
Conditional Random Fields for Named Entity Recognition

Feature Selection and Optimization in
Biology and Chemistry

Dissertation

zur Erlangung des Grades eines

Doktors der Naturwissenschaften

der Technischen Universität Dortmund
an der Fakultät für Informatik
von

Roman Klinger

Dortmund
2011

Conditional Random Fields for Named Entity Recognition

Feature Selection and Optimization in
Biology and Chemistry

Dissertation

zur Erlangung des Grades eines

Doktors der Naturwissenschaften

der Technischen Universität Dortmund

an der Fakultät für Informatik

von

Roman Klinger

Dortmund

2011

Tag der mündlichen Prüfung:

Dekanin: Prof. Dr. rer. nat. Gabriele Kern-Isberner

Gutachter:

Prof. Dr. rer. nat. Günter Rudolph

Prof. Dr. rer. nat. Martin Hofmann-Apitius

Contents

List of Figures	vii
List of Tables	xi
1 Introduction	1
1.1 Text Mining	1
1.2 Challenges in Processing Text	2
1.3 Named Entity Recognition	4
1.3.1 Dictionary-based	4
1.3.2 Rule-based	5
1.3.3 Machine Learning-based	5
1.3.4 Applications in the Domain of Biology, Medicine, and Chemistry	5
1.4 Motivation and Structure of this Thesis	7
I Fundamentals	11
2 Workflow of Named Entity Recognition	13
2.1 Introduction	13
2.2 Data Formats, Selection, and Preprocessing	13
2.3 Annotation	16
2.4 Sentence Splitting and Tokenization	18
2.5 Token and Character Normalization	19
2.6 Feature Extraction	20
2.7 Postprocessing and Entity Normalization	23
2.8 Evaluation	24
2.9 Visualization and Presentation	26
3 Graphical Models and Conditional Random Fields	31
3.1 Introduction and Previous Work	31
3.2 Probabilistic Models	32
3.2.1 Naïve Bayes	33
3.2.2 Hidden Markov Models	34
3.2.3 Maximum Entropy Model	35
3.3 Graphical Representation	39
3.3.1 Directed Graphical Models	40
3.3.2 Undirected Graphical Models	41

3.4	Conditional Random Fields	43
3.4.1	Basic Principles	43
3.4.2	Linear-chain CRFs	44
3.4.3	Arbitrarily Structured CRFs	53
3.5	Summary	56
II	Case Studies and Adaptions to the Biological and Chemical Domain	57
4	Recognition of IUPAC and IUPAC-like Chemical Names	59
4.1	Introduction	59
4.2	Dictionary-based Chemical Name Recognition	61
4.3	Named Entity Recognition Workflow	63
4.3.1	Entity Types	63
4.3.2	Corpus Selection and Annotation	64
4.3.3	Tokenization	65
4.3.4	Feature Extraction	66
4.3.5	Normalization of IUPAC names	66
4.4	Results	67
4.4.1	Parameter Selection	68
4.4.2	Evaluation of Named Entity Recognition	70
4.4.3	Annotation of full MEDLINE and Normalization	72
4.4.4	Extending the Model to other Chemical Nomenclatures	74
4.5	Summary and Discussion	74
5	Recognition of Gene and Protein Names	77
5.1	Introduction	77
5.2	Named Entity Recognition Workflow	78
5.2.1	Entity Types, Corpus Selection and Annotation	78
5.2.2	Tokenization and Feature Extraction	79
5.2.3	Postprocessing	79
5.3	Multi-Model approach	80
5.4	Results	81
5.4.1	Multi Model Combination	81
5.4.2	Parameter Selection	82
5.4.3	Evaluation of Named Entity Recognition	83
5.4.4	Generalizability and Stability over Time	85
5.5	Summary and Discussion	86
6	Recognition of Mentions of Single Nucleotide Polymorphisms	89
6.1	Introduction	89
6.2	Named Entity Recognition Workflow	91

6.2.1	Entity Types	91
6.2.2	Corpus Selection and Annotation	93
6.2.3	Tokenization and Feature Extraction	93
6.2.4	Normalization and Postprocessing	94
6.2.5	Dictionary-based and Rule-based Named Entity Recognition and Normalization	95
6.3	Results	96
6.3.1	Parameter Selection	97
6.3.2	Evaluation of Named Entity Recognition	98
6.3.3	Evaluation on Independent Test Set	98
6.4	Summary and Discussion	100
III	Further Enhancements	103
7	Feature Subset Selection	105
7.1	Introduction	105
7.2	Feature Selection Methods for CRFs	107
7.2.1	Filter	107
7.2.2	Iterative Feature Pruning	110
7.3	Results	111
7.3.1	Experimental Setting	113
7.3.2	Cross-Validation on the Training Sets	113
7.3.3	Results on independent test sets	115
7.3.4	Discussion	117
7.4	Conclusion and Future Work	120
8	User's Choice of Precision and Recall	121
8.1	Introduction	121
8.2	Methods	122
8.2.1	Approximation of Evaluation Measures	122
8.2.2	Non-dominated Sorting Genetic Algorithm II	123
8.2.3	Multi-Objective Optimization of CRF	123
8.3	Experiments	126
8.3.1	Experimental Setting	126
8.3.2	Comparison of Smoothed and Non-Smoothed Objective Function	126
8.3.3	Results	128
8.4	Conclusion and Future Work	131
9	Incorporating Distant Information via Automatically Selected Skip Edges	133
9.1	Introduction	133
9.1.1	Previous Work	133
9.1.2	Problem Description	135

9.2	Methods	137
9.2.1	Isolated filtering	137
9.2.2	Filtering via Markov Blanket based Graph Reduction	138
9.2.3	Best First Search on Templates	138
9.3	Results	140
9.3.1	Analysis of the Filtering Steps	140
9.3.2	Analysis of all Proposed Templates	143
9.4	Conclusion and Future Work	145
IV	Recapitulation	147
10	Summary and Conclusions	149
11	Future Work	151
A	Acknowledgements	153
B	Publications integrated in this Thesis	155
	Bibliography	157
	Index	177

List of Figures

1.1	Number of abstracts per year in Medline.	2
1.2	Number of articles per year in the PubMed Central Open Access data.	2
1.3	Example of a text with specified named entities of the class <i>virus</i>	4
1.4	Number of articles per year in Medline.	7
2.1	Workflow of named entity recognition.	14
2.2	Screenshot of the annotation tool WordFreak.	16
2.3	Screenshot of the annotation tool Knowtator in Protégé.	17
2.4	Example of a text with multiple named entity annotations of the class <i>virus</i>	18
2.5	Reflect web service augmenting Wikipedia website.	27
2.6	Example of a PDF document augmented with information about detected entities.	28
2.7	Screenshot of the search engine SCAIView.	29
3.1	Overview of probabilistic models.	33
3.2	Directed graphical model.	41
3.3	Naïve Bayes classifier.	41
3.4	Independency and factor graph for the hidden Markov model.	42
3.5	Maximum entropy classifier.	43
3.6	Linear chain conditional random field.	45
3.7	Alternative interpretation of a linear-chain CRF.	46
3.8	Example for a stochastic finite state automaton.	47
3.9	Message passing in the forward-backward algorithm.	52
3.10	Examples for structures of conditional random fields.	54
3.11	Example for an unrolled skip-chain CRF.	55
4.1	Distribution of classes in test corpus <i>Chemistry-Corpus</i> for dictionary based analysis.	62
4.2	Dictionary-based recall on <i>Chemistry-Corpus</i> (exact match).	63
4.3	Example abstract with tagged entities.	64
4.4	Results on the training data <i>IUPAC-Train-M</i> with 30-fold bootstrapping with different feature sets, different orders of the CRF, and offset conjunctions.	68
4.5	Results on the training data <i>IUPAC-Train-M</i> with some small feature sets on different orders of the CRF and offset conjunctions.	69
4.6	Results on the sampled MEDLINE corpus <i>IUPAC-Test-M</i>	71
4.7	Results of a CRF trained on different combinations of classes of chemical names.	75
5.1	Results for combination methods of the multi model output on the BioCreative II named entity recognition training data set, evaluated via 50-fold bootstrapping.	81

5.2	Differences in tokenization strategies for the BioCreative II named entity recognition training data set, evaluated via 50-fold bootstrapping.	82
5.3	Results for different features sets for the BioCreative II named entity recognition training data set, evaluated via 50-fold bootstrapping.	83
5.4	Results for combination methods of the multi model output on the BioCreative II named entity recognition test data set.	84
5.5	Named entity recognition models for gene and protein recognition trained on yearly sub-samples applied to independent sub-samples from different years. . .	86
5.6	Named entity recognition models for gene and protein recognition trained on yearly sub-samples applied to independent sub-samples only containing new entities from different years.	87
6.1	Example abstract with tagged entities.	92
6.2	Example for observation and label sequence for text snippet after tokenization. .	94
6.3	Results for different feature sets for the SNP seed set, evaluated via 50-fold bootstrapping.	97
7.1	Number of disjoint features per class in CRF models.	106
7.2	Distribution of weights in CRF models.	106
7.3	Depiction of instance building for feature filtering.	108
7.4	Iterative Feature Pruning Algorithm.	111
7.5	Visualization of Iterative Feature Pruning for CRF trained on CoNLL data.	112
7.6	Comparison of the average F_1 measure using 10-fold cross-validation.	114
7.7	Results on independent test sets.	115
7.8	Feature distribution per class in CRF models with feature selection.	117
7.9	Distribution of weights in CRF models with feature selection.	118
8.1	Workflow of an evolutionary algorithm.	124
8.2	Comparison of different parameters: Step size σ with and without smoothing. .	125
8.3	Results of the final population for $\sigma = 0.01$ without smoothing.	127
8.4	Results of the method proposed by Minkov, Wang, et al. (2006) for two data sets for comparison with MOCRF.	128
8.5	Results in F_β on the result obtained via L-BFGS and MOCRF.	129
9.1	Distribution of the length of three entity classes.	134
9.2	Example of a skip chain CRF structure as used by Sutton and McCallum (2007) (as factor graph depiction).	134
9.3	Different skip chain factor templates to choose additionally to the linear chain. .	137
9.4	Markov blanket (shaded nodes) of a node v	139
9.5	Examples for the Markov blankets of variables touched by the skip chain template. The skip chain factors are shown in red.	139
9.6	Information Gain-based filtering results compared to empirical results for a sampled subset of skip-chain templates.	141

9.7	Markov blanket-based filtering results compared to empirical results for a sampled subset of skip-chain templates.	141
9.8	Results for all proposed templates measured empirically.	143

List of Tables

2.1	Example of tokenization.	19
2.2	Example feature sequence for text fragment.	21
2.3	Common static features capturing characteristics of each token.	22
2.4	Contingency table for two classes.	24
2.5	Named entity recognition example input sequence with output sequences.	26
4.1	Chemical entity classes used for corpora annotation.	62
4.2	Features used in the CRF.	67
4.3	Top 15 found terms with their number of occurrences.	72
4.4	Top 5 found converted structures.	73
4.5	Number of normalizable IUPAC entities in MEDLINE.	74
5.1	Examples for gene names.	77
5.2	Examples for correction of brackets in gene name recognition.	80
5.3	Results on the trainingset averaged over 50 bootstrap replicates and on the test set.	84
6.1	Summarization of Corpora developed for the detection of Single Nucleotide Polymorphisms.	93
6.2	Performance of named entity recognition with conditional random fields in a 50-fold bootstrapping evaluation on seed set.	96
6.3	Results of the final SNP model on independent test sets (in %).	98
6.4	Results on independent test set.	100
7.1	2×2 contingency table for feature selection in CRF.	109
7.2	Minimal numbers of original features needed to lose maximally 0.01 of F_1 measure applying IFP.	119
8.1	Results for classic L-BFGS training in comparison to MOCRF.	130
9.1	Top 10 skip chain templates determined by information gain-based filter, Markov blanket-based filter, and by training and comparison.	142
9.2	Top 20 skip chain templates from all proposed templates occurring at least 10 times.	144

Chapter 1

Introduction

1.1 Text Mining

Most knowledge in our world is stored and communicated in the form of natural language text. For a human, it is typically easy to analyze and understand the topic and context, which was sufficient in times of (hand-)written letters between scientists, authors, or poets. In recent times, especially since the incredibly fast establishment of online communication in the 1980s and 1990s the number of textual communications is growing.

This development includes a shift of personal discussions to computer-aided real-time media like chats as the Internet Relay Chat¹ (IRC) or asynchronous media as the Usenet² or bulletin boards realized in the World Wide Web. Sending letters is often replaced by sending emails. Publishing is not limited to journalists but enabled for everybody by web-logs (“blogs”) or microblogging as implemented in social network sites like Facebook³ or Twitter⁴.

While these technologies are kind of a new development, the classical media is changing as well, especially in the scientific and research community. Databases including abstracts of journal articles or proceeding contributions are freely available (like PubMed with Medline⁵, Citeseerx⁶, ACM Digital Library⁷) and even the number of electronically accessible full text articles is growing rapidly (PubMed Central⁸, ACL Anthology⁹). This trend of making articles available without fees or costs for the reader is called *Open Access*. An increasing number is even published online exclusively, without a printed version. Figure 1.2 shows the growth of the PubMed Central Open Access data¹⁰, a set of full text articles under the Creative Common Licence¹¹ or similar allowing data mining.

Additionally to these developments, the growing number of digitalized (scanned, optionally with character recognition) data as news corpora, journals of several decades, the web itself including huge text collections like Wikipedia¹² makes it impossible to read all relevant or

¹ <http://www.irc.org> (This and all subsequent footnote URLs have been verified on 12/01/2010. All date mentions in this thesis follow the format mm/dd/yyyy.)

² <http://www.usenet.org>

³ <http://www.facebook.com/>

⁴ <http://www.twitter.com/>

⁵ <http://www.ncbi.nlm.nih.gov/pubmed>

⁶ <http://citeseerx.ksu.edu.sa/>

⁷ <http://portal.acm.org/dl.cfm>

⁸ <http://www.pubmedcentral.nih.gov/>

⁹ <http://www.aclweb.org/anthology/>

¹⁰ <http://www.ncbi.nlm.nih.gov/pmc/>

¹¹ <http://creativecommons.org/>

¹² <http://www.wikipedia.org/>

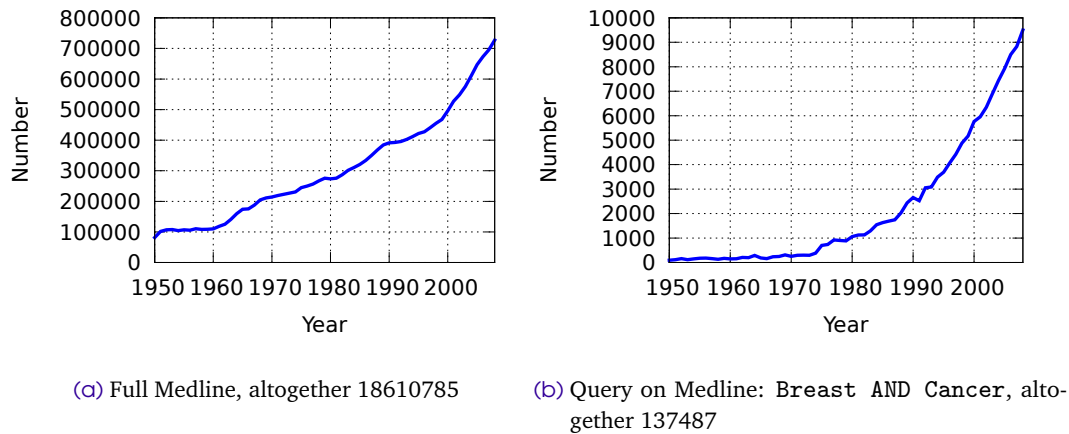


Figure 1.1: Number of abstracts per year in Medline (as of 07/26/2010).

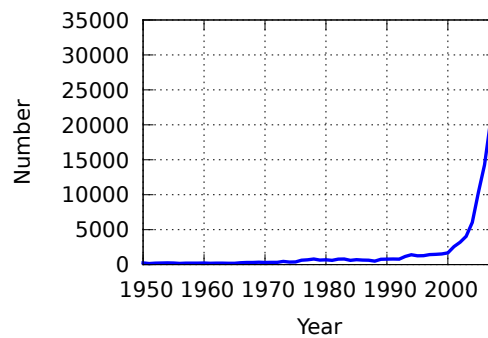


Figure 1.2: Number of articles per year in the PubMed Central Open Access data, altogether 188785 (as of 07/26/2010).

interesting articles available. The number of articles published in conferences or journals is growing so rapidly that it is not possible to follow, even in a comparatively limited or new topic. This is visualized in Figure 1.1. As depicted in Figure 1.1b, the number of articles dealing with for instance breast cancer was 9503 in 2008, to read all of them meant reading 26 articles per day.

This huge amount raises several challenges, some of them are briefly introduced in the following.

1.2 Challenges in Processing Text

These huge amounts of data collections in electronic form harbor huge chances as well as occurring problems. They include the following challenges (without being comprehensive).

Information Retrieval

Information retrieval (IR) deals with representing, storing, organizing of and, the most important topic, the access to information items (Baeza-Yates and Ribeiro-Neto, 1999). This includes the search for a document by query of a user as well as the request of documents being similar to a given one. Efficient processing and indexing are part of that as well as visualization of the results.

Typically, no semantic representations are incorporated in the IR process, statistical approaches based on the bag-of-words (see Section 2.6) are used. A combination of both is the so-called semantic search, on which light will be shed in Section 1.3.4.

Clustering

Clustering of documents is of interest to collect similar documents, therefore it can be applied for Information Retrieval (Section 1.2). Beside that, it helps structuring digital libraries and can filter for relevant documents. Statistics of the different clusters may be interesting, to find measures for the diversity of a text collection. Other applications include the organization of search results (Zamir, Etzioni, et al., 1997) or plagiarism detection (Frigui and Nasraoui, 2004). Commonly, the bag-of-words is used as in Information Retrieval as main component.

The problem of modeling topics (*Topic models*, Blei and Lafferty (2009)) can be understood as a special case of clustering. A common approach (Blei, Ng, et al., 2003) is formulated as probabilistic models to uncover the underlying semantic structure of a document providing the user with a set of topics, each specified by a set of most important words in this topic. Therefore, this method allows to get a good overview of a text collection and helps browsing it.

Text Classification

In contrast to clustering and topic modelling, *text classification* is typically performed in a supervised or semi-supervised manner (Joachims, 2002). The goal is to classify texts in two or more classes, e. g. “belongs to a topic A” or “does not belong to topic A”. This can be used for information retrieval or the content of the text can be used to solve other classification problems, as for instance shown by Shatkay, Höglund, et al. (2007) for the prediction of protein subcellular localizations.

Information Extraction

Information extraction is similar to information retrieval, but with the goal of extracting structured information while the retrieval task is typically fulfilled by returning a set of documents (Cowie and Wilks, 1996). Typical subtasks are *named entity recognition* (which will be introduced in Section 1.3) or coreference¹³ (Peng and McCallum, 2006). One of

¹³ Coreference occurs when expressions refer to the same entity. Example: “Peter said that he will be late.”. “Peter” and “he” refer to the same person.

Human immunodeficiency virus (HIV) and acquired immune deficiency ...

Figure 1.3: Example of a text with specified named entities of the class *virus* (text snippet from Peterson (2010)).

the most important challenges is relation extraction, the identification of relations between entities.

Named entity recognition is a prerequisite for other information extraction tasks and therefore of fundamental importance.

A more comprehensive overview of the different challenges and tasks in text mining in general is given in the text books by Berry (2004) and Srivastava and Sahami (2009) as well as Clark, Fox, and Lappin (2010) or Manning and Schütze (2003).

1.3 Named Entity Recognition

Named entity recognition (NER, Nadeau and Sekine (2007)) is a fundamental basis for many information extraction tasks, it is commonly understood as labeling the mentions of terms of a class of interest given a text, *e. g.* by specifying offsets relative to characters in the text or marking them graphically. An example of NER is given in Figure 1.3.

Methods can be divided into gazetteer or dictionary based, rule-based and machine-learning based, each having their advantages and disadvantages, which will be briefly discussed here. These classes typically overlap to a certain extent.

1.3.1 Dictionary-based

The simplest approach from the technical point of view are purely dictionary based systems performing a classification of a term by the occurrence in a list¹⁴. The prerequisite is the existence of such list; harvesting one from literature can be similarly tedious as the annotation of a training corpus for a machine learning-based approach (compare to Section 2.3). If the entity class of interest does not frequently contain ambiguous names, such an approach rewards the developer and user with a typically highly precise result of terms which can additionally easily be mapped to a data base identifier (so called normalization, compare to Section 2.7). Otherwise, disambiguation approaches need to be implemented which may be rule-based as well as machine learning-based. The main disadvantage is the disability to find terms not present in the dictionary. If the class of interest is rapidly evolving the recall may be limited and the system needs a lot of maintainance. Similarly, finding names of a class which are not finitely enumerable cannot be addressed with a dictionary. Additionally, if the

¹⁴ These lists are typically called gazetteer while this term originally refers only to lists of geographical places. Synonymously, the term dictionary is used, being more common in the domain of biological, medical and chemical language processing. These terms are used synonymously throughout this thesis.

class of interest harbors different writing variations or permutations of words, a dictionary based approach may be sophisticated.

Applications include the detection of chemical names (Hettne, Stierum, et al., 2009; Hettne, Williams, et al., 2010), gene/protein names (Hanisch, Fundel, et al., 2005), drug names and adverse effects (Leaman, Wojtulewicz, et al., 2010). In the non-biomedical domain, a widely addressed task is the recognition of person names, organization names, and places (Sang and De Meulder, 2003).

1.3.2 Rule-based

The manual specification of rules to detect names can be simple for classes like frequently mentioned data base identifiers in text (like “AB1234”) and more complicated if it comes to natural language formulations. Rule-based systems are successful in some domains, especially in combination with dictionaries. Note, that using regular expressions for detecting names is in this category.

The main advantage is the possible comprehension of such a system as hand-crafted rules include knowledge of the task. Additionally, names can be found without being limited to a dictionary which needs to be updated frequently.

This approach can easily be applied to detect morphologically notable entities like email addresses but also natural language formulations like sub-types of singular nucleotide polymorphism mentions (Caporaso, Baumgartner, et al., 2007).

1.3.3 Machine Learning-based

Machine learning-based named entity recognition has the advantage of being able to find names which have not been seen before without the disadvantage of the need of manually building rules. Though, this generalizability is dependent on the incorporated features; a system only relying on a single dictionary-based feature would not allow for that. The main disadvantage is the need for a representative training corpus.

This thesis is focusing on machine learning-based named entity recognition using graphical models, more specifically conditional random fields (Lafferty, McCallum, and Pereira, 2001), which are introduced in Chapter 3. An overview of the whole workflow is given in Chapter 2. Machine learning-based named entity recognition has been applied to a large number of domains—some of which will be presented in this thesis. Prior work will be referred to in the chapters in Part II respectively.

1.3.4 Applications in the Domain of Biology, Medicine, and Chemistry

Named entity recognition and information extraction in biology, medicine, and chemistry has special challenges in comparison to other domains. These will be explained in the application adaptations in Part II. In the following, some applications of named entity recognition exemplifying the use of such methods are mentioned.

Having the information in which documents an entity class of interest is used can support information retrieval for the so-called *semantic search*. An example query could be

```
Give me all documents mentioning "breast cancer" and a gene name!
```

Obviously, that would not be possible without named entity recognition. In comparison to Figure 1.1, the number of articles is limited as depicted in Figure 1.4 for the query above and analogously for chemical IUPAC names instead of gene names. This is an application in information retrieval, but can easily be extended to a simple version of information extraction by returning ranked lists of the respective entities, leading to an hypothesis of related genes or chemicals to breast cancer. Example implementations of such systems are SCAIView¹⁵ (Hofmann-Apitius, Fluck, et al., 2008), GoPubMed¹⁶ (Doms and Schroeder, 2005) or AliBaba¹⁷ (Plake, Schiemann, et al., 2006).

While this already supports information retrieval, the classical paradigm of using bag-of-words as features can be enhanced: Instead of using words to measure *e. g.* the similarity of documents, normalized named entities can be used, such that different synonyms are known to refer to the same real-world entity. Such an approach has been shown to be useful by Gurulingappa, Müller, et al. (2009).

In addition to this usage of named entities in further computations, they can be helpful alone. Data base curators read journal articles and pass the information manually into structured data bases. A visualization of the relevant entity classes in the document helps the curators to speed-up their work by up to 35% (Dowell, 2009). Visualization aspects will be briefly discussed in Chapter 2.

Other applications of named entity recognition include the generation of networks, *e. g.* by assuming that the co-occurrence of an entity shows a relation (Younesi, 2008). More sophisticated network analyzes are based on relation extraction methods using linguistic properties and have typically a higher precision, as incorporated by Leach, Tipney, et al. (2009). A visualization of such networks is included in the application AliBaba. Relation extraction methods are the topic of the shared tasks and challenges BioCreative (Hirschman, Yeh, et al., 2005; Hirschman, Krallinger, and Valencia, 2007; Bañeres, Cesareni, et al., 2009) and BioNLP 2009 (Kim, Ohta, et al., 2009; Bjerne, Heimonen, et al., 2009), typically having a named entity recognition as prerequisite, too.

Text mining in general has a lot more applications and benefits, the focus of this thesis is named entity recognition. For an overview of current topics in bio-medical text mining, the respective section of the journal *Bioinformatics*¹⁸ as well as the annual BioNLP Workshop of the Association of Computational Linguistics¹⁹ are recommended.

¹⁵ <http://www.scaiview.com/>

free version focused on animals: <http://www.scaiview.com/animal>

¹⁶ <http://www.gopubmed.org/>

¹⁷ <http://alibaba.informatik.hu-berlin.de/>

¹⁸ <http://bioinformatics.oxfordjournals.org>

¹⁹ Proceedings available online: <http://www.aclweb.org/anthology-new/>

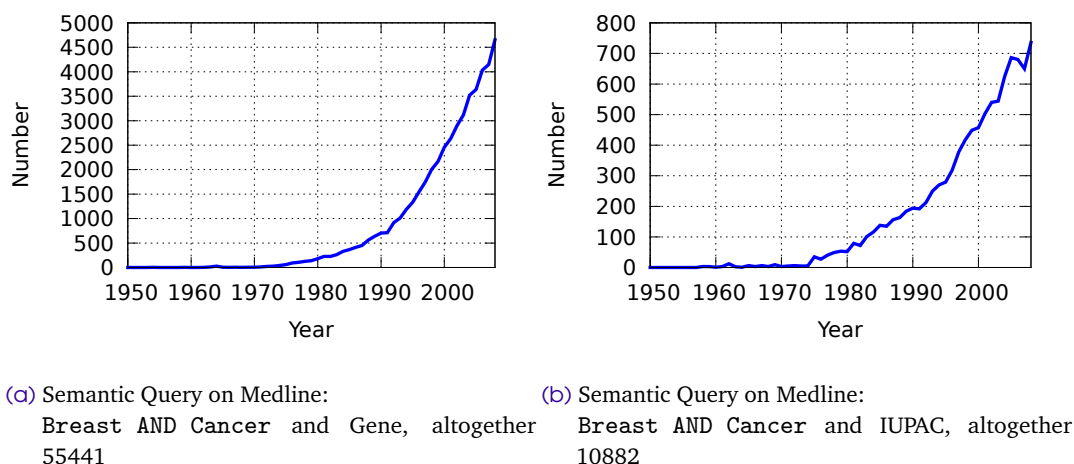


Figure 1.4: Number of articles per year in Medline.

1.4 Motivation and Structure of this Thesis

As described in Section 1.3, named entity recognition is the foundation for several applications and can help in different settings. Different techniques have been applied in the past, machine learning-based methods being one of the state-of-the-art approaches, especially linear-chain conditional random fields.

The work presented in this thesis was mainly performed in collaboration with the Bioinformatics Department at the Fraunhofer Institute for Algorithms and Scientific Computing²⁰ (SCAI). One focus of this group is “Information Extraction and Semantic Text Analysis”²¹, mainly dictionary and rule-based named entity recognition using ProMiner²² (Hanisch, Fundel, et al., 2005). As dictionary-focused methods have some limitations (as described in Section 1.3.1), one goal of this thesis is to prove the practicability of conditional random fields for classes from biology and chemistry which have largely not been addressed so far and where dictionary-based methods are problematic. The model selection is analyzed in detail with the goal to provide knowledge to support the process of building named entity recognition systems for new entity classes. Specific problems, presumably being typical for biological and chemical name classes, are addressed later. Although this thesis focuses on biology and chemistry, the proposed methods can be transferred to other domains as well.

In short, the work in this thesis provides the knowledge and methods to simplify the workflow of building named entity recognition systems, especially for the comparatively complex classes of interest from biological and chemical domains. Besides that, this simplification should lead to better, more appropriate models.

²⁰ <http://www.scai.fraunhofer.de/>

²¹ <http://www.scai.fraunhofer.de/en/business-research-areas/bioinformatics/research-development/information-extraction-semantic-text-analysis.html>

²² <http://www.scai.fraunhofer.de/prominer.html>

In more detail, the start are the **Fundamentals** in Part I: An introduction to named entity recognition and a summarization of the necessary steps from data preprocessing and building and applying a model to the presentation of results to a user is given in Section 2. The following Section 3 introduces graphical models in general and the development of conditional random fields (CRF), a class of conditional graphical models. Linear-chain conditional random fields (LCCRF) are highlighted here as one current state-of-the-art method. The relations of CRF to other graphical models is pointed out as this is beneficial to understand the underlying ideas of the model.

Case Studies and Adaptions to the Biological and Chemical Domain are presented in Part II, namely the **Recognition of IUPAC and IUPAC-like Chemical Names** in Chapter 4, the **Recognition of Gene and Protein Names** in Chapter 5, and the **Recognition of Mentions of Single Nucleotide Polymorphisms** in Chapter 6. The recognition of chemical names, especially names following the nomenclature of the International Union of Pure and Applied Chemistry IUPAC (McNaught and Wilkinson, 1997) is difficult because of the sheer amount of names. IUPAC names are even not finitely enumerable as they are generated from the actual chemical compound following a rule set. Unfortunately, authors of scientific publications do not stick to these rules consequently, which leads to the need of detecting IUPAC-like names, too. In this context, the detection of other chemical names will be discussed. A comparison with dictionary-based methods, a collaborative work with Corinna Klein is included (Klein, 2010) which especially motivates the approach chosen here.

The recognition of gene and protein names in Chapter 5 is highly relevant in bio-medical text mining and one of the first topics addressed. It is the main theme of the BioCreative competitions (Hirschman, Yeh, et al., 2005; Hirschman, Krallinger, and Valencia, 2007; Bañeres, Cesareni, et al., 2009). In BioCreative II (Hirschman, Krallinger, and Valencia, 2007), Named entity recognition was a separate task, addressing the question how to deal with multiple annotations. While 21 teams participated, only two actually made use of the provided data, therefore, this can be rated as an unsolved problem. Additionally, another question is discussed: As it is equally common to address gene and protein name recognition with machine learning as well as dictionary-based methods, it is analyzed to what extent the most often claimed advantage of the use of machine learning, the generalizability to newly invented names, really holds.

The challenge of recognizing Single Nucleotide Polymorphisms (SNP) mentions in text (described in Chapter 6) needs to be addressed by a machine learning approach as many formulations of the multiple classes of interest are given in natural language. While rule-based approaches and machine learning-based methods for a sub-class of abstracts have been published already, the generalization to be used on all MEDLINE abstracts as well as an approach to normalization is presented. The latter is especially challenging in this application, therefore common pitfalls are briefly presented.

In Part III, **Further Enhancements** motivated by Part II are addressed. In Chapter 7, the problem of selecting a feature subset is discussed. Typically, huge numbers of features are generated by automated feature extraction methods *e. g.* by using each word or prefix and suffix of a specified length. The goal incorporating feature selection is to generate a model

with the same or similar performance but a lower complexity. That leads to more efficient inference and training and allows a better understanding of the trained model. For the feature subset selection, known methods from classification settings are adapted. The proposed methods allow the design of a workflow to train a model on a domain with a huge number of predefined features which can presumably capture important characteristics. While training with all features would not be feasible, the feature selection simplifies the process of adapting a GRF to a new domain to a large extent.

Chapter 8 addresses an optimization issue of conditional random fields. In machine learning, the objective function to be trained is not necessarily the one to be evaluated later on. One reason is that the final evaluation function could be non-derivable which made optimization harder. Another reason may be that the training procedure has characteristics which demand another objective function for training. For conditional random fields, typically the log-likelihood of the model is used as an objective function, corresponding to the optimization of the accuracy of the model, while the F_1 measure²³ is used for evaluation. This measure is the harmonic mean of precision and recall, it seems to be natural to optimize both measures in a multi-criterial setting to allow the user to choose a model with desired β for F_β at inference time.

Chapter 9 goes beyond linear chain conditional random fields (while the previous methods can easily be applied to non-linear structures, too). Especially in biology and chemistry, named entities are frequently multi token terms. A hypothesis is that a linear-chain structure is not able to capture these intrinsic dependencies. While experiments with different dependencies for named entity recognition have been published, an automated selection of a structure has not been developed. This thesis presents automated approaches to find a meaningful structure which shows improvements in named entity recognition together with an evaluation of heuristics to speed up this search.

Part IV recapitulates this thesis and gives a summary of the main contributions.

²³ Evaluation measures will be presented in Section 2.8.

Part I
Fundamentals

Chapter 2

Workflow of Named Entity Recognition

2.1 Introduction

This chapter introduces the whole workflow of named entity recognition from reading and preprocessing data (Section 2.2) and preparation of training data (Section 2.3) over the core methods of recognition (Sections 2.4 to 2.7) to evaluation (Section 2.8) and presentation of the result of an applied model to a user (Section 2.9). The workflow is depicted in Figure 2.1 and a detailed description given in the next sections.

The first step ① is the *data selection* for training as well as for inference that can include a filtering of irrelevant documents or a selection of most informative documents for training. The second step ② is *preprocessing*, which means extraction of the plain text from the different input formats to have a common basis to apply the methods to. To generate training data for a machine learning model or to evaluate the model, text needs to be *annotated* in the third step ③. The fourth step ④ *sentence splitting* and *tokenization* splits the document into logical units and the fifth step ⑤ *Token/Character Normalization* is necessary to limit the dimensionality of the problem in obvious cases.

The order of steps ①–⑤ is loosely coupled: It could make sense to do the annotation on the original data format instead of preprocessed data (exchange ② and ③), splitting and tokenization could as well be done before the annotation of data (exchange ④ and ⑤ with ③). The presented order allows a standardized environment to annotate independent of the document source and annotation before tokenization as this could influence the annotator (if she gets to know about this) which is typically not desired.

The sixth step ⑥ is the actual *NER Method*, in our case a machine learning based method leading to a model which is used later at inference time (This step could be replaced by a dictionary-based or rule-based method without changing the principle of the workflow). It consists of a feature extraction, model training or application. If the model is applied to a data set, the seventh step ⑦ is a *postprocessing* and/or *normalization* of the result which can then be evaluated, visualized or used for further computations in step eight ⑧.

In the following, an introduction to the different steps in this workflow is presented to provide an understanding of the procedure as a whole.

2.2 Data Formats, Selection, and Preprocessing ① ②

Data sources playing a role in the bio-medical domain are *e. g.* abstracts of scientific articles from the data base MEDLINE or full text articles, *e. g.* from PubMed Central (see Section 1.1).

