features, the words in each sentence are prefaced by their branch locations, and other prominent feature types are added. The head node, which carries a large amount of information content regarding possible FPMEs, and its part-of-speech are also included as features. The tree or subtree-structure is separated into branches that contain or do not contain mutation terms, and the words in each branch prefixed as such. An exception is the mutation term itself, which is excluded as a feature. The dependency relationships between the head node and branches are also included as features.

Table 4.3 shows the list of features generated from subtree-structure I in Figure 4.6. In the subtree-structure, there is one mutation containing branch and three non-mutation branches. The root node is "sensitive", which is stemmed to "sensit" and prefixed with "headword" to "headword_sensit". The part-of-speech is an adjective, or "headpos_JJ". The mutation branch relationship is *prep_to*, while the non-mutation branch relationships are *nsubjpass*, *cop*, and *advmod*. The only mutation branch word feature is "mbword_exchang", stemmed from "exchange", and the non-mutation branch words are generated in the same fashion.

Support vector machine with dependency tree kernel

In contrast to the SVM BOW and dependency tree hybrid feature classifiers, which use the RBF kernel to compute the "similarity" between sentences, our dependency tree

Table 4.3: Hybrid dependency tree features from the subtree-structure in Figure 4.6

Feature Set	**Feature**
Head word	headword_sensit
Head pos	headpos_JJ
Mutation branch relationship	mbrelp_advmod
Non-mutation branch relationship	nmbrelp_nsubjpass nmbrelp_advmod nmbrelp_cop
Mutation branch words	mbword_exchang
Non-mutation branch words	nmbword_activit nmbword_GR nmbword_the nmbword_cellular nmbword_was nmbword_dramat

kernel function generates a similarity score between two trees based on tree content and structure. For each word in the sentence, the part-of-speech, chunk group, and dependency information is used to generate a list of features shown in Table 4.4. The *general_pos* feature of a node is the basal form of its part-of-speech feature. While the unigram and bigram features were the word(s) themselves, the features described here include the word name as well as multiple properties of the word. Once we created the dependency tree and associated features for each sentence, we trained and tested our datasets with libsvm using our own tree kernel function.

The tree kernel function computes the similarity of common subtrees between any two trees. We modeled our kernel method after other kernel methods [Collins and Duffy, 2001, Culotta and Sorensen, 2004, Moschitti, 2006] with the feature determination closely resembling that from [Culotta and Sorensen, 2004]. Let **T** represent a dependency tree with

Table 4.4: Dependency tree kernel features and examples.

Feature Set	**Example**
word	"activit", "E381K"
pos	NN, NNP, VB, VBP, MUT
general_pos	noun, verb, adjective, adverb
chunk_tag	NP, VP, ADJP
parent_dependency	nsubj, dobj, prep_of

the node list $\{t_1 \dots t_n\}$ where t_i refers to both a node in $\mathbf{T}$ and a feature vector $\{v_i^1 \dots v_i^d\}$ with length d. We also use the notation $v_i^{feature\_type}$ to refer to a feature by its feature type rather than vector index. We define two functions over tree nodes: a matching function $m(t_i, t_j) \in \{0, 1\}$ and a similarity function $s(t_i, t_j) \in (0, \infty]$. We define

$$m(t_i, t_j) = \begin{cases} 1 & \text{if } v_i^{word} = v_j^{word} \\ 1 & \text{if } v_i^{pos} = mut \text{ and } v_j^{pos} = mut \\ 1 & \text{if } v_i^{general\_pos} = v_j^{general\_pos} \text{ and} \\ & \quad v_i^{parent\_dependency} = v_j^{parent\_dependency} \\ 0 & \text{otherwise} \end{cases} \tag{4.1}$$

and

$$s(t_i, t_j) = \begin{cases} d & \text{if } v_i^{pos} = mut \text{ and } v_j^{pos} = mut \\ \sum\limits_{v_i^q \in t_i} \sum\limits_{v_j^r \in t_j} C(v_i^q, v_j^r) & \text{otherwise} \end{cases} \tag{4.2}$$

where $q = r$ and $C(v_i^q, v_j^r)$ is a compatibility function between two features such that

$$C(v_i^q, v_j^r) = \begin{cases} 1 & \text{if } v_i^q = v_j^r \\ 0 & \text{otherwise} \end{cases} \tag{4.3}$$

The matching and similarity functions provide separate means for node comparisons. We use the matching function first to see if two nodes match, then calculate the similarity of the nodes accordingly. The function m returns 1 if the word features are the same between two nodes, if the part-of-speech for both words are "MUT", or if both general POS and parent dependency are the same. Intuitively, we want two nodes to match if their words are the same or if they are both mutation terms, regardless of the other features. If those conditions are not met, then we can also allow for a match if the general part-of-speech and parent dependencies are equal between two nodes. The function s returns the length of the feature vector d if both nodes contain mutation terms or the sum of features with the same value between two nodes. Again, we want to place more weight on mutation-mutation matches than other matches containing a mutation term. Given two trees T_1 and T_2, we can then define the kernel function

$$K(T_1, T_2) = \sum_{t_i \in T_1} \sum_{t_j \in T_2} \Delta(t_i, t_j) \tag{4.4}$$

where $\Delta(t_i, t_j)$ calculates the aggregate similarity of nodes contained in two subtrees rooted at t_i and t_j. We define Δ by

$$\Delta(t_i, t_j) = \begin{cases} 0 & \text{if } m(t_i, t_j) = 0 \\ s(t_i, t_j) & \text{if } m(t_i, t_j) = 1 \\ & \text{and either } t_i \text{ or } t_j \text{ are leaf nodes} \\ s(t_i, t_j) + \lambda \sum_{c_{t_i}^x \in c_{t_i}} \sum_{c_{t_i}^y \in c_{t_i}} \Delta(c_i^x, c_j^y) & \text{if } m(t_i, t_j) = 1 \text{ and } t_1 \text{ and } t_j \text{ have} \\ & \text{child nodes } c_{t_i} \text{ and } c_{t_j} \end{cases} \tag{4.5}$$

where c_{t_i} are the child nodes of t_i and $c_{t_i}^x$ is the x^{th} child node. If the node does not match, there is no need to process its children so we return 0 immediately. If the nodes match and at least one is a leaf node, we return the similarity score of the two nodes. Otherwise, if the nodes match and they both have child nodes, we recursively sum the similarity score of the nodes with the Δ of their children. Additionally, we include the decay factor to penalize the similarity of matching nodes away from the root node. We tested λ of 0.25, 0.5, and 0.75 and found little difference in the resulting performance, but omitting λ hurt performance significantly. By iterating over each node in a tree, we implicitly sample all the subtrees of the tree since the Δ function recursively iterates through the children of every node. Larger subtrees will contribute a bigger Δ to the kernel function, and λ ensures that "child" subtrees will not be counted multiple times if their "parent" subtrees match. Finally, we normalize K to be between 0 and 1 by the function

$$K'(T_1, T_2) = \frac{K(T_1, T_2)}{\sqrt{K(T_1, T_1)K(T_2, T_2)}} \tag{4.6}$$

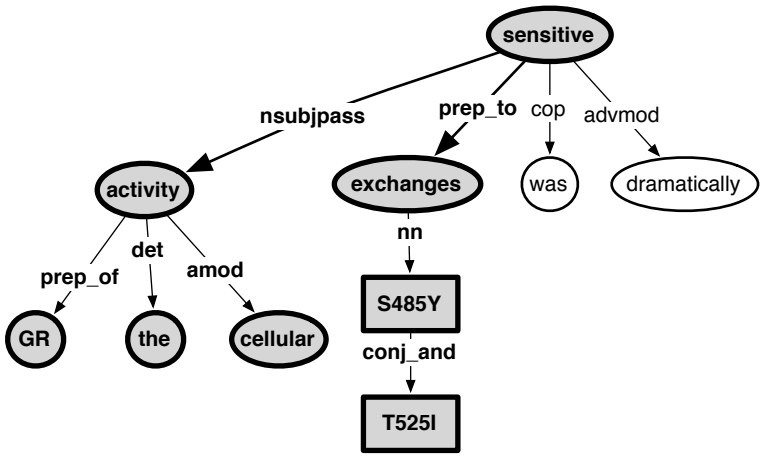


Figure 4.6: Subtree-structure containing an FPME from the original tree shown in Figure 4.1 and a detailed representation of subtree-structure J shown in Figure 4.3. This subtree-structure was generated by moving branches around a conjunction relationship node. The FPME-relevant nodes and dependencies are highlighted in bold.

