Technische Universität München

Faculty of Informatics

Bachelor's Thesis in Bioinformatics

PubSeq: Amino Acid-based Search Engine for MEDLINE Abstracts

PubSeq: Aminosäuresequenz basierte Suchmachine für MEDLINE Abstrakten

Author:
Pandu Raharja

Supervisor:
Prof. Dr. Burkhard Rost

Advisors:
Dr. Guy Yachdav
Juan Miguel Cajuela

August 2015

Declaration of Authorship

I confirm that this bachelor's thesis is my own work and I have documented all sources and material used.

Signed: ______________________________

Date: ______________________________

"People usually think that progress consists in the increase of knowledge, in the improvement of life, but that isn't so. Progress consists only in the greater clarification of answers to the basic questions of life. The truth is always accessible to a man. It can't be otherwise, because a man's soul is a divine spark, the truth itself. It's only a matter of removing from this divine spark (the truth) everything that obscures it. Progress consists, not in the increase of truth, but in freeing it from its wrappings. The truth is obtained like gold, not by letting it grow bigger, but by washing off from it everything that isn't gold."

L. N. Tolstoy

Abstract

Background

In genetic research, it is imperative for biomedical researcher to stay updated on the current state of identified proteins. It was hard – and is getting harder, especially after widespread use of Next-Generation Sequencing (NGS) – for researcher to keep updated on the research into protein he/she is currently investigating. This is furthermore exacerbated by the fact that existing search engines only allow querying abstracts using protein names.

Methods

In this project, I present the first search engine that allows user to find all publications mentioning proteins that are similar or identical to the one he/she's interested in. To achieve this, I created a Solr Index that lists down all gene names that were mentioned in each of MEDLINE abstracts and titles. I then populated the index by scanning the whole MEDLINE corpus, tagging protein names found in title and abstract, normalizing those names into UniProt IDs and pushing the ID mentions onto Solr index. Given user's sequence query, the program runs a BLAST on the sequence and normalizes blast results to UniProt IDs. The program then retrieves articles mentioning this ID and return these to user. For the good usability I offer the whole service in a web interface available in following address.

Zusammenfassung

Hintergrund

In genetischer Forschung ist es erzwingend, dass der/die biomedische ForscherIn mit der aktuellen Landschaft von identifizierten Protein sich ständig informiert. Es war schwierig – und wird immer schwieriger sein, vor allem nach dem verbreiteten Ansatz von Next Generation Sequencing (NGS) Technoligien, um der/die Forscherin mit dem Protein von der Interesse in aktuellem Zustand zu halten. Die Tatsache, dass die aktuelle Suchmaschine von den Artikeln nur Namenbasierte Suche unterstützt, hilft leider nicht weiter.

Methoden

In diesem Projekt stellen wir eine Suchmaschine vor, die erlaubt den Benützer, basiert auf Aminosäuresequenz nach den Artikeln suchen, die das Protein oder die Ähnliche erwähnen. Um dies zu erreichen hatten wir einen Solr Index erstellt, der alle erwähnte Proteine innerhalb jedes MEDLINE Artikels auflistet. Wir füllen sich diesen Index in dem wir den gesamten MEDLINE Corpus durchscannen und alle Proteinname mithilfe eines NLP-Programs detektieren. Wir wurden dann diese Namen in UniProt IDs normalisieren. Diese normalisierte Namen wurden schlielich in unserem Solr Index hinzufügen. Um die Benutzbarkeit dieser Dienstleistung zu maximieren hatten wir auch eine Webschnittstelle entwurfen, die in folgender Addresse verfügbar ist.

Acknowledgements

First and mostly, I would like to thank Prof. Burkhard Rost for the holistic supports provided, be it through the lab infrastructures or himself personally. I would also to thank two of my advisors, Dr. Guy Yachdav and Juan Miguel Cajuela, who have in spite of their busy schedules and great distances (and time differences) patiently advised me through the project. Knowing that both are about to finish their PhD programs, I wish them all the best of luck in their future endeavors. I would also like to thank Andre Ofner for helping with the evaluation of the systems.