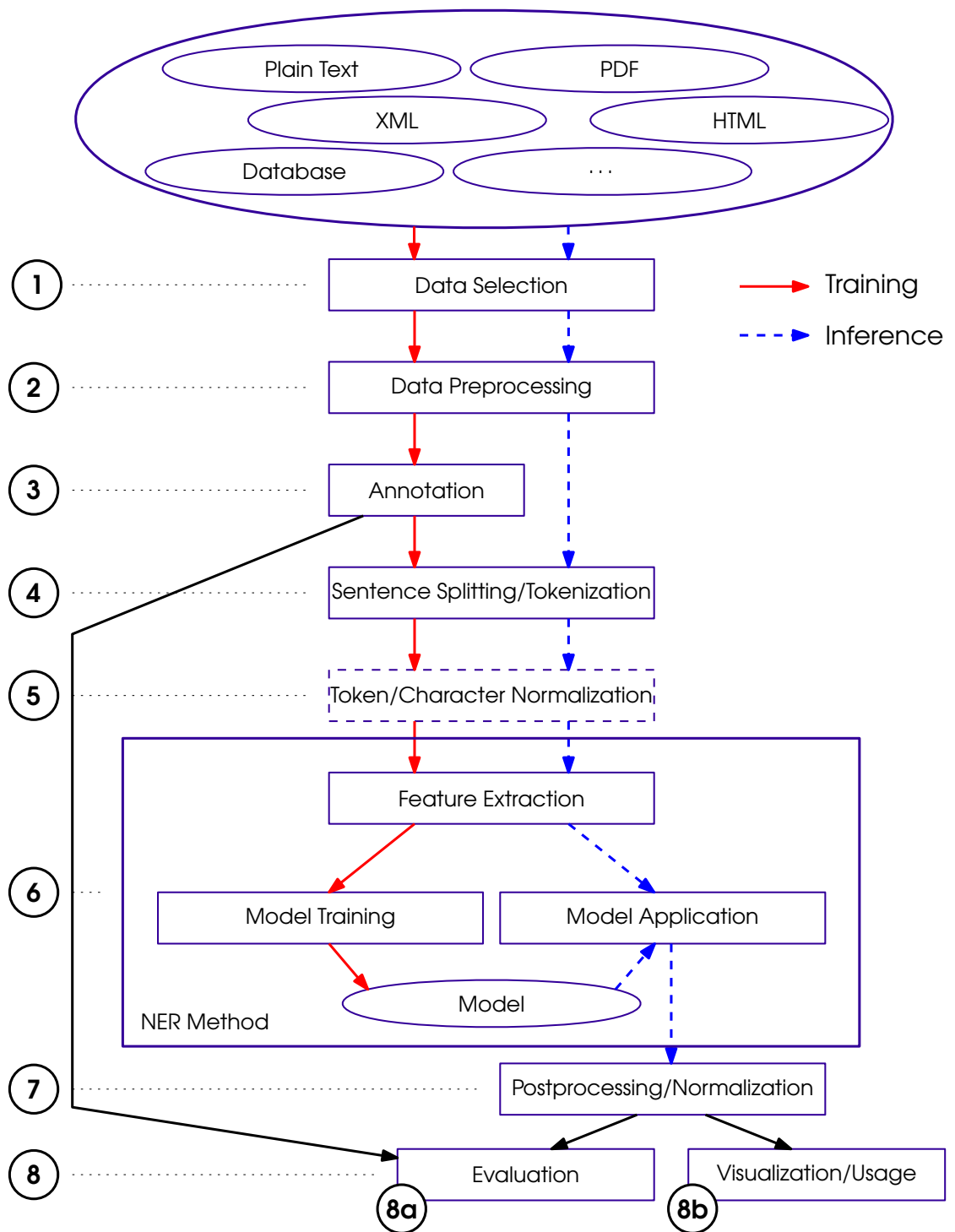


Figure 2.1: Workflow of named entity recognition.

Other sources of interest include patents (Lupu, Piroi, et al., 2009) or clinical data¹. These data are provided in a variety of formats, the most important ones are plain text (in different encodings like `latin1`, `ISO8859-15`, or `utf8`) and similarly simple text based formats like the MEDLINE format used *e. g.* for MEDLINE abstracts. For full text articles, eXtensible Markup Language (XML) files are commonly used, as in the open access subset of PubMed Central as well as files in HyperText Markup Language (HTML, World Wide Web Consortium (W3C) (1999)) for online reading. Evenly important is the Portable Document Format (PDF, Adobe Systems Incorporated (2007)), a graphical file format to allow exchange and view documents with a consistent layout on different devices (similarly to the Device Independent File format (DVI) or PostScript). This is the main format to exchange full text articles or retrieve patents.

After choosing a source for the documents, the relevancy of the different documents needs to be clarified. Therefore, a pre-classification of the documents can be beneficial, *e. g.* by a query to the database or a more sophisticated classification to filter relevant documents. As an example, it is typically not necessary to deal with abstracts talking about politics if gene name recognition should be applied. For the annotation of data as a training corpus the selection should be representative for the domain of application. A different philosophy is followed by *active learning*, which proposes to select the most informative training examples to avoid redundant annotation of data (Engelson and Dagan, 1996; Vlachos, 2008; Tomanek, 2010). Popular approaches are uncertainty-based sampling (Cohn, Atlas, and Ladner, 1994; Cohn, Ghahramani, and Jordan, 1996) and query-by-committee (Seung, Opper, and Sompolinsky, 1992).

The easiest format to process is plain text. XML is typically straight-forward to use as it is standardized and parsers are available for all popular programming languages. Documents in HTML are widely available as this is the main format for online publication. Cleaning the file and providing extracted plain text seems to be trivial, but it is not as filtering for the important parts of the web site needs to be performed. Additionally, information of the structure of the document should be preserved as these data can be used to filter for relevant paragraphs or postprocess the results. This was the task in the CLEANVAL shared task and competitive evaluation on the topic of cleaning arbitrary web pages to prepare web data for use as a corpus (Baroni, Evert, et al., 2007) and addressed by Meyer (2009) especially for biomedical texts. Frequent problems occur because the specifications are not strictly followed; improperly nested tags are for example not advised (`<a>`). Therefore, an interpretation of the data is necessary with a specialized HTML parser, representing the document as a well-formed tree. The plain text needs to be extracted and kept track of the HTML elements relative to the text. This is necessary to combine the information of the structure of the document with extracted semantic information.

The most challenging format to preprocess is PDF, as it is graphically structured (Klinger, Pesch, et al., 2009a). The main challenge is to reconstruct the reading order, as the included textual information (during generation of the PDF or by inclusion with optical character recognition) is not necessarily the logical one. This includes to recognize headers, footers,

¹ Compare to the currently ongoing i2b2/VA Shared Task (<http://www.i2b2.org/NLP>).

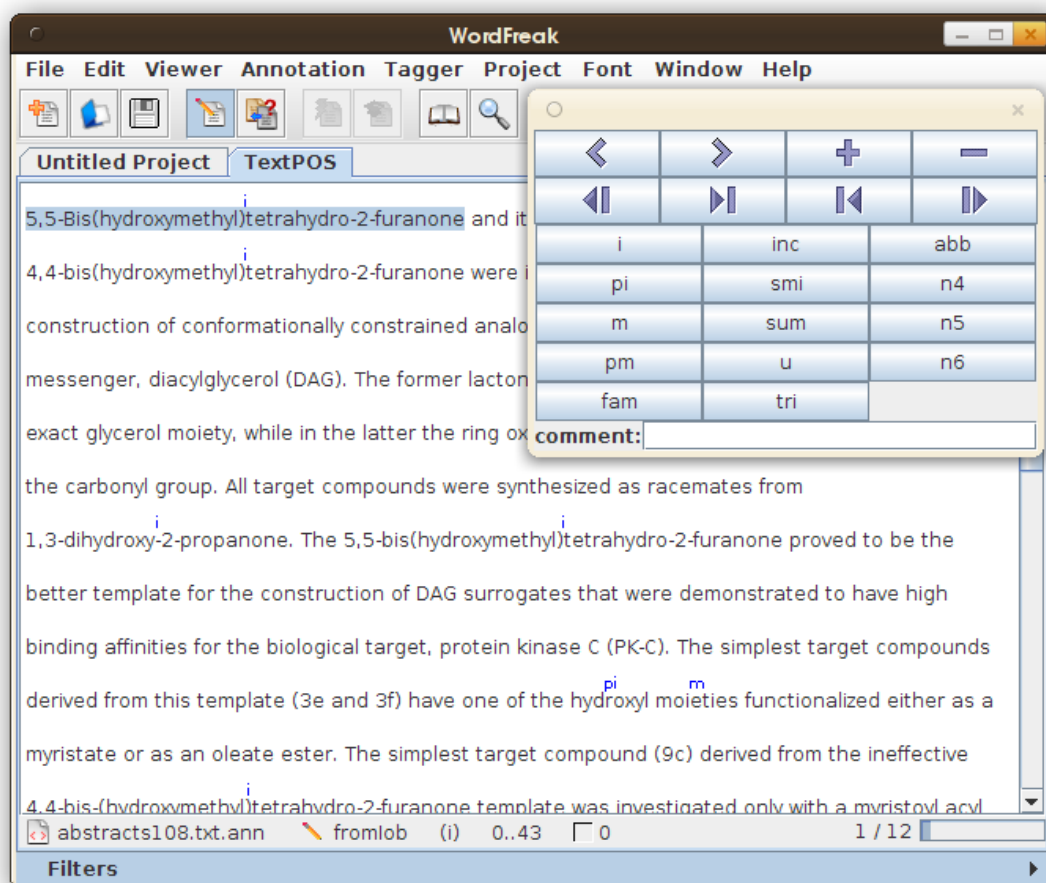


Figure 2.2: Screenshot of the annotation tool WordFreak.

tables, figures, captions, and footnotes. To keep track of the logical structure similarly to HTML, section headers should as well be recognized to detect common zones like introduction, methods, materials, results, or discussion. Assigning sentences to these sections has been evaluated by Agarwal and Yu (2009). A system addressing the described problems has been developed by Pesch (2010).

2.3 Annotation ③

To generate a training, validation, or evaluation set, manual annotation of text with the entity class of interest is necessary. Different tools are available to support the annotator, in this work, WordFreak (Morton and LaCivita, 2003) and Knowtator (Ogren, 2006) are used. A screenshot of WordFreak is shown in Figure 2.2, one of Knowtator in Figure 2.3. These tools were selected as WordFreak is light-weighted and can easily be used for simple

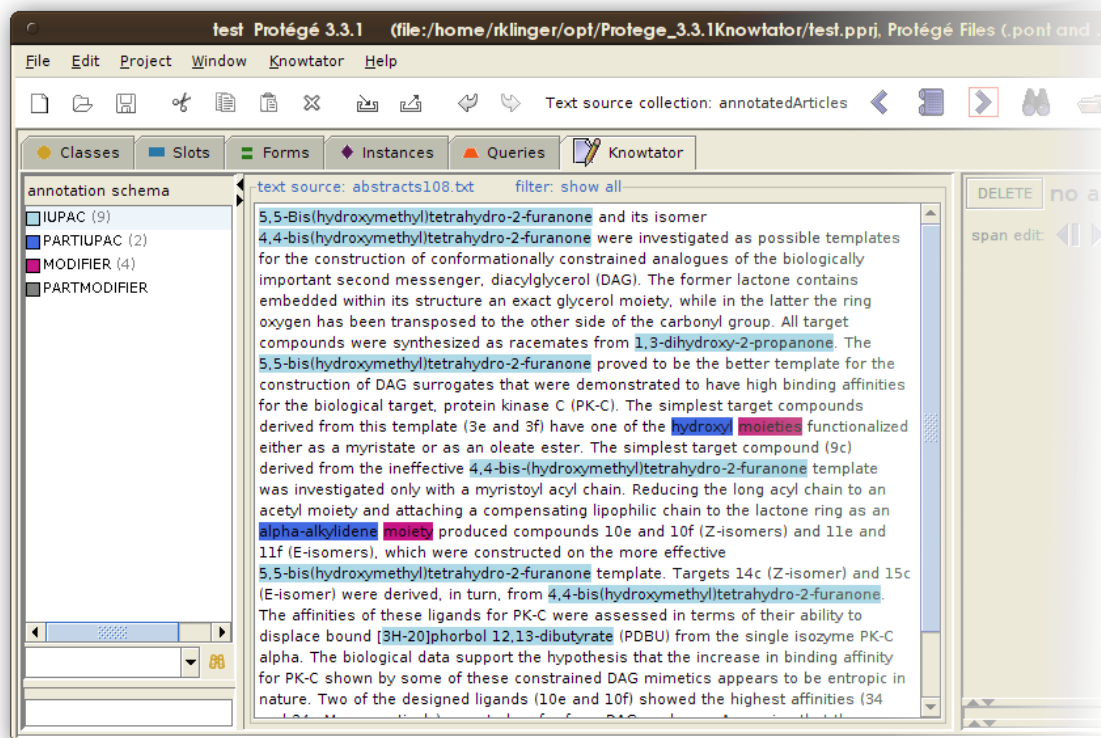


Figure 2.3: Screenshot of the annotation tool Knowtator in Protégé.

annotation tasks without installation. Knowtator is a plug-in for the ontology editor Protégé, therefore an installation is necessary. The advantage of Knowtator is its intuitiveness and flexibility. In contrast to WordFreak, it can easily handle overlaps between classes, add attributes to annotated entities (like comments or identifiers) and combine different entities to annotations of relations. Additionally, Protégé is established in the bio-medical information engineering community and therefore often known to potential annotators who are typically domain experts.

Though annotating data is typically time consuming and therefore cost intensive, multiple annotators should annotate the same documents to measure the quality of annotation. A measure for the consistency of two annotations, the inter-annotator agreement can be calculated. Typically the *Kappa Statistic* κ (Carletta, 1996; Siegel and Castellan, 1988) is used. Let $p(A)$ be the probability that annotators agree and $p(E)$ be the probability that they are expected to agree by chance. Then κ is defined as

$$\kappa = \frac{p(A) - p(E)}{1 - p(E)}. \quad (2.1)$$

This measure is developed for classification purposes, but it can be applied to named entity recognition by understanding the task as classification of each single word (or token) into *part*



Figure 2.4: Example of a text with multiple named entity annotations of the class *virus* specified by two annotators (text snippet from Peterson (2010)).

of *entity* and *outside an entity*. This measure does not take into account entity boundaries. But especially detecting those is a core task of named entity recognition. Therefore, a more intuitive solution is the application of the F_1 measure introduced in Section 2.8 in Equation 2.6 by assuming one of the annotators to represent the gold standard and the other to represent the guess. A disadvantage compared to the κ statistic is that more than 2 annotators cannot be taken into account.

A measure of inter-annotator agreement can be interpreted in two different ways. Firstly, how good the quality of the annotation is, secondly, how difficult an annotation task is. Having an automatically retrieved performance which is considerably higher than measured via inter-annotator agreement has to be at least critically reviewed. This may point out an overfitting problem.

2.4 Sentence Splitting and Tokenization 4

Syntactically, nearly all texts are divided into logical units. *Sentences* provide units which have an enclosed, often self-contained meaning to the author. Entities do not span over sentence borders, therefore, to keep separated instances small, it makes sense to work on separated sentences.

The process of separating sentences, typically referred to as *sentence splitting*, is not as trivial as it may seem on the first sight: Punctuation marks like “. ! ? ; : † . . .” are not always used at the end of a sentence (*e. g.* in figure captions or tables), especially not in an ellipsis. Additionally, these symbols are used in different context as well (*e. g.* in abbreviations like “Dear Mr. Miller,” or “*e. g.*”). In bio-medical entities, they can be part of a name (derived from abbreviations, like “*E. coli*” (Escherichia coli)). Most sentence splitters are based on regular expressions as they are very fast and of sufficient performance for most tasks, but machine learning based methods exist as well (Tomanek, Wermter, and Hahn, 2007c).

The process of splitting sentences into more fine-grained units is called *Tokenization*, leading to a list of *Tokens*. The idea of tokenization is typically to split as fine-grained as necessary to split logical units (like named entities) from the rest of the text and not too fine to keep logical units together if possible. This is especially important for machine learning methods which try to catch properties of the tokens to decide their role. Examples are shown in Table 2.1. Splitting at white space only is obviously not sufficient, as terms are not separated from brackets. Splitting at white space and punctuation marks splits floating point numbers. Separating tokens using all non-alpha symbols splits numbers. The last example is probably the most reasonable one here. Such a tokenizer can be specified by

Tokenization	Text
Original	Allele frequencies of DPYD*5 (A1627G) and DPYD*6 are 12.7% and 7.1% respectively.
Splitting at white space	Allele frequencies of DPYD*5 (A1627G) and DPYD*6 are 12.7% and 7.1% respectively.
Splitting at white space and punctuation marks	Allele frequencies of DPYD*5 (A1627G) and DPYD*6 are 12 . 7% and 7 . 1% respectively .
Splitting all non-alpha symbols	Allele frequencies of DPYD * 5 (A 1 6 2 7 G) and DPYD * 6 are 1 2 . 7 % and 7 . 1% respectively .
Splitting at white space and punctuation marks if followed by white space, always at brackets	Allele frequencies of DPYD*5 (A1627G) and DPYD*6 are 12.7% and 7.1% respectively .

Table 2.1: Example of tokenization (using | as token limiter, example text derived from Fredj, Gross, et al. (2007)). The last example is probably most reasonable.

a lexical analyzer generator like JLex². A set of regular expressions can be used to define which symbols should be kept together.

Having a token sequence derived from a text, named entity recognition can be understood as a token labeling process. Given the text sequence $\vec{x} = (x_1, \dots, x_n)^T$, a label sequence $\vec{y} = (y_1, \dots, y_n)^T$ needs to be found. This sequence is encoded in a label alphabet $\mathcal{L} = \{I-<entity>, O, B-<entity>\}$ where $y_i = O$ means that x_i is outside of an entity of interest, $y_i = B-<entity>$ means that x_i is the beginning of an entity and $y_i = I-<entity>$ means that x_i is an intermediate term of an entity. This format is therefore commonly known as IOB format. Variations are IO (not capturing multiple subsequent entities) and IOBE (explicitly coding the end of an entity).

In the applications in Part II, the importance of tokenization is discussed.

2.5 Token and Character Normalization ⑤

An optional step is the normalization of tokens and characters to remove obvious redundant encoding variants and therefore reduce dimensionality. Normalizing tokens can include the correction of frequent spelling errors (e. g. match feasible to feasible), frequent

² <http://www.cs.princeton.edu/~appel/modern/java/JLex/>

OCR errors or even replacing tokens with synonyms. Writing variants in British English and American English can be mapped to one form. Computing the stem or lemma of the word can be performed as well (“stemming”, “lemmatization”).

In a machine learning-based setting, such normalizations are typically performed during feature extraction and could be skipped. That holds for the character normalization as well, while it seems to be more natural to do that before feature extraction: A lot of symbols and characters with exactly the same semantic meaning can be expressed differently, from a technical point of view. That includes HTML and XML entities, different encodings, different writing variants of regional symbols like umlauts. As an example, the Greek letter β can be written as BETA or beta in text, as β or the variant β . Sometimes the German β is used wrongly, being a ligature for ß (ß). Mentions of Unicode (223) and HTML entities ($\&\text{beta}$) occur as well, even in plain text, due to publishers processing errors. Such Greek-letter conversion is especially necessary as such occur frequently in gene and protein names. Another example is the normalization of punctuation marks like the horizontal bar; the en dash “–”, the em dash “—”, the hyphen “-”, the minus sign “-” or the hyphen bullet - are commonly confused.

The normalization of tokens and characters could be seen as part of the preprocessing in step ②, but the advantage of doing that after the tokenization is that word boundaries are known and can be taken into account.

2.6 Feature Extraction ⑥

The set of features presented in the following can be found in a variety of applications from several domains. Publications using them in the context of probabilistic graphical models include McDonald and Pereira (2005); McDonald, Winters, et al. (2006); Narayanaswamy, Ravikumar, and Vijay-Shanker (2003); Settles (2005); Leaman and Gonzalez (2008); Liu, Huang, and Zhu (2010); McCallum (2003); Friedrich, Revillion, et al. (2006); Klinger, Friedrich, et al. (2007); Klinger, Kolářik, et al. (2008); Kolářik, Klinger, and Hofmann-Apitius (2009), and in different text processing methods *e. g.* Ando (2006); Joachims (2002). These features are fairly standard and partially used in rule-based systems and dictionary-based systems as well, therefore introduced here. More domain specific features are defined in the respective sections in Part II.

This set of features (which are typically boolean) can be divided into the categories

- *Statically defined morphological,*
- *Generated pattern-based from training data,*
- *Dictionaries*
 - *derived from training data,*
 - *from external resources,*
- *Grammatical features,*
- *Contextual feature patterns.*

SGPT	,	SGOT	,	and	...
INITCAPS	POS="NN"@1	INITCAPS	POS="NN"@-1	POS="NN"@1	
NP="B-NP"	NP="B-NP"@1	NP="B-NP"	NP="B-NP"@-1	NP="B-NP"@1	
POS="NN"	POS="NN"@-1	POS="NN"	INITCAPS@-1	PREFIX3=alk@1	
NP="O"@1	NP="B-NP"@-1	NP="O"@1	POS="CC"@1	PREFIX4=alka@1	
PUNCTUATION@1	INITCAPS@-1	NP="O"@-1	NP="O"@1	SUFFIX3=ine@1	
ALLCAPS	NP="O"	PUNCTUATION@1	WORD=and@1	SUFFIX4=line@1	
POS=","@1	PUNCTUATION	GENELIST1	NP="O"	WORD=alkaline@1	
WORD=,@1	INITCAPS@1	ALLCAPS	PUNCTUATION	WORD=and	
WORD=SGPT	ALLCAPS@1	PUNCTUATION@-1	GENELIST1@-1	NP="O"	
SUFFIX3=GPT	GENELIST1@1	POS=","@1	ALLCAPS@-1	POS="CC"	
PREFIX3=SGP	ALLCAPS@-1	WORD=,@1	WORD=,	GENELIST4@1	
	WORD=,	POS=","@-1	POS=","	NP="O"@-1	
	POS=","	WORD=,@-1	PREFIX3=SGO@-1	PUNCTUATION@-1	
	W=SGOT@1	WORD=SGOT	SUFFIX3=GOT@-1	GENELIST5@1	
	PREFIX3=SGO@1	SUFFIX3=GOT	WORD=SGOT@-1	POS=","@-1	
	SUFFIX3=GOT@1	PREFIX3=SGO		WORD=,@-1	
	WORD=SGOT@1				
	PREFIX3=SGP@-1				
	SUFFIX3=GPT@-1				
	WORD=SGPT@-1				

Table 2.2: Example feature sequence for text fragment. Grammatical features are blue, offset conjunction features red (@1 meaning derived from the next token, @-1 from the previous, belonging to the class of contextual features), statically defined in green, pattern-based generated in brown, and dictionary-based in cyan.

An example of a text snippet with features for each token is shown in Table 2.2.

Statically defined features typically rely on regular expressions capturing properties of the token. The set which is the fundament for the work presented in this thesis is shown in Table 2.3.

In contrast to those, by far most of the features are not manually defined but automatically generated by patterns on the training data. The so-called *bag-of-words* (BOW) generates a feature for each token x_i which is true for a token x_j if $x_i = x_j$. Analogously, morphological features testing prefixes and suffixes of different length (e.g. 2, 3, and 4) are generated. Another pattern-based feature extraction method is similar to building regular expressions, the so-called *word class* (Settles, 2005). This feature is a mapping of capital letters, lower case letters, and numbers to one instance respectively. In the variation *brief word class*, consecutive classes are collapsed. As an example, the string “Htr1b” belongs to the word class [A-Z] [a-z] [a-z] [0-9] [a-z] and to the brief word class [A-Z]+ [a-z]+ [0-9]+ [a-z]+.

Dictionaries can be derived directly from the training data combining all terms of a class in a list. The according feature is true for a token iff that token is in that list. Note that in contrast to the BOW the detection of all different words of a class forms one feature, weighted by one parameter in a learning model. This is meaningful if it can be assumed that unseen words are seldom in the test data but can be counterproductive if new entities should be found which did not occur in the training data. Dictionaries from external resources can

Name	RegEx	Examples	
		Positive	Negative
Init Cap	[A-Z] . *	Doctor ATG P	then a .
Init Cap Alpha	[A-Z] [a-z] . *	Doctor Michael	ATG P
All Caps	[A-Z] +	ATG I	P Doctor
Capitalized	[A-Z] [a-z] *	Doctor Michael	I P
Mixed Caps	[A-Z] [a-z] + [A-Z] [A-Za-z] *	BioCreative	ATG mRNA
Has Digit	. * [0-9] . *	59 MMP14 Cul4a	Michael
Single Digit	[0-9]	9 5	59 Cul4a
Double Digit	[0-9] [0-9]	59 25	9 5 7a
Natural Num.	[0-9] +	9 5 59 135	5.6 7a
Real Num.	[-0-9] + [\ . ,] [0-9] +	5.6 -5,6	9 56
Has Dash	. * - . *	2-methyl 5-6	methyl
Init Dash	[- . *]	-6.5 -like	5-6
End Dash	[. * -]	5- web-	5-6
Alphanumeric	[A-Za-z0-9] +	Cul4a Michael 6	5-6 Dr.
Punctuation	[. , ; : ? ! - +]	. ,	5.
Multi Dots	[\ . \ . +]
End Dot	[^ \ .] + . * \ .	Dr. Mr.M.
Has Dot	. + \ . . +	A.b	Dr.
Acronym	[A-Z] [A-Z \ .] * \ . [A-Z \ .] *	P U.S. L.A.S.E.R	Dr.
Initial	[A-Z] \ .	H.	Mr.
Single Char.	[A-Za-z]	A B c	Aa H.
Single Cap.	[A-Z]	A B	c
Quote	[" ' ‘]	" "	a
Slash	[\ \ /]	/	-
Bracket	[(\ [\])]	[]	{

Table 2.3: Common static features capturing characteristics of each token (examples separated by |).

support generalization exceeding the training data. Additionally, the flexibility of a machine learning system in comparison to a dictionary based system only using this dictionary can be beneficial as it can learn in which context dictionary entries should not be tagged or tagged in addition to the content of the dictionary.

Grammatical features are typically extracted with an external approach. So-called POS taggers detect the *part-of-speech* of a token. The Penn Treebank Project³ lists 36 classes, some are *Noun* (NN), *adverb* (RB), or *adjective* (JJ). *Shallow Parsing* (also called *light parsing* or *chunking*) goes beyond POS tagging and additionally detects connected phrases in a sentence, like a *Noun Phrase* (NP) or *Verb Phrase* (VP) (Marcus, Santorini, and Marcinkiewicz, 1994). As an example, in the sentence “I will go to the nice house.”, “will go” is a verb phrase and “the nice house” a noun phrase. For POS tagging and Chunking several tools are available, partly specialized for bio-medical texts. An overview is given by Feldman and Sanger (2007). Going further to a full grammatical analysis is called *parsing* but will not be reviewed here as parsing based features are not a core part of this thesis. Approaches exist for joint grammatical analyzes and named entity recognition, but demands for data annotated with both. While these methods are promising, in practice this remains a preprocessing step (Finkel and Manning, 2009; Sutton and McCallum, 2005; McCallum, Rohanimanesh, and Sutton, 2003).

Contextual features are generated from a subset of surrounding tokens for the current token x_i . With the so-called *offset conjunction*, features of the preceding and succeeding tokens are added, specified by a window size. Additionally, logical conjunctions of the boolean features can be added. This class is very important to catch context in the text sequence, but can lead to a very huge set.

The full method of training and inference of a machine learning model is described in Chapter 3.

2.7 Postprocessing and Entity Normalization ⑦

Postprocessing can increase the performance of a system significantly. One simple common approach to increase the recall is to do an exact string search for a recognized entity in the proximity of the entity and add found strings to the results. The underlying assumption is that the string of a found entity will not be mentioned nearby with a different meaning. The precision can be influenced by filtering obvious false positives.

This approach of starting with a high recall system makes especially sense in the *Normalization*, the process of mapping mentions of names to real-world entities (Borgman and Siegfried, 1992). It can be understood as finding a function $\text{map}(e_i) \rightarrow d_j$ where e_i is a name mention $e_i = (x_p, \dots, x_q)$ ($p \leq q, p, q \in \mathbb{N}$) and $d_j \in D$ is an entity in a dictionary D of entities to normalize to.

There are two central challenges in the normalization procedure: Firstly, finding a canonical form of a name mention and secondly, *disambiguating* it if multiple real world entities fit.

³ <http://www.cis.upenn.edu/~treebank>

One domain of need for disambiguation is the normalization of person names (Mann and Yarowsky, 2003): Is the mention of a *Michael Jordan* referring to the basketball player or the machine learning professor? Some approaches take the context into account, *e. g.* topic based (*cf.* Section 1.2, Song, Huang, et al. (2007)). Others use classification systems to cluster the text in a supervised manner (Phua, Lee, and Smith, 2006). For such approaches the dictionary does not need to be given explicitly but can emerge from the disambiguation system. For many entity classes of interest in the biological domain, that dictionary needs to be provided beforehand. Therefore, dictionary-based named entity recognition provides a straight-forward way to map entities to the dictionary as each synonym can be directly allocated to an identifier. The disambiguation problem naturally remains. That approach is common for gene and protein names (A. M. Cohen, 2005; Hanisch, Fundel, et al., 2004; Hanisch, Fundel, et al., 2005; Fluck, Mevissen, et al., 2007).

The first step for finding a name in the dictionary is, as mentioned above, the generation of a canonical name. Therefore, a list of synonyms representing an entity d_j can be incorporated. Additionally, variants of a mention need to be reduced. As stated by Savary and Jacquemin (2003), such variations can be classified into being *orthographic*, *morphologic*, *syntactic*, or *semantic*. Orthographic variations are *e. g.* the introduction of optional hyphens, slashes, or an upper or lower case change. Morphological changes are *e. g.* plural or singular forms. Syntactic changes are variations due to the grammatical context like coordinations or the use of propositions. Semantic changes are the use of a modifier which changes the meaning of a term.

This work does not deal with disambiguation, though building canonical forms and mapping them to dictionaries are part of the applications in Part II.

2.8 Evaluation 8a

The performance of a trained model needs to be evaluated, finally on an independent test set to get a measure for the performance on unseen data as well as for meta-parameter optimization (as in the methods in Part III) on a validation set.

All measures used in this work are based on the contingency table shown in Table 2.4 (Rijsbergen, 1979). The entries in the table denote frequencies of instances being true

		Correct	
		C_1	$\neg C_1$
Predict	C_1	TP	FP
	$\neg C_1$	FN	TN

Table 2.4: Contingency table for two classes C_1 and not C_1 ($\neg C_1$) used to compute different evaluation measures. The abbreviations denote frequencies of instances being true positives (TP), false positives (FP), false negatives (FN) or true negatives (TN) given model parameters and a data set.

positives (TP), false positives (FP), true negatives (TN), or false negatives (FN). These values are functions of a model configuration $\vec{\lambda}$ and some data $\mathcal{D} \ni (\vec{x}, \vec{y})$ consisting of text sequences \vec{x} and given label sequences \vec{y} . The *Accuracy* is defined as

$$\text{acc}(\vec{\lambda}, \mathcal{D}) = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}}. \quad (2.2)$$

Closely related is the *Precision*

$$\text{prec}(\vec{\lambda}, \mathcal{D}) = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad (2.3)$$

which is combined with *Recall*

$$\text{rec}(\vec{\lambda}, \mathcal{D}) = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad (2.4)$$

to form the F_β measure

$$F_\beta(\vec{\lambda}, \mathcal{D}) = \frac{(1 + \beta^2) \cdot \text{prec}(\vec{\lambda}, \mathcal{D}) \cdot \text{rec}(\vec{\lambda}, \mathcal{D})}{\beta^2 \cdot \text{prec}(\vec{\lambda}, \mathcal{D}) + \text{rec}(\vec{\lambda}, \mathcal{D})}. \quad (2.5)$$

The well-established F_1 measure is then

$$F_1(\vec{\lambda}, \mathcal{D}) = \frac{2 \cdot \text{prec}(\vec{\lambda}, \mathcal{D}) \cdot \text{rec}(\vec{\lambda}, \mathcal{D})}{\text{prec}(\vec{\lambda}, \mathcal{D}) + \text{rec}(\vec{\lambda}, \mathcal{D})}. \quad (2.6)$$

For application in named entity recognition, it can be intuitive to specify that measure directly on true positives, false positives, and false negatives:

$$F_1(\text{TP}, \text{FP}, \text{FN}) = \frac{2 \cdot \text{TP}}{2 \cdot \text{TP} + \text{FN} + \text{FP}}. \quad (2.7)$$

The discrepancy between accuracy and F_β measures is based on the fact that the first is not differentiating between false positives and false negatives nor between true positives and true negatives while the latter does by incorporating recall and precision. This is especially important when the classes are not similarly distributed.⁴

Measures in Text Segmentation

For segmentation tasks, some special properties hold. An example for an input sequence and possible output sequences is shown in Table 2.5 taken from data of Wilbur, Smith, and Tanabe (2007).

Assuming \vec{y}^* to be the correct segmentation and \vec{y}' to be the predicted sequence, the result is 1 TP (true positive), 1 FN (false negative) and 1 FP (false positive). Only predicting the first segment and not the second one leads to a better result with 1 TP and 1 FN (as \vec{y}' in Table 2.5). This is a reason why it is easier to get a high precision than a high recall. Given a predicted sequence and confidence scores, it is therefore easy to increase precision by removing unconfident entities. But it can easily be seen that adding entities to a result is not straight-forward, as searching for candidates is necessary.

⁴ As an example, assuming two classes A and B and 99 examples from A and 1 from B leads to an accuracy of 99% always guessing A.

$\vec{x} =$	(or	chicken	beta-actin	(cBA)	gene	were	injected) ^T
$\vec{y}^* =$	(O	<u>B</u>	<u>I</u>	O	<u>B</u>	O	O	O	O) ^T
$\vec{y}' =$	(O	<u>B</u>	<u>I</u>	O	O	O	O	O	O) ^T
$\vec{y}'' =$	(O	<u>B</u>	<u>I</u>	<u>B</u>	<u>I</u>	<u>I</u>	<u>I</u>	O	O) ^T

Table 2.5: Named entity recognition example input sequence with possible output sequences. For better perceptibility, segments have been underlined additionally. The correct sequence is \vec{y}^* , \vec{y}' and \vec{y}'' are possible predictions. (snippet from abstract of Lu, Chen, et al. (1992), annotations from Wilbur, Smith, and Tanabe (2007))

Estimating the Generalization Error

To estimate the generalization error of the whole workflow, two methods are used throughout this thesis. *k-fold Cross-Validation* divides the data \mathcal{D} into k subsets of approximately equal size. The training workflow and evaluation is then repeated k times on $k - 1$ subsets and tested on the remaining subset. The average performance and standard deviation gives information about the generalizability. The special case of $k = |\mathcal{D}|$ is called *Leave-One-Out*.

In many applications *Bootstrapping* (Efron and Tibshirani, 1993) performs better, but is then computationally more expensive (Kohavi, 1995). From the data \mathcal{D} , $|\mathcal{D}|$ training examples are sampled with replacement. The remaining examples are used as test. This sub-sampling is repeated, depending on the required accuracy.

2.9 Visualization and Presentation 8b

The final step of the workflow, following a successful evaluation, is the application of the model to new data to extract unseen information. As introduced in Section 1.3.4, several applications build on the fundament of named entity recognition. The named entities are not only useful for further computations but documents enriched with informations from those can provide a reader with valuable additional information speeding up the process of interpretation and understanding of a document. Three applications should be highlighted in the following, as they follow different ideas.

The web service Reflect⁵ highlights protein and small molecule names in web sites (Pafilis, O'Donoghue, et al., 2009). A browser plugin can be used to directly access the service from a web site. Entities are highlighted in the web site and a mouse-over pop-up shows additional information. A screenshot⁶ is shown in Figure 2.5.