4.5 Results

In this section, we show the evaluation of the SVM classifier using different feature sets for the identification and extraction of FPMEs from PubMed abstract sentences. Sentences that contained point mutation terms were extracted and randomly separated into training and test sets. 2003 sentences were identified and 1001 were used for the training set and 1002 for the test set. After manual annotation, the training set consists of 603 FPME containing, or true positive (TP), sentences and 398 non-FPME containing, or true negative (TN) sentences. The test set contains 526 TP and 476 TN sentences. For the identification task, the sentences themselves were classified as FPME containing or non-containing.

When decomposed into subtree-substructures for the extraction task, the training and test set sentences generated 5871 (858/5013) and 5676 (746/4930) subtree-structures, respectively. Manual annotation labeled 858 FPME containing, or true positive (TP)

Table 4.5: Results of FPME identification.

	SVM BOW			**SVM dependency tree kernel**
Feature Set	top	all	hybrid	N/A
Precision	0.835	0.827	0.797	0.692
Recall	0.730	0.781	0.738	0.631
F-measure	0.787	0.803	0.766	0.660

subtree-structures, and 5013 non-FPME containing, or true negative, subtree-structures. The test set subtree-structures divided into 746 TP and 4930 TN subtree-structures. We evaluated the classifiers using the standard precision, recall, and F-measure statistics.

4.5.1 FPME sentence identification

We tested the SVM classifier with both "top" and "all" feature sets, the hybrid dependency tree feature set, and the dependency tree kernel. In cross validation tests on the training sets, we found that using both unigram and bigram features increased the F-measure marginally from 0.841 to 0.851. As a result, we chose to include both types of features for the "top" and "all" sets. There were 20,931 in the "all" set, and using the IG filter for feature selection narrowed the number to 424 in the "top" set. Those results for all the classifiers and feature sets are shown in Table 4.5. The SVM BOW employing the "all" feature set yielded the best F-measure of 0.803. The SVM dependency tree kernel gave the lowest F-measure at 0.660.

4.5.2 FPME extraction

We tested the SVM classifier with the BOW and hybrid dependency tree features, in addition to the dependency tree kernel. While the BOW features generated a higher F-

Table 4.6: Results of FPME extraction.

	Bag-of-words	**Dependency tree hybrid**	**Dependency tree kernel**
TP	245	352	671
TN	4531	4371	1805
FP	439	399	3125
FN	508	401	75
Precision	0.358	0.467	0.177
Recall	0.325	0.469	0.899
F-measure	0.341	0.468	0.295

measure than the hybrid dependency tree features did in the identification task, the reverse was true for the extraction task. The results in Table 4.6 show that the hybrid dependency tree features gave the highest F-measure at 0.468, the BOW features an F-measure of 0.341, and using the dependency tree kernel yielded an F-measure of only 0.295.

4.6 Discussion

In our evaluation of SVM classifiers for the identification and extraction of FPMEs, it was expected that performance on the identification task would be more successful than the extraction task. The problem of sentence classification is simpler in nature, and certain words are good indicators for a FPME description. As a result, the classifiers using the BOW features performed much better than their more complicated counterparts. In our earlier tests, the addition of bigrams generated an F-measure increase of only 0.01 and provided higher performance gains in cross-validation. SVMs using the hybrid dependency tree features and the dependency tree kernel yielded much lower F-measures than the BOW features for the identification task.

Our initial intuition was that the combined presence of grammar relationships and

words would be good indicators for FPME descriptions. However, the results show that incorporating semantic relationships via the dependency tree kernel degrades performance, indicating that word presence or absence alone may be sufficient for identifying FPME containing sentences. The kernel function, which compares the similarity between subtrees of two dependency trees without regard to the significance of the content of the subtrees, may also be responsible for suboptimal results. It is possible that FPME-containing and non-containing sentences share many similar subtrees that are not related to the effects at all. Since the kernel function does not discriminate between subtree content when calculating similarity, the actual contribution of FPME-containing subtrees is diminished as a whole.

Also, because multiple pre-processing steps were made before the actual classification task, the results of the classifier are tied closely with the part-of-speech tagging, chunking, and dependency tree generation. Any error in the pre-processing steps will propagate errors into the classifier. An examination of 100 dependency tree parses in the training set showed that 37 of them could have been better structured for FPME-extraction purposes. Since the tree structure dominates the dependency tree kernel calculations, a suboptimal parse can render two otherwise similar FPME-containing sentences to have a low kernel score.

The hybrid dependency tree features performed better than the dependency tree kernel for the identification task, but still worse than the BOW features. One main explanation is the choice of the root, or head, node as a prominent feature in the hybrid feature set. If the root node has no relation to the FPME, the performance will suffer.

In contrast, for the extraction task, the original dependency tree for each sen-

tence is decomposed into many subtree-structures and the root node of the correct subtree-structure can be exposed as a feature. Referring again to Figure 4.1, the root node of the tree happens to be a component of an FPME description. However, if we do not decompose the tree into subtree-structures, the root node for substructure I would not be revealed as a feature to the classifier. Many such sentences have the FPME buried deeper within the dependency tree. By generating these subtree-structures for each dependency tree, we allow for the possibility of identifying more FPMEs at the cost of adding many true negative structures. We believe that our method of subtree-structure generation creates the most diversity of substructures without completely iterating over every substructure possibility. Again, the possible suboptimal generation of dependency trees by the parser will also have a large effect on the classification results.

Unlike the hybrid dependency tree features, the dependency tree kernel performed poorly for our extraction task for reasons more related to the framing of the problem than the kernel itself. The dependency tree kernel calculates similarities between trees based on shared subtrees. When many subtree-structures are generated from one dependency tree, those subtree-structures will have a high similarity from the perspective of the kernel method to each other and to the original dependency tree. Ultimately, the reasons for the poor performance in the identification task are exacerbated by the generation of so many subtree-structures. A filter for the removal of obvious non-FPME describing subtree-structures may indeed improve the precision values of our methods.

Other dependency tree kernel methods avoid this problem by employing a template approach to identifying target subtree-structures. Fundel et al. [2007] use this approach to

target different "effector-effectee" relationships between proteins. An option would be to target a select number of FPME types which are known, but this would leave the majority of FPME unidentified and predictably yield a high-precision and low-recall result.