Also, I would like to thank Tatyana Goldberg for administrative support during my stay at the lab. Also my gratitude for Tim Karl, our awesome system administrator, who has helped us tremendously in incorporating each of the cogs in our pipeline into one coherent system. While not involved in our project personally, I would like to thank Prof. Lars Juhl Jensen of University of Copenhagen for giving us access to his tagger program.

I would also to thank Andre Ofner for the help in validating the systems. I would also like to express my gratitude towards Robert Leaman and Zhiyong Lu from National Institue of Health. While we ended up using different implementation of normalizer in our system, their contributions during our earlier attempts in the project are not to understate.

Research wouldn't happen without grants and patrons. Therefore I would like to thank grant organizations that have contributed financial supports to the lab and its extension. I am full aware that without sufficient infrastructure and human capital support endowed by several grants, this project would be impossible to kick start and finish.

Finally I would also thank all Rostlab members and its extensions, without whom this work would all but possible.

Contents

Declaration of Authorship ii

Abstract iv

Zusammenfassung v

Acknowledgements vi

Contents vii

List of Figures ix

List of Tables xi

Abbreviations xiii

1 Introduction 1
1.1 An Easier Biomedical Research 1
1.2 Overview of This Thesis 2

2 Background 5
2.1 FASTA and BLAST 5
2.2 Natural Language Processing 8
2.2.1 Named Entity Recognition 9
2.3 Previous Works 11

3 Organizations and Components 13
3.1 Main Section 1 13
3.1.1 Subsection 1 13
3.1.2 Subsection 2 14
3.2 Main Section 2 14

4 Program Pipeline 15

5 Results and Analysis 17

6 Maintenance and Updates 19

7 Conclusions and Outlook 21

A Appendix Title Here 23

Bibliography 25

List of Figures

2.1 Overview of DNorm pipeline 11

List of Tables

Abbreviations

BLAST	**B**asic **A**lignment **S**earch **T**ools
MEDLINE	**Med**ical **Li**terature Analysis Retrieval Onli**ne**
NER	**N**amed **E**ntity **R**ecognition
UniProt	**Uni**versal **Prot**ein Resources

Chapter 1

Introduction

1.1 An Easier Biomedical Research

We'll try to present the main idea of this project in following story. Imagine you in the position as a biomedical researcher, are currently investigating some unknown enzymes that somehow were over-expressed in a patient with medical conditions. Upon some more investigating, you managed to get the sequence of several proteins. Without prior knowledge of the proteins, you would naturally BLAST the sequences and wait a little while while the BLAST is searching the sequence against your local database or some online service. Upon the results were coming, you would naturally want to check the resulting proteins one by one, at least the best matching ones. For each protein, you would want to search for articles that have dealt with this protein before.

Imagine that, instead of having to go through blasting the sequence manually and searching for articles one by one, you could just put in a sequence in a website, wait for a while and get the site returns a list of articles that mention the proteins with similar or exact sequence to the one you have. Not only you would save time and resource during the parts that were handled by website itself, you as a researcher could focus more on the substantial part of the research – that is, finding as much essential information about the unknown protein in as little overhead as possible. Therefore, we created a web service that realizes this. In the service, user would only have to put in the sequence of unknown protein, press the search button and receive at the end a list of articles that mention proteins with identical or similar sequence to queried proteins.

With this small contribution, we hope not only to bridge the gap between sequence and knowledge discovery in biomedical research, but also give researcher more flexibility and insights in their literature research. With also ongoing feature extensions and updates, we would also hope that the service would serve more researchers with more conveniences both in medium and long run.

1.2 Overview of This Thesis

In this thesis, we will describe how we came with the idea of creating PubSeq, how we did that and what we planned in the future regarding our implementation.