A similar approach is to augment documents in the Portable Document Format (PDF, Klinger, Pesch, et al. (2009a); Klinger, Pesch, et al. (2009b)). Currently, full text documents are read online or, mostly, read in the original layout the reader is used to. The presentation of full text is an important issue in contrast to showing abstracts. Understanding a full

⁵ <http://reflect.ws/>

⁶ Reflect applied on the Wikipedia site http://en.wikipedia.org/wiki/Sonic_hedgehog

The screenshot shows a Mozilla Firefox browser window displaying the Wikipedia article for "Sonic hedgehog". The Reflect web service is overlaid on the page, providing additional information about the protein. The overlay includes a search bar, a protein entry for "Shh (ENSMUSP00000002708)" from "M. musculus", and a 3D ribbon structure of the signaling domain of the murine Sonic hedgehog protein (PDB 1vhh). The overlay also shows a sequence viewer with the sequence "MLLLARCFLVILASSLLVCPGLACGPRGFGKRRHPKLTPLAYKQ" and a list of available structures and identifiers.

Figure 2.5: Reflect web service augmenting Wikipedia website.

article is strongly supported by the structure (Klinger, Pesch, et al., 2009b). Therefore, the PDF augments (Klinger, Pesch, et al., 2009a) takes a PDF file, extracts the plain text and processes it with named entity recognition (e.g. the applications presented in Part II as well as ProMiner (Hanisch, Fundel, et al., 2004)). The extracted named entities are then highlighted in a separate layer such that the original document can be viewed unchanged as well. Clicking on an entity shows a list of related informative links. The end of the document is enriched with short statistics of the entities in the present article. An example is depicted in Figure 2.6.

SCAView⁷ (Hofmann-Apitius, Fluck, et al., 2008) is a semantic search engine for MEDLINE abstracts and presentation application for named entity recognition. It performs a full text search and ranks the entities in the retrieved abstracts with respect to their relative entropy. Therefore, it can be used for information retrieval as well as information extraction. A screenshot is shown in Figure 2.7.

⁷ <http://www.scaiview.com/>

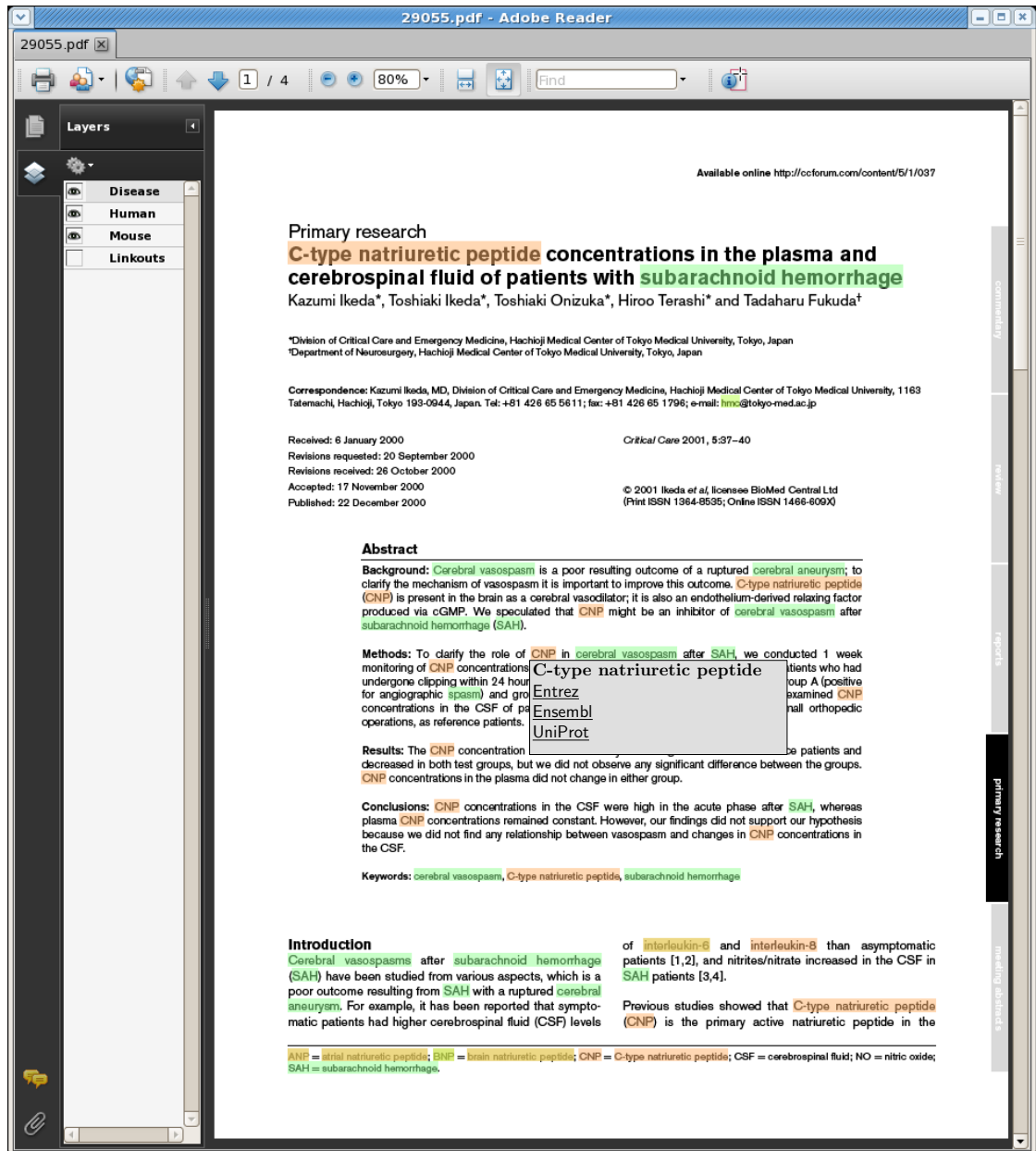


Figure 2.6: Example of a PDF document augmented with information about detected entities.

The screenshot displays the SCAIView web interface. The browser address bar shows the URL <http://www.scaiview.com/scaiview/>. The page title is "SCAIVIEW" and the subtitle is "SCAIVIEW: The Knowledge Discovery Framework".

The search bar contains the query "headache". Below the search bar, there are navigation tabs for "Help", "Documents", "Entity", "Analysis", and "About". The "Entity" tab is active, and the search results are displayed on page 0 of 19 documents, with 10 documents per page. The search took 36 ms.

On the left side, there is a sidebar with a tree view of entity classes. The "Human Genes / Proteins" category is expanded, showing various sub-classes such as "Chromosomal Location", "STS Marker", "non Normalized SNP", "Normalized SNP", "Normalized CRF SNP", "Drug Names", "+ IUPAC-like", "OMIM Reference", "Corpora", "Epigenetics", "Human miRNA", "Arabidopsis Genes", "Mouse Genes", "Interaction Verbs", "MeSH Disease", "Relations", "@neurIST Ontology", "Numerical Relations", "GO Component", "GO Function", "GO Process", "Statistical-Numeric Relations", and "Statistical-Unit Relations".

Below the search bar, there are three buttons: "Toggle Abstracts", "Select All Entity Classes", and "Deselect All Entity Classes". A list of entity classes is shown with checkboxes, including "Chromosomal Locations", "Drug Names", "Protein/Gene", "STS Marker", "OMIM Reference", "@neurIST", "non Normalized SNP", "Normalized SNP", "MeSH Disease", "Relations", "Interactions", "Corpora", "CRF SNP", "IUPAC", "Epigenetics", "Arabidopsis", "Mouse", "Human miRNA", "GO Component", "GO Function", "GO Process", "Numerical-Statistical Relations", and "Numerical-Units Relations".

The search results are displayed as a list of documents. The first document is titled "1. 5-hydroxytryptamine1B receptor and triptan response in migraine, lack of association with common polymorphisms." and is from PubMed. The authors listed are Daniela Velati, Michele Viana, Stefania Cresta, Paola Mantegazza, Lucia Testa, Diego Bettucci, Maurizio Rinaldi, Grazia Sances, Cristina Tassorelli, Giuseppe Nappi, Pier Luigi Canonico, Emilia Martignoni, Armando A Genazzani. The date is 2008-02, the journal is European journal of pharmacology, SciMag: 0.322, and the affiliation is DISCAFF and DFB Center, Via Bovio, 6, 28100 Novara, Italy.

The abstract text is as follows:

Triptans mediate vasoconstriction of meningeal vessels via stimulation of vascular 5-hydroxytryptamine (5-HT)_{1B} receptors. These drugs are recommended for acute treatment in patients with moderate-to-severe migraine attacks and in those patients with mid-to-moderate headache that are not controlled adequately by other agents. Yet, approximately 25% of all migraine users and 40% of all attacks do not respond to triptan treatment. Among the hypothesis to explain this is the possibility that genetic single nucleotide polymorphisms that alter the receptor, for example changing the transcriptional rate and therefore the amount of target protein might change the clinical response to these drugs. In the present contribution, we therefore decided to evaluate whether single nucleotide polymorphisms on the 5-HT_{1B} gene might contribute to inter-individual variability in clinical responses to triptans. Two polymorphisms in the promoter region of the 5-HT_{1B} receptor (T-261G and A-161T) and the synonymous variation G861C in the coding region were genotyped by restriction fragment length polymorphism in 105 migraine patients. In our sample population, 71% of patients responded to triptans. Allelic and diplotype frequencies were not significantly different between responders and non-responders. On the other hand, extrapolation of in vitro data on promoter activity would suggest that patients with higher copy number of receptors respond slightly better. Our data therefore do not support the involvement of 5-HT_{1B} single nucleotide polymorphisms in mediating the inter-individual variability to triptans.

Figure 2.7: Screenshot of the search engine SCAIView.

Chapter 3

Graphical Models and Conditional Random Fields

3.1 Introduction and Previous Work

This chapter gives an overview of the basic theory behind conditional random fields and illustrates how these are related to other probabilistic models. It is based on the report by Klinger and Tomanek (2007).

Classification is known as the assignment of a class $y \in \mathcal{Y}$ to an observation $x \in \mathcal{X}$. This task can be approached with probability theory by specifying a probability distribution to select the most likely class y for a given observation x .

A well-known example (Russell and Norvig, 2003) is the classification of weather observations into categories, such as *good* or *bad*, $\mathcal{Y} = \{\textit{good}, \textit{bad}\}$. For instance, let x be the weather observation on a special day, $\mathcal{X} = \{\textit{Monday}, \textit{Tuesday}, \dots, \textit{Sunday}\}$, x can be described by a set of features such as $f_{\textit{cloudy}}(x) = 1$, if and only if it is cloudy on day x , $f_{\textit{cloudy}}(x) = 0$ otherwise. Other features might be $f_{\textit{sunny}}$ or $f_{\textit{rainy}}$.

Modeling all dependencies in a probability distribution is typically very complex due to interdependencies between features. The naïve Bayes assumption of all features being conditionally independent is an approach to address this problem (see Section 3.2.1). In nearly all probabilistic models independence assumptions are made for some variables to make necessary computations manageable.

In the structured learning scenario, multiple and typically interdependent class and observation variables are considered which implicates an even higher complexity in the probability distribution. This is the case for image or music data as well as for natural language text. As for images, pixels near to each other are very likely to have a similar color or hue. In music, different succeeding notes follow special laws, they are not independent, especially when they sound simultaneously. Otherwise, music would not be pleasant to the ear. In text, words are not an arbitrary accumulation, the order is important and grammatical constraints hold.

A typical task in natural language processing is known as text segmentation, named entity recognition can be seen as such. The IOB format introduced in Section 2.4 is an approach for segmenting text token sequences into entities of classes of interest and non-entities.

One approach for modeling such linear sequence structures, as can be found in natural language text, are hidden Markov models (Rabiner, 1989). For the sake of complexity reduction, strong independence assumptions between the observation variables are made. This impairs the accuracy of the model. Conditional random fields (CRFs, Lafferty, McCallum, and Pereira (2001)) are developed exactly to fill that gap. While CRFs make similar assumptions on the dependencies among the class variables, no assumptions on the dependencies among

observation variables are made (see Section 3.4).

CRFs have found application in many domains which deal with structured data. Despite the frequent application of linear-chain CRFs, other underlying structures have been used to model the respective dependencies among the class variables. Especially in natural language processing, CRFs are currently a state-of-the-art technique for many of its subtasks including basic text segmentation (Tomanek, Wermter, and Hahn, 2007a), part-of-speech tagging (Lafferty, McCallum, and Pereira, 2001), shallow parsing (Sha and Pereira, 2003), the resolution of elliptical noun phrases (Buyko, Tomanek, and Hahn, 2007). CRFs have been proven to be very useful in named entity recognition, for example on documents from the biomedical domain (Settles, 2004; McDonald and Pereira, 2005; McDonald, Winters, et al., 2004). Additional applications will be shown in Part II. Furthermore, CRFs have been applied to gene prediction (DeCaprio, Vinson, et al., 2007), image labeling (He, Zemel, and Carreira-Perpinan, 2004), and object recognition (Quattoni, Collins, and Darrell, 2005), in telematics for intrusion detection (Gupta, Nath, and Ramamohanarao, 2007), and sensor data management (Zhang, Aberdeen, and Vishwanathan, 2007).

This chapter is organized as follows. In Section 3.2, a brief overview of three classical and well-established probabilistic models is given: Naïve Bayes, hidden Markov, and maximum entropy. The relations between and graphical representations of these different approaches are discussed in Section 3.3. In Section 3.4, the basic concepts of CRFs (3.4.1) are explained. This section is mainly focused on the special case of linear-chain CRFs (3.4.2) and methods for training (3.4.2) and inference (3.4.2). Moreover, building upon these explanations, a generalization to arbitrarily structured CRFs is given in Section 3.4.3.

For further reading, the tutorials of Wallach (2004) and Sutton and McCallum (2007) are recommended. They approach the theory behind CRFs from a different perspective.

3.2 Probabilistic Models

In this section, some well-known probabilistic models are discussed. conditional random fields are founded on the underlying ideas and concepts of these approaches.

The naïve Bayes model is an approach to classify single class variables in dependence of several feature values. In that model, the input values are assumed to be conditionally independent. It is a so called generative approach, modeling the joint probability $p(y, \vec{x})$ of the input values \vec{x} and the class variable y . The hidden Markov model is an extension to the naïve Bayes model for sequentially structured data also representing the dependencies of the variables \vec{x} and \vec{y} as a joint probability distribution.

Modeling joint probabilities has disadvantages due to computational complexity. The maximum entropy model, in contrast, is based on modeling the conditional probability $p(y|x)$. Like the naïve Bayes model, it is an approach to classify a single class variable in dependence of several feature values. The difference is the consideration of conditional probability $p(y|x)$ instead of the joint probability.

While a hidden Markov model is a sequential extension to the naïve Bayes model, condi-

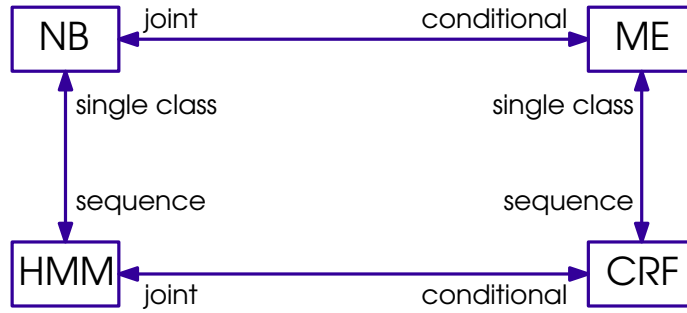


Figure 3.1: Overview of probabilistic models: naïve Bayes model (NB), hidden Markov model (HMM), maximum entropy model (ME), and conditional random field (CRF). Depicted aspects are joint versus conditional probability, single class prediction versus prediction on sequential data.

tional random fields can be understood as a sequential extension to the maximum entropy model. Both maximum entropy models and conditional random fields are known as discriminative approaches.

A graphical comparison of these models is given in Figure 3.1. In the following, an explanation of these models is given based on Bishop (2006) and Russell and Norvig (2003).

3.2.1 Naïve Bayes

A conditional probability model is a probability distribution $p(y|\vec{x})$ with an input vector $\vec{x} = (x_1, \dots, x_m)$, where x_i ($1 \leq i \leq m$) are features and y is the class variable to be predicted. That probability can be formulated with Bayes' law:

$$p(y|\vec{x}) = \frac{p(y)p(\vec{x}|y)}{p(\vec{x})}. \quad (3.1)$$

The denominator $p(\vec{x})$ is not important for classification as it can be understood as a normalization constant which can be computed by considering all possible values for y . The numerator can also be written as a joint probability

$$p(y)p(\vec{x}|y) = p(y, \vec{x}), \quad (3.2)$$

which can be too complex to be computed directly (especially when the number of components in \vec{x} is high). A general decomposition of that probability can be formulated applying the chain rule $p(x_1, \dots, x_m) = \prod_{i=2}^m p(x_i|x_{i-1}, \dots, x_1)$:

$$p(y, \vec{x}) = p(y) \prod_{i=2}^m p(x_i|x_{i-1}, \dots, x_1, y). \quad (3.3)$$

In practice, it is often assumed, that all input variables x_i are conditionally independent of each other which is known as the naïve Bayes assumption (Hand and Yu, 2001). That means

that $p(x_i|y, x_j) = p(x_i|y)$ holds for all $i \neq j$. Based on this simplification, a model known as the naïve Bayes classifier is formulated as

$$p(y|\vec{x}) \propto p(y, \vec{x}) = p(y) \prod_{i=1}^m p(x_i|y). \quad (3.4)$$

This probability distribution is less complex than the one formulated in Equation 3.3. Dependencies between the input variables \vec{x} are not modeled, probably leading to an imperfect representation of the real world. Nevertheless, the naïve Bayes model performs surprisingly well in many real world applications, such as email classification (Androutsopoulos, Koutsias, et al., 2000; Androutsopoulos, Paliouras, et al., 2000; Kiritchenko and Matwin, 2001).

3.2.2 Hidden Markov Models

In the naïve Bayes model, only single output variables are considered. To predict a sequence of class variables $\vec{y} = (y_1, \dots, y_n)$ for an observation sequence $\vec{x} = (x_1, \dots, x_n)$, a simple sequence model can be formulated as a product over single naïve Bayes models. Dependencies between single sequence positions are not taken into account. Note, that in contrast to the naïve Bayes model there is only one feature at each sequence position, typically the identity of the respective observation:

$$p(\vec{y}, \vec{x}) = \prod_{i=1}^n p(y_i) \cdot p(x_i|y_i). \quad (3.5)$$

Each observation x_i depends only on the class variable y_i at the respective sequence position¹. Due to this independence assumption, transition probabilities from one step to another are not included in this model. This assumption is hardly met in practice, resulting in limited performance. It is reasonable to assume that there are dependencies between the observations at consecutive sequence positions. To model this, state transition probabilities are added:²

$$p(\vec{y}, \vec{x}) = \prod_{i=0}^n p(y_i|y_{i-1})p(x_i|y_i). \quad (3.6)$$

This leads to the well-known *hidden Markov model* (HMM, Rabiner (1989))

$$P(\vec{x}) = \sum_{y \in \mathcal{Y}} \prod_{i=0}^n p(y_i|y_{i-1})p(x_i|y_i), \quad (3.7)$$

where \mathcal{Y} is the set of all possible label sequences \vec{y} .

Dependencies between output variables \vec{y} are modeled. A shortcoming is the assumption of conditional independence (see Equation 3.6) between the input variables \vec{x} due to complexity issues. As we will see later, CRFs address exactly this problem.

¹ Recall that x_i are different observations at different sequence positions. In Equation 3.4, in contrast, x_i specifies different observations at the same position.

² The initial probability distribution is assumed to be included as $p(y_0|y_{-1}) = p(y_0)$

3.2.3 Maximum Entropy Model

The two models introduced in Section 3.2.1 and 3.2.2 are trained to maximize the joint likelihood. In the following, the *maximum entropy model*³ is discussed in more detail as it is fundamentally related to CRFs. The maximum entropy model is a conditional probability model. It is based on the *Principle of maximum entropy* (Jaynes, 1957) which states that if incomplete information about a probability distribution is available, the only unbiased assumption that can be made is a distribution which is as uniform as possible given the available information. Under this assumption, the proper probability distribution is the one which maximizes the entropy given the constraints from the training material. For the conditional model $p(y|x)$ the conditional entropy $H(y|x)$ (Korn and Korn, 2000; Bishop, 2006) is applied, which is defined as

$$H(y|x) = - \sum_{(x,y) \in \mathcal{Z}} p(y,x) \log p(y|x). \quad (3.8)$$

The set $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$ consists of \mathcal{X} , the set of all possible input variables x , and \mathcal{Y} , the set of all possible output variables y . Note that \mathcal{Z} contains not only the combinations of x and y occurring in the training data, but all possible combinations.

The basic idea behind maximum entropy models is to find the model $p^*(y|x)$ which on the one hand has the largest possible conditional entropy but is on the other hand still consistent with the information from the training material. The objective function, later referred to as *primal problem*, is thus

$$p^*(y|x) = \operatorname{argmax}_{p(y|x) \in P} H(y|x), \quad (3.9)$$

where P is the set of all models consistent with the training material. What is meant with “consistent” will be explained in detail later on page 36.

The training material is represented by features. Here, these are defined as binary-valued functions $f_i(x, y) \in \{0, 1\}$ ($1 \leq i \leq m$) which depend on both the input variable x and the class variable y . An example for such a function is:

$$f_i(x, y) = \begin{cases} 1 & \text{if } y = \text{name and } x = \text{Mister} \\ 0 & \text{otherwise} \end{cases} \quad (3.10)$$

The expected value of each feature f_i is estimated from the empirical distribution $\tilde{p}(x, y)$. The empirical distribution is obtained by simply counting how often the different values of the variables occur in the training data:

$$\tilde{E}(f_i) = \sum_{(x,y) \in \mathcal{Z}} \tilde{p}(x, y) f_i(x, y). \quad (3.11)$$

³ These explanations of maximum entropy models are based on the maximum entropy tutorial by Berger, Pietra, and Pietra (1996).

All possible pairs (x, y) are taken into account here. As the empirical probability for a pair (x, y) which is not contained in the training material is 0, $\tilde{E}(f_i)$ can be rewritten as

$$\tilde{E}(f_i) = \frac{1}{N} \sum_{(x,y) \in \mathcal{T}} f_i(x, y). \quad (3.12)$$

The size of the training set is $N = |\mathcal{T}|$. Thus, $\tilde{E}(f_i)$ can be calculated by counting how often a feature f_i is found with value 1 in the training data $\mathcal{T} \subseteq \mathcal{X}^4$ and dividing that number by the size N of the training set.

Analogously to Equation 3.11, the expected value of a feature on the model distribution is defined as

$$E(f_i) = \sum_{(x,y) \in \mathcal{X}} p(x, y) f_i(x, y). \quad (3.13)$$

In contrast to Equation 3.11 (the expected value on the empirical distribution), the model distribution is taken into account here. Of course, $p(x, y)$ cannot be calculated in general because the number of all possible $x \in \mathcal{X}$ can be enormous. This can be addressed by rewriting $E(f_i)$ by

$$E(f_i) = \sum_{(x,y) \in \mathcal{X}} p(x) p(y|x) f_i(x, y) \quad (3.14)$$

and substituting $p(x)$ with the empirical distribution $\tilde{p}(x)$. This is an approximation to make the calculation of $E(f_i)$ possible (see Lau, Rosenfeld, and Roukos (1993) for a more detailed discussion). This results in

$$E(f_i) \approx \sum_{(x,y) \in \mathcal{X}} \tilde{p}(x) p(y|x) f_i(x, y), \quad (3.15)$$

which can (analogously to Equation 3.12) be transformed into

$$E(f_i) = \frac{1}{N} \sum_{x \in \mathcal{T}} \sum_{y \in \mathcal{Y}} p(y|x) f_i(x, y). \quad (3.16)$$

Only x values occurring in the training data are considered ($x \in \mathcal{T}$) while all possible y values are taken into account ($y \in \mathcal{Y}$). In many applications the set \mathcal{Y} typically contains only a small number of variables. Thus, summing over y is possible here and $E(f_i)$ can be calculated efficiently.

Equation 3.9 postulates that the model $p^*(y|x)$ is consistent with the evidence found in the training material. That means, for each feature f_i its expected value on the empirical distribution must be equal to its expected value on the particular model distribution, these are the first m constraints

$$E(f_i) = \tilde{E}(f_i). \quad (3.17)$$

⁴ In fact, \mathcal{T} is a multiset as the elements from \mathcal{X} can be contained several times. So the subset relation $\mathcal{T} \subseteq \mathcal{X}$ only holds in a special case.

Another constraint is to have a proper conditional probability ensured by

$$p(y|x) \geq 0 \text{ for all } x, y \quad \sum_{y \in \mathcal{Y}} p(y|x) = 1 \text{ for all } x. \quad (3.18)$$

Finding $p^*(y|x)$ under these constraints can be formulated as a constrained optimization problem. For each constraint a Lagrange multiplier λ_i is introduced. This leads to the following Lagrange function $\Lambda(p, \vec{\lambda})$:

$$\Lambda(p, \vec{\lambda}) = \underbrace{H(y|x)}_{\substack{\text{primal problem} \\ \text{Equation 3.9}}} + \sum_{i=1}^m \lambda_i \underbrace{(E(f_i) - \tilde{E}(f_i))}_{\substack{\doteq 0 \\ \text{constraints from} \\ \text{Equation 3.17}}} + \lambda_{m+1} \underbrace{\left(\sum_{y \in \mathcal{Y}} p(y|x) - 1 \right)}_{\substack{\doteq 0 \\ \text{constraint from} \\ \text{Equation 3.18}}}$$

(3.19)

This is maximized to get the model formulation $p_{\vec{\lambda}}^*(y|x)$ in Equation 3.28 on page 38. In the following, a detailed derivation is given.

In the same manner as done for the expectation values in Equation 3.15, $H(y|x)$ is approximated:

$$H(y|x) \approx - \sum_{(x,y) \in \mathcal{X}} \tilde{p}(x) p(y|x) \log p(y|x). \quad (3.20)$$

The derivation of Equation 3.20 is given by

$$\begin{aligned} \frac{\partial}{\partial p(y|x)} H(y|x) &= -\tilde{p}(x) \left(\log p(y|x) + \frac{p(y|x)}{p(y|x)} \right) \\ &= -\tilde{p}(x) (\log p(y|x) + 1). \end{aligned} \quad (3.21)$$

The derivation of the first m constraints in Equation 3.19 is

$$\begin{aligned} &\frac{\partial}{\partial p(y|x)} \sum_{i=1}^m \lambda_i (E(f_i) - \tilde{E}(f_i)) = \\ &= \frac{\partial}{\partial p(y|x)} \sum_{i=1}^m \lambda_i \left(\sum_{(x,y) \in \mathcal{X}} \tilde{p}(x) p(y|x) f_i(x, y) - \left(\sum_{(x,y) \in \mathcal{X}} \tilde{p}(x, y) f_i(x, y) \right) \right) \\ &= \sum_{i=1}^m \lambda_i \tilde{p}(x) f_i(x, y). \end{aligned} \quad (3.22)$$

The complete derivation of the Lagrange function from Equation 3.19 is then:

$$\frac{\partial}{\partial p(y|x)} \Lambda(p, \vec{\lambda}) = -\tilde{p}(x)(1 + \log p(y|x)) + \sum_{i=1}^m \lambda_i \tilde{p}(x) f_i(x, y) + \lambda_{m+1}. \quad (3.23)$$

Equating this term to 0 and solving by $p(y|x)$ leads to

$$\begin{aligned} 0 &= -\tilde{p}(x)(1 + \log p(y|x)) + \sum_{i=1}^m \lambda_i \tilde{p}(x) f_i(x, y) + \lambda_{m+1} \\ \tilde{p}(x)(1 + \log p(y|x)) &= \sum_{i=1}^m \lambda_i \tilde{p}(x) f_i(x, y) + \lambda_{m+1} \\ 1 + \log p(y|x) &= \sum_{i=1}^m \lambda_i f_i(x, y) + \frac{\lambda_{m+1}}{\tilde{p}(x)} \\ \log p(y|x) &= \sum_{i=1}^m \lambda_i f_i(x, y) + \frac{\lambda_{m+1}}{\tilde{p}(x)} - 1 \\ p(y|x) &= \exp\left(\sum_{i=1}^m \lambda_i f_i(x, y)\right) \cdot \exp\left(\frac{\lambda_{m+1}}{\tilde{p}(x)} - 1\right). \end{aligned} \quad (3.24)$$

The second constraint in Equation 3.18 is given as

$$\sum_{y \in \mathcal{Y}} p(y|x) = 1. \quad (3.25)$$

Substituting Equation 3.24 into 3.25 results in

$$\begin{aligned} \sum_{y \in \mathcal{Y}} \exp\left(\sum_{i=1}^m \lambda_i f_i(x, y)\right) \cdot \exp\left(\frac{\lambda_{m+1}}{\tilde{p}(x)} - 1\right) &= 1 \\ \exp\left(\frac{\lambda_{m+1}}{\tilde{p}(x)} - 1\right) &= \frac{1}{\sum_{y \in \mathcal{Y}} \exp\left(\sum_{i=1}^m \lambda_i f_i(x, y)\right)}. \end{aligned} \quad (3.26)$$

Substituting Equation 3.26 back into Equation 3.24 results in

$$p(y|x) = \exp\left(\sum_{i=1}^m \lambda_i f_i(x, y)\right) \cdot \frac{1}{\sum_{y \in \mathcal{Y}} \exp\left(\sum_{i=1}^m \lambda_i f_i(x, y)\right)}. \quad (3.27)$$

This is the general form the model needs to have to meet the constraints. The maximum entropy model can then be formulated as

$$p_{\vec{\lambda}}^*(y|x) = \frac{1}{Z_{\vec{\lambda}}(x)} \exp\left(\sum_{i=1}^m \lambda_i f_i(x, y)\right), \quad (3.28)$$

and $Z_{\vec{\lambda}}(x)$ then is

$$Z_{\vec{\lambda}}(x) = \sum_{y \in \mathcal{Y}} \exp \left(\sum_{i=1}^m \lambda_i f_i(x, y) \right). \quad (3.29)$$

This formulation of a conditional probability distribution as a log-linear model and a product of exponentiated weighted features is discussed from another perspective in Section 3.3. In Section 3.4, the similarity of conditional random fields, which are also log-linear models, with the conceptually closely related maximum entropy models becomes evident.

For a more detailed discussion of maximum entropy models and related approaches, the book by Pearl (1988) and the maximum entropy tutorial by Berger, Pietra, and Pietra (1996) are recommended.

In this section, two kinds of probabilistic models have been introduced. On the one hand *generative* models, such as naïve Bayes and hidden Markov models, which are based on joint probability distributions. As can be seen in Equation 3.4 and 3.6, in such models the observation variables x_i topologically precede, also called “generate”, the input variables y . This characteristic can be seen in the graphical representation (see Section 3.3), of these models in Figures 3.3a and 3.4a. On the other hand, *discriminative* models, such as maximum entropy models, are based on conditional probability distributions. In the next section, both groups of models are reviewed from a different perspective, their graphical representations.

3.3 Graphical Representation

The underlying probability distributions of probabilistic models can be represented in a graphical form, this is why they are often called probabilistic *graphical* models. The following explanations are inspired by Bishop (2006).

A *probabilistic graphical model* is a diagrammatic representation of a probability distribution. In such a graph there is a node for each random variable. The absence of an edge between two variables represents *conditional independence* between those variables. Conditional independence means that two random variables a and b are independent given a third random variable c if they are independent in their conditional probability distribution, formally $p(a, b|c) = p(a|b)p(b|c)$.⁵ From such graphs, also called *independency graphs*, one can read the conditional independence properties of the underlying distribution. Note that a fully connected independency graph does not contain any information about the probability distribution, only the absence of edges is informative: Conditional independence in the probability distribution does not induce the absence of the edge in the graph.⁶

⁵ Note that in contrast two random variables a and b are *statistically* independent if and only if $p(a, b) = p(a)p(b)$.

⁶ A graph is called dependency graph if the independence of two variables implicates separability of the corresponding nodes in the graph (Beierle and Kern-Isberner, 2003, pp. 337f.).

Conditional independence is an important concept as it can be used to decompose complex probability distributions into a product of factors, each consisting of the subset of corresponding random variables. This concept makes complex computations (which are for example necessary for learning or inference) much more efficient. In general, the *decomposition*, in fact a *factorization* of a probability distribution, is written as the product of its factors Ψ_s , with \vec{v}_s the subset of the respective random variables constituting such a factor

$$p(\vec{v}) = \prod_s \Psi_s(\vec{v}_s). \quad (3.30)$$

Let $G = (V, E)$ be a graph with vertexes V and edges E . In an *independency graph* (for example the one shown in Figure 3.2a), the vertexes $V = X \cup Y$, with X and Y sets of random variables, are depicted by circles. X will typically be considered as the set of input or observation variables (shaded circles), and Y as the set of output variables (empty nodes). An independency graph can have directed or undirected edges, depending on the kind of graphical model it represents (see Sections 3.3.1 and 3.3.2).

In a *factor graph* (Kschischang, Frey, and Loeliger, 2001), such as the one shown in Figure 3.2b, the circles represent, as in an independency graph, the random variables of the underlying distribution, depicted by circles. Further, a factor graph contains factor nodes, depicted by small, filled squares, which represent the factors Ψ_s (compare with Equation 3.30).⁷ In a factor graph, the edges are always undirected, linking the random variables to the factor nodes. A factor Ψ_s includes all random variables to which the respective factor node is directly connected by an edge. Thus, a factor graph represents more explicitly the factorization of the underlying probability distribution. Independency graphs of both directed and undirected graphical models can be transformed into factor graphs.

As an example, assume a probability distribution $p(x_1, x_2, y)$ to factorize as $p(\vec{x}) = p(x_1) \cdot p(x_2) \cdot p(y|x_1, x_2)$. It has the factors $\Psi_1(x_1) = p(x_1)$, $\Psi_2(x_2) = p(x_2)$, and $\Psi_3(y) = p(y|x_1, x_2)$. Here, x_1 and x_2 are conditionally independent given y . Figure 3.2 shows an independency graph and a factor graph representing this distribution.

In the following, directed and undirected graphical models are discussed. Naïve Bayes and hidden Markov models fall into the first group, the maximum entropy model falls into the second group of graphical models.

3.3.1 Directed Graphical Models

A joint distribution $p(\vec{v})$ can be factorized into the product of conditional distributions for each node v_k , so that each such conditional distribution is conditioned on its set of parent nodes v_k^p :

$$p(\vec{v}) = \prod_{k=1}^K p(v_k | v_k^p) \quad (3.31)$$

⁷ A factor graph consists of two sets of nodes: variable and factor nodes. There are no edges between nodes of the same set, so a factor graph is always bipartite.

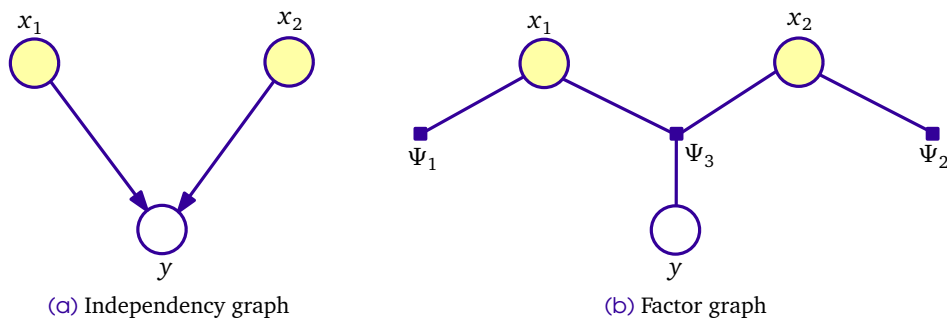


Figure 3.2: Directed graphical model.