Each method generated an F-measure below 0.5, which can be misleading without the proper context. As previously mentioned, many subtrees are generated and evaluated for each sentence and the majority of subtrees will be true negatives. As such, even a random classifier would result in an extremely low precision proportional to the TP:TN ratio because it would label a large amount of false positives. In this instance, less than 10% of our subtrees contain a FPME, and a random classifier should perform with a precision of approximately 0.10. Conversely, if a random classifier missed half the true positive subtrees, the recall would be expected to be approximately 0.50. Thus, a random classifier would be expected to perform at an F-measure of 0.17. Comparatively, our dependency tree hybrid F-measure of 0.468 is a vast improvement.

To our knowledge, there is no prior work in FPME identification and extraction. There have been evaluations of dependency trees as applied to the text mining of entity relationships, but differences in methodological approaches makes direct comparisons difficult. RelEx [Fundel et al., 2007] achieves an F-measure of 0.82 for extracting gene and protein relations from the LLL-challenge dataset, but this dataset contains only 55 training and 80 test sentences. In [Erkan et al., 2007], dependency trees were used to solve the BioCreAtIvE Protein Protein Interaction (PPI) and Protein Interaction Sentences (PIS) subtasks, and they achieved F-measures in the range of 0.08 to 0.23, which are low by normal standards. Dependency tree kernel approaches are also varied in terms of methods and

goals. In [Moschitti, 2006] the more general linguistics task of classifying predicate argument structures was attempted using a SVM dependency tree kernel method, which yielded accuracy scores of 0.8-0.9. In [Zelenko et al., 2003], dependency tree kernels were used to identify person-affiliation and organization-location relations at a 0.87 and 0.83 F-measure performance level, respectively. In a dissimilar application of tree kernels, [Yamanishi et al., 2007] represented human glycans, or chains of sugar molecules, as trees and classified them with their blood components at accuracy levels in the 0.7-0.95 range.

For all of these previous applications except for the glycan classification, a binary relationship was being identified or extracted. FPMEs are sometimes, but not always represented as a binary entity relationship, and approaching it as such would result in a low recall performance evaluation. Given the difficulty of our problem, we feel that an F-measure of 0.468 is a reasonable result for the extraction of FPMEs.

4.7 Conclusion

We have explored multiple methods for identifying and extracting functional point mutation effects from biomedical literature. These effects are valuable pieces of information used to explore the sequence-structure-function relationship in proteins, and ultimately can provide insight into many disease systems. Our methods use standard supervised machine learning techniques along with publicly available natural language processing tools. While the most promising method, the SVM employing a dependency tree kernel, was ill-suited for our task, with further modifications it can be tailored to fit the problem of functional point mutation effect extraction. In the meanwhile, we have also presented a method that

uses the dependency tree structure to provide independent variables that can be fed into many different machine learning classifiers. While this method does not produce as high an F-measure as most template based approaches, it is not limited to extracting primarily binary relationships between known entities.

Chapter 5

Application on Cystic Fibrosis Literature

5.1 Abstract

Point mutations are a vital source of information that relates changes in protein sequence and structure to functional phenotypes at the organism level. A universal application of point mutation identification is to elucidate the mechanisms of diseases. Inheritable diseases and cancers, amongst others, have their foundation of action rooted in changes in DNA and protein sequences. Often times, deletions, insertions, and point mutations generate a dysfunctional protein which changes a critical cellular mechanism. Cystic fibrosis is a genetic disease whereby a mutation in the Cystic Fibrosis Transmembrane Conductance Regulator (CFTR) affects the ability of the ion channel to make its way to the cellular membrane and function. This results in the clinical manifestations of cystic fibrosis symptoms. We present here an application of Mutation GraB on cystic fibrosis full-text literature. We show that a computational approach to identifying point mutations in literature results in a far larger corpus of literature analyzed and a more diverse set of point mutations extracted when compared to manual curation. If used to supplement manual curation methods, Mutation GraB could significantly save both time and effort in the upkeep of mutation databases.

5.2 Introduction

The ultimate test in utility of a text mining application is not its performance on randomly generated test sets, but its ability to increase the productivity, coverage, and efficiency of end users. In the case of point mutation extraction tools, one metric of utility is the ability to discover new information in the text which was previously unknown or unseen.

While manual curation has proven useful for many electronic respositories, its dependence on external participation can be perceived as a strength and a weakness. When a critical mass participates in the addition and curation of information to a database, its contents are of great value to other researchers. However, if not enough participants are active in the curation and upkeep of a database, or if the desired information is fragmented between multiple competing databases, the results are often incomplete and outdated databases of little value.

Curators whose sole purpose is to manually identify and extract information for inclusion in these databases cannot keep up with the amount of information generated. Text-mining applications, in this context, can be of great benefit to database curators and administrators. Yeh et al. [2003] explored the challenges and benefits of using text mining for database curation and found difficulties in document accessibility and quantifying the actual benefits of text-mining aided curation. Research done by Miotto et al. [2005] proposed a framework of document classification and machine learning methods to classify relevant literature according to a search subject. Ideally, by implementing an application that can identify literature of interest or filter out non-target literature, database curators can spend less time searching through literature and more time identifying their desired information. A type of information which is the target of many different databases are protein-protein interactions. The MIPS [Mewes et al., 2006], BIND [Bader et al., 2003], and DIP [Xenarios et al., 2002] databases are few examples of expert curated databases for protein-protein interactions. Other curated databases exist for a diverse amount of information such as enzymatic reactions [Barthelmes et al., 2007], transcription factor binding sites [Matys

et al., 2003], and genetic polymorphisms [Sherry et al., 2001]. We will focus on the promise of text-mining applications to help the curation of point mutation databases using our Mutation GraB [Lee et al., 2007] application.

MutationGraB was developed to identify and extract point mutations from biomedical literature, assigning a protein of origin to the mutation term as well. We compare the protein point mutations extracted by Mutation GraB to those manually curated in the Cystic Fibrosis Mutation Database (CFMD) [Tsui, 1992] and in Swiss-Prot. We chose to study the cystic fibrosis system and those two databases for a number of reasons. First, the body of literature for cystic fibrosis is large, but not prohibitively so. If we were to search for mutations in all proteins, the size of the corpora would be too large. Second, the CFMD and Swiss-Prot databases are actively and accurately curated, and the contained information can generally be trusted. Finally, point mutations have a specific context in heritable diseases, and the analysis of disease related point mutations can lead to new insights into disease treatment and drug research. We limit the comparable mutations to the protein sequence because the DNA sequence often varies between literature sources and makes sequence comparison and validation troublesome.

The goals of this project are to:

1. Compare the number of point mutations extracted by an automated (Mutation GraB) versus a manual (CFMD) system.

2. Explore the increase in coverage and efficiency that a combined automated and manual curation system can achieve over each alone.

5.3 Background

5.3.1 Cystic Fibrosis

Cystic fibrosis (CF) is an autosomal recessive inherited disease caused by a defect in the cystic fibrosis conductance regulator (CFTR) gene. This defective gene results in the secretion of a thick mucous from epithelial cells, leading to clogged airways and infections in the lungs and the inability to digest and absorb food in the intestinal tract. This disease is most common in children and affects 70,000 people worldwide. There is no cure for the disease, and aggresive treatment is necessary to prolong and improve the quality of life for patients.

Whether a defective CFTR protein affects the lungs or the digestive tract, the primary cause of CF symptoms is the mucosal obstruction of exocrine glands [Rowe et al., 2005]. In the lung, CFTR is expressed by the submucosal glands, and defective CFTR mutations result in the build-up of a thick mucous that blocks airways. Besides creating breathing difficulties, the mucosal build-up also creates an environment where bacteria can readily grow. Pathogens find the warm and moist environment to be hospitable for growth, and *P. Aeruginosa* even secretes a circular polysaccharide which blocks antimicrobial agents from reaching the infections. In addition to pathogen infection, tissue inflammation is another cause of decreased lung function. The mucosal build-up causes elevated levels of inflammatory proteins and cytokines, both of which result in a persistant inflammatory response in the lung.