In **Chapter 2**, we will discuss how bridging the knowledge gap has been attempted in the past and how our contribution would fit in the bigger picture. We also discuss some of the methods that are relevant in our project. Also, we would look into how our project builds upon existing knowledge and technology.

Chapter 3 introduces the system as a whole. How we organize the sub-components together. We would also delve deeper into technological side of the projects here, while keeping the reader aware of the bigger picture. We would discuss our rationale behind selecting some of technology stacks that we used. All the while,we would also show some the visual examples from our component here. By the end of the chapter it is hoped that the reader would understand how each single component interacts with others within our system.

Chapter 4 tries to look the program from the persepctive of end user. Having some abstractions hidden, we would show how convenient would that be for a researcher to use our application. We would also make our case for value proposition of PubSeq search engine in this chapter.

Chapter 5 covers quantitative measurement of the quality of our website. We would focus mostly on how our system performs, especially with regards to the sensitivity and specificity. We would focus mostly on the the quality of protein tagging within our data (see Chapter 5 for detail). We would also muse on how our system would have an edge over similar UniProt ID-based abstract search service provided by UniProt [1] [2].

Chapter 6 explains our update and maintenance design for the website. Here the reader would be aware on how we attempt to make our website up-to-date to the latest protein landscape. We would, again, delve into how this is realized within our system.

Chapter 7 covers our conclusion of the system so far. There we presented our own ideas on how the website could and would be improved. we would again reiterate the the merits of using the PubSeq as the search engine for abstracts based on protein sequence.

Chapter 2

Background

This chapter introduces the concepts and techniques that are relevant throughout this thesis. First, the concept of similarity search, especially the two software suite FASTA and BLAST would open our chapter. And then, we would introduce various contemporary concepts in bioinformatics and bioinformatics-related infrastructure such as UniProt and MEDLINE. Additionally, we would introduce the concept of named entity recognition (NER) within the field of Natural Language Processing and how it would be relevant for us. Finally we would see how our project relates to previous works in similar topics and how it would improve, provide alternative or give additional insight to them.

2.1 FASTA and BLAST

As the title of this thesis already conveyed, the main idea of this project is to bridge the accessibility and knowledge gap between sequence and the main source of knowledge and reference of previous discoveries – a vast corpora of publications in natural sciences – through a modern search engine. Given a sequence of amino acids, it would be impossible for a human to directly identify directly the protein, let alone the characteristics and the functions and the characteristics of the protein.

Several attempts on bridging one component of the gap, specifically between sequence and other known sequences, was done in eighties and earlier nineties. In 1981, Smith and Walterman published the algorithm computing complete local sequence alignment, which was further improved by Gotoh in 1982 [3] and Altschul (Altschul and Erickson, 1986 [4]). This was however deemed too slow, especially if used for the purpose of one-against-all search, which was heavily (and still is) used for sequence-based knowledge discovery in biomedical research.

In 1985, Lipman and Pearson published the first paper mentioning the DNA and protein sequence alignment program FASTA [5]. During the first publication, FASTA was designed and intended to search for similar protein sequences. It takes a sequence of amino acids and searches against entries within a corresponding database by using local sequence alignment to find similar sequences. In general, FASTA takes four steps in computing three scores that characterize sequence similarity [6]:

1. Finding identify regions with high density of sequence identities and pair identities between two sequences. FASTA achieved a fast computation in this step by using a look up table, a map that describes for each character where it appears within sequence. In conjunction with the lookup table, FASTA also uses the diagonal method to find all regions of similarity between the two sequences, counting matches and penalizing for intervening mismatches. This diagonal could be visually seen in two sequence alignment as series of matches ('dots') in match matrix between two sequences.

2. Rescanning of the 10 regions with highest sequence identities using PAM250 matrix. PAM250 matrix refers to assumed point accepted mutation (PAM) matrix after 250 mutations, which is basically the 250-th power of initial PAM matrix. The probability of each entry within PAM matrix was acquired from analysis of phylogenetic trees (Dayhoff, 1978 [7]).