This is the same kind of factorization as shown in Figure 3.2 for the example distribution $p(x_1, x_2, y)$. As another example, take the naïve Bayes classifier which is discussed in Section 3.2.1. Figure 3.3 graphically represents such a model with three observation variables. The corresponding probability distribution factorizes as $p(y, x_1, x_2, x_3) = p(y) \cdot p(x_1|y) \cdot p(x_2|y) \cdot p(x_3|y)$, following the naïve Bayes assumption. Analogously, Figure 3.4 shows an HMM classifier for a sequence of three input variables x_1, x_2, x_3 . The factorization is $p(x_1, x_2, x_3, y_1, y_2, y_3) = \Psi_1(y_1) \cdot \Psi_2(x_1, y_1) \cdot \Psi_3(x_2, y_2) \cdot \Psi_4(x_3, y_3) \cdot \Psi_5(y_1, y_2) \cdot \Psi_6(y_2, y_3)$ which corresponds⁸ to the HMM (see Equation 3.6).

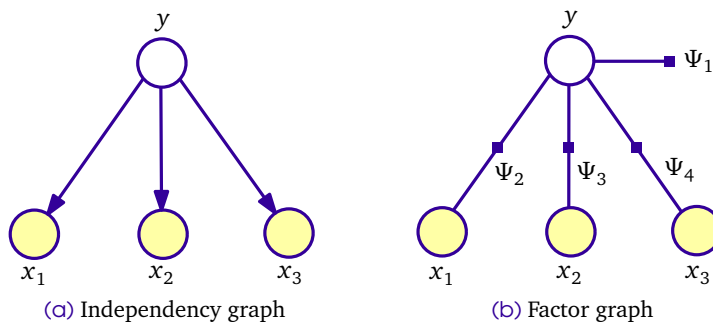


Figure 3.3: Naïve Bayes classifier.

3.3.2 Undirected Graphical Models

A probability distribution can be represented by an undirected graphical model using a product of non-negative functions of the maximal cliques of G . The factorization is performed in a way that conditionally independent nodes do not appear within the same factor, that

⁸ A dedicated start value $y_0 = \perp$ is assumed, so that $\Psi(y_1) = p(y_0 = \perp, y_1)$.

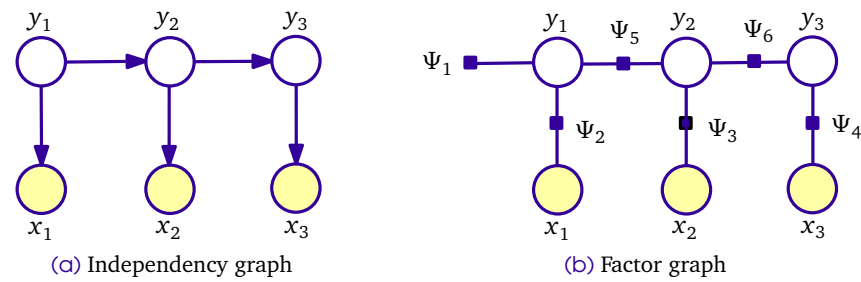


Figure 3.4: Independence and factor graph for the hidden Markov model.

means that they belong to different *cliques*:

$$p(\vec{v}) = \frac{1}{Z} \prod_{C \in \mathcal{C}} \Psi_C(\vec{v}_C). \quad (3.32)$$

The factors $\Psi_C \geq 0$ are so-called *potential functions* of the random variables \vec{v}_C within a clique $C \in \mathcal{C}$.

A potential function may be any arbitrary function, not necessarily probability functions. This is in contrast to directed graphs where a joint distribution is factorized into a product of conditional distributions. Thus, normalization of the product of potential functions is necessary to achieve a proper probability measure. This is yielded by a normalization factor Z . Calculating Z is one of the main challenges during parameter learning as summing over all possible variables is necessary:

$$Z = \sum_{\vec{v}} \prod_{C \in \mathcal{C}} \Psi_C(\vec{v}_C). \quad (3.33)$$

In Section 3.2.3 the maximum entropy model was discussed which can be formulated by such a product of non-negative potential functions (compare to Equation 3.28)

$$p_{\vec{\lambda}}(y|x) = \frac{1}{Z_{\vec{\lambda}}(x)} \prod_{i=1}^m \exp(\lambda_i f_i(x, y)). \quad (3.34)$$

In such log-linear models, potential functions are formulated as the exponential function of weighted features. Such a formulation is frequently used because it fulfills the requirement of strict positivity of the potential functions. Figure 3.5a shows the independency graph for a maximum entropy classifier with an observation variable x , a corresponding factor graph with three features is shown in Figure 3.5b.

Directed and undirected graphical models differ in the way the original probability distribution is factorized. The factorization into a product of conditional probability distributions as done in a directed graphical model is straight-forward. In undirected graphical models a factorization into arbitrary functions is done. This does not require an explicit specification how the variables are related. But it comes at the expense of having to calculate the normalization factor.

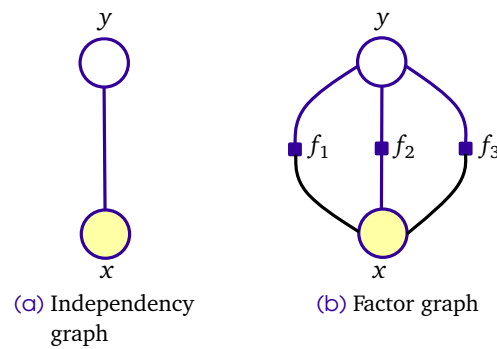


Figure 3.5: Maximum entropy classifier.

3.4 Conditional Random Fields

In the previous section, some well-known probabilistic models were discussed from a mathematical perspective. Moreover, the graphical representation, which characterizes the underlying probability distribution of the model, was shown.

A hidden Markov model can be understood as the sequence version of a naïve Bayes model, instead of single independent decisions, a hidden Markov model represents a linear sequence of decisions. Accordingly, conditional random fields can be understood as the sequence version of maximum entropy models, that means they are discriminative models, too. Furthermore, in contrast to hidden Markov models, conditional random fields are not tied to the linear-sequence structure but can be arbitrarily structured.

In the following, the idea and foundation of conditional random fields is illustrated. First, a general formulation of conditional random fields is given followed by a discussion of a popular form of CRFs, those with a linear sequence structure. A focus are aspects of training and inference. This section closes with a short discussion of arbitrarily structured CRFs, being used in Chapter 9.

3.4.1 Basic Principles

Introduced by Lafferty, McCallum, and Pereira (2001), *conditional random fields* (CRF) are probabilistic models for computing the probability $p(\vec{y}|\vec{x})$ of a possible output $\vec{y} = (y_1, \dots, y_n) \in \mathcal{Y}^n$ given the input $\vec{x} = (x_1, \dots, x_n) \in \mathcal{X}^n$ which is also called the *observation*. A CRF in general can be derived from Equation 3.32:

$$p(\vec{v}) = \frac{1}{Z} \prod_{C \in \mathcal{C}} \Psi_C(\vec{v}_C). \quad (3.35)$$

The conditional probability $p(\vec{y}|\vec{x})$ can be written as

$$\begin{aligned}
 p(\vec{y}|\vec{x}) &= \frac{p(\vec{x}, \vec{y})}{p(\vec{x})} \\
 &= \frac{p(\vec{x}, \vec{y})}{\sum_{\vec{y}'} p(\vec{y}', \vec{x})} \\
 &= \frac{\frac{1}{Z} \prod_{C \in \mathcal{C}} \Psi_C(\vec{x}_C, \vec{y}_C)}{\frac{1}{Z} \sum_{\vec{y}'} \prod_{C \in \mathcal{C}} \Psi_C(\vec{x}_C, \vec{y}'_C)}. \tag{3.36}
 \end{aligned}$$

Reducing by $\frac{1}{Z}$ and taking the denominator as normalization factor leads to the general model formulation of conditional Markov networks, also known as conditional random fields:

$$p(\vec{y}|\vec{x}) = \frac{1}{Z(\vec{x})} \prod_{C \in \mathcal{C}} \Psi_C(\vec{x}_C, \vec{y}_C). \tag{3.37}$$

As described in Section 3.3, Ψ_C are the different factors corresponding to maximal cliques in the independency graph (see Kschischang, Frey, and Loeliger (2001)). See Figure 3.6 for an example of a linear-chain CRF. Each factor corresponds to a potential function which combines different features f_i of the considered part of the observation and the output. As mentioned, the normalization follows from the denominator of Equation 3.36:

$$Z(\vec{x}) = \sum_{\vec{y}'} \prod_{C \in \mathcal{C}} \Psi_C(\vec{x}_C, \vec{y}'_C). \tag{3.38}$$

In practice, factor graphs often use the same parameters for several factors, called *parameter tying*. A *factor template* (also called *clique template*) θ_j consists of parameters $\{\lambda_{jk}\}$, feature functions $\{f_{jk}\}$, and a description of a relationship between variables, yielding a set of tuples $\{(\vec{x}_j, \vec{y}_j)\}$. For each of these variable tuples (\vec{x}_i, \vec{y}_i) that fulfil this relationship, the factor template instantiates a factor that shares $\{\lambda_{jk}\}$ and $\{f_{jk}\}$ with all other instantiations of θ_j (Singh, Schultz, and McCallum, 2009; Sutton and McCallum, 2007). With \mathcal{C} as set of clique templates, the probability distribution is

$$p(\vec{y}|\vec{x}) = \frac{1}{Z(\vec{x})} \prod_{\theta_j \in \mathcal{C}} \prod_{(\vec{x}_i, \vec{y}_i) \in \theta_j} \exp \left(\sum_{k=1}^{K_j} \lambda_{jk} f_{jk}(\vec{x}_i, \vec{y}_i) \right). \tag{3.39}$$

Examples for clique templates are given in Section 3.4.2 and 3.4.3, as well as Chapter 9.

3.4.2 Linear-chain CRFs

A special form of a CRF, which is structured as a linear chain, models the output variables as a sequence (therefore, such model can be seen as emerging from one template). Figure 3.6

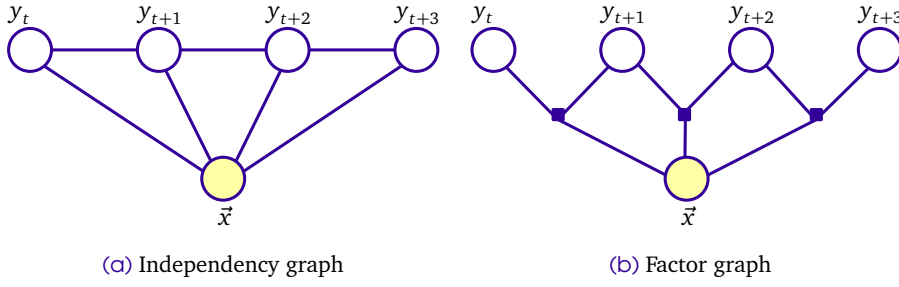


Figure 3.6: Linear chain conditional random field.

shows the respective independency and factor graphs. The CRF introduced in Equation 3.37 can be formulated as

$$p(\vec{y}|\vec{x}) = \frac{1}{Z(\vec{x})} \prod_{j=1}^n \Psi_j(\vec{x}, \vec{y}), \quad (3.40)$$

with

$$Z(\vec{x}) = \sum_{\vec{y}'} \prod_{j=1}^n \Psi_j(\vec{x}, \vec{y}'). \quad (3.41)$$

Given the factors $\Psi_j(\vec{x}, \vec{y})$ in the form

$$\Psi_j(\vec{x}, \vec{y}) = \exp \left(\sum_{i=1}^m \lambda_i f_i(y_{j-1}, y_j, \vec{x}, j) \right), \quad (3.42)$$

and assuming $n + 1$ to be the length of the observation sequence⁹, a linear-chain CRF can be written as

$$p_{\vec{\lambda}}(\vec{y}|\vec{x}) = \frac{1}{Z_{\vec{\lambda}}(\vec{x})} \cdot \exp \left(\sum_{j=1}^n \sum_{i=1}^m \lambda_i f_i(y_{j-1}, y_j, \vec{x}, j) \right). \quad (3.43)$$

The index j is needed in comparison to the maximum entropy model because a label sequence is considered instead of a single label to be predicted. In Equation 3.43, j specifies the position in the input sequence \vec{x} . Note that the weights λ_i are not dependent on the position j . This technique, known as parameter tying, is applied to ensure a specified set of variables to have the same value.

The normalization to $[0, 1]$ is given by

$$Z_{\vec{\lambda}}(\vec{x}) = \sum_{\vec{y} \in \mathcal{Y}} \exp \left(\sum_{j=1}^n \sum_{i=1}^m \lambda_i f_i(y_{j-1}, y_j, \vec{x}, j) \right). \quad (3.44)$$

⁹ Note, that the number of factors is n because any two consecutive positions y_{j-1} and y_j are combined in a factor.

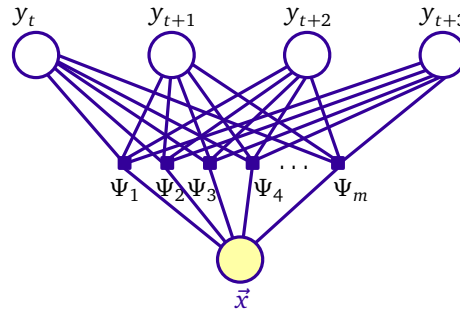


Figure 3.7: Alternative interpretation of a linear-chain CRF.

Summation over \mathcal{Y} , the set of all possible label sequences, is performed to get a feasible probability.

In Equation 3.43 a formulation of a linear-chain CRF is given. Moving the sum over the sequence positions in front of the exponential function, the actual factorization typically done for a CRF gets more evident:

$$p_{\vec{\lambda}}(\vec{y}|\vec{x}) = \frac{1}{Z_{\vec{\lambda}}(\vec{x})} \cdot \prod_{j=1}^n \exp \left(\sum_{i=1}^m \lambda_i f_i(y_{j-1}, y_j, \vec{x}, j) \right). \quad (3.45)$$

The factor graph in Figure 3.6b corresponds to this factorization. One could also move the sum over the different features in front of the exponential function

$$p_{\vec{\lambda}}(\vec{y}|\vec{x}) = \frac{1}{Z_{\vec{\lambda}}(\vec{x})} \cdot \prod_{i=1}^m \exp \left(\sum_{j=1}^n \lambda_i f_i(y_{j-1}, y_j, \vec{x}, j) \right). \quad (3.46)$$

In this interpretation, the factors are not “running” over the sequence but over the features. The factor graph with factors $\Psi_i = \exp \left(\sum_{j=1}^n \lambda_i f_i(y_{j-1}, y_j, \vec{x}, j) \right)$ corresponding to features f_i is given in Figure 3.7. This interpretation is less intuitive but shows the relation to the maximum entropy model (in Figure 3.5).

The model can be interpreted with even more factors by moving both sums to the front of the exponential function

$$p_{\vec{\lambda}}(\vec{y}|\vec{x}) = \frac{1}{Z_{\vec{\lambda}}(\vec{x})} \cdot \prod_{i=1}^m \prod_{j=1}^n \exp \left(\lambda_i f_i(y_{j-1}, y_j, \vec{x}, j) \right). \quad (3.47)$$

The corresponding factor graph is not shown here because of the large number of factors in the graph.

The factorization based on maximal cliques (see Equation 3.45) is the one usually applied for a linear-chain CRF. The other two factorizations (see Equations 3.46 and 3.47) do not adhere to this maximality. In general, factorizing according to cliques consisting of less

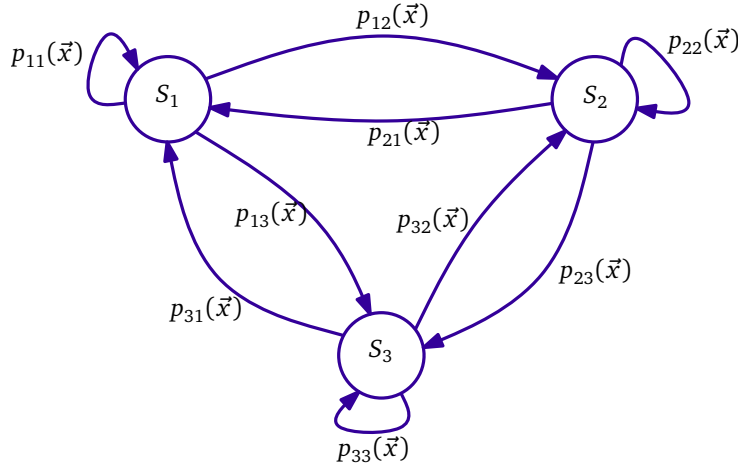


Figure 3.8: Example for a stochastic finite state automaton.

variable nodes than the maximal clique lead to inaccuracies because not all dependencies are correctly considered. In this case, however, it leads to redundant computations as can be seen in Equation 3.47. The rest of the paper is based on the idea of the first factorization.

Linear Chain CRFs have exactly one clique template $C \in \mathcal{C}$: It specifies the independency graph to consist of connections between y_j and y_{j-1} and \vec{x} : $C = \{\Psi_j(y_j, y_{j-1}, \vec{x}) \mid \forall j \in \{1, \dots, n\}\}$. Because of that special structure, it is possible to represent a linear-chain CRF by a stochastic finite state automaton (SFSA) similar to hidden Markov models. This is beneficial for implementation purposes. In that automaton the transition probabilities depend on the input sequence \vec{x} . Its structure is in general arbitrary but the most straight-forward approach is to use a fully connected automaton with states S_l where $l \in \mathcal{Y}$. One state is used for every symbol in the label alphabet. Such automaton with $|\mathcal{Y}| = 3$ is depicted in Figure 3.8.

As stated in Equation 3.43, the features are dependent on the label sequence and herewith on the state transitions in the finite state automaton. So it is important to point out that only a subset of all features f_i is used in every transition in the graph.

The strategy to build a linear-chain CRF can now be summarized:

1. Construct an SFSA $\mathcal{S} = (S, T)$ out of the set of states S (with transitions $T = (s, \dot{s}) \in S^2$). It can be fully connected but it is also possible to forbid some transitions.¹⁰
2. Specify a set of feature templates¹¹ $F = \{g_1(\vec{x}, j), \dots, g_h(\vec{x}, j)\}$ on the input sequence. These are not used directly but for the generation of the features f_i .
3. Generate set of features $\mathcal{F} = \{\forall s, \dot{s} \in S. \forall g_o \in F : f_k(s, \dot{s}, g_o)\}$

¹⁰ Such a decision might depend on the training data and the transitions contained there.

¹¹ An example for such a feature template is $g_1(\vec{x}, j) = \begin{cases} 1 & \text{if } x_j = v \\ 0 & \text{else} \end{cases}$.

Until now, only first-order linear-chain CRFs have been considered. To define linear-chain CRFs with a higher order (see McDonald and Pereira, 2005), the features need to have the form

$$f_i(\vec{y}, \vec{x}, j) = f_i(h_j(\vec{y}), \vec{x}, j) \quad (3.48)$$

with

$$h_j(\vec{y}) = (y_{j-k+1}, \dots, y_j). \quad (3.49)$$

The order is given by k . For higher orders¹² ($k > 2$), the same probabilistic state automaton is used by combining different previous output values y_i in special states. For example, for $k = 3$ the set of states would be $S' = \{(S_i, S_j)\}$ for all $i, j \in \{1, \dots, |S|\}$ (according to the first-order SFSFA in Figure 3.8).

For that special linear-chain structure of the CRF, training and inference are formulated in a similar way as for hidden Markov models as basic problems (Rabiner, 1989):

- I) Given observation \vec{x} and a CRF \mathcal{M} : How to find the most probably fitting label sequence \vec{y} ?
- II) Given label sequences \mathcal{Y} and observation sequences \mathcal{X} : How to find parameters of a CRF \mathcal{M} to maximize $p(\vec{y}|\vec{x}, \mathcal{M})$?

Problem I is the most common application of a conditional random field, to find a label sequence for an observation. Problem II is the question how to train, to adjust the parameters of \mathcal{M} which are especially the feature weights λ_i .

In discriminative approaches, the probability $p(\vec{x}|\mathcal{M})$ is not modeled. Estimating this is another basic question in context of hidden Markov models and not considered here.

Training

For all types of CRFs, as well as for maximum entropy models, the maximum-likelihood method can be applied for parameter estimation. That means, that training the model is done by maximizing the log-likelihood \mathcal{L} on the training data \mathcal{T} :

$$\begin{aligned} \mathcal{L}(\mathcal{T}) &= \sum_{(\vec{x}, \vec{y}) \in \mathcal{T}} \log p(\vec{y}|\vec{x}) \\ &= \sum_{(\vec{x}, \vec{y}) \in \mathcal{T}} \left[\log \left(\frac{\exp \left(\sum_{j=1}^n \sum_{i=1}^m \lambda_i f_i(y_{j-1}, y_j, \vec{x}, j) \right)}{\sum_{\vec{y}' \in \mathcal{Y}} \exp \left(\sum_{j=1}^n \sum_{i=1}^m \lambda_i f_i(y'_{j-1}, y'_j, \vec{x}, j) \right)} \right) \right]. \end{aligned} \quad (3.50)$$

¹² Note that first order means $k = 2$.

This approach corresponds to maximization of accuracy (compare with Chapter 8 for an alternative approach). To avoid overfitting, the likelihood is penalized with the term $-\sum_{i=1}^n \frac{\lambda_i^2}{2\sigma^2}$ (a so-called Gaussian Prior). This technique is established for use in maximum entropy models and can also be applied here (see explanations by Chen and Rosenfeld, 2000). The parameter σ^2 models the trade-off between fitting exactly the observed feature frequencies and the squared norm of the weight vector (McDonald and Pereira, 2005). The smaller the values are, the smaller the weights are forced to be, so that the chance that few high weights dominate is reduced. Additionally, other regularization terms tested in this context are exponential priors or hyperbolic priors (Peng and McCallum, 2004). For feature selection, the l_1 prior $\sum_{i=1}^m |\lambda_i|$ (Vail, Lafferty, and Veloso, 2007; Vail, 2008) has been used. In the following, this work focuses on the Gaussian prior; more feature selection methods are discussed in Chapter 7, **Feature Subset Selection**.

For the derivation, the notation of the likelihood function $\mathcal{L}(\mathcal{T})$ is reorganized:

$$\begin{aligned}
 \mathcal{L}(\mathcal{T}) &= \sum_{(\vec{x}, \vec{y}) \in \mathcal{T}} \left[\log \left(\frac{\exp \left(\sum_{j=1}^n \sum_{i=1}^m \lambda_i f_i(y_{j-1}, y_j, \vec{x}, j) \right)}{\sum_{\vec{y}' \in \mathcal{Y}} \exp \left(\sum_{j=1}^n \sum_{i=1}^m \lambda_i f_i(y'_{j-1}, y'_j, \vec{x}, j) \right)} \right) \right] - \sum_{i=1}^m \frac{\lambda_i^2}{2\sigma^2} \\
 &= \sum_{(\vec{x}, \vec{y}) \in \mathcal{T}} \left[\left(\sum_{j=1}^n \sum_{i=1}^m \lambda_i f_i(y_{j-1}, y_j, \vec{x}, j) \right) - \right. \\
 &\quad \left. - \log \left(\sum_{\vec{y}' \in \mathcal{Y}} \exp \left(\sum_{j=1}^n \sum_{i=1}^m \lambda_i f_i(y'_{j-1}, y'_j, \vec{x}, j) \right) \right) \right] - \sum_{i=1}^m \frac{\lambda_i^2}{2\sigma^2} \\
 &= \underbrace{\sum_{(\vec{x}, \vec{y}) \in \mathcal{T}} \sum_{j=1}^n \sum_{i=1}^m \lambda_i f_i(y_{j-1}, y_j, \vec{x}, j)}_{\mathcal{A}} - \underbrace{\sum_{(\vec{x}, \vec{y}) \in \mathcal{T}} \log \left(\sum_{\vec{y}' \in \mathcal{Y}} \exp \left(\sum_{j=1}^n \sum_{i=1}^m \lambda_i f_i(y'_{j-1}, y'_j, \vec{x}, j) \right) \right)}_{\mathcal{B}}}_{\mathcal{Z}_{\vec{\lambda}}(\vec{x})} - \underbrace{\sum_{i=1}^m \frac{\lambda_i^2}{2\sigma^2}}_{\mathcal{C}}. \quad (3.51)
 \end{aligned}$$

The partial derivations of $\mathcal{L}(\mathcal{T})$ by the weights λ_k are computed separately for the parts \mathcal{A} , \mathcal{B} , and \mathcal{C} . The derivation for part \mathcal{A} is given by

$$\frac{\partial}{\partial \lambda_k} \sum_{(\vec{x}, \vec{y}) \in \mathcal{T}} \sum_{j=1}^n \sum_{i=1}^m \lambda_i f_i(y_{j-1}, y_j, \vec{x}, j) = \sum_{(\vec{x}, \vec{y}) \in \mathcal{T}} \sum_{j=1}^n f_k(y_{j-1}, y_j, \vec{x}, j). \quad (3.52)$$

The derivation for part \mathcal{B} , which corresponds to the normalization, is given by

$$\begin{aligned}
 \frac{\partial}{\partial \lambda_k} \sum_{(\vec{x}, \vec{y}) \in \mathcal{T}} \log Z_{\vec{\lambda}}(\vec{x}) &= \sum_{(\vec{x}, \vec{y}) \in \mathcal{T}} \frac{1}{Z_{\vec{\lambda}}(\vec{x})} \frac{\partial Z_{\vec{\lambda}}(\vec{x})}{\partial \lambda_k} \\
 &= \sum_{(\vec{x}, \vec{y}) \in \mathcal{T}} \frac{1}{Z_{\vec{\lambda}}(\vec{x})} \sum_{\vec{y}' \in \mathcal{Y}} \exp \left(\sum_{j=1}^n \sum_{i=1}^m \lambda_i f_i(y'_{j-1}, y'_j, \vec{x}, j) \right) \cdot \\
 &\quad \cdot \sum_{j=1}^n f_k(y'_{j-1}, y'_j, \vec{x}, j). \\
 &= \sum_{(\vec{x}, \vec{y}) \in \mathcal{T}} \sum_{\vec{y}' \in \mathcal{Y}} \frac{1}{Z_{\vec{\lambda}}(\vec{x})} \underbrace{\exp \left(\sum_{j=1}^n \sum_{i=1}^m \lambda_i f_i(y'_{j-1}, y'_j, \vec{x}, j) \right)}_{=p_{\vec{\lambda}}(\vec{y}'|\vec{x}) \text{ see Equation (3.43)}} \cdot \sum_{j=1}^n f_k(y'_{j-1}, y'_j, \vec{x}, j) \\
 &= \sum_{(\vec{x}, \vec{y}) \in \mathcal{T}} \sum_{\vec{y}' \in \mathcal{Y}} p_{\vec{\lambda}}(\vec{y}'|\vec{x}) \sum_{j=1}^n f_k(y'_{j-1}, y'_j, \vec{x}, j) \tag{3.53}
 \end{aligned}$$

Part \mathcal{C} , the derivation of the penalty term, is given by

$$\frac{\partial}{\partial \lambda_k} \left(- \sum_{i=1}^m \frac{\lambda_i^2}{2\sigma^2} \right) = - \frac{2\lambda_k}{2\sigma^2} = - \frac{\lambda_k}{\sigma^2}. \tag{3.54}$$

The log-likelihood function in Equation 3.51 is concave: The first term is linear (see Equation 3.52) the second term belongs to the normalization. Hence, it does not change the concavity of the function and the last term is concave (see Equation 3.54), so is the whole function.

Equation 3.52, the derivation of part \mathcal{A} , is the expected value under the empirical distribution of a feature f_i :

$$\tilde{E}(f_i) = \sum_{(\vec{x}, \vec{y}) \in \mathcal{T}} \sum_{j=1}^n f_i(y_{j-1}, y_j, \vec{x}, j). \tag{3.55}$$

Accordingly, Equation 3.53, the derivation of part \mathcal{B} , is the expectation under the model distribution:

$$E(f_i) = \sum_{(\vec{x}, \vec{y}) \in \mathcal{T}} \sum_{\vec{y}' \in \mathcal{Y}} p_{\vec{\lambda}}(\vec{y}'|\vec{x}) \sum_{j=1}^n f_i(y'_{j-1}, y'_j, \vec{x}, j). \tag{3.56}$$

The partial derivations of $\mathcal{L}(\mathcal{T})$ can also be interpreted as

$$\frac{\partial \mathcal{L}(\mathcal{T})}{\partial \lambda_k} = \tilde{E}(f_k) - E(f_k) - \frac{\lambda_k}{\sigma^2}. \quad (3.57)$$

Note the relation of Equations 3.55 and 3.56 to Equations 3.12 and 3.16 which were formulated for the maximum entropy model. Besides the fact that for the CRF several output variables are considered, these Equations correspond. A difference is the absence of the factor $\frac{1}{N}$, which is irrelevant for finding the maximum by the approximation of the first derivation $\tilde{E}(f_k) - E(f_k) - \frac{\lambda_k}{\sigma^2} = 0$.

Computing $\tilde{E}(f_i)$ is easily done by counting how often each feature occurs in the training data (McDonald and Pereira, 2005). Computing $E(f_i)$ directly is impractical because of the high number of possible tag sequences ($|\mathcal{Y}|$). Recall, that for the maximum entropy models, $E(f_i)$ can be computed efficiently due to the small number of different output variables y in most applications. In a CRF, sequences of output variables lead to enormous combinatorial complexity. Thus, a dynamic programming approach is applied, known as the forward-backward algorithm originally described for hidden Markov models (Rabiner, 1989). This algorithm can also be used for linear-chain conditional random fields in a slightly modified form.

According to McDonald and Pereira (2005), a function $T_j(s)$ is defined, which maps a single state s at an input position j to a set of allowed next states at position $j + 1$, and the inverse function $T_j^{-1}(s)$, which maps the set of all states of possible predecessors to s . Special states \perp and \top are defined for start and end of the sequence. An example for the states in Figure 3.8 is $T_j(S_1) = \{S_1, S_2, S_3\}$. Forward (α) and backward (β) scores will be used, which can be understood in general as messages sent over the network, in the following assumed to be a linear chain (Bishop, 2006):

$$\alpha_j(s|\vec{x}) = \sum_{s' \in T_j^{-1}(s)} \alpha_{j-1}(s'|\vec{x}) \cdot \Psi_j(\vec{x}, s', s) \quad (3.58)$$

$$\beta_j(s|\vec{x}) = \sum_{s' \in T_j(s)} \beta_{j+1}(s'|\vec{x}) \cdot \Psi_j(\vec{x}, s, s'). \quad (3.59)$$

In relation to the definition of the potentials in Equation 3.42 the features are defined on special states: $\Psi_j(\vec{x}, s, s') = \exp\left(\sum_{i=1}^m \lambda_i f_i(y_{j-1} = s, y_j = s', \vec{x}, j)\right)$.

The α functions are messages sent from the beginning of the chain to the end. The β functions are messages sent from the end of the chain to the beginning. They are initialized by

$$\alpha_0(\perp|\vec{x}) = 1 \quad (3.60)$$

$$\beta_{|\vec{x}|+1}(\top|\vec{x}) = 1. \quad (3.61)$$

With these messages, it is possible to compute the expectation under the model distribution

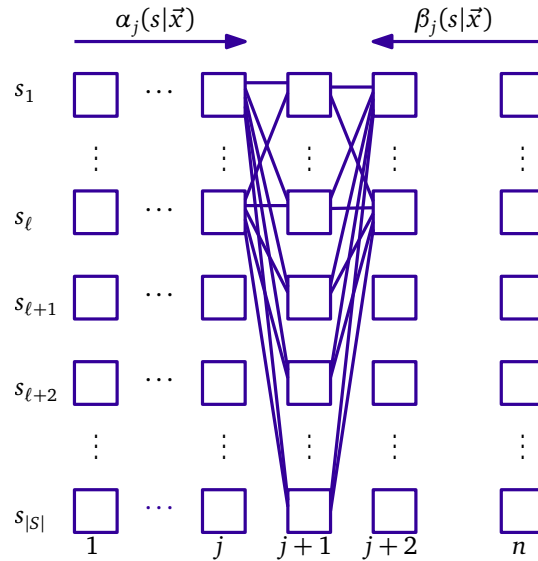


Figure 3.9: Message passing in the forward-backward algorithm. Each column represents one variable, each box in a row one possible value of that variable.

efficiently (Lafferty, McCallum, and Pereira, 2001; McDonald and Pereira, 2005) by

$$E(f_i) = \sum_{(\vec{x}, \vec{y}) \in \mathcal{D}} \frac{1}{Z_{\vec{\lambda}}(\vec{x})} \sum_{j=1}^n \sum_{s \in S} \sum_{s' \in T_j(s)} \underline{f_i(s, s', \vec{x}, j) \cdot \alpha_j(s|\vec{x}) \Psi_j(\vec{x}, s, s') \beta_{j+1}(s'|\vec{x})}. \quad (3.62)$$

The underlined part of Equation 3.62 can be understood as computing the potentials in all combinations of state sequences in the training data. A nice visualization is the so-called *lattice diagram* in Figure 3.9 (Bishop, 2006) in which possible paths of messages sent are depicted. The values for α and β are stored after one iteration so that they have to be computed only once.

The normalization factor is computed by

$$Z_{\vec{\lambda}}(\vec{x}) = \beta_0(\perp|\vec{x}). \quad (3.63)$$

The forward-backward algorithm has a run-time of $O(|S|^2 n)$, so it is linear in the length of the sequence and quadratic in the number of states.

Inference

The problem of inference is to find the most likely sequence \vec{y} for given observations \vec{x} . Note, that this is not about to choose a sequence of states, which are individually most likely. That would be the maximization of the number of correct states in the sequence. In contrast, for finding the most likely sequence the Viterbi Algorithm is applied (Rabiner, 1989). The Viterbi Algorithm is similar to the forward-backward algorithm. The main difference is that instead of summing a maximization is applied.

The quantity $\delta_j(s|\vec{x})$, which is the highest score along a path, at position j , which ends in state s , is defined as

$$\delta_j(s|\vec{x}) = \max_{y_1, y_2, \dots, y_{j-1}} p(y_1, y_2, \dots, y_j = s|\vec{x}). \quad (3.64)$$

The induction step is

$$\delta_{j+1}(s|\vec{x}) = \max_{s' \in S} \delta_j(s') \cdot \Psi_{j+1}(\vec{x}, s, s'). \quad (3.65)$$

The array $\psi_j(s)$ keeps track of the j and s values. The algorithm then works as follows:

1. Initialization:

The values for all steps from the start state \perp to all possible first states s are set to the corresponding factor value.

$$\begin{aligned} \forall s \in S : \quad \delta_1(s) &= \Psi_1(\vec{x}, \perp, s) \\ \psi_1(s) &= \perp \end{aligned} \quad (3.66)$$

2. Recursion:

The values for the next steps are computed from the current value and the maximum values regarding all possible succeeding states s' .

$$\begin{aligned} \forall s \in S : 1 \leq j \leq n : \quad \delta_j(s) &= \max_{s' \in S} \delta_{j-1}(s') \Psi(\vec{x}, s', s) \\ \psi_j(s) &= \operatorname{argmax}_{s' \in S} \delta_{j-1}(s') \Psi(\vec{x}, s', s) \end{aligned} \quad (3.67)$$

3. Termination:

$$p^* = \max_{s' \in S} \delta_n(s') \quad (3.68)$$

$$\vec{y}_n^* = \operatorname{argmax}_{s' \in S} \delta_n(s') \quad (3.69)$$

4. Path (state sequence) backtracking:

Recompute the optimal path from the lattice using the track keeping values ψ_t .

$$\vec{y}_t^* = \psi_{t+1}(\vec{y}_{t+1}^*) \quad t = n-1, n-2, \dots, 1 \quad (3.70)$$

Steps 1-3 are very similar to the forward-backward algorithm. A lattice is filled with the “best” values. Step 4 reads the best path from that lattice.