While the lung is the most commonly affected tissue in CF patients, other tissues that express CFTR are also summarily affected. In men, cystic fibrosis usually leads

to infertility because of the obstruction of the vas deferens, while in women, cervical mucous or malnutrition can result in the same. Cystic fibrosis also affects the pancreas by blocking enzyme secretion. This results in the diminished uptake of fatty acids and can lead to malnutrition, fatty diarrhea, and pancreatitis. The intestines may also be blocked by the mucous, preventing absorption of food and leading to increased fecal volume and malnutrition.

5.3.2 The CFTR Protein

The CFTR gene was isolated using linkage-based mapping [Riordan et al., 1989] to chromosome 7q and found to contain little resemblance to other ion channels. CFTR is 1480 amino acids in length and contains two membrane spanning domains, two nucelotide binding domains, and a regulatory domain. It is a member of the ATP-binding cassette (ABC) gene family. While its function is believed to be ion transport, CFTR's role as a signal transducer influences the expression of other proteins involved in inflammatory responses, maturational processing, and cell signaling [Rowe et al., 2005]. Primarily, CFTR is a PKA-regulated chloride transporter, but it also regulates sodium and potassium transport as well.

5.3.3 CFTR Mutations

While the CFTR protein alone is responsible for cystic fibrosis, over 1,500 different CFTR mutations have been discovered that can affect the clinical phenotype of the disease. Additionally, many modifier proteins have been found which interact with CFTR to also produce wide differences in the severity of cystic fibrosis manifestations.

The most prevalent CFTR mutation responsible for cystic fibrosis is the ΔF508

mutation in the nucleotide binding domain, which is found in approximately 70% of defective CFTR alleles and in 90% of cystic fibrosis patients in the United States [Rowe et al., 2005]. While the ΔF508 retains chloride transport activity in membranes, it is recognized as misfolded and destroyed before it can reach the cellular membrane. Generally speaking, cystic fibrosis can be categorized into six different classes.

1. Class I: Absence of synthesis.
2. Class II: Defective protein maturation and premature degradation.
3. Class III: Disordered regulation.
4. Class IV: Defective ion conductance.
5. Class V: Reduced transcription due to promoter or splicing abnormality.
6. Class VI: Accelerated turnover from the cell membrane.

ΔF508 is the predominant Class II mutation, but other clinically relevant mutations such as G85E and G91R also result in misfolded and prematurely degraded proteins. Premature stop codons compose of the majority of Class I mutations. This early termination of transcription is prevalent in the Ashkenazi Jewish population. Other point mutations in the nucleotide binding and regulatory domains result in a properly transported protein, but one whose function is compromised. These mutations result in the Class IV and Class III mutations, respectively.

The Cystic Fibrosis Mutation Database, an online and manually curated database, serves as the standard repository for CFTR mutations. As of this analysis, it contains

1534 unique mutations, including missense, frameshift, splicing, nonsense insertions, and deletions. Because it is actively growing and curated, this database provides an excellent comparison between manually curated and automatically extracted point mutations.

5.4 Methods

5.4.1 External Database Data Retrieval

We retrieved CFTR point mutation information from the CFTR Mutation Database and the Swiss-Prot databases. The CFMD provided its mutation information, consisting of point mutations, insertions, and deletions in the form of an HTML document. This document was retrieved and parsed to identify the point mutation term and its type. The Swiss-Prot database was downloaded in XML format and the entry for the human CFTR protein identified (Accession # P13569). The CFTR Swiss-Prot sequence is annotated with a "VARIATION" flag at specific positions to denote point mutations, deletions, and insertions. All the point mutations denoted with a "VARIATION" tag were identified and saved.

5.4.2 Extraction of CFTR point mutations from PubMed literature

The search term "cystic fibrosis[mh] CFTR[mh] point mutation[mh]" was used to identify articles which discussed CFTR point mutations. Using the PubMed EUtils and the provided LinkOuts, we were able to retrieve 536 full-text PDF articles. These articles were converted to ASCII text and then processed by Mutation GraB to identify and extract human CFTR point mutations. The extracted point mutations were verified manually for

accuracy. For consistency purposes, we only used protein point mutations extracted by Mutation GraB for comparison against the CFMD and Swiss-Prot sources.

5.4.3 Comparison of Point Mutation Coverage between Data Sources

5.5 Results

We evaluated the utility of identifying and extracting point mutations from biomedical literature by using the MutationGraB application on a set of full-text cystic fibrosis articles. In this section, we show the common and different point mutations contained between the Mutation GraB, Swiss-Prot, and CFMD datasets. We took every protein point mutation from each dataset and identified the unique point mutations and the amino acid positions at which point mutations were found. For example, within one dataset, the S42F, and S42A point mutations are considered unique and different point mutations, but position 42 is only counted to contain a point mutation once. This measures the number of point mutations found, as well as the coverage of point mutation positions between the datasets.

Table 5.1 shows the number of protein point mutations from each source. MutationGraB was able to extract 609 unique protein point mutations at 401 distinct amino acid positions from the 536 full-text articles. The CFTR entry in the Swiss-Prot database contained 184 protein point mutations from 150 amino acid positions, and the CFMD contained 750 protein point mutations from 541 positions. Figure 5.1A shows the number of common and different point mutations found in the different datasets. For example, MutationGraB found 246 mutations that were not found in the CFMD, and the CFMD contained 387 mutations that were not recovered by MutationGraB.

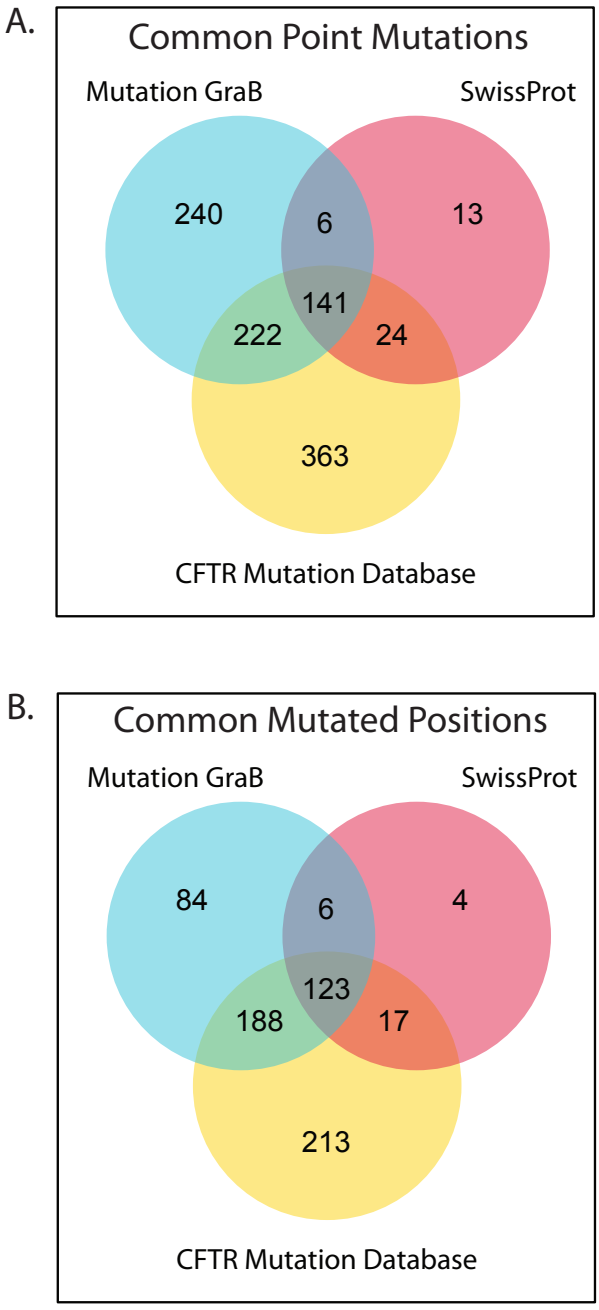


Figure 5.1: (A) A Venn diagram of the number of unique point mutations common to each dataset. (B) A Venn diagram of the amino acid positions to contain a point mutation from each dataset. The union of all positions to contain a point mutation between every dataset would represent 835 different positions.