3. Annealing of both ends of alignment and calculating similarity score is the sum of the joined initial regions minus a penalty (usually 20) for each gap [6].

4. Construction of optimal alignment using Needleman-Wunsch Algorthm [8] on the best matching region. The program would then return the similarity score of this alignment along with the best score from step 2 and 3.

In 1988, Pearson and Lipman improved the software by adding support and improvement, among others, for nucleic acid similarity search, translated nucleic acid search [9]. This allowed researchers to do trans-domain search between nucleic and amino acids.

Further down the road, in 1990, Altschul et al. published the Basic Alignment Research Too [10], better known in its acronym as BLAST. The algorithm, like FASTA, is based on heuristics search and is structured in similar manner to BLAST. BLAST takes a sequence to search for and a sequence or a set of sequences to search against. In modern usage, the set of sequences is provided by some database. The algorithm would then run in following main steps [11]:

1. Removal of low complexity regions or sequence repeats from query sequence. Low complexity refers to sequence with few elements.

2. Creation of k-gram sequences from query sequence.

3. For each word from step 2, listing of possible matching words and selection of high scoring words. Matching words are the all possible combinations of words with same length as the k-gram word. For each possible word a score is calculated, which is based on substitution matrix. The best scoring words are then passed onto next step. This differs from FASTA, which focuses more on common words in database.

4. Organization of remaining high scoring words into efficient search three. Both step 3 and 4 would be repeated for each word from step 2.

5. Scanning of database for exact matches with remaining high-scoring words.

6. Extension of database match to high-scoring-segment pair (HSP). This is done by annealing both ends of match until the matching score begins to decrease.

7. Listing of all HSPs that are significant enough.

8. Evaluation of statistical significance of the HSPs. BLAST models statistical significance using Gumbel extreme value distribution [12], in which the probability of observing score S higher than equal to x is defined as

   $$P(S \leq x) = 1 - exp(-e^{-\lambda(x-\mu)})$$

   with

   $$\mu = log(Km'n')/t$$

   The parameters μ and K are fitted from the distribution of results from high scoring pairs. m' and n' are effective length of the query and database sequences.

9. Make two or more HSP regions into one alignment. In a given hit sequence from database, the algorithm would attempt merging the regions into one had the score of combined region is larger than individual score.

10. Computation of sequence alignments using Smith-Walterman Algorithm [13].

Nowadays, both FASTA and BLAST were distributed not only locally but also online by various providers such as National Center of Biotechnology Information (NCBI) [1] and European Bioinformatics Institute (EBI) [2] [3].

2.2 Natural Language Processing

The rapid development in sequence similarity search coupled with explosion of genome-wide sequencing, which was even more augmented by the advent of post-Sanger and – recently – New Generation Sequencing (NGS), means that the problem of identifying sequence is more or less explained. There is however one part that is missing from our picture: how to get the information on how the sequence was mentioned in previous publications?

Come Natural Language Processing (NLP). Natural Language Processing is a interdisciplinary field that deals with the interaction between computer and human languages (hence the natural language). The aspects of natural languages such as named entities recognition (NER) [14], morphological segmentation [15], speech recognition and analysis [16] fall into the auspice of natural language processing. The methods used in natural language processing is mostly statistical-based [17] and some of the methods have been known to be used in other fields such as Conditional Random Fields [18] and its special case Hidden Markov Chain, which is one of the more commonly used methods in bioinformatics (e.g. Salzberg, et al. [19] and Burge, et al. [20]).

In this thesis we would make use of one aspect of natural language processing: named entity recognition (NER).