3.4.3 Arbitrarily Structured CRFs

In Section 3.4.2, efficient training and inference for the special case of a linear-chain CRF have been discussed. In the following, CRFs with an arbitrary graphical structure, such as a tree or a grid structure, are explained. Different CRF structures have been proposed by Sutton and McCallum (2007), Finkel, Grenager, and Manning (2005), Lafferty, McCallum, and Pereira (2001), Sutton and McCallum (2004), and Tsai, Sung, et al. (2006).

Moving from a linear-chain CRF to a general CRF basically means to abolish the restrictions that the clique templates (see Section 3.4.1) model a linear structure. Hence, more general algorithms for training and inference have to be applied.

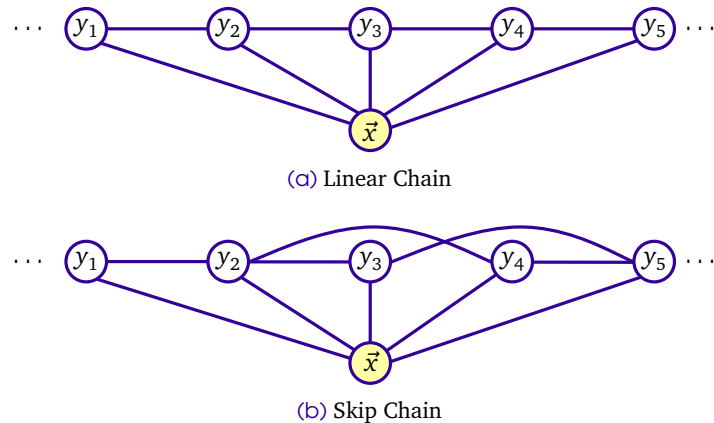


Figure 3.10: Examples for structures of conditional random fields.

Unrolling the graph

In some publications, different CRF structures are depicted (see citations above and examples shown in Figure 3.10). It has to be emphasized, that these structures are meant exemplarily because the actual graph is instantiated separately for each instance for training or inference with the help of the clique templates. Hence, the actual graph structure depends on the considered instance and the specific type of the CRF. The potential functions Ψ_j in the model (compare with Equation 3.40) are associated with the clique templates, but *not* to the cliques in the graph. The process of building the graph for a specific instance is called “unrolling the graph”.

As already discussed in Section 3.4.2 the set of clique templates \mathcal{C} for a linear-chain CRF is given by

$$\mathcal{C} = \{C\} \text{ with } C = \{\Psi_j(y_j, y_{j-1}, \vec{x}) \mid \forall j \in \{1, \dots, n\}\}. \quad (3.71)$$

Accordingly, for a linear-chain CRF there is only one clique template resulting in a linear structure for every possible input value. Only because of this linear sequence structure, an SFSA can be used as a basis for an efficient implementation.

Another possible set of clique templates is

$$\mathcal{C} = \{C_1, C_2\} \text{ with } C_1 = \{\Psi_j(y_j, y_{j-1}, \vec{x}) \mid \forall j \in \{1, \dots, n\}\} \quad (3.72)$$

$$\text{and } C_2 = \{\Psi_{ab}(y_a, y_b, \vec{x}) \mid (a, b) \in \{1, \dots, n\}^2\}, \quad (3.73)$$

where a and b are indexes that specify labels with special attributes on the input sequence. For example in a sequence $\vec{x} = (x_1, \dots, x_n) \in \mathbb{N}^n$ the indexes a and b could specify all items where b is divisible by a . Given a concrete sequence 2, 3, 4, 5, 6 this leads to a CRF with the structure shown in Figure 3.11. In real life applications parameter tying is often applied, in this example, the value of the weights λ_j is the same for factors from the same clique templates, independent of the position in the sequence.

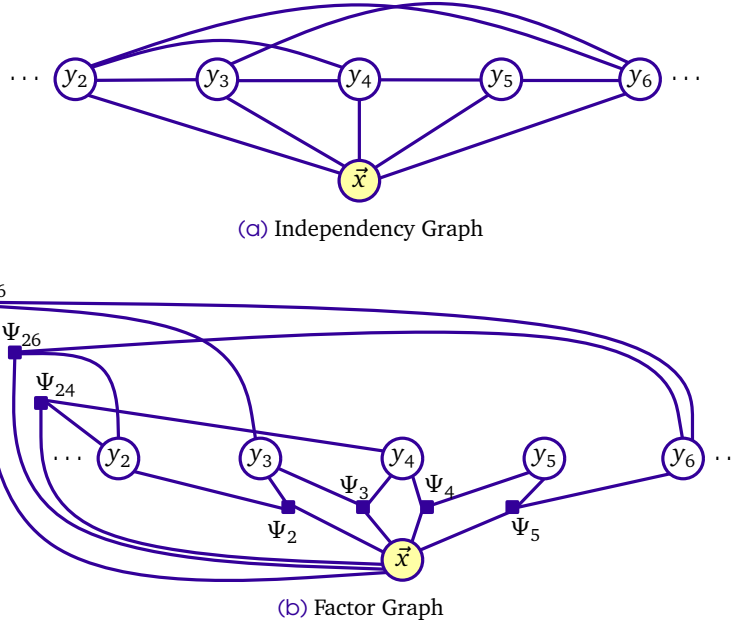


Figure 3.11: Example for an unrolled skip-chain CRF for the sequence $\vec{x} = (2, 3, 4, 5, 6)$ according to Equations 3.72 and 3.73.

Formally, a set of factor templates $\mathcal{C} = \{T_1, \dots, T_n\}$ consists of templates T_k describing a set of tuples $\{(\vec{x}_k, \vec{y}_k)\}$ on which factors are instantiated for which a property $p_k(\vec{x}_k, \vec{y}_k)$ holds and shares $\vec{\lambda}_k$ and $f(\cdot)_k$ between all instantiated factors on the tuples. K_j is the number of parameters of the j th template. The probability distribution on a factor graph with templates \mathcal{C} becomes

$$P(\vec{y}|\vec{x}) = \frac{1}{Z(\vec{x})} \prod_{T_j \in \mathcal{C}} \prod_{(\vec{x}_i, \vec{y}_i) \in T_j} \exp \left(\sum_{k=1}^{K_j} \lambda_{jk} f_{jk}(\vec{x}_i, \vec{y}_i) \right). \quad (3.74)$$

Training and Inference

In sequence structures such as the HMM or the linear-chain CRF (which is a simple chain globally conditioned on the input values \vec{x}) the forward-backward and the Viterbi Algorithm, which are based on sending messages along the chain in the only two possible directions, can be applied. Besides conceptually different algorithms, such as sampling methods, there are generalizations of these algorithms for tree-structured graphs, namely the sum-product and the max-sum algorithm (Kschischang, Frey, and Loeliger, 2001).

The basic idea is that the messages sent along the graph are collected from different directions before forwarding them. This generalization can also be used on arbitrary graphs. The basic idea is to compute a so-called junction tree from the original graph. The algorithms can then be applied in a slightly modified form.

3.5 Summary

A detailed overview of conditional random fields and the theory behind and related to it was presented in this chapter. Conditional random fields can be considered as an extension to the maximum entropy model (a structured learning model instead of a single-position classification model) on the one hand, and the hidden Markov model (discriminative model instead of a generative model) on the other hand.

Probabilistic models can be represented graphically by means of independency and factor graphs. It is distinguished between directed and undirected graphical models which result in different types of factorization. Such factorizations can be read from the graphical representation of a model. They are often a crucial prerequisite for efficiently performing complex computations, as needed for training or inference. Aspects of how a CRF can be factorized and the resulting graphical representations are a focus of this chapter.

Based on the concepts of graphical representation and factorization, a general formulation of CRFs has been introduced. For the special case of a CRF based on a linear sequence structure, an in-depth explanation of methods for training and inference was given. These methods were originally developed for HMMs, but can be reused in a slightly modified way for linear-chain CRFs.

Finally, issues which arise for CRFs with an arbitrary graphical structure were discussed. This includes aspects of how the potential functions of the model can be defined by means of clique templates and why the training and inference algorithms used for a linear-chain CRF cannot be used in a non-sequential scenario. For further reading and detailed explanations on algorithms for training and inference on general probabilistic graphical models the interested reader might refer to Bishop (2006); Lepar and Shenoy (1999); Jordan and Weiss (2002); Mooij and Kappen (2007); Borgelt and Kruse (2002).

Part II

Case Studies and Adaptions to the Biological and Chemical Domain

Chapter 4

Recognition of IUPAC and IUPAC-like Chemical Names¹

4.1 Introduction

Finding mentions of chemical compounds in text is of interest for several reasons: An annotation of the entities enables a search engine to return documents containing elements of this entity class (semantic search), *e. g.* together with a disease. This can be helpful to find relations *e. g.* to adverse reactions or diseases. Mapping the found entities to corresponding structures leads to the possibility to search for relations between different chemicals. This enables a chemist to search for similar structures or substructures and combine the knowledge in the text with knowledge from databases or to integrate other tools handling chemical information (*cf.* Section 1.3.4 and 2.9).

Chemical names can be distinguished into different classes. To deal with complex structures, different methods of nomenclature are used, *e. g.* mentions of sum formulæ or names according to the *Simplified molecular input line entry specification* (SMILES, Weininger (1988)) or the successor of SMILES, the *IUPAC International Chemical Identifier* (InChI). Because of a limited readability of such specifications for humans, trivial names and the nomenclature published by the *International Union of Pure and Applied Chemistry* (IUPAC, McNaught and Wilkinson (1997)) is commonly applied in text (Eller, 2006). Combinations of different types of names as well as abbreviations are in use, too.

The main innovation in this chapter is the presentation of an adaption of conditional random fields to the chemical domain, especially to IUPAC-like names. This includes specific aspects of corpus generation (Section 4.3.2), tokenization (Section 4.3.3), and feature extraction (Section 4.3.4) The need for a machine learning-based method is motivated (Section 4.2). In addition to the correct recognition of IUPAC and IUPAC-like names, the aim is to transform these names using name-to-structure converters to allow the usage of chemical tools on the extracted data (Section 4.3.5). That is compared with a normalization by simple dictionary mapping which shows to work surprisingly well. Additionally to the focus on IUPAC-like names, a problem addressed for the first time, a generalization to different classes of chemical names is shown, which has a 25 % better result in F_1 score than dictionary-based methods published recently (Section 4.4.4).

Additionally to the presentation of the results, an exhaustive analysis of different feature sets on the results is evaluated and discussed, enabling system designers for morphologically similar classes to use the experiences in designing the presented system.

¹ This chapter is based on the work by Klinger, Kolářik, et al. (2008) and Kolářik, Klinger, et al. (2008).

Related Work

Trivial names can be searched for with a dictionary based approach and directly mapped to the corresponding structure at the same time. For instance, Hanisch, Fundel, et al. (2005) use the dictionary extracted from DrugBank² (Wishart, Knox, et al., 2006) to recognize drug names in MEDLINE abstracts (Kolářik, Hofmann-Apitius, et al., 2007). Other systems use the drug dictionary from MedlinePlus³ (e. g. EbiMed, (Rebholz-Schuhmann, Kirsch, et al., 2007)) for the recognition of drug names. Hettne, Stierum, et al. (2009) present a dictionary merged from several sources, evaluated on the corpus presented in Section 4.2.

For other representations of chemical structures like SMILES, InChI or IUPAC names an enumeration is only possible for the most common substances. The full chemical space is countably infinite. Several systems address that problem regarding chemical entities with a variety of approaches. Narayanaswamy, Ravikumar, and Vijay-Shanker (2003) describe a manually developed set of rules relying upon lexical information, linguistic constraints of the English language and contextual information for the detection of several entity classes. The reason for choosing this approach is stated as the lack of an annotated corpus. The evaluation was done on a small hand-selected corpus containing 55 MEDLINE abstracts selected by searching for *acetylates*, *acetylated* and *acetylation*. They found 158 chemical names from which 22 were ambiguous and classified into different classes and 13 chemical parts with two ambiguous ones. The F_1 measure for the first is 90.86 % (93.15 % precision, 86.08 % recall). The latter has an F_1 measure of 91.67 % (100 % precision, 84.62 % recall).

Similarly, Kemp and Lynch (1998) identify chemical names in patent texts with handcrafted rules using dictionaries with chemical name fragments. They claim to identify 97.4 % from 14855 specific chemical names in 70 patent descriptions taken from documents from the IPC class CO 7D. The false positive rate is reported to be 4.2 %.

The concept described by Anstein, Kremer, and Reyle (2006) for which the preconditions are described by Reyle (2006) uses a grammar for the analysis of fully specified (e. g. 7-hydroxyheptan- 2-one), trivial (e. g. benzene) and semi-systematic (e. g. benzene-1,3,5-triacetic acid) as well as underspecified (e. g. deoxysugar) compound names. The advantage of that approach is that the grammatical analysis can be used as a basis for a conversion to the chemical structure. A possible problem is the difficulty to recognize names not following the specification to a certain degree as well as the completeness and maintenance of a changing standard.

A molecular similarity search is used by Rhodes, Boyer, et al. (2007) to enable a user to “search for related Intellectual Property” in U.S. patents based on a specified drawn molecule. They report to find 3,623,248 unique chemical structures from 4,375,036 U.S. patents. The absolute numbers of found patents for the top 25 drugs listed by Humana (2005) are shown.

The program developed by Sun, Tan, et al. (2007) focuses on finding sum formulæ like $CH_3(CH_2)_2OH$ in text using support vector machines (Schölkopf and Smola, 2002) and conditional random fields (Lafferty, McCallum, and Pereira, 2001).

² <http://redpoll.pharmacy.ualberta.ca/drugbank/>

³ U.S. National Library of Medicine (2007), <http://mplus.nlm.nih.gov/medlineplus/druginformation.html>

In the work of Corbett, Batchelor, and Teufel (2007), first-order hidden Markov models (Rabiner, 1989) implemented in the toolkit LingPipe⁴ are combined with other methods for the identification of chemical entities. The program finds *e. g.* structural classes, atoms, and elements, fragments, trivial names, SMILES and InChI as well as IUPAC names. They give an inter-annotator F_1 measure of 93 % for chemical names on their annotated corpus. The performance is evaluated on different corpora, recall rates are between 69.1 % and 80.8 % and precision rates between 64.1 % and 75.3 % (Corbett and Murray-Rust, 2006). A separate evaluation of the included named entity recognition modules from the toolkit LingPipe results in an F_1 measure of 74 % (Corbett, Batchelor, and Teufel, 2007). Their implementation, the open source program OSCAR3⁵ (*Open Source Chemistry Analysis Routines*, Corbett, 2007), is available to the academic community.

4.2 Dictionary-based Chemical Name Recognition

Dictionary-based methods naturally provide a normalization to a database identifier and the related real-world compound, which is important for chemical names as the variety of mentions for the same compound can be very high. Klein (2010) states the number of synonyms for *e. g.* “Aspirin” to be 233 in a dictionary merged from several sources.

Thus, the first step towards chemical named entity recognition needs to be an analysis of resources for dictionary-based recognition and an evaluation of their comprehensiveness (Kolářik, Klinger, et al., 2008). For such inspection, a corpus of 100 articles has been annotated with entity classes corresponding to the ones mentioned in Section 4.1, namely IUPAC, PART, TRIVIAL, ABB, SUM, and FAMILY. Examples are depicted in Table 4.1. The separation between TRIVIAL and IUPAC names is based on the term length, names with only one word were classified as TRIVIAL even if they were IUPAC names. Multi word systematic and semi-systematic names are always annotated as IUPAC. This includes names that imply only an IUPAC-like part (*e. g.* 17-alpha-E) or names including a labeling (*e. g.* 3H-testosterone). This does not follow strictly the definition of IUPAC, but can be followed more easily and consequently by annotators. For the correct resolution of enumerations, partial chemical names have been annotated separately as PART, but chemical names were not tagged in other entities (*e. g.* in protein names). Names were only tagged as FAMILY if they describe well defined chemical families but not pharmacological families (*e. g.* glucocorticoid was labeled but not anti-inflammatory drug). Substances used as base for building various derivatives and analogs were tagged as IUPAC, not as FAMILY (*e. g.* 1,4-dihydronaphthoquinones). This corpus is referred to as *Chemistry-Corpus*. The inter-annotator agreement between two independent annotators measured via F_1 is 89 %. The annotated corpus has 1206 entities with 391 IUPAC, 92 PART, 414 TRIVIAL, 161 ABB, 49 SUM, and 99 FAMILY entities. This distribution of the different classes is shown in Figure 4.1.

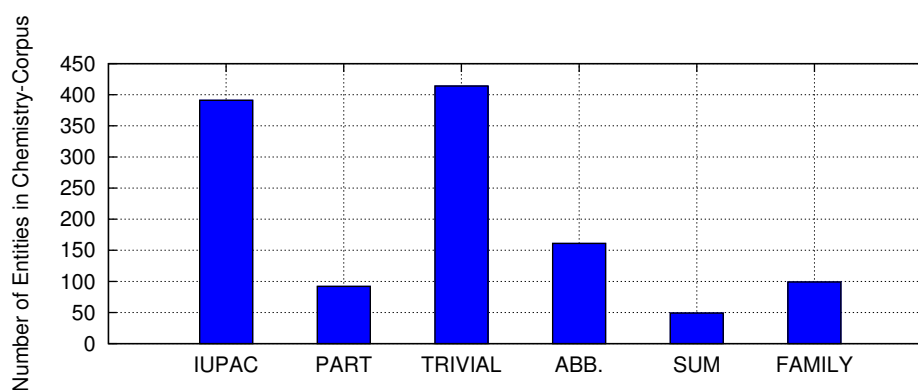
Different resources have been used to generate chemical name dictionaries, namely the

⁴ <http://alias-i.com/lingpipe/>

⁵ <http://oscar3-chem.sourceforge.net>

Chemical Class	Description	Example Annotation
IUPAC	IUPAC names, IUPAC-like names, systematic, and semi-systematic names	1-hexoxy-4-methyl-hexane
PART	partial IUPAC class names	17beta-
TRIVIAL	trivial names	aspirin, estragon
ABB	abbreviations and acronyms	TPA
SUM	sum formulae, atoms, and molecules, SMILES, InChI	KOH
FAMILY	chemical family names	disaccharide

Table 4.1: Chemical entity classes used for corpora annotation.

Figure 4.1: Distribution of classes in test corpus *Chemistry-Corpus* for dictionary based analysis.

freely available database *PubChem*, the database *Chemical Entities of Biological Interest ChEBI*, *Medical Subject Headings MeSH*, the *Kyoto Encyclopedia of Genes and Genomes KEGG*, the *DrugBank*, and *Human metabolome Database HMDB*. A detailed description of these resources is presented by Kolářik, Klinger, et al. (2008). Analyzing the comprehensiveness of dictionaries extracted from these resources corresponds to measuring the recall. The results are shown in Figure 4.2.⁶ TRIVIAL names are most comprehensively found. Most problematic is the recognition of IUPAC with an exact-match recall of at most 0.23 (cf. Figure 4.2). Accepting partial matches leads to a maximum recall of 0.94: This shows the challenge of detecting IUPAC names: Dictionaries can only find parts of an IUPAC name as full names are mainly not included; detecting the boundaries correctly is a problem.

Therefore, a method identifying IUPAC and IUPAC-like names only is presented, additional approaches can be used to recognize other chemical name classes (e. g. brand names or elements): IUPAC and IUPAC-like names can be recognized based on their morphological structure with higher performance than with methods based on dictionaries (Kolářik, Klinger,

⁶ MESH_C and MESH_T are different parts from MESH, KEGG_C and KEGG_D are different parts of KEGG. A detailed description can be found in Kolářik, Klinger, et al. (2008)

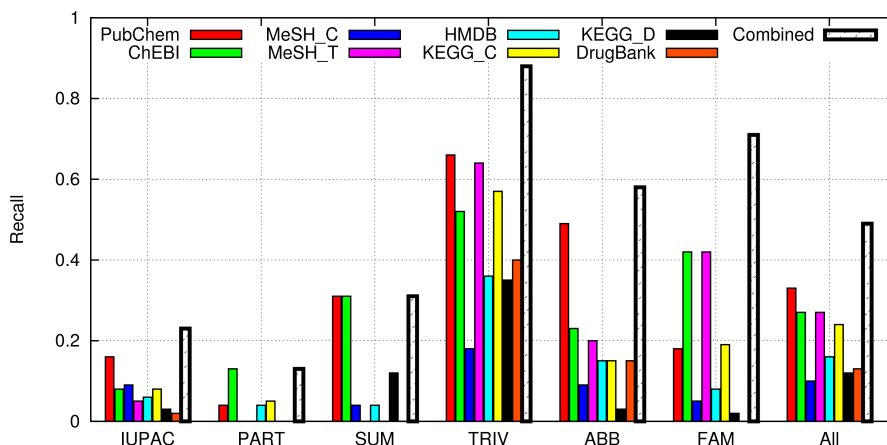


Figure 4.2: Dictionary-based recall on *Chemistry-Corpus* (exact match).

et al., 2008) as we will see. These IUPAC-like terms do not only include correct IUPAC names but also names not following the nomenclature strictly. This enables a higher recall regarding mentioned chemicals, which is important for document retrieval purposes.

4.3 Named Entity Recognition Workflow

In the following, the annotated entities and training corpora are described as well as the configuration of a linear-chain CRF as a model for finding IUPAC entities and IUPAC-related MODIFIER entities. The inter-annotator agreement of two independent annotators is given. The model selection is performed by bootstrapping (Efron and Tibshirani, 1993) and evaluated on two independent test corpora, one consisting of sampled abstracts from MEDLINE, the other one on hand selected paragraphs from bio-chemical patents. Finally, the use of *name-to-structure* converters as a basis of a possible normalization is analyzed. This conversion to a unique structure is compared to straight-forward string matching normalization against dictionaries.

4.3.1 Entity Types

The entities of interest described and motivated in Sections 4.1 and 4.2 are IUPAC and MODIFIER mentions. Chemical entities in general are named following different nomenclatures which are also combined by the authors of biomedical texts. Only concentrating on correct IUPAC terms is not sufficient, hence an IUPAC entity is defined to be a chemical substance mentioned in a IUPAC-like manner. Additionally to correct IUPAC names, it includes IUPAC names in which a part is abbreviated, fragments and group names. In Figure 4.3 an example abstract from MEDLINE with annotations of the two entity classes is shown.

Next to full names like “1,2,3,4-tetrahydronaphthalene-1-carboxylic acid” or “4-[(3-chlorophenyl)amino]methyl]-6,7-dihydroxychromen-2-one”, fragments, e. g. in enumera-

Synthesis of racemic 6,7,8,9-tetrahydro-3-hydroxy-1H-1-benzazepine-2,5-diones as antagonists of N-methyl-d-aspartate (NMDA) and α -amino-3-hydroxy-5-methylisoxazole-4-propionic acid (AMPA) receptors.

The synthesis and pharmacological properties of several racemic 6,7,8,9-tetrahydro-3-hydroxy-1H-1-benzazepine-2,5-diones (THHBADs) are described. Synthesis was accomplished via a Schmidt reaction with 5,6,7,8-tetrahydro-2-methoxynaphthalene-1,4-diones (THMNDs) followed by demethylation. THMNDs were prepared via a Diels-Alder reaction with 2-methoxybenzoquinone (5) or 2-bromo-5-methoxybenzoquinone (14) and substituted 1,3-butadienes. The pharmacology of THHBADs was characterized by electrical recordings in *Xenopus* oocytes expressing rat brain NMDA and AMPA receptors. THHBADs are antagonists of NMDA and AMPA receptors with functional potency being dependent upon the substitution pattern on the tetrahydrobenzene moiety. The 7,8-dichloro-6-methyl (18a) and 7,8-dichloro-6-ethyl (18b) analogs are the most potent THHBADs prepared and have apparent antagonist dissociation constants (K_b values) of 0.0041 and 0.0028 μM , respectively, for NMDA receptors and 0.51 and 0.72 μM , respectively, for AMPA receptors.

Figure 4.3: Example abstract with tagged entities (PMID 9240357, (Guzikowski, Whittemore, et al., 1997)). IUPAC entities are depicted in red while MODIFIER entities are shown in blue.

tions, are tagged separately like “acridine-4-” and “phenazine-1-carboxamide” in “. . . both the acridine-4- and phenazine-1-carboxamide series. . .” or “3 α -[bis(4-fluoro-” and “4-chlorophenyl)methoxy] tropane” in “. . .N- and 2-substituted-3 α -[bis(4-fluoro- or 4-chlorophenyl)methoxy]tropane. . .”.⁷

The alternative to the separate way of annotating parts in enumerations would have been an annotation including the connecting word (in that example “or”). This is not meaningful because parts of names are sometimes divided by long text passages. With this kind of annotation, an enumeration resolution of found parts in the text is prepared.

The MODIFIER entity describes similarities to a mentioned substance as in “[IUPAC-entity] analogues” or “[IUPAC-entity] modifier” or “3-substituted-[IUPAC-entity]”.

4.3.2 Corpus Selection and Annotation

Three main corpora are generated additionally to the *Chemistry-Corpus* built for Section 4.2 for building the model and evaluating the approach following the annotation guideline. A training corpus consisting of MEDLINE abstracts (abbreviated as *IUPAC-Train-M*), a test corpus containing MEDLINE abstracts (*IUPAC-Test-M*) and a test corpus made up of parts of patents (*IUPAC-Test-P*).

The training corpus is built in two steps. First, a preliminary corpus (abbreviated as *IUPAC-Prelim*) is built in the same manner as described by Friedrich, Revillion, et al. (2006). For that, in the BioCreative training corpus (Hirschman, Krallinger, and Valencia, 2007), the gene and protein names are replaced by randomly selected correct IUPAC names from PubChem⁸. This leads to an artificial corpus with 15000 sentences with 1216341 tokens. It

⁷ The colors show the entity: red for IUPAC entities, blue for MODIFIER entities

⁸ NCBI (2007), <ftp://ftp.ncbi.nlm.nih.gov/pubchem/Compound/CURRENT-Full/XML/>

includes 24325 entities. On that corpus a CRF is trained and used for tagging 10000 sampled abstracts from MEDLINE. From these, 463 abstracts are selected which include 161591 tokens in 3700 sentences⁹ with 3712 IUPAC and 1039 MODIFIER entities.

For evaluation of the system, 1000 MEDLINE records with 124122 tokens in 5305 sentences are sampled equally distributed from full MEDLINE containing 151 IUPAC and 14 MODIFIER entities resulting in the corpus *IUPAC-Test-M*.

Passages from 26 patents dealing with chemical processes were hand selected according to occurring enumerations of chemicals, especially using different and mixed nomenclatures to detect possible problems. These paragraphs consist of 4309 words in 152 sentences with 411 IUPAC entities forming the corpus *IUPAC-Test-P*.

The training corpus is annotated by two independent annotators. The inter-annotator F_1 measure for the IUPAC entity is relatively low with 78% (in contrast to 93% claimed by Corbett, Batchelor, and Teufel (2007)). One reason for the difference in comparison to Corbett and his colleagues is my differentiation of the IUPAC entity to other chemical mentions, which is not always easy to decide while all chemical mentions in the corpus generated by Corbett are combined in one entity. Another reason is the differing experience level of the annotators. While the first annotator collaborated on the development of the annotation guideline and annotated several corpora the second annotator based his annotations directly on the provided guideline.

For building the conclusive training corpus both annotations are combined by a third person. The F_1 measure between the resulting training corpus (*IUPAC-Train-M*) and the first-annotated corpus is 94%.

Additionally, a single person annotation of *IUPAC-Train-M* with all classes presented in Section 4.2 has been performed to evaluate a generalization of the presented model to other classes of chemical nomenclature.

4.3.3 Tokenization

IUPAC entities may be differing in structure from texts used to build sentence splitters. Therefore, the accuracy of such methods is unknown; in this setting whole abstracts are used for training and testing to overcome that possible limitation.

The straight-forward approach for tokenization as discussed in Section 2.4 is to keep entities unsplit. In the case of IUPAC names this is difficult. Firstly, symbols like brackets or hyphens are used as parts of the names. Secondly, due to the sheer length of the names, wrong white spaces are regularly included in IUPAC names, probably due to the processing steps to provide the text in different formats. Both issues together lead to a dilemma of separating such symbols or not. To be able to detect names despite of their length and context, a fine-grained tokenization is applied, splitting on all special symbols only keeping alphanumerical strings together, but separating between numbers and letters. The disadvantage of such

⁹ Number of sentences is detected with the JULIELab sentence splitter (<http://www.julielab.de/>, Tomanek, Wermter, and Hahn, 2007b).

approach is that the length of the entity is not captured and the model needs to take care of dependencies between distant tokens.

Note that this tokenization splits words, which are understood as the smallest semantically meaningful units in text by most humans. Therefore, the features need to be adapted as described in the next section.

4.3.4 Feature Extraction

Many of the evaluated features are extracted by standard methods, especially the morphological ones (see Section 2.6). Next to these commonly used features special IUPAC-related features are incorporated. These are the membership of a token to a list of often used prefixes and suffixes of length four in IUPAC names or a list of typical last tokens of the names. These lists are extracted from all IUPAC names mentioned in the data available from PubChem⁸.

The list of prefixes of length 4 has 714 members, the list of suffixes of the same length has 661 members. Another list includes 300 suffixes of the last tokens of IUPAC names to improve the detection of the end of a IUPAC name. The general idea of these lists is to provide the system with a possibility to generalize in excess of the training data.

As described in Section 4.3.3, a fine-grained tokenization is applied. To be able to keep track of originally concatenated tokens, a feature to detect preceding or succeeding white space is incorporated. This is not common in approaches to detect other entity classes as the tokenization is able to take care of keeping semantically logical units together.

The typically long entities demand to be modelled with contextual information.¹⁰ Therefore, different orders of offset conjunction (*cf.* Section 2.6) are tested, modelling the context on feature level and different orders of the CRF (*cf.* Equation 3.49), modelling the context together with the label variables.

4.3.5 Normalization of IUPAC names

To normalize found names, one solution is to convert them to a structure representation. Several tools have been developed for that task. Eigner-Pitto, Eiblmaier, et al. (2007) show a short evaluation of three commercial tools. One is *LexiChem*¹¹ (OpenEye, 2007) by *OpenEye*, a product capable of conversions from IUPAC names as well as other names to structures and vice versa. Another program is *ACDName* by *ACDLabs*, which generates chemical structures from systematic names, derivatives, semi-systematic, and trivial names as well as incorrect names, not strictly following the nomenclature (ACDLabs, 2007), but it focuses more on correct names than the program *Name=Struct* by *CambridgeSoft* (Eigner-Pitto, Eiblmaier, et al., 2007; CambridgeSoft, 2007).

The Open Source converter included in OSCAR3, called OPSIN¹² is used, the only software to my knowledge, which is freely available for academic evaluation purposes. It converts

¹⁰ In Chapter 9, Figure 9.1 on page 134 depicts the distribution of the length of entities comparing different classes.

¹¹ <http://www.eyesopen.com/lexichem-tk>

¹² Version of October 11, 2006, <http://oscar3-chem.sourceforge.net>.

Name	Explanation
<hr/>	
Static morphol. features	Reg.Ex.
<hr/>	
All Caps	[A-Z]+
Number Seq.	[-0-9]+[,]+[0-9.,]+
Is Dash	[- — -]
Is Quote	[, “ ” ‘ ’]
Is Slash	[\ /]
<hr/>	
Autom. generated features	
Autom. Prefixes/Suffixes	Automatic generation of a feature for every token: match that prefix or suffix (length 2)
Bag-Of-Words	Automatic generation of a feature for every token: match that token
<hr/>	
Spaces	
Spaces_left	white space preceding token
Spaces_right	white space following token
<hr/>	
Lists	
Prefix/Suffix lists	Prefixes and suffixes (length 4) of intermediate or last words generated from IUPAC names

Table 4.2: Features used in the CRF.

names to the *Chemical Markup Language* (CML, Murray-Rust (1997)) which can be translated to SMILES using the Chemistry Development Kit¹³ (CDK, Steinbeck, Han, et al. (2003)).

An alternative approach is to map the detected names to dictionaries as the ones presented in Section 4.2. For the experiments presented in Section 4.4, a merged dictionary retrieved from ChEBI and MeSH is used. Each term is mapped to a canonical form retrieved by stemming¹⁴ of each token, mapping ph and f to the same symbol, removing lower and upper case differences, ignoring white space in the term. The tokens in a term are not sorted, as this would lead to differences in the meaning of IUPAC names. This process does not include any semantic interpretation of the term. The advantage of this conservative approach is that normalizable entities are correctly identified (assuming correctness of the data base).

4.4 Results

In a first step a CRF is trained on the preliminary, tweaked corpus *IUPAC-Prelim* mentioned in section 4.3.2 and evaluated by 50-fold bootstrapping. This allows to evaluate how far the mere use of correct IUPAC names leads to an applicable system. The result is an F_1 measure of 97.92% (98.08% precision, 97.76% recall) with a first-order CRF with first-order offset conjunction and the same parameter set as described in section 4.4.1. These results are comparable to those published by Friedrich, Revillion, et al. (2006). Evaluating this model on

¹³ Version 1.0.1 of June 26, 2007, cdk.sourceforge.net/

¹⁴ Using the Snowball Stemmer <http://snowball.tartarus.org/>

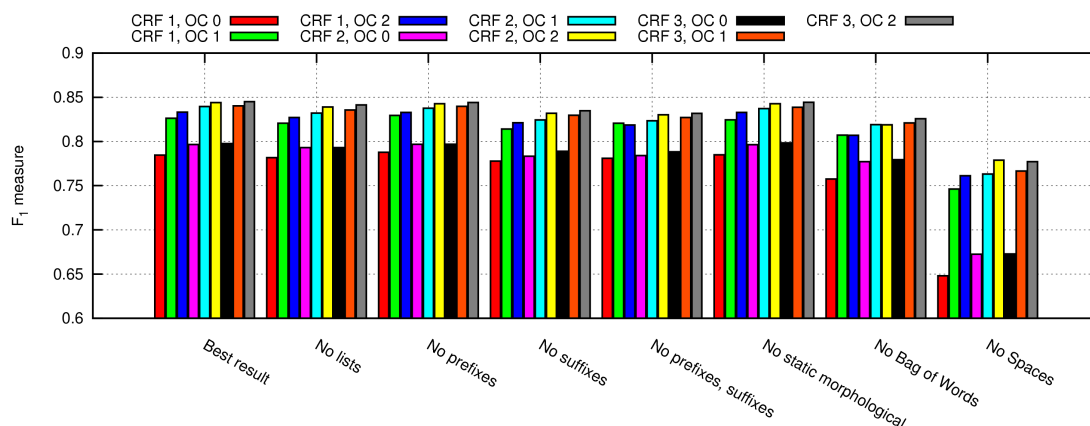


Figure 4.4: Results on the training data *IUPAC-Train-M* with 30-fold bootstrapping with different feature sets, different orders q of the CRF (given as CRF q above), and different orders p of the offset conjunction (given as OC p). The best results were obtained with the feature set presented in Table 4.2. For more details see sections 4.3.4 and 4.4.1.

the annotated MEDLINE training corpus *IUPAC-Train-M* shows a low F_1 measure (with 19.5 %) and 38.4 % recall. The performance on the sampled MEDLINE test corpus *IUPAC-Test-M* is even worse with 1.1 % F_1 measure and a recall of 29.1 %. These results show that there is a fundamental difference in tagging the tweaked corpus *IUPAC-Prelim* (which seems to be simple, considering the F_1 measure) and real world texts (as *IUPAC-Train-M* and *IUPAC-Test-M*). The analysis of the different corpora shows two main problems. On the one hand only correct IUPAC names are included in *IUPAC-Prelim*, but fragments occur frequently in real text. On the other hand, a big problem are missing negative examples in the tweaked training data representing what is *not* an IUPAC name: A lot of isolated numbers, single letters, expressions in or around brackets are found wrongly.