Table 5.1: Number of Point Mutations by Source

	MutationGraB	**Swiss-Prot**	**CFMD**
# Unique Point Mutations	609	184	750
# Amino Acid Positions	401	150	541

5.6 Discussion

We have shown a comparision of the quantity and coverage of point mutations gathered by manual curation versus automatic extraction methods. Our goal was to gauge the utility of a text mining application with regards to its ability to discover information which was previously unknown. In this instance, we examined the number of point mutations extracted by Mutation GraB against those found in manually curated databases.

In order to measure the potential effect of text mined point mutations on the entire body of CFTR point mutations, we can compare the point mutations that were found by MutationGraB against those contained in manually curated databases. Of the 593 protein point mutations found by MutationGraB, 357 of them were already present in the CFMD, and 263 were not found in the manually curated database. Table 5.2 shows the recall for Mutation GraB and the CFMD. In this case, the recall represents the ability of each curation method to retrieve all existing mutations in the literature, given that the total number of unique point mutations between both datasets is 999. The CFMD, which contained more point mutations than Mutation GraB was able to extract, had the higher recall of 0.764 to 0.594. Even though MutationGraB identified 593 unique point mutations, 236 of them were not found in the CFMD. Therefore, the combining of automatic and manual curation methods to create one unified set of CFTR point mutations would generate 26% more

Table 5.2: Recall % of Mutation GraB and CFMD on all Literature Extractable CFTR Point Mutations

	MutationGraB	**CFMD**
Recall	0.594 (593/999)	0.764 (763/999)

unique point mutations than manual curation alone.

Sequence position coverage is also another factor in comparing the database curation methods, as many mutations at fewer positions is less helpful than many mutations spread over a large number of positions. If we examine the union of amino acid positions where point mutations were found, a total of 835 different positions were covered. Figure 5.1B shows the unique and overlapping positions where point mutations were found in each dataset. The CFMD contained the most unique point mutation positions with 213 while SwissProt had only 4 unique positions. Looking at the intersection between all three datasets, 123 amino acid positions were found to contain a point mutation in each of the three datasets.

Another consideration for using automated methods to aid in database curation is the man-hours saved in manually screening literature by hand. While search methods such as PubMed can help identify texts related to CFTR mutations, a human reader must still manually read each target article for the presence of point mutations. The curators of the CFMD do not state how many articles were processed to identify all their point mutations, but the number cannot be trivial. In comparison, the process time of Mutation GraB on the 536 full-text articles was on the time frame of 2-3 hours. This direct comparison is unfair because it doesn't factor the development time of MutationGraB, but it still reflects the enormous power that computational methods have in terms of processing speed.

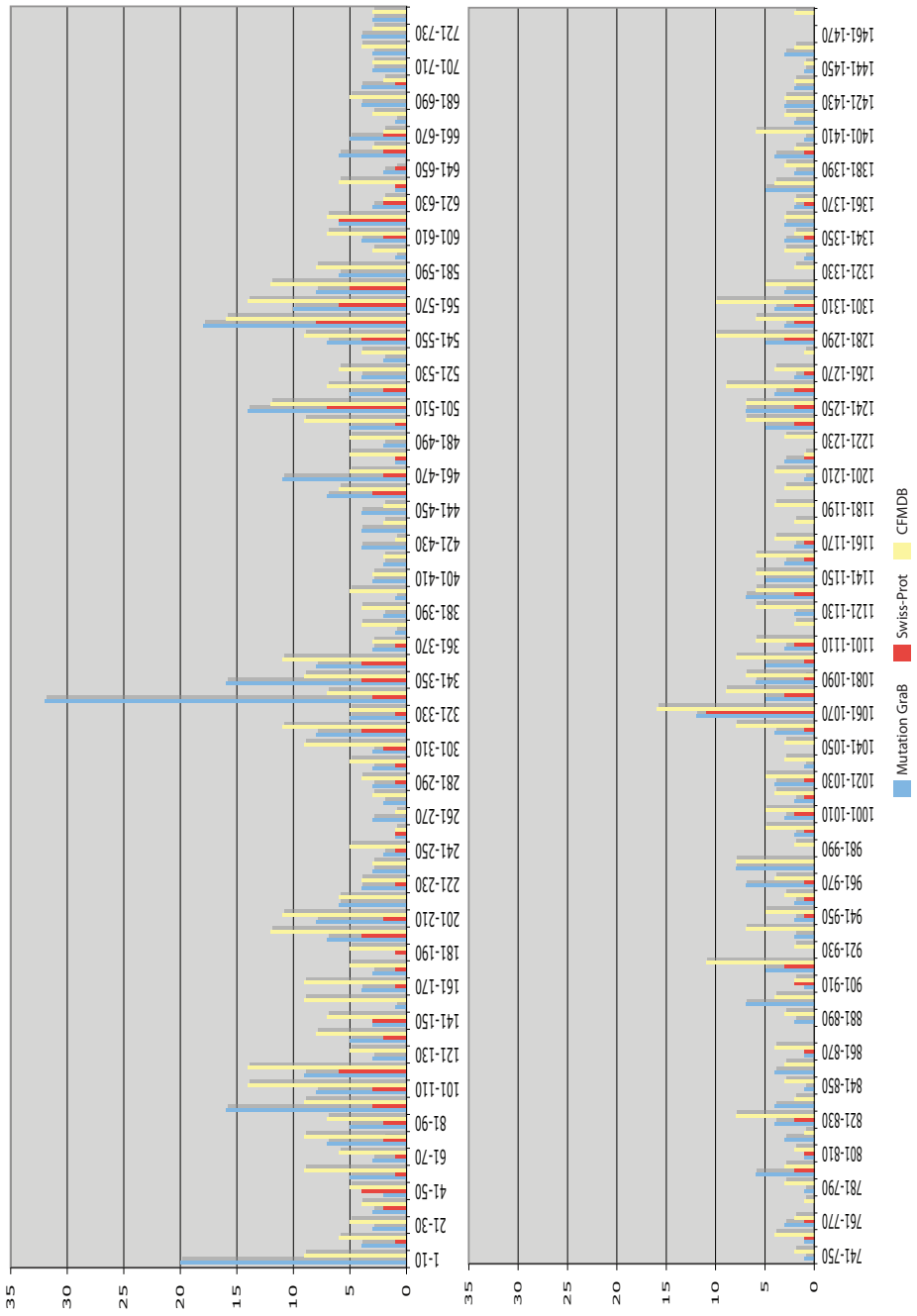


Figure 5.2: A histogram of the point mutations at each position of CFTR grouped into bins of 10-residue size. The blue bars represent the mutations found by Mutation GraB, red from Swiss-Prot, and yellow from CFMD.