[1] `http://blast.ncbi.nlm.nih.gov/Blast.cgi`

[2] `http://www.ebi.ac.uk/Tools/sss/wublast/`

[3] `http://www.ebi.ac.uk/Tools/sss/fasta/`

2.2.1 Named Entity Recognition

The term named entity recognition was first coined at the Sixth Message Understanding Conference (MUC-6) in 1996 [14] [21]. The problem statement of named entity recognition roughly goes as follow:

> *Given an input text, locate and classify elements of text according to defined categories of entity (a Named Entity)*

The pre-defined set of categories range from unique identifier such as name and location to expression of times and numeric values such as date and percent expression [14]. To take an example, consider the first paragraph of following recent article from the Wall Street Journal [4]:

> Uber Valued at More Than $50 Billion
>
> **Ride-sharing app, which just closed a funding round, reaches mark faster than Facebook**
>
> Uber Technologies Inc. has completed a new round of funding that values the five-year-old ride-hailing company at close to $51 billion, according to people familiar with the matter, equaling Facebook Inc.'s record for a private, venture-backed startup.

A named entity recognition program trained for company names and monetary values would identify and tag following annotations from the text:

> **Uber** `(company)` Valued at More Than **$50 Billion** `(monetary_value)`
>
> **Ride-sharing app, which just closed a funding round, reaches mark faster than Facebook** `(company)`
>
> **Uber Technologies Inc.** has completed a new round of funding that values the five-year-old ride-hailing company at close to **$51 billion** `(monetary_value)`, according to people familiar with the matter, equaling **Facebook Inc.** `(company)`'s record for a private, venture-backed startup.

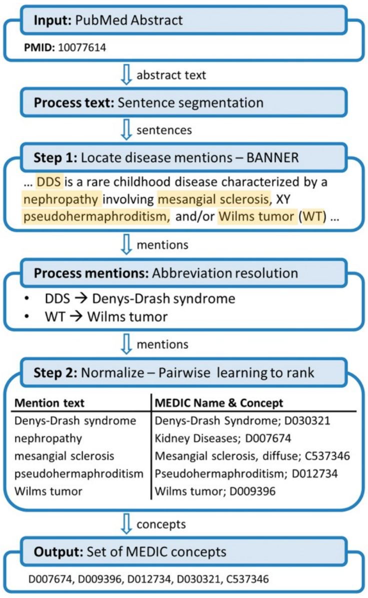
[4] http://www.wsj.com/articles/uber-valued-at-more-than-50-billion-1438367457

Here we see how various formats of company names could be tagged in this hypothetical case. Indeed, a good named-entity recognition tagger should be able to perform exactly this kind of task.

There are various ways of implementing Named Entity Recognition (NER), all ranging from supervised, semi-supervised to unsupervised learning [14]. Nowadays, most NER relies on supervised learning methods [14] such as Hidden Markov Model (HMM) [22], Decission Trees [23], Maximum Entropy Model [24], Support Vector Machines (SVM) [25] and Conditional Random Fields (CRF)[26], which is a discriminative generalization of HMM[27].

In this project, we attempted on two distinct NER implementations: **DNorm**in the earlier phase and non-publicly available NER tagger given to us by Lars Juhl Jensen at University of Copenhagen (which would for our convenience would be called **Lars Tagger** or **PubSeq NER Tagger** from now on) later on. DNorm (Leaman and Lu, 2013 [28]) is a NER designed to tag and normalize disease names within PubMed abstract. Given a PubMed abstract text, it tags the disease entities within text and normalize said entities into standardized form (in this case, MEDIC concepts [29]). The tagging component of DNorm is based on BANNER, which in turn is a CRF-based NER tagger [30]. To normalize the entities, the program utilizes pairwise approach of Learning to Rank [31] (see, for example, Joachims 2002 [32], which is used to rank the results of web search). The idea of ranking as a method of normalization could indeed be explained analogously in light of search engine itself, given a tagged string s, a set of standardized entity names N and a scoring function $M(s, n) \in R$, we would like to find out to which standardized name this string maps best to.