Based on the experiences on the tweaked training corpus *IUPAC-Prelim*, a CRF is trained on the annotated training corpus based on MEDLINE abstracts *IUPAC-Train-M* using a selected parameter set. The evaluation of the different parameters is given.

4.4.1 Parameter Selection

For model selection, the impact of the following parameters of the CRF are evaluated by applying 30-fold bootstrapping on the training set *IUPAC-Train-M*:

- Features representing the text like *Bag-Of-Words* or *morphological features* (cf. Table 4.2),
- the order of the CRF, and
- the order of the offset conjunction.

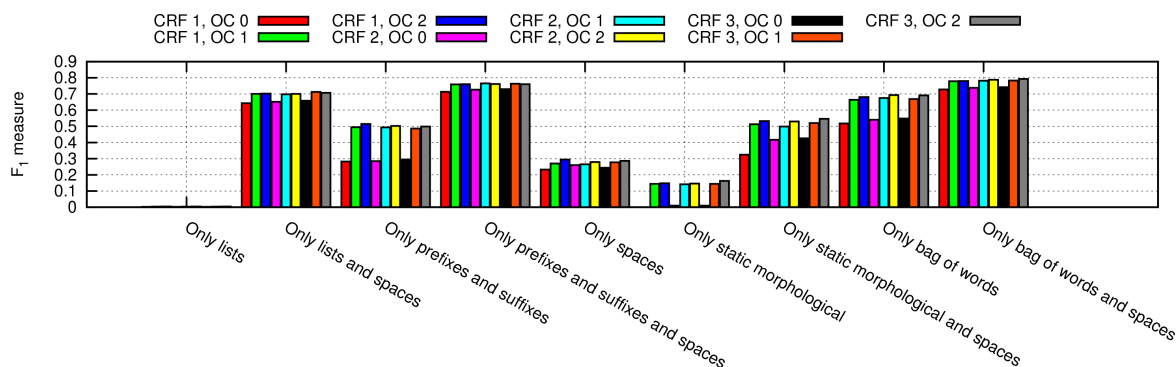


Figure 4.5: Results on the training data *IUPAC-Train-M* with some small feature sets on different orders of the CRF and offset conjunctions. In contrast to Figure 4.4, where variations of a larger feature set are shown, the importance of features is presented in the context of very small feature sets (note the different scales between Figures 4.4 and 4.5).

The feature set of the system with the best performance consists of automatically added features based on *Bag-Of-Words* as well as *Autom. Prefixes/Suffixes* of length two. Additionally the membership to *Prefix/Suffix lists* containing prefixes or suffixes of length four of last or intermediate tokens from IUPAC names is considered. From the set of *static morphological features*, *All Caps*, *Number Seq.*, *Is Dash*, *Is Slash* and *Is Quote* are used. The *Spaces* features to determine if the token is preceded or succeeded by white space is also included. Many other features, mainly from the field of gene and protein recognition were also tested, e. g. mapping the token to regular expressions representing Greek letters, combinations of alpha-numerical symbols, natural numbers etc.

To evaluate the impact of the different features, one from the best feature set is omitted in several experiments (cf. Figure 4.4) and models are trained only with small feature sets (depicted in Figure 4.5). The automatically generated features *Bag-Of-Words* and *Autom. Prefixes/Suffixes* have the highest impact on the performance together with the *Spaces* feature. Especially the last one is essential to obtain good results with impacts between 6.5% (CRF 2, OC 2) and 13.64% (CRF 1, OC 0). In contrast, the *static morphological* features and the *Prefix/Suffix lists* bring nearly no loss omitting them and low results when used as the only feature. Nevertheless, together with the feature *Spaces*, the results are surprisingly high (70% F_1 measure). Interesting is that using only *Autom. Prefixes/suffixes* or *Bag-Of-Words* together with the *Spaces* feature and CRF order 3 and offset conjunction order 2 results in an F_1 measure of 76.03% or 79.31% respectively.

Different configurations of the features with different orders of offset conjunction (adding context in form of features of the last p and next p tokens, where p is the order of the offset conjunction) are evaluated as well as the order of the CRF, which includes information from the last q labels (q is the order of the CRF). The results of some of the features for different orders of offset conjunction and CRF can also be seen in Figure 4.4. The importance of the different features is nearly the same for all the different orders. The divergence in the results is high for different feature sets, but it is also very important to have the context information

provided by the offset conjunction. The best F_1 measure can be obtained with an offset conjunction of order 2 and a CRF-order of 2 or 3. The difference between a CRF without an offset conjunction (*i. e.* order 0) to a CRF with order 1 offset conjunction are much higher than between order 1 and order 2 offset conjunctions. The increase of the order comes along with a high increase in the number of weights λ_i (*cf.* Equation 3.37 and 3.43): We have (for the CRF with order 3) 119884 weights without an offset conjunction, 315377 with an offset conjunction order of 1 and 521179 weights with an offset conjunction order of 2. This corresponds to the training and tagging durations depicted in Figure 4.6 (including reading the data from disk and feature extraction). Noteable here is the comparatively high tagging time for OC2 in an order-2 CRF in comparison to the respective order 3 OC2 duration. This difference is not fully understood; presumably interactions with the Java garbage collectors are a reason for this behaviour.

Inspecting the tagging errors exposes that especially bounding errors at the end or at the beginning of the name are more frequent for a lower order of the offset conjunction. Other taggings that can be correctly identified with an offset conjunction order of 2 are formulations like "... through the 7- or 12-methylene carbon with ..." where the high context information is necessary to classify "7-" correctly. A similar example is "... 2,3-substituted ..." with a tagging of "2,3" as IUPAC with an offset conjunction of 1 but a correct result with an offset conjunction of 2.

4.4.2 Evaluation of Named Entity Recognition

Using the best configuration identified in the previous section, the resulting model is evaluated on the sampled MEDLINE test corpus *IUPAC-Test-M*. In Figure 4.6 different orders of the CRF and the offset conjunction together with tagging and training durations are depicted. Similar to the results estimated with bootstrapping on the training corpus *IUPAC-Train-M*, highest performance is obtained with the most context information included by a CRF order of 3 and an offset conjunction order of 2. The F_1 measure for IUPAC entities is 85.6% with a precision of 86.5% and a recall of 84.8%. The MODIFIER entities are found with an F_1 measure of 84.6% (91.7% precision and 78.6% recall). Higher orders have not been applied because of prohibitive training durations. However, it can be seen that the best result is obtained at the expense of a high training time and, what is more important, on a higher tagging time of 307 seconds than other configurations of the CRF. For tagging a higher amount of data like the full MEDLINE database one could prefer to use a faster configuration like the one with order 2 and offset conjunction of 1 which only takes 215 seconds for tagging the test corpus. The F_1 measure for IUPAC entities is lower with 77.7%, but the recall is nearly on the same level with 82.1% (MODIFIER: 55% precision, 78.6% recall). It can be concluded, that there is a trade-off between tagging time and performance, so it depends on the application which configuration should be preferred. That topic is revisited in Chapter 7.

The analysis of the errors shows frequent problems in the recognition of short chemical names. On the one hand chemical names are recognized by the system which are not specified as IUPAC-like. On the other hand, short names, similar to trivial names, specified as

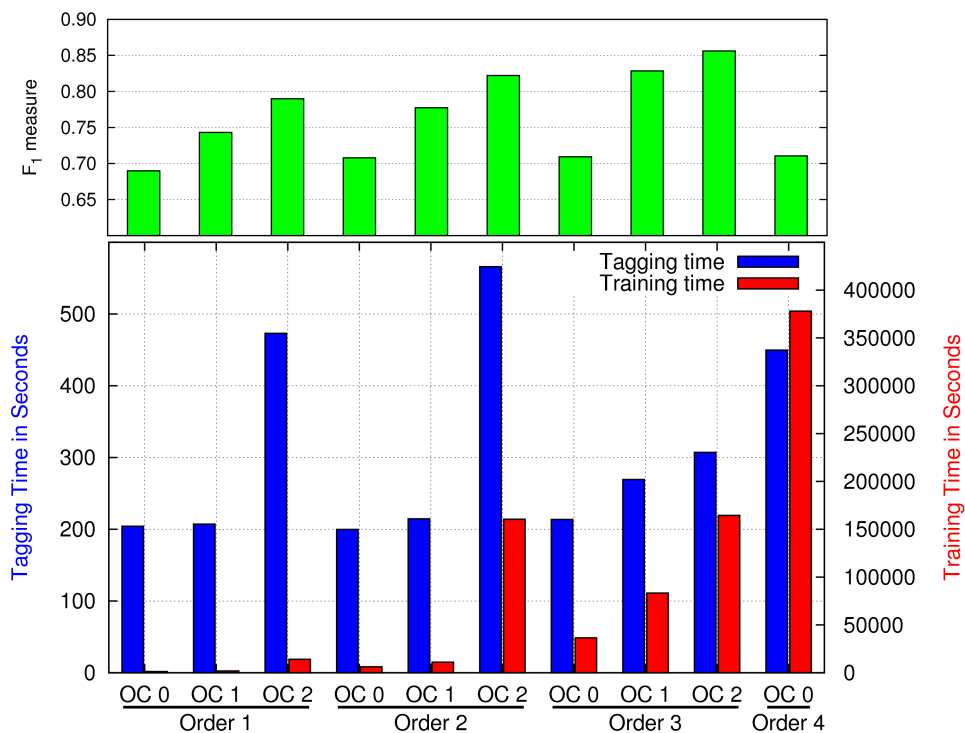


Figure 4.6: Results on the sampled MEDLINE corpus *IUPAC-Test-M* with different orders q of the CRF (given as Order q) and orders p of the offset conjunction (given as OC_p). The upper chart shows the F_1 measures for the different configurations, the lower one the tagging and training durations.

IUPAC-like by the annotators are most frequently unrecognized by the system. Nearly 50 % of the other false positive errors are boundary errors. In addition, names morphologically similar to IUPAC names like enzymes (e. g. “2-phospho-D-glyceratehydro-lyase” or “pyruvate O2-phosphotransferase”) are detected as false positive matches.

Applying the best system trained on the MEDLINE training corpus *IUPAC-Train-M* for tagging the patent test corpus *IUPAC-Test-P* shows a decrease in F_1 measure in comparison to the MEDLINE test corpus *IUPAC-Test-M* due to the bias of hand selecting difficult paragraphs instead of sampling from a set of sentences or text snippets: The F_1 measure is 81.5 % with a precision of 77.2 % and a recall of 86.4 %.

Applied on the CHEMISTRY-CORPUS (with several classes of chemical names) accepting all entities as true positive leads to 91.41 % precision and 29.04 % recall. The recall is naturally low as this corpus contains more and different entities than the system is trained for. The reason for the higher precision are detected trivial names only. Such occur as parts of IUPAC names and are therefore partly detected in addition.

Frequency	Name	Normalized
16811	N-methyl-D-aspartate	No
15275	5-hydroxytryptamine	Yes
11690	5-fluorouracil	Yes
9001	6-hydroxydopamine	No
7023	glucose-6-phosphate	No
6685	N-ethylmaleimide	Yes
5932	N-acetylcysteine	Yes
5178	12-O-tetradecanoylphorbol-13-acetate	No
5032	methyl	Yes
4742	N-acetylglucosamine	No
4311	benzo[a]pyrene	Yes
4164	3-methylcholanthrene	Yes
3991	4-aminopyridine	Yes
3931	2,3,7,8-tetrachlorodibenzo-p-dioxin	No
3979	5-hydroxyindoleacetic acid	No

Table 4.3: Top 15 found terms with their number of occurrences.

4.4.3 Annotation of full Medline and Normalization

Performing a run of the best CRF model on the full MEDLINE with 16,848,632 MEDLINE article entries (version as of July 13, 2007) belongs to the main motivations to develop such a model. In these entries, we have 8,975,073 abstracts. In titles and abstracts, altogether $2.2 \cdot 10^9$ tokens, are 1,715,263 IUPAC entities in 875,102 MEDLINE database entries. The tagging is performed on a computer cluster using 48 machines with two Opteron AMD double core processors with 2.6 GHz and 8 GB main memory on each machine in 76.65 hours (3.19 days). The operating system is Suse Linux Enterprise Server 9 (x86 64) with the Sun N1 Grid Engine 6.

From the found IUPAC entities, only 142,181 (16.24%) could be transformed to a structure with Opsin (see Section 4.3.5, without checking the correctness of the detected structure). Mapping to dictionaries as described in Section 4.3.5 leads to 517,746 normalized entities, 30.18% of all detected entities. Combining these normalizations with the ones from Opsin results in 562,229 normalized entities (32.78%). Inclosed are 44,483 entities which could not be normalized by mapping to a dictionary. These numbers are summarized in Table 4.5.

The top 15 found terms from MEDLINE are shown in Table 4.3, the top 5 of the converted structures in Table 4.4 together with the most often used terms which lead to the normalization.

To get an upper bound of convertible IUPAC names, 100,000 correct names are sampled from data provided by NCBI. From these, 30,028 (30%) are converted to structure information by OPSIN. This difference to the rate of names with a structure conversion with Opsin shows the

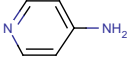
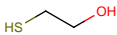
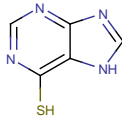
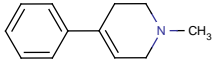
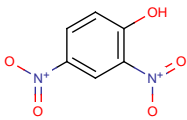
Frequency	SMILES and example names	Term freq.
4099	<chem>NC1=CC=NC=C1</chem>	
	4-aminopyridine	3991
	4-amino-pyridine	60
	4-Aminopyridine	36
		
3770	<chem>OCCS</chem>	
	2-mercaptoethanol	3696
	2-mercapto-ethanol	47
	2-Mercaptoethanol	19
		
2799	<chem>C1=NC2=NC=NC(=C2(N1))S</chem>	
	6-mercaptapurine	2766
	6-Mercaptopurine	20
	6-mercapto-purine	7
		
2607	<chem>CN1CCC(=CC1)C2=CC=CC=C2</chem>	
	1-methyl-4-phenyl-1,2,3,6-tetrahydropyridine	2416
	1-Methyl-4-phenyl-1,2,3,6-tetrahydropyridine	170
	methyl-4-phenyl-1,2,3,6-tetrahydropyridine	10
		
2457	<chem>OC=1C=CC(=CC=1[N+])(=O)[O-])[N+](=O)[O-]</chem>	
	2,4-dinitrophenol	2383
	2,4-Dinitrophenol	53
	(2,4-dinitrophenol)	11
		

Table 4.4: Top 5 found converted structures (applying OPSIN and CDK, drawn with Marvin (ChemAxon, 2007)) with their frequency and the frequency of occurrences of the top 3 terms which lead to the SMILES string.

Method	Number	%
All entities	1,715,263	100.00
Opsin	142,191	8.29
Dictionary Mapping	517,746	30.18
Opsin \cup Dic.Map.	562,229	32.78
Opsin \setminus Dic.Map.	44,483	2.59

Table 4.5: Number of normalizable IUPAC entities in MEDLINE.

difficulties to recognize partial names and incorrect use of IUPAC names.

4.4.4 Extending the Model to other Chemical Nomenclatures

To get an impression of the problem to find general chemical entity mentions in text, the annotation of *IUPAC-Train-M* with all classes described in Section 4.2 has been analyzed without modifications of the best performing IUPAC name detecting model. In such model with the classes ABB, FAMILY, IUPAC, PART, MODIFIER, SUM, and TRIVIAL, the questions arises, which classes should be combined and which should be separately handled, as they may be used in similar context and meaning.

This analysis is performed in a 10 fold cross validation. Different combinations of the classes are tested in one model for training, while during inference the classes are not respected. For instance the prediction of a FAMILY mention is counted as true positive if was labeled as TRIVIAL. The different settings are

- All classes separately labeled,
- All classes combined to one class,
- FAMILY and TRIVIAL combined, all others separated,
- FAMILY, TRIVIAL, and PART combined, all others separated,
- IUPAC and PART combined, all others separated.

The results are shown in Figure 4.7. Using the information which entity is part of which chemical nomenclature is beneficial, none of the tested combinations gave better results then separating all classes during training. That is surprising as it was assumed that chemical names share properties in usage despite of the nomenclature.

4.5 Summary and Discussion

In this chapter an approach of finding IUPAC-like terms in text with CRF is presented. The need for a machine learning based system, at least for the entity class of IUPAC and IUPAC-like

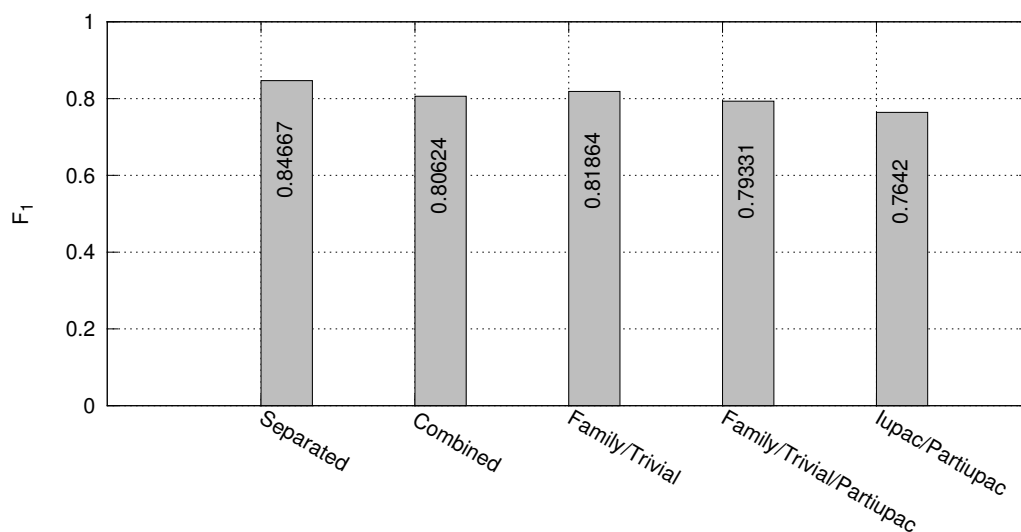


Figure 4.7: Results of a CRF trained on different combinations of classes of chemical names.

names has been motivated as dictionaries perform especially bad on those. The developed recognizer identifies IUPAC entities with an F_1 measure of 86.5% on a sampled independent test corpus built from MEDLINE. This corpus gives an estimation for all available abstracts from that database.

Using a tweaked corpus with correct IUPAC names shows that incorporating only complete IUPAC names in the training corpus is not sufficient. Obviously, the challenge is to recognize fragments and parts of IUPAC names. An error analysis of the final system on MEDLINE shows that boundary problems and the recognition of shorter chemical names lead to the main performance loss. This may be founded in ambiguities in the training data regarding these names and should be considered in a further extension of the training corpus. Preliminary results on an extended annotation of short names show an increase of precision to 91.4% (Kolářik, Klinger, et al., 2008).

When the IUPAC recognizer is applied to a hand-sampled patent corpus containing long enumerations and mixtures of different chemical nomenclatures the drop in performance is unexpectedly low with an F_1 measure of 81.5%. Apparently, the loss of F_1 measure in comparison to the MEDLINE corpus is due to a loss of precision rather than recall. Typical problems are finding the right borders of the chemical names in enumerations. It cannot be generalized from these experiments that it is harder to find IUPAC names in patents than in abstracts.

In the feature evaluation it is shown that automatically generated features like *Bag-of Words* and *Autom. Prefixes/Suffixes* together with *Space* information are the most important features influencing the performance of the system. The usage of combinations of these features alone e. g. *Space* together with *prefixes and suffixes* result in an F_1 measure of 76.03%. In contrast,

the *static morphological features* which are usually very important for a good generalization (together with other morphological features), in particular on the entity class of genes and proteins (as it will be discussed in Chapter 5 and by Klinger, Friedrich, et al. (2007)) do not have such a high impact here. Remarkably, the *Prefix/Suffix lists* (used for generalization purposes) appear to be of very low importance indicated by nearly no loss when left out. When used as the only feature, no positive result can be obtained. However, when combined with the feature *Spaces*, the results are surprisingly high (70.71 % F_1 measure).

Higher orders of the CRF in combination with high order offset conjunctions lead to the best results observed (F_1 measure 85.6 %) on the MEDLINE test corpus with an CRF order three and an offset conjunction of two. On this corpus also the direct dependency of training and labeling durations to the orders of CRF and offset conjunction are shown (cf. Figure 4.6).

Despite of the analysis given here for IUPAC entities, the open question remains, in which cases a representation of context information on the labels should be preferred in comparison to a representation of context information in the text, in form of features, used here by incorporating offset conjunction, a question which arises by the possibilities of the flexible formulation of CRFs. To my knowledge, no deeper analysis was published about that topic so far. Results for these parameters have been shown here for one class. Experiments with different orders of CRF and offset conjunction in the field of gene and protein names showed that with higher orders the results tend to get worse, probably because of more needed training data when more complex dependencies are modeled. Therefore, it can be concluded that such long entities of IUPAC names harbour special difficulties.

In a final test, the full MEDLINE was labeled showing the scalability of the implementation. The highest frequency (without normalization) is almost 17000 mentions of one term (Table 4.3). A conversion of the names to its corresponding structure with Opsin show that only a minor part (below 20 %) can be processed (without evaluating the correctness of the conversion). Straight-forward dictionary mapping to ChEBI and MeSH shows a much higher result with 30 %. But it is important to note, that this result is probably a characteristic of MEDLINE data, as entities in this data base are well known: Dictionary mapping will probably work worse on text introducing new chemical compounds.

The experiment of training with different types of chemical nomenclature emerged to be very promising. Recent work in the field of dictionary-based chemical NER showed the remaining problems with that approach (Hettne, Stierum, et al., 2009; Klein, 2010) with less than 55 % F_1 on CHEMISTRY-CORPUS, therefore it is proposed that chemical named entity recognition should be approached in general with machine learning (showing here a result¹⁵ of more than 80 % F_1). The normalization has been shown to be promising with dictionary mapping and is expected to work well for other chemical mentions than IUPAC, too.

¹⁵ Note that the evaluation was performed on different abstracts. But the huge difference in results is nevertheless promising.

Chapter 5

Recognition of Gene and Protein Names¹

5.1 Introduction

The recognition of IUPAC and IUPAC-like names (in Chapter 4) is difficult because of the infinite number of names and their special structure which makes it hard to find the correct starting and ending offsets. In contrast, the challenge of detecting gene and protein mentions (between which commonly is not differentiated as gene names may be used for the related protein and vice versa) is that such names hardly follow comprehensible rules. Terms describing such entities may additionally be common words, acronyms with multiple meanings or names for protein families. Therefore, disambiguation using context is generally necessary. Additionally, an automated system should be able to detect newly invented names (Leser and Hakenberg, 2005). Examples for gene name mentions are depicted in Table 5.1 illustrating the differing morphology and the lack of naming conventions². Next to these names, abbreviations are commonly used.

The BioCreative competitions I and II (Hirschman, Yeh, et al., 2005; Wilbur, Smith, and Tanabe, 2007; Smith, Tanabe, et al., 2008) focused on named entity recognition of gene names in one sub-task. The top scoring results were reasonable with 83 % F_1 measure in BioCreative I (Yeh, Morgan, et al., 2005).

In comparison with the task of person and organization name recognition, this value is still

Name	Synonyms
cheap date	amn, amnesiac
maggie	mge
I'm not dead yet	INDY
Pray for Elves	PfE
really interesting new gene	ring
forkhead box D4-like 3	FOXD6, FOXD413
Alopecia with mental retardation syndrom 2	APMR 2
peroxisome proliferator activated receptor alpha	PPAR, Ppara

Table 5.1: Examples for gene names.

¹ This chapter includes the work by Klinger, Friedrich, et al. (2007) and Smith, Tanabe, et al. (2008).

² Examples partly from <http://jpetrie.myweb.uga.edu/genes.html>, access date 08/23/2010; and K. Cohen (2009); Azov (2005)

comparatively low, where the top result in 2003 was 88.76 % F_1 measure (Florian, Jing, and Zhang, 2003). One problem in gene name recognition is to produce a consistent annotation (Leser and Hakenberg, 2005). To address this issue, the task in BioCreative II was defined with multiple annotations, presumably by multiple annotators. This led to a top scoring result of 87.21 % F_1 because of the acceptance of fuzzy entity boundaries (by multiple annotations of an entity)—the top scoring system did not use multiply annotated training data (Ando, 2006).

The question arises if the information of multiple annotators can be used in a system for gene name recognition—only two out of 21 participating systems used this additional information. The work presented in this chapter was the better scoring one, ranking 4th of all contributions. The 3rd and 2nd ranking systems incorporated dependencies in the opposite direction as common (features had the form $f(y_i, y_{i+1}, \vec{x}, i)$ instead of $f(y_{i-1}, y_i, \vec{x}, i)$) (Huang, Lin, et al., 2006; Kuo, Chang, et al., 2006), detailed information can be found in a later publication (Hsu, Chang, et al., 2008). The top ranking system used unlabeled data in a semi-supervised scenario based on support vector machine classification (Ando, 2006).

Therefore, the main innovation in this chapter is the proposal of a method to combine data from multiple annotators which are often available anyway to assess inter-annotator agreement. An adaption of the CRF workflow including an exhaustive parameter analysis is presented, similarly to the previous Chapter 4, which allows to learn about class specific characteristics. Additionally, an analysis of one important claimed advantage of machine learning-based gene name recognition, namely the ability to detect newly invented names which were not existing during system engineering or training time, really holds. For that, artificial data sets for several years are generated and an evaluation is given how stable a system remains.

5.2 Named Entity Recognition Workflow

5.2.1 Entity Types, Corpus Selection and Annotation

The entity type defined in the BioCreative II data set is GENE/PROTEIN. The data consists of 15,000 sentences as training data and 5000 sentences as test data, both from MEDLINE. There is no information available how the instances were collected.

The annotations are split into a gold standard and acceptable alternatives. The training gold standard has 18,265 annotations and 14,499 alternatives. The test data has 6,331 annotations with 5,068 alternatives.

One characteristic of this gold standard and alternatives is that there seems to be no rule which annotation is an alternative or in the gold set. For example in the sentence “*On the other hand factor IX activity is decreased in coumarin treatment with factor IX antigen remaining normal.*” the gold standard is the twice annotation of *factor IX*. The alternative annotation gives the information that finding *factor IX antigen* is just as well. But in “*The arginyl peptide bonds that are cleaved in the conversion of human factor IX to factor IXa by factor XIa were identified as Arg145-Ala146 and Arg180-Val181.*” the gold standard is defining *human factor*

IX and *factor IXa* and *factor XIa* (as highlighted in blue) but the alternative allows *factor IX* instead of *human factor IX* (as underlined).

5.2.2 Tokenization and Feature Extraction

To split sentences into tokens, three different variants are taken into account. Firstly, the one proposed by Settles (2005). His available implementation ABNER splits at all special symbols, including number-letter changes in words, brackets, and hyphens. Hyphens are not recognized as singular tokens but stay connected with the succeeding token. Therefore, 5-nucleotidase is tokenized as 5 -nucleotidase. Additionally, separating dashes is tested (leading to 5 - nucleotidase). The last tokenization configuration is to keep number and letter changes together, as gene and protein names frequently contain such.

The feature set includes all the ones presented in Section 2.6. Part-of-Speech tags are extracted with GeniaTagger³ (Tsuruoka, Tateishi, et al., 2005). The output of the dictionary and rule-based gene and protein recognition system ProMiner, namely if a token is part of a gene or protein name by means of an EntrezGene-based dictionary is a domain specific feature. Next to this feature of external source, the inclusion of a token in different other dictionaries is used: A positive guess is provided by a list extracted from the HUGO gene nomenclature committee⁴. Negative examples of entities which could have a similar morphology are in lists of amino acids, enzymes, cell lines, organisms, and units, additionally a list of stopwords, all provided by the distribution of ABNER. Compiled in our group at SCAI is another list of species, one of acronyms, and one of chemical names. For all dictionary features, another feature is generated which holds if a token in the proximity of 5 tokens of the current one is part of the dictionary.

5.2.3 Postprocessing

Gene names frequently include brackets, for instance as in NF(H) promoter, C3(D) epitopes, Lp(a), or d(T2AG3T). As a CRF can capture only a limited context around a token without an explosion of the feature number, the problem of not detecting the whole pair of opening and closing brackets is addressed in a postprocessing step, analogously with quotes, several cases are differentiated.

If there is a lonely closing bracket at the end of an annotation, it is removed. If there is a missing closing bracket and the next character after the annotation is a closing bracket, this is added. If there is a missing bracket at the beginning, this is added. Examples are shown in Table 5.2.

³ <http://www-tsujii.is.s.u-tokyo.ac.jp/GENIA/tagger/>

⁴ <http://www.genenames.org/>

Text Fragment	Guess	Correction
(anti-IIa) and Holmer et al.	anti-IIa)	anti-IIa
a plant homeodomain (PHD) finger and Y15172 (Surf-5).]	plant homeodomain (Y15172 (surf-5	plant homeodomain Y15172 (surf-5)
contained a (dG-dC)7 segment	dG-dC)7 segment	(dG-dC)7 segment

Table 5.2: Examples for correction of brackets in gene name recognition.

5.3 Multi-Model approach

The workflow description in Sections 5.2.1 to 5.2.3 focused on the general process for gene and protein name recognition. The question how to deal with multiple annotations as introduced in Section 5.1 was not addressed so far. Keeping in mind the problem of ambiguities in the different gold annotations explained in Section 5.2.1, the approach presented here is heading into this direction. It is assumed that annotators tend to repeat their differences between each other. For instance, one annotator always marks the word gene behind a gene name and the other does not.

Thus, the approach is to build one annotation out of the shortest possibilities and one out of the longest ones, each without overlaps. If no alternatives are available, the annotation is used in the set of long as well as of short annotations. One CRF is trained on each set, respectively. At inference time, both models are applied and the results need to be combined to one single annotation. Three different strategies are tested:

1. Use long annotation first, then add short annotation (without overlaps)
2. Use short annotation first, then add long annotation (without overlaps)
3. Greedy: Combine both (with overlaps)

For instance let us assume to have `fibrinogen degradation products` as the annotation from the model trained on long annotations and `fibrinogen` and `FDP` the annotations from the model trained on short annotations on the text part `fibrinogen degradation products (FDP)`.

The results from the three methods are

1. `fibrinogen degradation products` and `FDP`
2. `fibrinogen` and `FDP`
3. `fibrinogen` and `FDP` and `fibrinogen degradation products`

In the second method nothing is added because the long annotation overlaps with the short annotations. The greedy method does not necessarily downgrade the results as alternatives are available for evaluation as well. The first method and last method seem to be appropriate

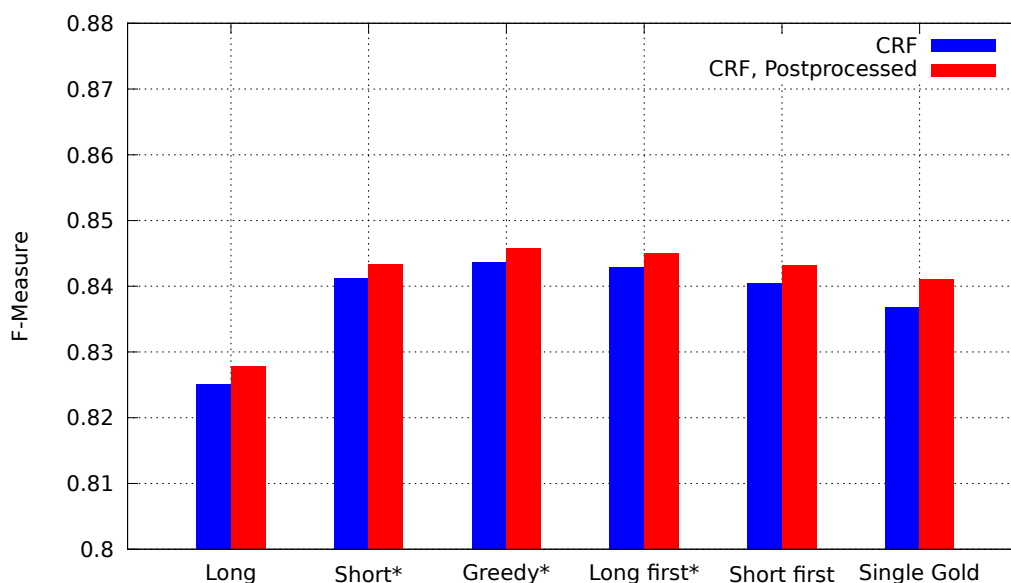


Figure 5.1: Results for combination methods of the multi model output on the BioCreative II named entity recognition training data set, evaluated via 50-fold bootstrapping.

because of a possible improvement in comparison to the two separate annotations as we will see in Section 5.4.

5.4 Results

For evaluation, the F_1 measure (Equation 2.6) is used. Identifying the values for true positives, false positive, and false negatives works slightly different with multiple acceptable annotations: If a guess annotation matches one of the given gold or alternative annotations, it is counted as true positive, as false positive otherwise. If another guess matches another alternative, while an annotation exists which overlaps both, it is not counted, neither as false positive nor as false negative.

5.4.1 Multi Model Combination

As the first step in evaluation, the different combination approaches are tested. The results of the multi model output is shown in Figure 5.1, evaluated with 50-fold bootstrapping (the ones marked with a * were submitted to the competition). The combination with the long form as a basis or the greedy combination are superior to all other approaches, better than using only the single provided gold standard. Interestingly, using only the short annotation is similarly good. Comparing the values for precision and recall (Table 5.3) of these three best results, the greedy combination has the highest recall, as expected. Using the long annotation first is a good trade-off and the short annotation alone is precision-focused. Using

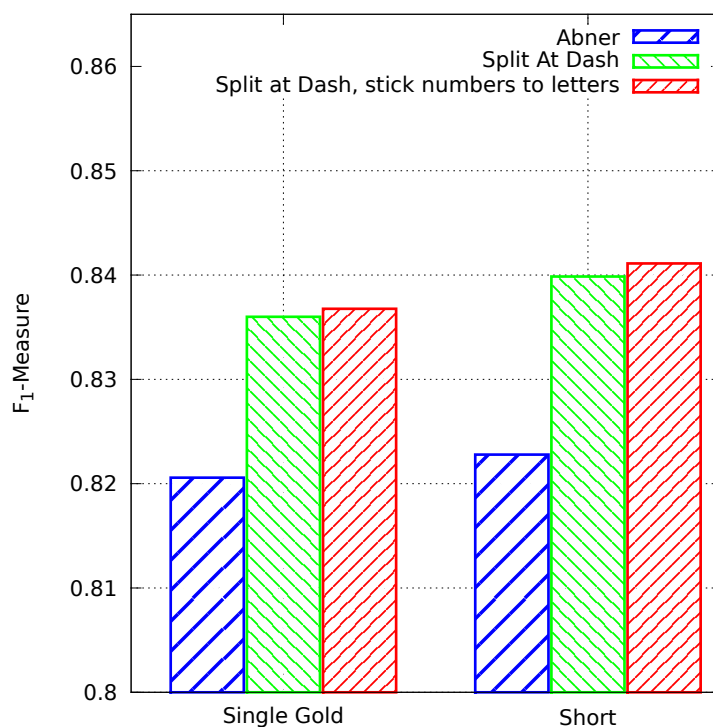


Figure 5.2: Differences in tokenization strategies for the BioCreative II named entity recognition training data set, evaluated via 50-fold bootstrapping.

the long annotation alone is much worse; the differences between long annotations and short annotations which are used because of no alternative in the data may be too large.