5.7 Conclusion

In this study, we have compared the utility of using an automatic extraction method targeted at point mutations against the traditional method of manual point mutation curation. We have chosen a set of literature focused on a hereditary disease, as it relates to the most common applications of a point mutation search - elucidating the mechanisms of disease and protein function by mutational analysis. Using publicly available cystic fibrosis literature, we compared the mutations extracted by Mutation GraB to those of manually curated databases. We found that while the Cystic Fibrosis Mutation Database contained a larger and more thorough set of point mutations, the set of mutations extracted by Mutation GraB contained a large number not represented in the CFMD. We believe that a combined approach using automated extraction methods in concert with manual curation will yield the most efficient means for identifying and cataloging biological information from literature sources. As publicly available literature sources increase and the nature of biological research migrates towards quantitative approaches, the amount of capturable information in text will only increase. In order for this information to be available in an electronic format, a combination of text-mining and manual curation methods are required to efficiently and effectively extract the data to online databases.

Chapter 6

Conclusion

Within this dissertation, I have presented a body of work for the development of a text mining methodology for identifying and extracting point mutation information from biomedical literature. These methods and their applications represent a small, but significant part of the growth of biomedical text mining as a field in bioinformatics. Biomedical text mining represents an interface between biological, computational, and linguistics research in which a diverse set of hypotheses and goals are targeted. While other researchers often view biomedical text mining challenges in a purely computational or linguistic light, it is important to balance theoretical researach with practical applications to benefit to biology researchers.

With that in mind, we created our text mining methods and applications with the purpose of being computationally novel and noteworthy while having a practical use for end users. As described in Chapter 3, Mutation GraB uses a graph theoretical approach to associate terms in the text. This term association metric is used to find the correct protein and organism terms which correspond with a found point mutation. Graph methods are further used as dependency trees to identify sentences containing point mutations which also describe the functional effect of the mutation. In Chapter 4, we show how kernel methods can be combined with the graph structure of a sentence to predict the part of a sentence which contains the functional effect. To examine the practical application of Mutation GraB, we extracted the point mutations from a set of cystic fibrosis journal articles and compared the results with the contents of a manually curated database. The results, shown in Chapter 5, provide evidence that a combined approach of semi-automated text mining with manual curation can yield greater coverage of information than a single method alone.

As text mining can be generalized into methods which find the relationships between words or phrases in the text, graphical methods such as the ones presented in this dissertation should have a wide range of applications. Graph theory has a long standing history in math and computer science research, and its applications to biomedical research is inevitable. Text, having a regular structure but a diverse content, is ideal for graphical analysis. By naturally forming relationships between entities, graphical methods can be used to extract all manners of relationships, ranging from protein-protein to disease-drug interactions.

The importance of biomedical text mining to the scientific community will only increase as the open accessibility of research literature becomes more and more prevalent. We are confident that the ideas, methods, and applications described in this disseration will be of great value to downstream text mining research. Ultimately, we hope that the research presented in this dissertation will aid scientists in discovering the molecular mechanisms behind protein function and genetic diseases and allow for a greater interrogation into alleviating human diseases.

Bibliography

Gary D Bader, Doron Betel, and Christopher W V Hogue. Bind: the biomolecular interaction network database. *Nucleic Acids Res*, 31(1):248–250, 2003. ISSN 1362-4962 (Electronic).

C. J. O. Baker and R. Witte. Enriching protein structure visualizations with mutation annotations by text mining the protein engineering literature. In *IBM Center for Advanced Studies CASCON 2004 workshop reports*, 2004.

C. J. O. Baker and R. Witte. Mutation mining - a prospector's tale. *Information Systems Frontiers*, (8):47–57, 2006.

Jens Barthelmes, Christian Ebeling, Antje Chang, Ida Schomburg, and Dietmar Schomburg. Brenda, amenda and frenda: the enzyme information system in 2007. *Nucleic Acids Res*, 35(Database issue):D511–4, 2007. ISSN 1362-4962 (Electronic). doi: 10.1093/nar/gkl972.

D. A. Benson, I. Karsch-Mizrachi, D. J. Lipman, J. Ostell, and D. L. Wheeler. Genbank: update. *Nucleic Acids Res*, 32 Database issue:D23–6, 2004. 1362-4962 Journal Article.

C. Blaschke, M. A. Andrade, C. Ouzounis, and A. Valencia. Automatic extraction of

biological information from scientific text: protein-protein interactions. *Proc Int Conf Intell Syst Mol Biol*, pages 60–7, 1999. Journal Article.

C. Blaschke, E. A. Leon, M. Krallinger, and A. Valencia. Evaluation of biocreative assessment of task 2. *BMC Bioinformatics*, 6 Suppl 1:S16, 2005. Blaschke, Christian Leon, Eduardo Andres Krallinger, Martin Valencia, Alfonso Evaluation Studies Research Support, Non-U.S. Gov't England BMC bioinformatics BMC Bioinformatics. 2005;6 Suppl 1:S16. Epub 2005 May 24.

B. Boeckmann, A. Bairoch, R. Apweiler, M. C. Blatter, A. Estreicher, E. Gasteiger, M. J. Martin, K. Michoud, C. O'Donovan, I. Phan, S. Pilbout, and M. Schneider. The swiss-prot protein knowledgebase and its supplement trembl in 2003. *Nucleic Acids Res*, 31(1): 365–70, 2003. 1362-4962 Journal Article.

Emmanuel Boutet, Damien Lieberherr, Michael Tognolli, Michel Schneider, and Amos Bairoch. Uniprotkb/swiss-prot: The manually annotated section of the uniprot knowledgebase. *Methods Mol Biol*, 406:89–112, 2007. ISSN 1064-3745 (Print).

J. G. Caporaso, Jr. Baumgartner, W. A., D. A. Randolph, K. B. Cohen, and L. Hunter. Mutationfinder: a high-performance system for extracting point mutation mentions from text. *Bioinformatics*, 23(14):1862–5, 2007a. Caporaso, J Gregory Baumgartner, William A Jr Randolph, David A Cohen, K Bretonnel Hunter, Lawrence R01-NLM-009254/United States PHS Research Support, N.I.H., Extramural England Bioinformatics (Oxford, England) Bioinformatics. 2007 Jul 15;23(14):1862-5. Epub 2007 May 11.

J Gregory Caporaso, William A Baumgartner, David A Randolph, K Bretonnel Cohen,

and Lawrence Hunter. Rapid pattern development for concept recognition systems: application to point mutations. *J Bioinform Comput Biol*, 5(6):1233–1259, 2007b. ISSN 0219-7200 (Print).

Chih-Chung Chang and Chih-Jen Lin. Libsvm: a library for support vector machines, 2001. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

J. T. Chang, H. Schutze, and R. B. Altman. Gapscore: finding gene and protein names one word at a time. *Bioinformatics*, 20(2):216–25, 2004. 1367-4803 (Print) Evaluation Studies Journal Article Validation Studies.

H. W. Chun, Y. Tsuruoka, J. D. Kim, R. Shiba, N. Nagata, T. Hishiki, and J. Tsujii. Extraction of gene-disease relations from medline using domain dictionaries and machine learning. *Pac Symp Biocomput*, pages 4–15, 2006. Chun, Hong-Woo Tsuruoka, Yoshimasa Kim, Jin-Dong Shiba, Rie Nagata, Naoki Hishiki, Teruyoshi Tsujii, Jun'ichi Singapore Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing Pac Symp Biocomput. 2006;:4-15.