The reason why we initially resorted to DNorm is because of its good performance with regard to precision, recall and F-measure [28]. Also, the underlying tagging mechanism BANNER appeared to perform well in BioCreative benchmarks for tagging various biological entities, most notably gene names [33] [30]. Trying to extend DNorm to support gene names recognition *and normalization*, however, proved to be very difficult. We figured out that there was no publicly available data set that we could use to train our model for gene normalization. Secondly, since the DNorm uses matrix to represents both the normalization and similarity model [28], creating matrix models for the whole set of proteins would have the program create a matrix with millions of row – which wouldn't scale in normal cluster environment. Also, the fact that DNorm was optimized for disease names means that there is no guarantee that the resulting


Input: PubMed Abstract
PMID: 10077614
abstract text
Process text: Sentence segmentation
sentences
Step 1: Locate disease mentions – BANNER
… DDS is a rare childhood disease characterized by a nephropathy involving mesangial sclerosis, XY pseudohermaphroditism, and/or Wilms tumor (WT) …
mentions
Process mentions: Abbreviation resolution
• DDS → Denys-Drash syndrome
• WT → Wilms tumor
mentions
Step 2: Normalize – Pairwise learning to rank
concepts
Output: Set of MEDIC concepts
D007674, D009396, D012734, D030321, C537346

Mention text	MEDIC Name & Concept
Denys-Drash syndrome	Denys-Drash Syndrome; D030321
nephropathy	Kidney Diseases; D007674
mesangial sclerosis	Mesangial sclerosis, diffuse; C537346
pseudohermaphroditism	Pseudohermaphroditism; D012734
Wilms tumor	Wilms tumor; D009396

FIGURE 2.1: Overview of DNorm pipeline. The figure was taken from Leaman, Dogan and Lu, 2013 [28].

benchmark for protein names would be as good as disease tagging and normalization benchmark.

Later on, we switched to Lars Tagger. Unlike NER methods that are already mentioned in this chapter, our current NER Tagger relies on mapping between normalized entity names and their corresponding alternative names to annotate and normalize named entities within in the text. It checks whether a part of text closely resembles one or more elements in the mapping. It would then calculate the probability of the part of text being the instance of one of the entities (see more on Chapter 3). While there is no publication specifically dedicated for gene annotations, the underlying engine was used to tag and annotate taxonomic names in PubMed text (Pacifilis, et al., 2013 [34]). Moreover, we used part of STRING database [35] as the dictionary between normalized entity names and its variation of names that would be used to identify named entities in text and at the same time normalize them.

2.3 Previous Works

There are several publications and tools that attempt on similar goals as we do that have not been reviewed in previous section. While we are not aware of other project that realized end-to-end article search based on sequence, there are several (very successful) attempts on doing parts of it.

In the realm of **gene/protein names tagging and/or normalization**, there are various programs that are known to tag and normalize well such as GNAT (Hackenberg, et al. 2011 [36]), ABNER (Burr, 2015 [37]) and LINNEAUS (Gerner, et al. 2010 [38]). The latter was originally designed to detect and annotate taxonomy names within text.

For searching for protein mentions within PubMed corpus we are for example aware of. Given a UniProtKB-ID, a user can search for the list of articles mentioning the protein. User could, for example, search articles that mention human tumor protein P53 (`P53_HUMAN`). We are unfortunately not aware how it was done (manually annotated or done using NLP methods) or whether it covers only parts of article that are visible within PubMed corpus or the whole article.

Chapter 3

Organizations and Components

We discuss various components of the program in this chapter

3.1 Main Section 1

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aliquam ultricies lacinia euismod. Nam tempus risus in dolor rhoncus in interdum enim tincidunt. Donec vel nunc neque. In condimentum ullamcorper quam non consequat. Fusce sagittis tempor feugiat. Fusce magna erat, molestie eu convallis ut, tempus sed arcu. Quisque molestie, ante a tincidunt ullamcorper, sapien enim dignissim lacus, in semper nibh erat lobortis purus. Integer dapibus ligula ac risus convallis pellentesque.

3.1.1 Subsection 1

Nunc posuere quam at lectus tristique eu ultrices augue venenatis. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Aliquam erat volutpat. Vivamus sodales tortor eget quam adipiscing in vulputate ante ullamcorper. Sed eros ante, lacinia et sollicitudin et, aliquam sit amet augue. In hac habitasse platea dictumst.