5.4.2 Parameter Selection

The selection of features or feature generating patterns (*cf.* Section 2.6 and 5.2.2) and the type of tokenization (*cf.* Section 5.2.2) to gain an optimized result are evaluated via 50-fold bootstrapping (Section 2.8).

The differences in tokenization are depicted in Figure 5.2. The approach to stick the dashes to the next token as in Abner is counter-productive. Splitting numbers from letters only leads to a small decrease in performance.

The results for different feature sets are shown in Figure 5.3. The histogram shows the decrease in performance leaving out a feature or feature generating pattern. Morphological features are of overwhelming importance, leaving them out leads to a decrease of about 18%. Offset conjunction has a high impact of 2%. Interestingly, ProMiner has only a minor impact on the training set but improves results on the test set (by 2%). The other dictionaries influence the result only a bit—while it makes the model much more complex with the need to store the whole dictionaries. Note that prefixes and suffixes of length 3 and 4 have

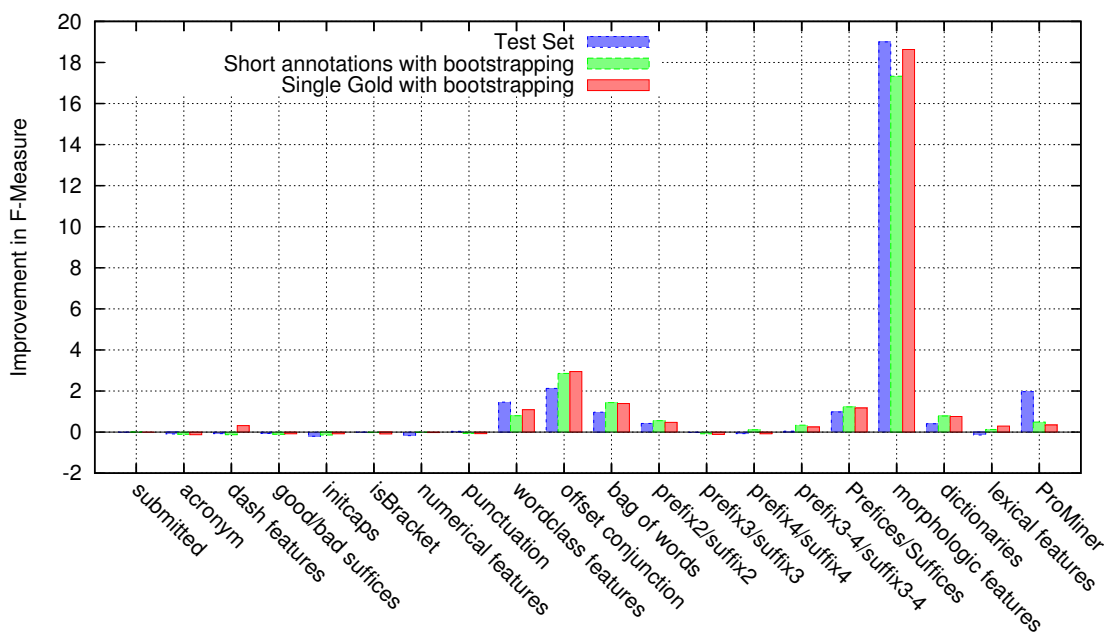


Figure 5.3: Results for different features sets for the BioCreative II named entity recognition training data set, evaluated via 50-fold bootstrapping.

no positive impact, but the combination of both—that demonstrates the non-monotony in feature search.

5.4.3 Evaluation of Named Entity Recognition

The results on the final test set are shown in Table 5.3 and in Figure 5.4. The best result is achieved using the long annotation first, adding the short ones, with 86.33%. The result using only the provided single gold data is 85.64%, therefore the multi-model approach shows a slight increase in performance. Additionally, with the greedy approach a much higher recall can be achieved (7.1 more in %) with only a small loss in F_1 . It can be summarized, that the approach to select annotations with the shared property of maximal or minimal length for a model respectively is meaningful.

The source for errors are mainly in two classes: Enumerations and abbreviations. Frequently only the first or last entity is found, while enumerated parts of the name are not detected. To solve that, the enumerations needed to be resolved to mentions of full entity names. Abbreviations are challenging as their morphology does not provide sufficient information to distinguish between genes/proteins and other classes. Therefore the model learns that the concatenation of capital letters is more likely to be not an entity of interest.

Model	Bootstrapping on Trainingset			On Testset		
	Precision	Recall	F_1	Precision	Recall	F_1
Long	86.30 (0.0065)	79.53 (0.0094)	82.78 (0.0064)	87.41	80.29	83.70
Short*	86.87 (0.0054)	81.94 (0.0106)	84.33 (0.0069)	88.57	83.83	86.13
Greedy*	80.21 (0.0069)	89.47 (0.0057)	84.58 (0.0047)	82.02	90.63	86.11
Long first*	85.38 (0.0060)	83.63 (0.0079)	84.50 (0.0055)	87.27	85.41	86.33
Short first	83.83 (0.0063)	84.81 (0.0065)	84.32 (0.0048)	85.50	85.61	85.56
Single Gold	86.61 (0.0071)	81.76 (0.0123)	84.11 (0.0076)	87.86	83.53	85.64

Table 5.3: Results on the trainingset (in %, averaged over 50 bootstrap replicates) and on the test set after postprocessing and disambiguation (Standard deviation is given in brackets).

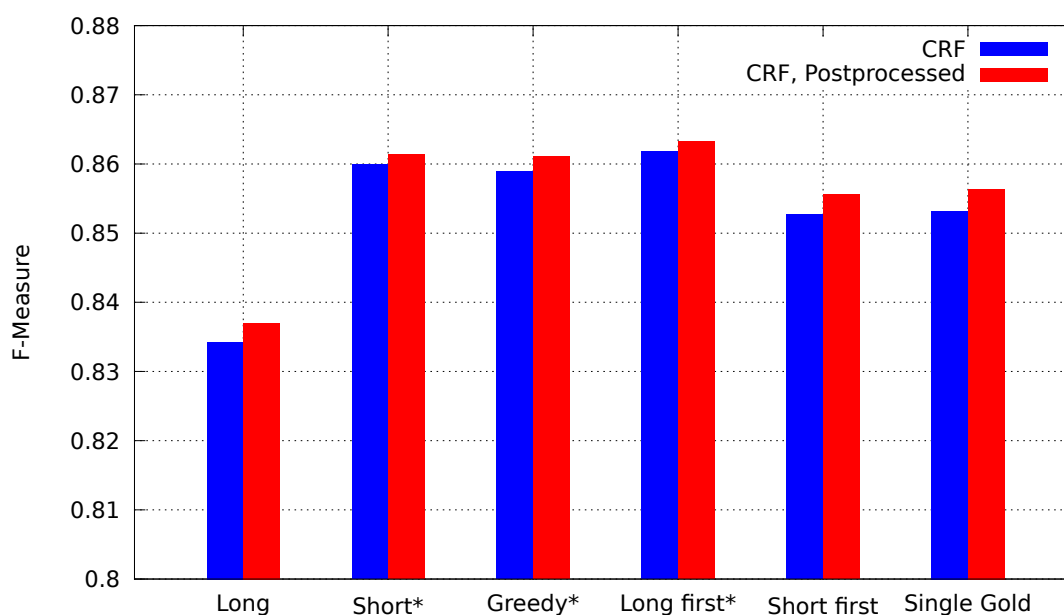


Figure 5.4: Results for combination methods of the multi model output on the BioCreative II named entity recognition test data set.

Disambiguation approaches could help here, as well as lists of frequent false negatives as a feature. Additionally, finding the correct boundaries of an entity is a problem, although in the setting of this task multiple annotations are available. Especially long mentions including numbers and hyphens seem to be challenging.

5.4.4 Generalizability and Stability over Time

One of the most often claimed advantages of a machine learning-based approach compared to a dictionary-based approach is that newly invented names can be found. This is especially interesting in Gene and Protein name recognition, where dictionary-based approaches gain comparable results. Why should one accept the disadvantage of the normalization being an additional step?

In the following, an analysis how stable the presented machine learning-based named entity recognition remains over years and decades is presented. The experimental setting is as follows: The whole MEDLINE data is tagged with ProMiner (Hanisch, Fundel, et al., 2005) with gene and protein names. From all sentences in all abstracts published in the years 1980, 1985, 1990, 1995, 2000, 2005, and 2009, exactly 10,000 instances are sampled respectively, which contain at least one entity. This leads to 7 data sets with 10,000 sentences. This procedure supports the same distribution of entities in all years while they are actually more sparsely spread in earlier abstracts.

The data from each of the 7 selected years is split into half training and half validation data. On each training set a model with the feature configuration as described in this chapter (except the external dictionary feature based on ProMiner) is trained and tested on each of the 7 validation sets. The result is depicted in Figure 5.5. Each line shows one model trained on 1980 to 2009 respectively, specified by color and line type and additionally by a cross. The values shown are the results on independent test sets from the different years.

The F_1 measure drops clearly. The system trained on 1980 data loses about 3% in 5 years, 12% in 10 years, and 27% in 29 years. The one trained in 1995 loses 4%, 6%, and 7% in 5, 10, and 14 years respectively. An up-to-date system applied to data from the past is performing worse, but not as bad as a system applied to future data.

Comparing precision and recall shows that mainly the recall is decreasing, *e. g.* 12% for the 1995-system till 2009. Therefore, it may be assumed that especially newly introduced entities are problematic to be found. The graph in Figure 5.6 shows the results on filtered test sets. All entities on which a model was trained are removed, identified by string identity. Therefore, only for “newly invented” mentions is tested. Here, the recall does not decrease, but is lower in general. That shows, that 40% to 60% of new entities can be found. Such machine learning system can generalize to new entities, though only to a limited degree.

Note, that the use of ProMiner with dictionaries extracted from up-to-date databases may be a limitation of this analysis. Nevertheless, it supports the hypothesis that a model only is stable over time to a certain degree. The approach of data generation can also be assumed to be the reason for the observed increasing performance from past to present as it can be seen in Figure 5.6: The recall of ProMiner may be higher on newer data.

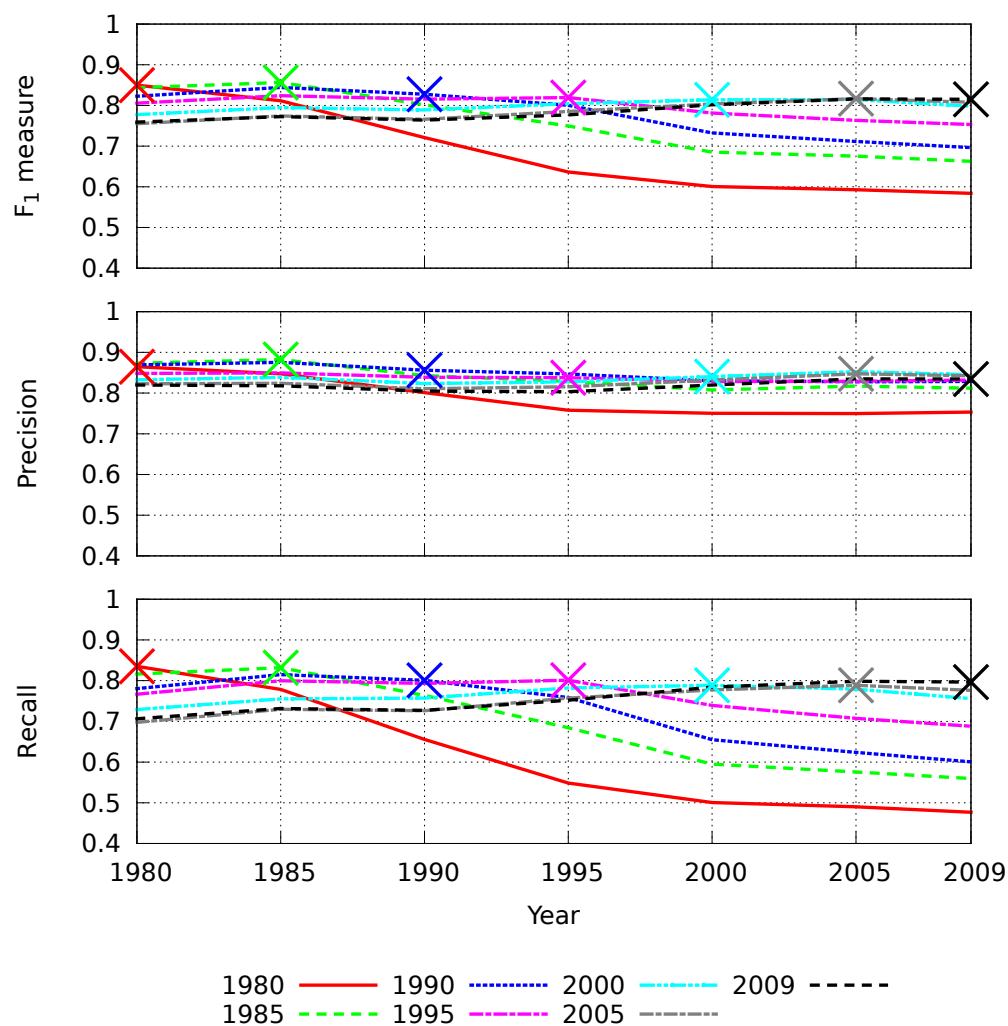


Figure 5.5: Named entity recognition models for gene and protein recognition trained on yearly sub-samples applied to independent sub-samples from different years.

5.5 Summary and Discussion

In this chapter an approach of finding gene and protein mentions in text with CRF is presented. The advantage of a machine learning based system is motivated by the ability to detect newly invented names. It is explained how multiple annotation on the same text can be incorporated into the workflow to build and apply a model. The developed recognizer identifies entities with an F_1 measure of 86.33% on the test corpus provided by BioCreative II, built from MEDLINE. This result is ranking 4th of 21 in the competition and the best incorporating the provided multiple annotations.

The feature evaluation shows that morphological features are the most important ones, catching structural characteristics in contrast to a dictionary-based approach. Together

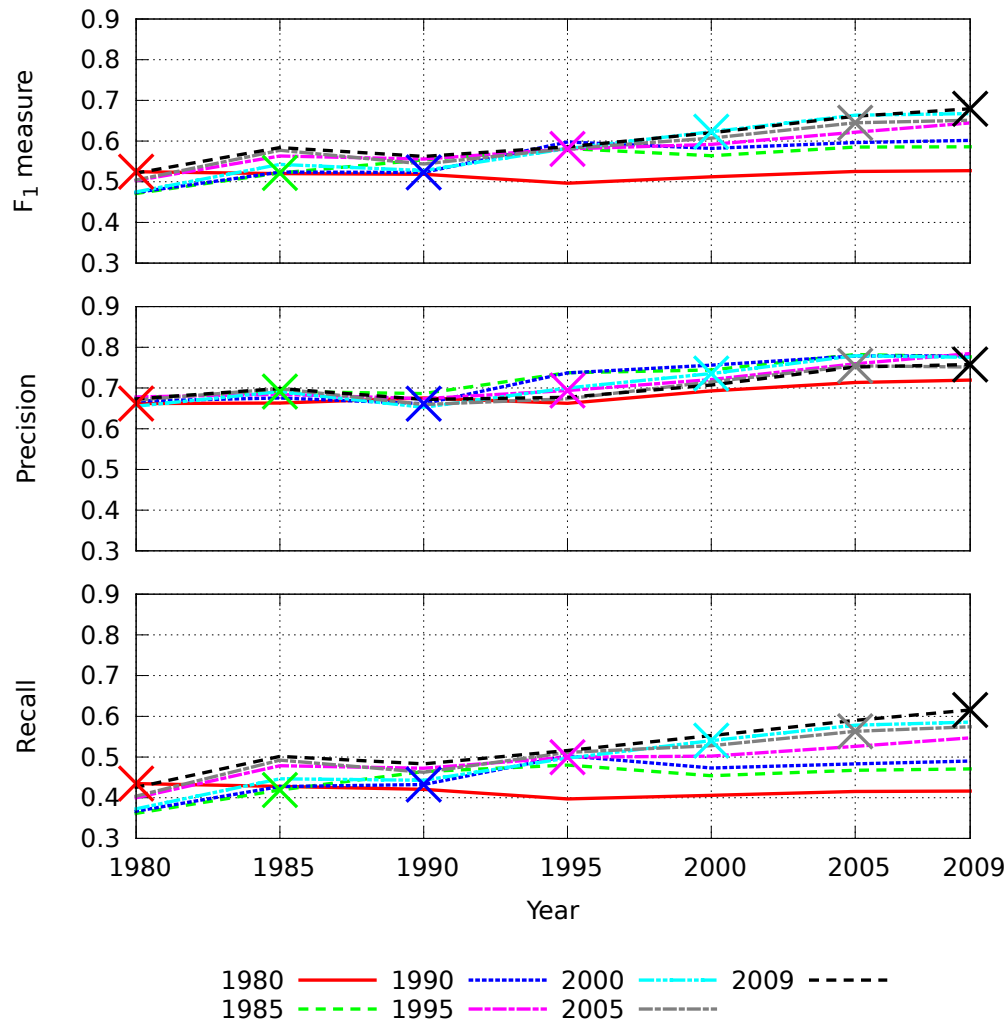


Figure 5.6: Named entity recognition models for gene and protein recognition trained on yearly sub-samples applied to independent sub-samples only containing new entities from different years.

with the offset conjunctions which measure contextual information, it belongs to the most important features. The intelligent selection of the available annotations for different models shows an impact, especially the ability to build methods with a higher recall (choosing between precision and recall-focused systems is discussed in more detail in Chapter 8).

The analysis how stable a model stays over years and decades shows that most problems are hidden in the invention of newly introduced gene names. Not surprisingly respecting those, the recall goes down to nearly half of the result of all mentions which occur in text. Nevertheless, it is presented that the model introduced in this chapter is able to catch characteristics of the mentions which go beyond dictionary-approaches. While there was no success to increase the generalizability with domain adaption methods taking the distribution of each feature into account, it can be assumed that this should be investigated further as future work. Otherwise training sets need to be updated regularly and models be retrained.

Chapter 6

Recognition of Mentions of Single Nucleotide Polymorphisms¹

6.1 Introduction

Sequence variations, in particular Single Nucleotide Polymorphisms (SNPs), are considered key elements in fields such as genetic epidemiology and pharmacogenomics. Researchers in these areas are interested in finding genes associated with clinically relevant phenotypes, such as diseases or drug responses, as well as in selecting the relevant sequence variants on candidate genes for genotyping studies. Information on sequence variations can be found at public resources such as dbSNP (Smigielski, Sirotkin, et al., 2000) and HapMap (International HapMap Consortium, 2005). The NCBI database dbSNP serves as a central repository for both SNPs and short deletion and insertion polymorphisms. It currently contains 64,607,439 variations for 58 different organisms, each represented by a unique identification code (the “refSNP” or “rs number”)²

The mapping of variations mentioned in texts to a unique database identifier (normalization) is important from a biomedical perspective, because it provides the biological context to the variations. Mapping a variation entity to a dbSNP identifier allows to link a text entity to a database entry and thus enriches context. In consequence, all the information available for this variation can be obtained: organism, genome location, validation status, populations in which the variant has been sequenced, biological sequences where the variant has been mapped (gene, mRNA, protein) and other pieces of information (*cf.* Section 2.7).

Finding the variation mention alone is interesting for reasons similarly to the entities in Chapters 4 and 5: It can improve information retrieval tasks for database curators (McDonald, Winters, et al., 2006) and allow semantic searches, as described in Section 1.3.4.

From the linguistic point of view, this class of entities is interesting because the relevant information is commonly described in natural language. Similarly to IUPAC names, one could think of an arbitrary number of possible mentions, therefore a dictionary-based method is likely to fail. In contrast to gene and protein names, the relevant information is not frequently specified with a given name. A more detailed analysis of linguistic aspects of mutation mentions is given by K. B. Cohen (2008).

The main contribution in this chapter is the adaption of the workflow to the class of SNPs and the discussion of necessary steps for a successful normalization. Similarly to the other

¹ This chapter includes partially the work by Klinger, Furlong, et al. (2007) and by Thomas, Klinger, et al. (2011).

² http://www.ncbi.nlm.nih.gov/projects/SNP/snp_summary.cgi, accessed 09/08/2010

chapters of Part II, a feature and parameter analysis is shown, revealing characteristics of the specific classes of interest.

Related Work

Contrasting to the extensive research carried out in the field of gene and protein name entity recognition, only few initiatives have been directed to the task of retrieval of SNPs and other types of sequence variants from the literature. Quite similarly to the problem of detection of gene and protein names from biomedical literature, the identification of sequence variants is hampered by the lack of use of a standardized nomenclature³ (Dunnen and Antonarakis, 2001) and by the ambiguity of the terms under use. Even though some journals like “Human Mutation” enforce the use of the variation nomenclature⁴, not all journals do so and we are confronted with a large backtrack of articles lacking these standards.

The first report on this subject was the approach called MuteXt, which was focused in collecting single point mutations for two pharmaceutically interesting protein families, nuclear hormone receptors (NR) and G-protein coupled receptors (GPCR), with the aim to populate a database (Horn, Lau, and Cohen, 2004). The method searched full text articles for mutations on these protein families using regular expressions. The authors reported a recall of 49.3 % and 64.5 %, and a precision of 87.9 % and 85.8 % (for GPCR and NR, respectively). A continuation of this work is the approach of the application “Mutation GraB” (Lee, Horn, and Cohen, 2007), aimed at the identification of protein point mutation across different protein families. The system identifies terms representing point mutations, organism and protein names using regular expressions, and then associates those terms by means of a graph bigram approach, achieving an overall F_1 measure of 75 %.

A related approach has been implemented in MEMA (Rebholz-Schuhmann, Marcel, et al., 2004). In this work, regular expressions are used for the extraction of polymorphism-gene pairs from MEDLINE abstracts. A difference with MuteXt is that it considers polymorphisms of the substitution type both at the nucleotide and the amino acid levels. Nevertheless, the MEMA system achieved a higher performance (75 % recall and 98 % precision) for the extraction of allelic variants from texts.

Similarly, the application *Mutation Finder* (Caporaso, Baumgartner, et al., 2007) uses a set of regular expressions to extract point mutations. It maps different types of mentions to a standard form, while no association with a database is performed. The stated performance for recognition is a recall of 92 % and a precision of 98 %.

The entity tagger Vtag (McDonald, Winters, et al., 2004) was developed for the retrieval of several types of polymorphisms and mutations (point mutations, translocations and deletions) related to cancer from the literature. It is based on CRFs. The reported performance is 85 % precision, 79 % recall and 82 % F_1 -measure.

Although these methods achieve good results at the performance level, none of them incorporate allelic variation data from sequence databases (e. g. dbSNP) and neither do they

³ <http://www.hgvs.org/mutnomen/>

⁴ <http://www3.interscience.wiley.com/cgi-bin/jabout/38515/ForAuthors.html#CONVENTION>

tackle the problem of the normalization of the variation entities identified so far. These features, however, are incorporated in the OSIRIS system (Bonis, Furlong, and Sanz, 2006). OSIRIS integrates different sources of information and incorporates ad-hoc tools for synonymy generation with the aim of retrieving literature about the SNPs of a gene. The retrieval is performed using the PubMed search engine. Although the recall was not assessed, it achieved a high precision level (82%). In addition, it provides a first way of linking a dbSNP entry with the articles referring to it. The OSIRIS system uses a query expansion approach, which starts from the entries of dbSNP. This approach increases precision but limits the possible recall to normalizable variants.

In the following, a workflow of gene and protein name recognition with a conditional random field to recognize variation mentions is presented. That allows for identifying normalizable mentions as well as those which can be found in dbSNP.

6.2 Named Entity Recognition Workflow

The method described is aimed at the conditional random field-based identification (CRF, as described in Section 3.4) of variation terms in biomedical texts which are combined with gene and protein mentions to prepare a normalization. For the gene mentions, the dictionary-based named entity recognition system ProMiner (Hanisch, Fundel, et al., 2005) is used. The workflow of the system involves two steps: first, several entities (described in Section 6.2.1) are identified and tagged using CRFs, ProMiner, and regular expressions. This forms the foundation for the second step, the normalization.

Figure 6.1 shows an example abstract with highlighted entities.

6.2.1 Entity Types

The entities of interest are GENE/PROTEIN and VARIATION. For the variation entities, the description and annotation guidelines⁵ defined by the Institute for Research in Cognitive Science at the University of Pennsylvania are followed, which are used for the BioTagger (McDonald, Winters, et al., 2004) (which includes the functionality of VTag). A variation is defined as a small change in the nucleotide sequence of the genome. Variations can be mapped to a gene locus in its coding or non-coding regions, and thus exert effect at the level of protein function or gene function. Examples of variations are SNPs, short insertions and deletions, named variations as Alu sequences, and other types of variations represented in the dbSNP database. From the point of view of a NER system, a VARIATION entity is defined by the combination of tokens that specify the following pieces of information: location of the variation, alternate alleles of the variation (original or/and altered) and the type of the variation. While LOCATION and STATE are obligate requirements to define a normalizable VARIATION entity, *type* can be missing. For the normalization an entity from the class GENE/PROTEIN is typically required. In Figure 6.1, the underlined terms illustrate the

⁵ http://bioie ldc.upenn.edu/wiki/index.php/Main_Page

BACKGROUND AND PURPOSE: The collagen alpha2(I) gene (COL1A2) on chromosome 7q22.1, a positional and functional candidate for intracranial aneurysm (IA), was extensively screened for susceptibility in Japanese IA patients. METHODS: Twenty-one single nucleotide polymorphisms (SNPs) of COL1A2 were genotyped in genomic DNA from 260 IA patients (including 115 familial cases) (mean age, 59.9 years) and 293 controls (mean age, 61.6 years). Differences in allelic and genotypic frequencies between the patients and controls were evaluated with the chi(2) test. Circular dichroism spectrometry was monitored with collagen-related peptides that mimic triple-helical models of type I collagen with Ala-459 and Pro-459 to estimate the conformation and stability of alterations. RESULTS: Significant genotypic association in the dominant model was observed between an exonic SNP of COL1A2 and familial IA patients (chi(2)=11.08; df=1; P=0.00087; odds ratio=3.19; 95% CI, 2.22 to 6.50). This SNP induces Ala to Pro substitution at amino acid 459, located on a triple-helical domain. Circular dichroism spectra showed that the Pro-459 peptide had a higher thermal stability than the Ala-459 peptide. CONCLUSIONS: The variant of COL1A2 could be a genetic risk factor for IA patients with family history.

state, location, gene, type

Figure 6.1: Example abstract (Yoneyama, Kasuya, et al., 2004) with tagged entities STATE, LOCATION and TYPE, and GENE which form the entity set VARIATION (underlined in one example).

entity set VARIATION. Accordingly, the selected entity classes of interest for the annotation of variations are

- TYPE like *deletion, single nucleotide polymorphism or insertion*
- LOCATION like *codon 6* or *position 30* in “A/G single nucleotide polymorphism (SNP) at position 30”, or *-1131* in “-1131T>C”
- STATE-ORIGINAL like *Gly* in “Gly->Ala”, *A* in “A/G single nucleotide polymorphism (SNP) at position 30”
- STATE-ALTERED like *Ala* in “Gly->Ala”, *G* in “A/G single nucleotide polymorphism (SNP) at position 30”
- STATE-GENERIC when it is unclear if the original or the altered state is meant, like *Pro* in “Pro-459”

Corpus	# Abstracts	Numbers of entities		
		Type	State	Location
Seed	207	635	1057	734
Train	440	1688	3166	2120
GENETIC-TEST	247	107	58	58
RANDOM-TEST	1000	79	74	40

Table 6.1: Summarization of Corpora developed for the detection of Single Nucleotide Polymorphisms.

6.2.2 Corpus Selection and Annotation

Variation mentions are comparatively rare in MEDLINE, therefore, a seed corpus was manually assembled from the results of a PubMed query⁶ for

```
"Pathological Conditions, Signs and Symptoms"[MeSH] AND "Polymorphism,
Single Nucleotide"[MeSH] AND "Humans"[MeSH] AND hasabstract[text] AND
English[lang] AND ("2004/01/01"[PDAT] : "2005/01/01"[PDAT]) AND
"Chemicals and Drugs Category"[MeSH]
```

These instances were filtered to contain a GENE/PROTEIN entity (recognized with ProMiner). An intermediate system with these 105 MEDLINE abstracts produced frequent false positives—from the set of abstracts containing those, another 102 abstracts are included in the seed set. This seed set is used in the following for parameter selection and adaption of the model. Approximately half of the abstracts with positive examples (61/105) refer to variations that can be mapped to a dbSNP identifier. For the final training, an enriched set is assembled via confidence-based active learning (*cf.* Section 2.2). It contains 440 abstracts, including the seed set.⁷

For evaluation of the named entity recognition, two test corpora are generated: One sampled randomly from MEDLINE, containing 1000 abstracts (referred to as RANDOM-TEST) and one containing 247 instances from MEDLINE abstracts with the MeSH classification “Genetic” being more specific to the domain of interest (referred to as GENETIC-TEST).

Table 6.1 summarizes the numbers of entities in the different corpora.

6.2.3 Tokenization and Feature Extraction

As the system should mainly be applied to full MEDLINE, no sentence splitting is applied but full abstracts are used as instances. The tokenization is similar to the one proposed for IUPAC and IUPAC-like entities as described in Section 4.3.3; the main difference is the need to split at number-letter changes. The frequently occurring structure of the text example 214C->T points that out: Here, 214 should be annotated as LOCATION, C and T as STATES.

⁶ <http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?db=PubMed>

⁷ Active learning performed in a Master’s Thesis co-supervised by the author (Thomas, 2008).

Labels	...	O	B-state	B-location	B-state	O	B-type
Text	...	or	A	55	V)	single
Labels	I-type	I-type	I-type	O	B-location	I-location	...
Text	-	nucleotide	polymorphism	in	exon	4	...

Figure 6.2: Example for observation and label sequence for the text snippet: “...or A55V) single-nucleotide polymorphism in exon 4 ...” after tokenization.

To enable a model to separate the string into these entities, such fine-grained tokenization is necessary. Note, that splitting at hyphens is analogously applied. The tokenization is additionally exemplified with IOB (cf. Section 2.4) in Figure 6.2.

Next to the commonly used morphological features as described in Section 2.6, special variation related features are incorporated (mostly inspired by the BioTagger (McDonald, Winters, et al., 2004)). These are the membership of a token to a list of different types of variations (*deletion, duplication, insertion, inversion, transition, ...*), and the use of different regular expressions matching to frequently used terms for locations (e.g. nucleotide $[0-9]^+$, amino acid $[0-9]^+$, chr|chromosome $[1-9]|1[0-9]|2[0-2]|X|Y, \dots$). In the case of the entity *state*, the case-insensitive membership to the long and short forms of amino acids is important (*Alanine, Ala, Asparagine, Asn, ...*). This is especially useful for finding natural language formulations like “... induces Ala to Pro substitution at amino acid 459...”.

6.2.4 Normalization and Postprocessing

The normalization workflow is explained in more detail by Thomas, Klinger, et al. (2011) while the description here is limited to discuss the most important tasks and challenges in the following, as the case of mapping a variation mention harbors some special challenges.

The process of normalization allows the assignment of a *variation* entity found in the text to a dbSNP identifier, and in consequence the biological information about the variation can be obtained (such as organism, associated gene, etc). Once a dbSNP identifier is assigned to a *variation* entity set, it is unambiguously associated to a genomic location. In most of the cases the variation is mapped within the sequence of a gene (in its coding or non-coding regions) or in its proximity, and thus is associated to a single gene. In other cases, the variation is mapped to intergenic regions of the genome in the proximity of more than one gene. In such cases, database curators annotate the variation as associated to more than one gene in the genome.

Therefore, the information needed are associated genes, the variation location, and state change. This implies the need to detect the relation between these entities. Starting with the recognized gene, all stored variations in dbSNP are candidates to be mapped to the variation mention in question—this limits the possible associations to a handable number instead of starting with state-location combinations. Difficulties to address in this process include the

following:⁸

Which entities form a variation mention? Mentions following a nomenclature or convention can be easily associated (for instance in c. -225C>A or 32C→T) while this is non-trivial for natural language formulations (cf. Figure 6.1). The association with a gene may be solved by straight-forward proximity measuring while a higher precision can be expected by more linguistically motivated methods (Lee, Horn, and Cohen, 2007).

Is the mutation correctly stated? Due to the processing of a publication as well as through typos of the authors, mutations may not be correctly stated. Because of short entities and number-dependent normalization, this can happen more easily than in an entity class where author-given names are more frequent.

Which convention of naming is the author following? The nomenclature for mutations changed several times in the past (Beaudet and Tsui, 1993; Ad Hoc Committee on Mutation Nomenclature, 1996; Beutler, McKusick, et al., 1996; Antonarakis, 1998; Dunnen and Antonarakis, 2000; Dunnen and Antonarakis, 2001; Ogino, Gulley, et al., 2007). The normalization workflow should be able to handle all these rules (what means that they need to be found by the named entity recognition and can be associated correctly). Additionally, counting conventions need to be taken into account.

Is the variation described on the protein sequence or on nucleotide sequence? Genetic sequence changes occurring on the DNA may be propagated to mRNA and therefore change the protein sequence. The same mutation can be described on these different sequences, leading to differing counting of the location.

Which version of the genome is used as reference? As the sequenced human genome is updated regularly due to sequencing errors, it is important to know which one is used to identify the location of the variation. Authors commonly do not provide this information—a hint may be the publication date, but nevertheless this issue remains hard to solve.

Is the mutation stored in the data base? It seems to be obvious to not normalize a mention if it seems not to be in the data base. Nevertheless, due to problems to solve the prior mentioned issues, it makes, on the one hand, sense to allow differences and apply a kind of fuzzy search for an entry: That may, on the other hand, lead to normalization errors.

Examples for these cases and additional explanations to pitfalls in normalization can be found in the work by Thomas, Klinger, et al. (2011).

6.2.5 Dictionary-based and Rule-based Named Entity Recognition and Normalization

To be able to normalize variation mentions, a normalization solution for gene and protein names is needed. The workflow presented in Chapter 5 is optimized for pure named entity recognition—a normalization is developed additionally.⁹ The search for variations in the database is starting with the gene/protein identifier, therefore a high recall is needed. The established ProMiner system (Hanisch, Fluck, et al., 2003) with a gene and protein dictionary

⁸ Following the descriptions by Thomas, Klinger, et al. (2011)

⁹ This normalization is not part of this thesis.

extracted from the ENTREZ GENE database (Maglott, Ostell, et al., 2005) and the UniProt database (The UniProt Consortium, 2006) is used. This system consists of three different steps: The first step covers the generation and curation of a gene/protein name dictionary, which associates each biological entity with all known synonyms including automated cleaning of general terms like family names. The second, core, part of the system is an approximate search procedure focusing on a high recall. In a last step filters are applied to increase precision of the search results including disambiguation—this last step is not performed in the workflow of variation normalization to gain a high recall.

To detect direct mentions of database identifiers from dbSNP in texts, regular expressions are used as these are morphologically different from the descriptions searched for with the CRF approach. They are formed by the letters “rs” or “RS” followed by a natural number, as for instance rs1234567. Therefore, the regular expression `[rR][sS][]*[0-9][0-9]*` matches these mentions. As this leads to positive matches for *e. g.* cell-lines (“RS1”), computer names (“rs6000”), interfaces (“RS485”), indian rupees (“Rs1000”), and chemical compounds (“RS61433”), these names are only accepted if the keywords “mutation” or “SNP” occur in the same abstract. Frequent false positives are additionally never accepted.