Michael Collins and Nigel Duffy. New ranking algorithms for parsing and tagging: kernels over discrete structures, and the voted perceptron, 2001. 1073128 263-270.

R. G. Cotton and O. Horaitis. The hugo mutation database initiative. human genome organization. *Pharmacogenomics J*, 2(1):16–9, 2002. 1470-269X (Print) Journal Article.

J. Crim, R. McDonald, and F. Pereira. Automatically annotating documents with normalized gene lists. *BMC Bioinformatics*, 6 Suppl 1:S13, 2005. 1471-2105 Journal Article.

Aron Culotta and Jeffrey Sorensen. Dependency tree kernels for relation extraction. In *ACL '04: Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 2004.

N. Daraselia, A. Yuryev, S. Egorov, S. Novichkova, A. Nikitin, and I. Mazo. Extracting human protein interactions from medline using a full-sentence parser. *Bioinformatics*, 20 (5):604–11, 2004. 1367-4803 Journal Article.

Gunes Erkan, Arzucan Ozgur, and Dragomir R Radev. Extracting interacting protein pairs and evidence sentences by using dependency parsing and machine learning techniques. In *Proceedings of the Second BioCreative Challenge Evaluation Workshop*, Madrid, Spain, 2007.

C. Friedman, P. Kra, H. Yu, M. Krauthammer, and A. Rzhetsky. Genies: a natural-language processing system for the extraction of molecular pathways from journal articles. *Bioinformatics*, 17 Suppl 1:S74–82, 2001. 1367-4803 Evaluation Studies Journal Article.

C. Friedman, T. Borlawsky, L. Shagina, H. R. Xing, and Y. A. Lussier. Bio-ontology and text: bridging the modeling gap. *Bioinformatics*, 22(19):2421–9, 2006. Friedman, Carol Borlawsky, Tara Shagina, Lyudmila Xing, H Rosie Lussier, Yves A 1K22 LM008308-01/LM/United States NLM 1U54CA121852-01A1/CA/United States NCI R01 LM07659/LM/United States NLM R01 LM08635/LM/United States NLM Research Support, N.I.H., Extramural England Bioinformatics (Oxford, England) Bioinformatics. 2006 Oct 1;22(19):2421-9. Epub 2006 Jul 26.

K. Fundel, D. Guttler, R. Zimmer, and J. Apostolakis. A simple approach for protein name

identification: prospects and limits. *BMC Bioinformatics*, 6 Suppl 1:S15, 2005. 1471-2105 (Electronic) Journal Article.

K. Fundel, R. Kuffner, and R. Zimmer. Relex–relation extraction using dependency parse trees. *Bioinformatics*, 23(3):365–71, 2007. Fundel, Katrin Kuffner, Robert Zimmer, Ralf Research Support, Non-U.S. Gov't England Bioinformatics (Oxford, England) Bioinformatics. 2007 Feb 1;23(3):365-71. Epub 2006 Dec 1.

A. Hamosh, A. F. Scott, J. Amberger, C. Bocchini, D. Valle, and V. A. McKusick. Online mendelian inheritance in man (omim), a knowledgebase of human genes and genetic disorders. *Nucleic Acids Res*, 30(1):52–5, 2002. 1362-4962 Journal Article.

D. Hanisch, K. Fundel, H. T. Mevissen, R. Zimmer, and J. Fluck. Prominer: rule-based protein and gene entity recognition. *BMC Bioinformatics*, 6 Suppl 1:S14, 2005. 1471-2105 Journal Article.

Mark Hepple. Independence and commitment: assumptions for rapid training and execution of rule-based pos taggers. In *ACL '00: Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, pages 278–277, Morristown, NJ, USA, 2000. Association for Computational Linguistics. doi: http://dx.doi.org/10.3115/1075218.1075254.

William Hersh, Ravi Teja Bhupatiraju, and Sarah Corley. Enhancing access to the bibliome: the trec genomics track. *Medinfo*, 11(Pt 2):773–777, 2004.

L. Hirschman, M. Colosimo, A. Morgan, and A. Yeh. Overview of biocreative task 1b: normalized gene lists. *BMC Bioinformatics*, 6 Suppl 1:S11, 2005a. 1471-2105 Journal Article.

L. Hirschman, A. Yeh, C. Blaschke, and A. Valencia. Overview of biocreative: critical assessment of information extraction for biology. *BMC Bioinformatics*, 6 Suppl 1:S1, 2005b. 1471-2105 (Electronic) Journal Article.

F. Horn, A. L. Lau, and F. E. Cohen. Automated extraction of mutation data from the literature: application of mutext to g protein-coupled receptors and nuclear hormone receptors. *Bioinformatics*, 20(4):557–68, 2004. 1367-4803 Evaluation Studies Journal Article Validation Studies.

Dan Jurafsky and James H. Martin. *Speech and language processing : an introduction to natural language processing, computational linguistics, and speech recognition*. Prentice Hall, Upper Saddle River, N.J., 2000. ISBN 0130950696.

S Katrenko and P Adriaans. Learning relations from biomedical corpora using dependency trees. *In KDECB (Knowledge Discovery and Emergent Complexity in BioInformatics) Lecture Notes in Bioinformatics*, 4366, 2006.

S. Kim, J. Yoon, and J. Yang. Kernel approaches for genic interaction extraction. *Bioinformatics*, 24(1):118–26, 2008. Kim, Seonho Yoon, Juntae Yang, Jihoon Research Support, Non-U.S. Gov't England Bioinformatics (Oxford, England) Bioinformatics. 2008 Jan 1;24(1):118-26. Epub 2007 Nov 14.

Asako Koike and Toshihisa Takagi. Gene/protein/family name recognition in biomedical literature. In *HLT-NAACL 2004 Workshop: Biolink, Linking Biological Literature, Ontologies and Database*, pages 9–16, 2004.

L. C. Lee, F. Horn, and F. E. Cohen. Automatic extraction of protein point mutations

using a graph bigram association. *PLoS Comput Biol*, 3(2):e16, 2007. Lee, Lawrence C Horn, Florence Cohen, Fred E GM039900-15/GM/United States NIGMS LM008883-01/LM/United States NLM Research Support, N.I.H., Extramural United States PLoS computational biology PLoS Comput Biol. 2007 Feb 2;3(2):e16.

Johann Lenffer, Frank W Nicholas, Kao Castle, Arjun Rao, Stefan Gregory, Michael Poidinger, Matthew D Mailman, and Shoba Ranganathan. Omia (online mendelian inheritance in animals): an enhanced platform and integration into the entrez search interface at ncbi. *Nucleic Acids Res*, 34(Database issue):D599–601, 2006. ISSN 1362-4962 (Electronic). doi: 10.1093/nar/gkj152.

D. Maglott, J. Ostell, K. D. Pruitt, and T. Tatusova. Entrez gene: gene-centered information at ncbi. *Nucleic Acids Res*, 33(Database issue):D54–8, 2005. 1362-4962 (Electronic) Journal Article.

Christopher D Manning and Hinrich Schütze. *Foundations of statistical natural language processing*. MIT Press, Cambridge, Mass., 1999. ISBN 0262133601.

E. M. Marcotte, I. Xenarios, and D. Eisenberg. Mining literature for protein-protein interactions. *Bioinformatics*, 17(4):359–63, 2001. 1367-4803 Journal Article.

V Matys, E Fricke, R Geffers, E Gossling, M Haubrock, R Hehl, K Hornischer, D Karas, A E Kel, O V Kel-Margoulis, D-U Kloos, S Land, B Lewicki-Potapov, H Michael, R Munch, I Reuter, S Rotert, H Saxel, M Scheer, S Thiele, and E Wingender. Transfac: transcriptional regulation, from patterns to profiles. *Nucleic Acids Res*, 31(1):374–378, 2003. ISSN 1362-4962 (Electronic).