3.1.2 Subsection 2

Morbi rutrum odio eget arcu adipiscing sodales. Aenean et purus a est pulvinar pellentesque. Cras in elit neque, quis varius elit. Phasellus fringilla, nibh eu tempus venenatis, dolor elit posuere quam, quis adipiscing urna leo nec orci. Sed nec nulla auctor odio aliquet consequat. Ut nec nulla in ante ullamcorper aliquam at sed dolor. Phasellus fermentum magna in augue gravida cursus. Cras sed pretium lorem. Pellentesque eget ornare odio. Proin accumsan, massa viverra cursus pharetra, ipsum nisi lobortis velit, a malesuada dolor lorem eu neque.

3.2 Main Section 2

Chapter 4

Program Pipeline

Chapter 5

Results and Analysis

Chapter 6

Maintenance and Updates

Chapter 7

Conclusions and Outlook

Appendix A

Appendix Title Here

Write your Appendix content here.

Bibliography

[1] UniProt Consortium et al. Ongoing and future developments at the universal protein resource. *Nucleic acids research*, 39(suppl 1):D214–D219, 2011.

[2] The UniProt Consortium. Uniprot citation mapping, 2015. URL http://www.uniprot.org/citationmapping/.

[3] Osamu Gotoh. An improved algorithm for matching biological sequences. *Journal of molecular biology*, 162(3):705–708, 1982.

[4] Stephen F Altschul and Bruce W Erickson. Optimal sequence alignment using affine gap costs. *Bulletin of mathematical biology*, 48(5-6):603–616, 1986.

[5] David J Lipman and William R Pearson. Rapid and sensitive protein similarity searches. *Science*, 227(4693):1435–1441, 1985.

[6] William R Pearson. Rapid and sensitive sequence comparison with fastp and fasta. *Methods in enzymology*, 183:63–98, 1990.

[7] Margaret O Dayhoff and Robert M Schwartz. A model of evolutionary change in proteins. In *In Atlas of protein sequence and structure*. Citeseer, 1978.

[8] Saul B Needleman and Christian D Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology*, 48(3):443–453, 1970.

[9] William R Pearson and David J Lipman. Improved tools for biological sequence comparison. *Proceedings of the National Academy of Sciences*, 85(8):2444–2448, 1988.

[10] Stephen F Altschul, Warren Gish, Webb Miller, Eugene W Myers, and David J Lipman. Basic local alignment search tool. *Journal of molecular biology*, 215(3):403–410, 1990.

[11] David W Mount and David W Mount. *Bioinformatics: sequence and genome analysis*, volume 2. Cold spring harbor laboratory press New York:, 2001.

[12] Emil Julius Gumbel and Julius Lieblein. *Statistical theory of extreme values and some practical applications: a series of lectures*, volume 33. US Government Printing Office Washington, 1954.

[13] Temple F Smith and Michael S Waterman. Identification of common molecular subsequences. *Journal of molecular biology*, 147(1):195–197, 1981.

[14] David Nadeau and Satoshi Sekine. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26, 2007.

[15] Fernand Meyer and Serge Beucher. Morphological segmentation. *Journal of visual communication and image representation*, 1(1):21–46, 1990.

[16] Lawrence Rabiner and Biing-Hwang Juang. Fundamentals of speech recognition. 1993.

[17] Christopher D Manning and Hinrich Schütze. *Foundations of statistical natural language processing*. MIT press, 1999.

[18] Charles Sutton and Andrew McCallum. An introduction to conditional random fields for relational learning. *Introduction to statistical relational learning*, pages 93–128, 2006.

[19] Steven L Salzberg, Arthur L Delcher, Simon Kasif, and Owen White. Microbial gene identification using interpolated markov models. *Nucleic acids research*, 26 (2):544–548, 1998.

[20] ChristopherB Burge. Modeling dependencies in pre-mrna splicing signals. *New Comprehensive Biochemistry*, 32:129–164, 1998.