H W Mewes, D Frishman, K F X Mayer, M Munsterkotter, O Noubibou, P Pagel, T Rattei, M Oesterheld, A Ruepp, and V Stumpflen. Mips: analysis and annotation of proteins from whole genomes in 2005. *Nucleic Acids Res*, 34(Database issue):D169–72, 2006. ISSN 1362-4962 (Electronic). doi: 10.1093/nar/gkj148.

Olivo Miotto, Tin Wee Tan, and Vladimir Brusic. Supporting the curation of biological databases with reusable text mining. *Genome Inform*, 16(2):32–44, 2005. ISSN 0919-9454 (Print).

T. Mitsumori, S. Fation, M. Murata, K. Doi, and H. Doi. Gene/protein name recognition based on support vector machine using dictionary as features. *BMC Bioinformatics*, 6 Suppl 1:S8, 2005. 1471-2105 (Electronic) Journal Article.

Alessandro Moschitti. Making tree kernels practical for natural language learning. In *International Conference on European Association for Computational Linguistics*, Trento, Italy, 2006.

S. Novichkova, S. Egorov, and N. Daraselia. Medscan, a natural language processing engine for medline abstracts. *Bioinformatics*, 19(13):1699–706, 2003. 1367-4803 Evaluation Studies Journal Article Validation Studies.

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

S. Pyysalo, F. Ginter, T. Pahikkala, J. Boberg, J. Jarvinen, and T. Salakoski. Evaluation of two dependency parsers on biomedical corpus targeted at protein-protein interactions. *Int J Med Inform*, 75(6):430–42, 2006. Pyysalo, Sampo Ginter, Filip Pahikkala, Tapio Boberg, Jorma Jarvinen, Jouni Salakoski, Tapio Comparative Study Evaluation Studies

Research Support, Non-U.S. Gov't Ireland International journal of medical informatics Int J Med Inform. 2006 Jun;75(6):430-42. Epub 2005 Aug 11.

D. Rebholz-Schuhmann, S. Marcel, S. Albert, R. Tolle, G. Casari, and H. Kirsch. Automatic extraction of mutations from medline and cross-validation with omim. *Nucleic Acids Res*, 32(1):135–42, 2004. 1362-4962 Journal Article.

J R Riordan, J M Rommens, B Kerem, N Alon, R Rozmahel, Z Grzelczak, J Zielenski, S Lok, N Plavsic, and J L Chou. Identification of the cystic fibrosis gene: cloning and characterization of complementary dna. *Science*, 245(4922):1066–1073, 1989. ISSN 0036-8075 (Print).

Steven M Rowe, Stacey Miller, and Eric J Sorscher. Cystic fibrosis. *N Engl J Med*, 352(19): 1992–2001, 2005. ISSN 1533-4406 (Electronic). doi: 10.1056/NEJMra043184.

J. Schafer and K. Strimmer. An empirical bayes approach to inferring large-scale gene association networks. *Bioinformatics*, 21(6):754–64, 2005. Schafer, Juliane Strimmer, Korbinian Evaluation Studies Research Support, Non-U.S. Gov't England Bioinformatics (Oxford, England) Bioinformatics. 2005 Mar;21(6):754-64. Epub 2004 Oct 12.

S. T. Sherry, M. H. Ward, M. Kholodov, J. Baker, L. Phan, E. M. Smigielski, and K. Sirotkin. dbsnp: the ncbi database of genetic variation. *Nucleic Acids Res*, 29(1): 308–11, 2001. 1362-4962 (Electronic) Journal Article.

L. Smith, T. Rindflesch, and W. J. Wilbur. Medpost: a part-of-speech tagger for biomedical text. *Bioinformatics*, 20(14):2320–1, 2004. Smith, L Rindflesch, T Wilbur, W J

Evaluation Studies England Bioinformatics (Oxford, England) Bioinformatics. 2004 Sep 22;20(14):2320-1. Epub 2004 Apr 8.

Padmini Srinivasan. Text mining: Generating hypotheses from medline. *Journal of the American Society for Information Science and Technology*, 55(5):396–413, 2004.

B. J. Stapley, L. A. Kelley, and M. J. Sternberg. Predicting the sub-cellular location of proteins from text using support vector machines. *Pac Symp Biocomput*, pages 374–85, 2002. Journal Article.

L. Tanabe and W. J. Wilbur. Tagging gene and protein names in biomedical text. *Bioinformatics*, 18(8):1124–32, 2002. 1367-4803 (Print) Journal Article.

L C Tsui. The spectrum of cystic fibrosis mutations. *Trends Genet*, 8(11):392–398, 1992. ISSN 0168-9525 (Print).

J. Watkinson, X. Wang, T. Zheng, and D. Anastassiou. Identification of gene interactions associated with disease from gene expression data using synergy networks. *BMC Syst Biol*, 2:10, 2008. Watkinson, John Wang, Xiaodong Zheng, Tian Anastassiou, Dimitris England BMC systems biology BMC Syst Biol. 2008 Jan 30;2:10.

R. Witte and C. J. O. Baker. Combining biological databases and text mining to support new bioinformatics applications. In A. Montoyo, editor, *10th International Workshop on Applications of Natural Language to Information Systems*, pages 310–321, 2005.

Ioannis Xenarios, Lukasz Salwinski, Xiaoqun Joyce Duan, Patrick Higney, Sul-Min Kim, and David Eisenberg. Dip, the database of interacting proteins: a research tool for studying

cellular networks of protein interactions. *Nucleic Acids Res*, 30(1):303–305, 2002. ISSN 1362-4962 (Electronic).

Yoshihiro Yamanishi, Francis Bach, and Jean-Philippe Vert. Glycan classification with tree kernels. *Bioinformatics*, 2007.

A. Yeh, A. Morgan, M. Colosimo, and L. Hirschman. Biocreative task 1a: gene mention finding evaluation. *BMC Bioinformatics*, 6 Suppl 1:S2, 2005. 1471-2105 (Electronic) Journal Article.

Alexander S Yeh, Lynette Hirschman, and Alexander A Morgan. Evaluation of text data mining for database curation: lessons learned from the kdd challenge cup. *Bioinformatics*, 19 Suppl 1:i331–9, 2003. ISSN 1367-4803 (Print).

D. Zelenko, C. Aone, and A. Richardella. Kernel methods for relation extraction. *Journal of Machine Learning Research*, 2003.

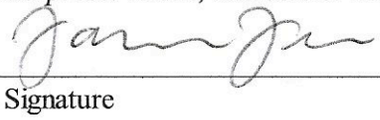
G. Zhou, J. Zhang, J. Su, D. Shen, and C. Tan. Recognizing names in biomedical texts: a machine learning approach. *Bioinformatics*, 20(7):1178–90, 2004. 1367-4803 (Print) Evaluation Studies Journal Article Validation Studies.

Publishing Agreement

It is the policy of the University to encourage the distribution of all theses and dissertations. Copies of all UCSF theses and dissertations will be routed to the library via the Graduate Division. The library will make all theses and dissertations accessible to the public and will preserve these to the best of their abilities, in perpetuity.

Please sign the following statement:

I hereby grant permission to the Graduate Division of the University of California, San Francisco to release copies of my thesis or dissertation to the Campus Library to provide access and preservation, in whole or in part, in perpetuity.



Author Signature

12/1/08

Date