[21] Ralph Grishman and Beth Sundheim. Message understanding conference-6: A brief history. In *COLING*, volume 96, pages 466–471, 1996.

[22] Daniel M Bikel, Scott Miller, Richard Schwartz, and Ralph Weischedel. Nymble: a high-performance learning name-finder. In *Proceedings of the fifth conference on Applied natural language processing*, pages 194–201. Association for Computational Linguistics, 1997.

[23] Satoshi Sekine, Ralph Grishman, and Hiroyuki Shinnou. A decision tree method for finding and classifying names in japanese texts. In *Proceedings of the Sixth Workshop on Very Large Corpora*, 1998.

[24] Andrew Borthwick, John Sterling, Eugene Agichtein, and Ralph Grishman. Exploiting diverse knowledge sources via maximum entropy in named entity recognition. In *Proc. of the Sixth Workshop on Very Large Corpora*, volume 182, 1998.

[25] Masayuki Asahara and Yuji Matsumoto. Japanese named entity extraction with redundant morphological analysis. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 8–15. Association for Computational Linguistics, 2003.

[26] Andrew McCallum and Wei Li. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*, pages 188–191. Association for Computational Linguistics, 2003.

[27] John Lafferty, Andrew McCallum, and Fernando CN Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. 2001.

[28] Robert Leaman, Rezarta Islamaj Doğan, and Zhiyong Lu. Dnorm: disease name normalization with pairwise learning to rank. *Bioinformatics*, page btt474, 2013.

[29] Allan Peter Davis, Thomas C Wiegers, Michael C Rosenstein, and Carolyn J Mattingly. Medic: a practical disease vocabulary used at the comparative toxicogenomics database. *Database*, 2012:bar065, 2012.

[30] Robert Leaman, Graciela Gonzalez, et al. Banner: an executable survey of advances in biomedical named entity recognition. In *Pacific Symposium on Biocomputing*, volume 13, pages 652–663. World Scientific, 2008.

[31] Tie-Yan Liu. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*, 3(3):225–331, 2009.

[32] Thorsten Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 133–142. ACM, 2002.

[33] Larry Smith, Lorraine K Tanabe, Rie J Ando, Cheng-Ju Kuo, I-Fang Chung, Chun-Nan Hsu, Yu-Shi Lin, Roman Klinger, Christoph M Friedrich, Kuzman Ganchev, et al. Overview of biocreative ii gene mention recognition. *Genome biology*, 9(Suppl 2):S2, 2008.

[34] Evangelos Pafilis, Sune P Frankild, Lucia Fanini, Sarah Faulwetter, Christina Pavloudi, Aikaterini Vasileiadou, Christos Arvanitidis, and Lars Juhl Jensen. The species and organisms resources for fast and accurate identification of taxonomic names in text. *PloS one*, 8(6):e65390, 2013.

[35] Damian Szklarczyk, Andrea Franceschini, Michael Kuhn, Milan Simonovic, Alexander Roth, Pablo Minguez, Tobias Doerks, Manuel Stark, Jean Muller, Peer Bork, et al. The string database in 2011: functional interaction networks of proteins, globally integrated and scored. *Nucleic acids research*, 39(suppl 1): D561–D568, 2011.

[36] Jörg Hakenberg, Martin Gerner, Maximilian Haeussler, Illés Solt, Conrad Plake, Michael Schroeder, Graciela Gonzalez, Goran Nenadic, and Casey M Bergman. The gnat library for local and remote gene mention normalization. *Bioinformatics*, 27(19):2769–2771, 2011.

[37] Burr Settles. Abner: an open source tool for automatically tagging genes, proteins and other entity names in text. *Bioinformatics*, 21(14):3191–3192, 2005.

[38] Martin Gerner, Goran Nenadic, and Casey M Bergman. Linnaeus: a species name identification system for biomedical literature. *BMC bioinformatics*, 11(1): 85, 2010.