6.3 Results

In the following, the evaluation of the developed workflow is presented. As it consists of several steps, the assessment is split into parameter selection for the CRF, the evaluation of the NER of variation mentions, and of the whole system. Additionally, the NER and regular expression-based RS number detection are analyzed. It is important to point out that the evaluation of the normalization is difficult because the effort to contrast each of the dbSNP identifiers assigned to a variation identified in the text with the entries in the database is high.

Entity	precision (%)		recall (%)		F_1 -Measure (%)	
Location	69.9	(0.0432)	67.2	(0.0417)	67.9	(0.0347)
Type	73.6	(0.0355)	51.2	(0.0395)	60.3	(0.0297)
State	78.0	(0.0275)	80.1	(0.0289)	79.2	(0.0205)
States separately						
State-altered	71.2	(0.0449)	72.6	(0.0440)	71.7	(0.0299)
State-Generic	10.0	(0.0436)	6.0	(0.0530)	6.9	(0.0434)
State-Original	71.8	(0.0637)	73.9	(0.0357)	72.6	(0.0368)

Table 6.2: Performance of named entity recognition with conditional random fields in a 50-fold bootstrapping evaluation on seed set (in parentheses the standard deviation is given).

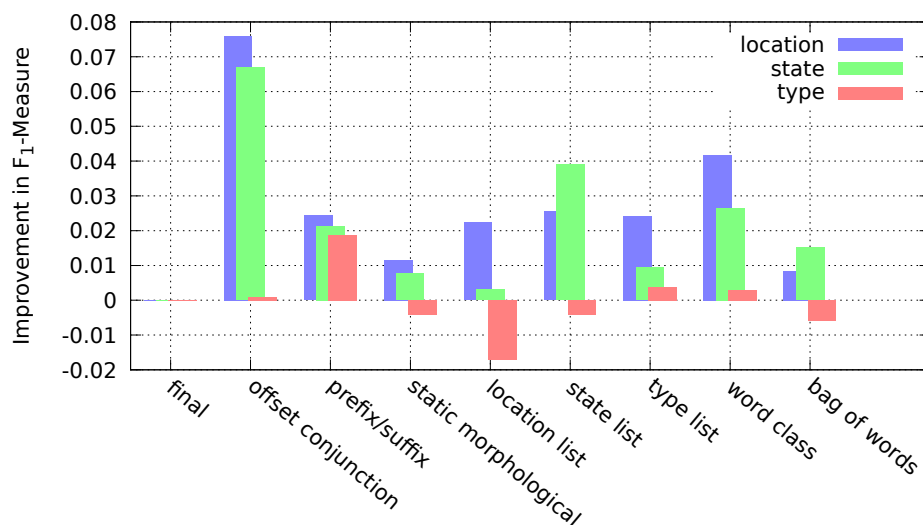


Figure 6.3: Results for different feature sets for the SNP seed set, evaluated via 50-fold bootstrapping.

6.3.1 Parameter Selection

The parameter selection is performed on the seed set presented in Section 6.2.2. Data inspection and discussions with the annotators revealed that it is hard to differentiate between the different types of states. Table 6.2, utilizing 50-fold bootstrapping (*cf.* Section 2.8) proves that: Separating the states leads to a considerably lower F_1 measure than in a combined setting, presumably because they cannot be differentiated by the features incorporated. In other words: They are used in a very similar context. Therefore, in contrast to the annotation guidelines of the University of Pennsylvania, the different *state*- entities (*-original*, *-altered* and *-generic*) are combined to a single entity class *state*, as this is sufficient for normalization purposes.

Figure 6.3 depicts the importance of the different features. In contrast to the according histograms for IUPAC names and gene/protein names (in Figure 4.4 and 5.3) three entity classes are shown, instead of one main class of interest. This highlights the problem that different features do not contribute positively to all classes.¹⁰ The morphological features do not show a high impact. Very important are the domain specific lists of entities for the respective classes. Notably, the location list has a negative impact on the detection of entities from the class *TYPE*, exemplifying the difficult to interpret and understand relationships between features and labels. This is accepted for the final model as this entity is not essential for variation normalization. Most important is the offset conjunction, as the different classes are depending on each other. Second most important is the word class: This captures the structure of nomenclatures. Feature classes not shown in this graph have no or nearly no

¹⁰ This could lead to the idea that a model for each class separately should be trained. An experiment with this setting showed very limited performance and is therefore not investigated further in this thesis.

impact.

6.3.2 Evaluation of Named Entity Recognition

The final trained model using the parameters as described in the previous Section 6.3.1 is evaluated on two independent test sets. RANDOM-TEST is the set sampled from whole MEDLINE, sparsely populated with entities, GENETIC-TEST has more entities as it is sampled from abstracts dealing with genetics only.

The results on both test sets is shown in Table 6.3. The values are much higher than seen in the parameter evaluation due to active learning and sufficiently high to serve as a basis for normalization. On the GENETIC-TEST corpus, the recall of the LOCATION entity class is the lowest and could therefore limit the association to full variation mentions—the value on RANDOM-TEST is higher due to a smaller variety of complicated mutation mentions. The class state could nearly be called to be perfect with an F_1 measure of 95.65 % on the relevant GENETIC-TEST corpus and good on RANDOM-TEST with 80 %—the same good value holds for LOCATION mentions on the latter.

As described in Section 6.2.5, rs number mentions can be found with regular expressions. Sub-sampling 300 abstracts from whole MEDLINE matching the expression for rs numbers naturally lead to a recall of 100 %. An assessment of the precision showed that the same structure of the dbSNP identifiers occurs in other mentions, leading to a precision of only 74 %. Including the additional lists of terms which need to occur to be accepted as mutation mentions finally leads to a precision of 97 % and a recall of 98 %.

6.3.3 Evaluation on Independent Test Set

The whole workflow of recognizing the entities (GENE/PROTEIN, STATES, LOCATION), combining them as a complete variation mention, and finally normalizing them to dbSNP needs to be evaluated as a whole. In this evaluation, a model trained on the seed set is used because further annotation was performed in a follow-up work. Nevertheless, the results give an insight to the whole workflow and the increase in performance of the model with the active learning based training set does not change the results dramatically, as will be presented

Set	Class	Precision	Recall	F_1 measure
GENETIC-TEST	TYPE	92.55	81.31	86.57
	LOCATION	81.25	67.24	73.58
	STATE	96.49	94.83	95.65
RANDOM-TEST	TYPE	71.26	78.48	74.70
	LOCATION	75.56	85.00	80.00
	STATE	76.54	83.78	80.00

Table 6.3: Results of the final SNP model on independent test sets (in %).

later.

The evaluation of the whole process of entity recognition and normalization is performed on a set consisting of a corpus of 100 abstracts selected randomly from a database of citations about genes related to the disease intracranial aneurysms.¹¹ The database of citations contains 2476 abstracts automatically annotated using OSIRIS v1.2 all containing normalized variation mentions. The abstracts refer to the allelic variants of genes related to the disease intracranial aneurysms, and each of the abstracts contains at least one occurrence of a normalized variation. Note, that this corpus is biased towards the recognition of OSIRIS.

The evaluation is inspired by the approach for the gene normalization task of the BioCre-AtIvE assessment (Hirschman, Krallinger, and Valencia, 2007). Thus, only the disjunct occurrences of variation as well as false positives are counted. An alternative way of assessment would have been to count all found annotations, which would be biased by articles redundantly (from the point of view of normalization) mentioning the same variation more than once, as in the following example: "... showed a heterozygous single base-pair transition from G to A (codon 53), resulting in a glycine for glutamic acid substitution (G53E)." (Ellie, Camou, et al., 2001). In this way, it is not necessary for the named entity recognition to find all mentions of a variation, it is sufficient to have the information for a normalization once. Actually, it often finds all mentions of a variation. This approach measures the use of the developed workflow for information retrieval. To extract relations between variations and other entities (like *e. g.* diseases), the latter approach would have been beneficial for estimation of the performance.

The sample of 100 abstracts used for the evaluation is manually reviewed by counting the following quantities:

- (a) Number of disjunct mentions of variations
- (b) Number of disjunct normalized variations obtained with OSIRIS v1.2
- (c) Number of disjunct mentions tagged by the CRF trained on seed set
- (d) Number of disjunct normalized variations by the presented system, based on (c)
- (e) Number of disjunct mentions tagged with the CRF trained on training set

A variation mention is counted for each given triplet of LOCATION, GENE/PROTEIN, and STATE which is sufficient for normalization. The measure (e) is given to compare the influence of the improved named entity recognition model on the normalization, although it is not evaluated in the workflow here together with a succeeding normalization.

The results are displayed in Table 6.4. Although the corpus was defined by mentions of the OSIRIS v1.2 system, the newly developed method can normalize more variations in the abstracts (142 to 136). Another advantage is that much more variations are found, even if

¹¹ Generated in the European Commission-funded project @neurist, <http://www.aneurist.org/>, made available via SCAIView, <http://www.scaiview.com/>.

	absolute number	%
(a) # mentions	264	100.00
(b) # normalized variations OSIRIS	136	51.52
(c) # mentions tagged CRF	216	81.82
(d) # normalized variations	142	53.79
(e) # mentions tagged improved CRF	232	87.88

Table 6.4: Results on independent test set (all counts are disjunct per article).

they are not normalizable (216 to 136). From all of the 142 variations, 127 are found with the CRF and 15 are rs numbers directly mentioned in the text.

Testing the model which is trained on the augmented seed set based on active learning showed notably better performance in named entity recognition. Nevertheless, this better model is not able to find a lot more mentions per abstract. Additional 18 mentions can be detected, 2 mentions are missed which are found with the earlier model. Although it is not tested, the positive influence on the result of the normalization can be assumed to be limited for information retrieval purposes.

An analysis of the false positives show typical errors in the different classes

State single capital letters ‘A’, ‘T’, ‘C’, ‘G’ in wrong context,

Type dates and other numbers, spans like ‘period 1945–1986’,

Location Words with all letters in capital at the beginning of a title.

6.4 Summary and Discussion

In this chapter a named entity recognition system suitable to serve as a basis for normalization for variation mentions in biomedical text is presented. The system is aimed at identifying different types of variations, from SNPs to small insertions and deletions, both at the nucleotide and the protein levels. The association of a variation with a gene and its products (mRNA, proteins) is automatically obtained once a variation is normalized, by the association at the sequence level of each dbSNP entry with a gene feature in the genome. The system Mutation GraB (Lee, Horn, and Cohen, 2007) also uses a sequence-based approach, comparing the wild-type amino acid of a point mutation with the sequence of the possible associated proteins. However, this strategy is limited to point mutations that are located on protein sequences. As mentioned before, this approach is not limited to protein point mutations, but covers a wide range of changes within the coding region of a gene as well as in introns and adjacent regions of the gene (promoter regions, 5’ and 3’ UTR). In addition, through its linkage to dbSNP, biological contextual information can be obtained as well, for instance if the variation alters protein function, or the frequency of the variation in certain population.

Although the system is aimed at the identification of allelic variants of human genes, the approach is easily applicable to other organisms with variation data available (*e. g.* mouse). Moreover, the CRF-based tagger could be applied to site-directed mutagenesis data as well, if the proper training data is provided (in this case, the entity `TYPE` specific for allelic variations should not be used).

The system is able to identify two types of variants, those that can be mapped to a dbSNP identifier and those that are not present in the database, and therefore are not linkable to dbSNP entries. Both resulting sets are of value for database curators.

Similarly to other methods previously published (Horn, Lau, and Cohen, 2004; Rebholz-Schuhmann, Marcel, et al., 2004; Bonis, Furlong, and Sanz, 2006; Lee, Horn, and Cohen, 2007) the presented method identifies simple representations of the variation entities such as A12T, A-T 12, A(12)T, but contrasting to previous reports, it also identifies and normalizes more complex representations like “A/G single nucleotide polymorphism (SNP) at position 49”.

Part III

Further Enhancements

Chapter 7

Feature Subset Selection¹

7.1 Introduction

In Part II, several applications of named entity recognition have been presented. All of them show huge numbers of features. The time complexity for training and inference in a CRF per iteration as explained in Section 3.4.2 is $\mathcal{O}(|\mathcal{L}|^2mn)$, where $|\mathcal{L}|$ is the number of possible labels, m the average number of features per factor, and n the number of factors emerging from data. While the features are typically very sparse, the question arises if the huge number of overall features (e. g. 492,611 for the short annotation-based model in Chapter 5) limits the performance.

Feature Selection is well established for many machine learning methods, for instance for feed-forward neural networks (Bishop, 1995) or decision trees (Breiman, Friedman, et al., 1984; Quinlan, 1986). The main advantages are an improvement of prediction performance, faster training and prediction as well as a better understanding of the models (Guyon and Elisseeff, 2003). Methods can be distinguished between filters not using the learning algorithm and wrappers using the learning algorithm as a black box (Kohavi and John, 1997). An overview of approaches for classification tasks is given by Liu and Motoda (2008), more specifically for text classification by Yang and Pederson (1997).

Such feature selection methods are not well established for conditional random fields (cf. Section 3.4, Lafferty, McCallum, and Pereira (2001)). In this chapter, methods coping with the task of handling sequential data represented by a huge number of features are introduced. Reported numbers are even higher than the ones in the applications in Part II, for instance 1,686,456 for a gene name tagger (Hsu, Chang, et al., 2008). Due to these high numbers, training and inference times can explode. These high numbers of features are generated by automated methods, *i. e.*, for every token in the training set, features are generated. Then, all other tokens are tested for these features. This method is typically applied to determine the identity of words as well as for prefixes or suffixes of different length or for learning schemata of regular expression-like patterns (cf. Section 2.6). Figure 7.1 shows for two data sets (namely the BioCreative II data set used in Chapter 5 and the CoNLL 2003 data set, the latter is introduced in Section 7.3.1) which classes of features lead to high numbers. Obviously, the automated procedures to generate features in a pattern-based manner lead to the huge numbers of features to maintain. Figure 7.2 shows the empirical distribution of the weights. Most of the weights are around zero (note that the depiction has a logarithmic scale): This motivates the assumption that a large fraction of features has little or no meaning.

¹ This chapter is based on Klinger and Friedrich (2009a).

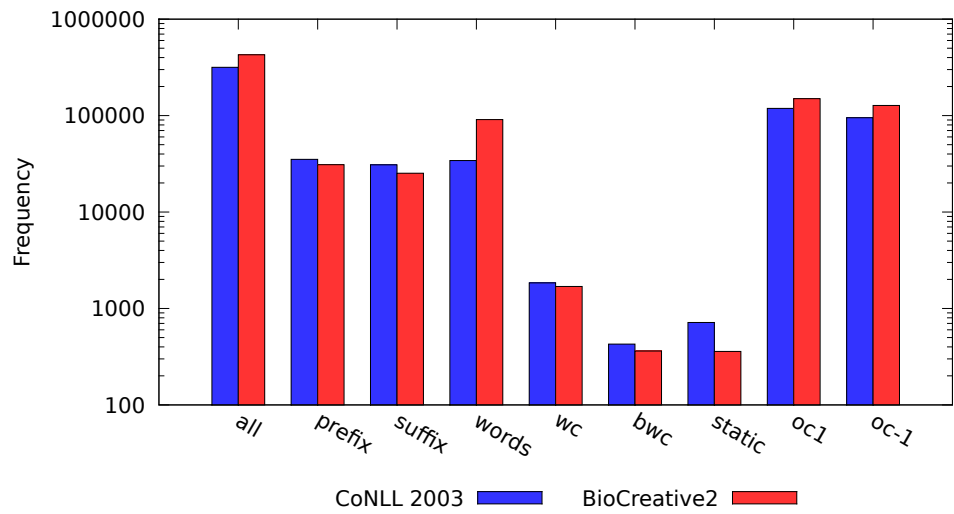


Figure 7.1: Number of disjoint features per class in CRF models.

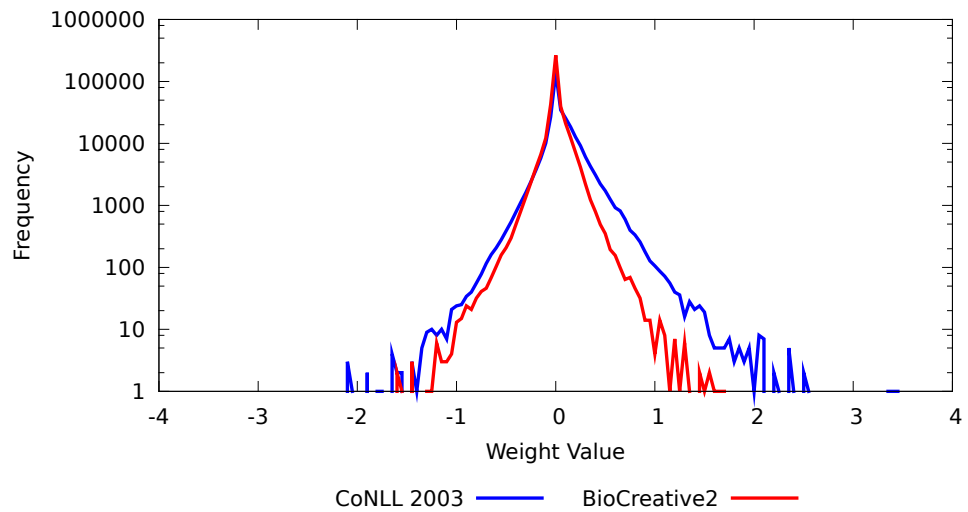


Figure 7.2: Distribution of weights in CRF models.

Only a few approaches dealing with feature handling for CRFs are published. The work by McCallum (2003) demonstrates a method for iteratively constructing feature conjunctions that would increase conditional log-likelihood if added to the model. An analysis of different penalty terms for regularization is shown by Peng and McCallum (2004). Goodman (2004) presents a related analysis of exponential priors for maximum entropy models. The recent work of Vail, Lafferty, and Veloso (2007) (see additionally Vail, 2008) shows feature selection in conditional random fields by L_1 -norm regularization in the robotics domain, a work which is related to the selection in maximum entropy models proposed by Koh, Kim, and Boyd (2007).

These methods incorporate the training procedure in the selection process. In contrast, different filter methods for feature selection are presented to limit the complexity before starting the training. It is demonstrated how the sequential structure of text can be respected by filtering approaches originally developed for classification problems (especially in Section 7.2.1). These filter methods are compared to an iterative approach, based on the idea that features associated with a low weight in the model have a lower impact. This idea is related to approaches known as *optimal brain damage* and *Optimal Brain surgeon* for artificial neural networks (Cun, Denker, and Solla, 1990).

In the following, different approaches for feature selection in CRF are introduced in Section 7.2, namely the adaption of methods inspired from classification to filter features and an iterative approach to remove features with low weights, determined empirically during training. In Section 7.3, an evaluation of the feature selection methods is given. Using such fast methods is the main innovation of this chapter. The results show a reduction of complexity leading to improved speed and better explainability, allowing more general workflows to adapt a CRF to a new class of interest (to be discussed in Part IV).

7.2 Feature Selection Methods for CRFs

An exhaustive search to find the optimal feature subset is not possible due to the large number of features and comparatively long training times. Hence, a wrapper approach considering the CRF as a black box is impractical. Therefore, in Part II groups of features, for instance consisting of patterns to generate features, were analyzed. In the following, we are dealing with each single feature instead. Different approaches for feature subset selection of sequential data in CRFs are introduced, *i. e.*, filtering methods (in Section 7.2.1) and an iterative method (in Section 7.2.2).

7.2.1 Filter

To apply filter methods for classification tasks, the sequence data have to be represented as classification instances. For a pair of sequences (\vec{y}, \vec{x}) this is done with respect to the incorporated factors in the CRF. For every factor $\Psi_j(\vec{y}, \vec{x})$, an instance is built. The labels are all dependencies on \vec{y} , the features have the values at the corresponding position for \vec{x} . For the factors in a linear-chain CRF as shown in Equation 3.42, the instance $\mathcal{I}_j = (\mathcal{L}_j, \vec{\varphi}_j)$

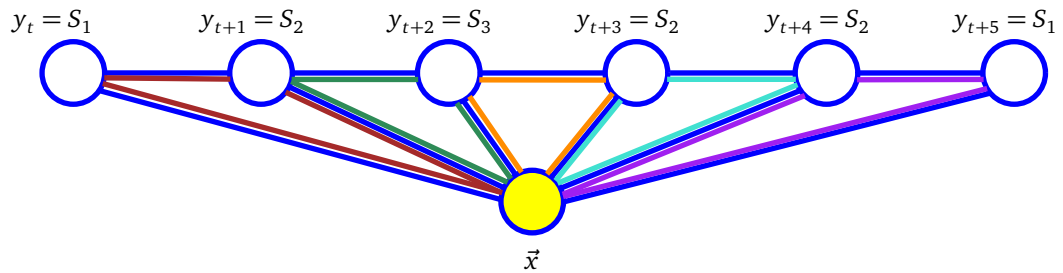


Figure 7.3: Depiction of instance building for feature filtering. The linear structure is splitted into instances for classification representing the transition between two labels. Each instance extracted is marked with a different color.

at position j ($0 < j \leq n$) has the label $\mathfrak{L}_j = (y_{j-1}, y_j)$ from a set of all possible transitions² $\mathfrak{L}_j \in \mathcal{L}^2$. The feature values are $\varphi_k(x_j)$. Instances are built for all positions in all training examples from \mathcal{T} . This process is exemplified in Figure 7.3. The instances built in this example have the labels $S1 \rightarrow S2$, $S2 \rightarrow S3$, $S3 \rightarrow S2$, $S2 \rightarrow S2$, and $S2 \rightarrow S1$, respectively.

The features are ranked for each factor generating template respectively by measures presented below. In the case of a linear chain CRF presented in this chapter, there is only one template, in the case of multiple templates ranked lists are generated for each template as these are the basis for the parameter binding between factors. The best p_{filter} features (where p_{filter} is a parameter specifying the percentage of kept features) are selected to represent the text data.

In the following, the number of generated instances is denoted with h , the number of instances with feature $\varphi(x_j)$ with value 1 with h_j^1 and with value 0 with h_j^0 . The number of instances with label \mathfrak{L}_ℓ is $h(\ell)$, with feature values 1 or 0 of those with $h_j^1(\ell)$ and $h_j^0(\ell)$ respectively.

Simple Information Gain

The first approach for measuring the quality of a feature is the use of information gain of a feature $IG(\varphi(x_j))$ to differentiate between all possible labels \mathfrak{L}_ℓ . It is defined as

$$IG(\varphi(x_j)) = I\left(\frac{h_j^1}{h}, \frac{h_j^0}{h}\right) - R(\varphi(x_j)) \quad (7.1)$$

where $I(\cdot)$ is the information content

$$I(p_1, p_2) = -p_1 \log_2 p_1 - p_2 \log_2 p_2 \quad (7.2)$$

² For the linear-chain CRF of order 1. In general, $\mathfrak{L}_j \in \mathcal{L}^{c+1}$ with order c holds.

with probabilities p_1 and p_2 . $R(\varphi(x_j))$ is the remainder of bits of information after testing feature $\varphi(x_j)$:

$$R(\varphi(x_j)) = \sum_{\ell=1}^{|\mathcal{L}^2|} \left(\frac{h(\ell)}{h} I \left(\frac{h_j^1(\ell)}{h(\ell)}, \frac{h_j^0(\ell)}{h(\ell)} \right) \right). \quad (7.3)$$

For comparing the features it is sufficient to compute $R(\cdot)$ because $I(\cdot)$ is constant for one feature (Russell and Norvig, 2003). To rate all m features with $|\mathcal{L}|$ labels, $\mathcal{O}(m|\mathcal{L}^2|)$ calculations of the information content are needed.

The impact of all features on all transitions is examined with this approach. Therefore, the result is a feature set for all transitions; no differences for specific labels are made. Therefore, this approach is referred to as *Simple IG*.

Information Gain One-Against-All

The limitation of *Simple IG* is the disregard of differences between transitions in CRF. To cope with that, a list of the best p_{filter} features are assigned to every transition \mathcal{L}_ℓ . In general, every clique in the graph has its own evaluation of features $\varphi(\cdot)$.

The remainder, the measure for the quality of a feature in Equation 7.3, changes slightly to

$$R_{OAA}(\varphi(x_j), \mathcal{L}_\ell) = \left(\frac{h(\ell)}{h} I \left(\frac{h_j^1(\ell)}{h(\ell)}, \frac{h_j^0(\ell)}{h(\ell)} \right) \right) + \left(\frac{\bar{h}(\ell)}{h} I \left(\frac{\bar{h}_j^1(\ell)}{\bar{h}(\ell)}, \frac{\bar{h}_j^0(\ell)}{\bar{h}(\ell)} \right) \right) \quad (7.4)$$

where

$$\bar{h}(\ell) = \sum_{l \in \{1, \dots, |\mathcal{L}^2|\} \setminus \ell} h(l),$$

and $\bar{h}_j^1(l)$ and $\bar{h}_j^0(l)$ analogous. This approach is applied to rank the features $\varphi(x_j)$ for every transition \mathcal{L}_ℓ separately. It is referred to as *Information Gain One-Against-All (IG-OAA)*. To rank all features, the information content needs to be determined $\mathcal{O}(m|\mathcal{L}|)$ times.

χ^2 -Statistics

Another well-known and often incorporated ranking method are χ^2 -statistics (Pearson, 1900; Plackett, 1983). The 2×2 contingency table is defined for each feature $\varphi(x_j)$ and each

	$\varphi(x_j) = 1$	$\varphi(x_j) = 0$	Σ
\mathcal{L}_ℓ	$h_j^1(\ell)$	$h_j^0(\ell)$	$h(\ell)$
$\mathcal{L}_{\neq \ell}$	$\bar{h}_j^1(\ell)$	$\bar{h}_j^0(\ell)$	$\bar{h}(\ell)$
Σ	h_j^1	h_j^0	h

Table 7.1: 2×2 contingency table for feature selection in CRF.

transition \mathfrak{L}_ℓ compared to all other transitions (cf. Table 7.1). The χ^2 -statistic is then computed by

$$\chi^2(\varphi(x_j), \mathfrak{L}_j) = \frac{\left(h_j^1(\ell) \cdot \bar{h}_j^0(\ell) - h_j^0(\ell) \cdot \bar{h}_j^1(\ell)\right)^2 \cdot h}{h(\ell) \cdot \bar{h}(\ell) \cdot h_j^0 \cdot h_j^1} \quad (7.5)$$

Similar to *IGOAA*, this is performed to rank the features $\varphi(x_j)$ for every transition \mathfrak{L}_ℓ separately. Therefore, also $\mathcal{O}(m|\mathcal{L}|)$ ranking measure calculations are needed. This method is referred to as χ^2 *OAA*.

Random

The most simple method is a *random* ranking and selection of features. This is used as a baseline to evaluate the other measures presented in the previous sections.

7.2.2 Iterative Feature Pruning

Training a CRF is commonly performed by the iterative algorithm L-BFGS to assign weights λ_i to all feature functions $f_i \in \mathcal{F}$ such that $\mathcal{L}(T)$ is maximized (compare to Equation 3.50). Typically, many weights are close to 0, as depicted in Section 7.1 in Figure 7.2. The idea of Iterative Feature Pruning (IFP) is that feature functions with low absolute weight value have a low impact on the output sequence.

Based on this assumption, the algorithm (see pseudo code in Figure 7.4) starts with a fully optimized CRF using all features representing the training data (Line 2). The next step is the removal of features with lowest absolute value (3). Let S be the set of remaining features after one iteration of IFP. Then the parameter $p = 1 - \frac{|S|}{|\mathcal{F}|}$ specifies the percentage of features to be removed in each iteration (13–15).

In each step, a retraining with L-BFGS is performed to allow an adaptation of the model by adjusting the weights for the remaining features (16). After that, the pruning is repeated (17) until no features are left (11) or another criterion is reached (see Section 7.3.3).

During this process, at each iteration of pruning, the current performance of the model is evaluated and stored (8). This information can be used to select the final feature set (Line 4, an heuristic is shown in Section 7.3.3) and to train a full model with the identified feature subset (Line 5).

To illustrate the process of IFP, it is shown exemplarily by means of extreme examples for one data set³ in Figure 7.5. On the horizontal axis, all L-BFGS training iterations are shown consecutively with the intermediate pruning steps. The blue dotted line shows the decrease of the number of features, the red solid line the F_1 measure for the training data, the green dashed line the F_1 measure for one validation data set. Removing 40% of the features in each step ($p = 0.4$) clearly depicts the process of removing and retraining of the model. Experiments have shown that in general lower numbers of features can be achieved with

³ CoNLL data set introduced in Section 7.3.1

```

1: function IFP(crf,trainData, valData, p)
2:   crf = BFGS(crf, trainData, valData)
3:   log = PRUNING-STEP(crf, trainData, valData, p)
4:   crf = SELECTFEATURESET(log)
5:   crf = BFGS(crf, trainData+valData,null)
6: end function
7: function PRUNING-STEP(crf, trainData, valData, p)
8:   log ← EVALUATE(trainData, valData, crf)
9:    $\mathcal{F}$  = GETFEATURESET(crf)
10:  if  $\mathcal{F} = \emptyset$  then
11:    return log
12:  end if
13:   $S$  = features with lowest weights such that
       $p = 1 - |S|/|\mathcal{F}|$ 
14:   $\mathcal{F} = \mathcal{F} \setminus S$ 
15:  SETFEATURESET(crf, $\mathcal{F}$ )
16:  crf = BFGS(crf, trainingData)
17:  return PRUNING-STEP(crf, trainData, valData)
18: end function

```

Figure 7.4: Iterative Feature Pruning Algorithm (starting with method IFP(·) in Line 1).

comparable F_1 measures if smaller values of p are used. The drawback is the higher number of iterations needed. In the following, $p = 0.1$ is used which leads to good results as shown in Section 7.3.

7.3 Results

In this section, the methods proposed in Section 7.2 are evaluated on the data sets and configurations of the CRFs described in Section 7.3.1. The hypotheses to be analyzed are:

Selecting a reasonable subset of features:

- A1** Improves explainability of the CRF model,
- A2** Improves training time and tagging time which is beneficial for developing as well as applying the model,
- A3** Improves performance in F_1 measure or does not decrease it dramatically.

Additionally, it is assumed that one method is superior to all others:

- B** One method outperforms the others.

The evaluations in the following form the basis for the discussion of these hypotheses in Section 7.3.4.

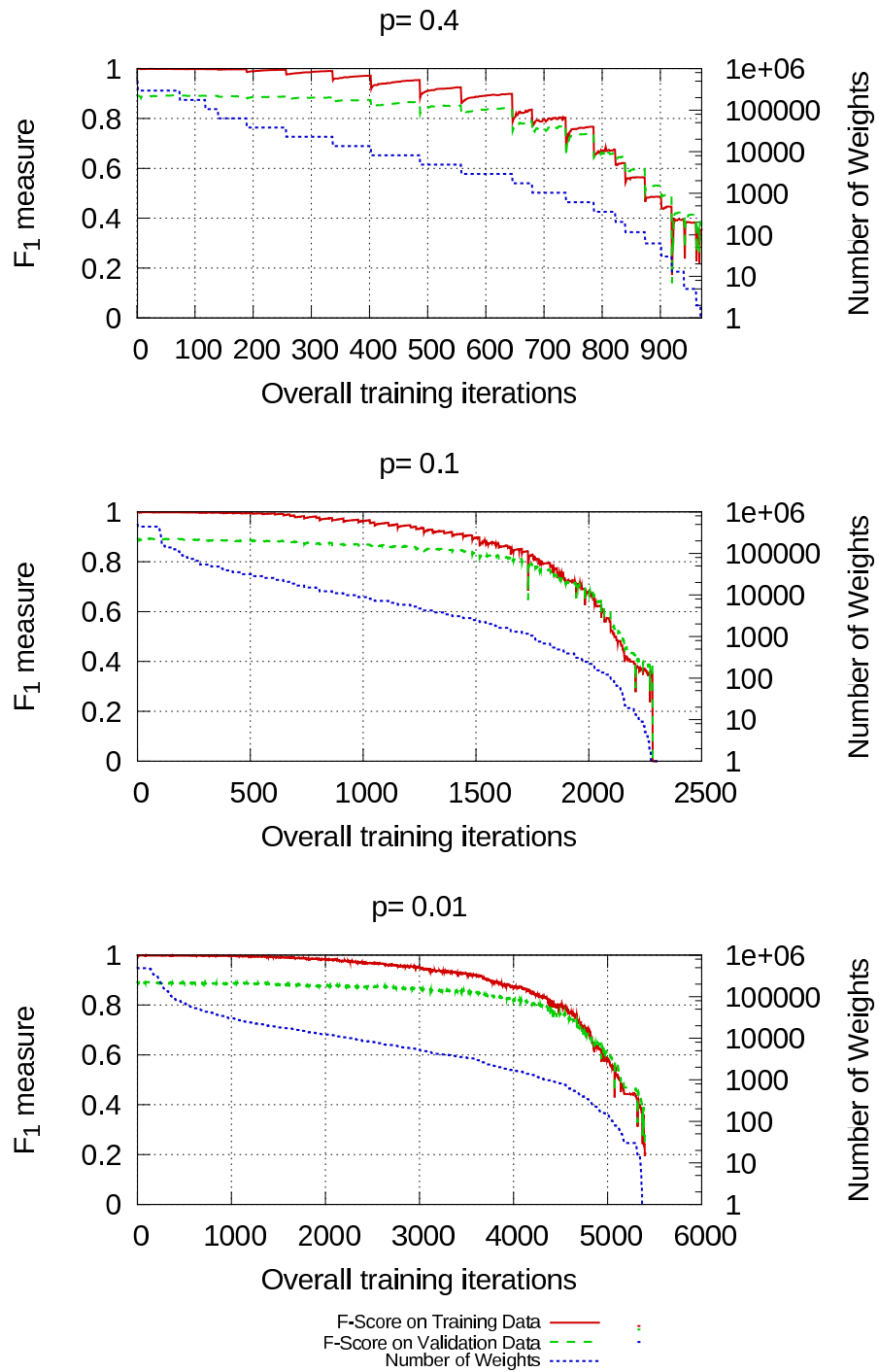


Figure 7.5: Visualization of Iterative Feature Pruning for CRF trained on CoNLL data with different percentages of features pruned in each iteration.

7.3.1 Experimental Setting

The results and evaluations are shown on the basis of three data sets with slightly different configurations of the CRF.

The BioCreative 2 Gene Mention Task data (BC2) contains entities of the class *Gene/Protein* with the specialty of acceptance of several boundaries for entities. The configuration of the CRF as described in Chapter 5 using the shortest possible annotation as exact true positive per entity is used.

The data set described in Chapter 4 contains entities which are IUPAC or IUPAC-like names. For the tests shown in this chapter, the configuration with a CRF order and offset conjunction order of 1 is used for the benefit of a simpler model to experiment with.⁴

The CoNLL data (Sang and De Meulder, 2003) is an annotation of the Reuters corpus (Lewis, Yang, et al., 2004) containing the classes *person*, *organization*, *locations* and *misc*. An order-one CRF with offset conjunction combining features of one preceding and succeeding token for each position in the text sequence is used. The feature set is fairly standard with Word-As-Class, prefix and suffix generation of length two, three and four as well as regular expressions detecting capital letters, numbers, dashes and dots separately and as parts of tokens. The combination of the provided sets “train” and “testa” is used for training and “testb” for testing. This data set is an example for a problem from a non-biomedical domain.

For evaluating the inference time on a larger set, a uniform sample from the Medline⁵ database of 10,000 entries is used additionally. Each one comprises titles, author names, and abstracts. The number of tokens is 958,869 for BC2 and 960,744 for IUPAC and CoNLL.

7.3.2 Cross-Validation on the Training Sets

As a basis for parameter selection (presented in Section 7.3.3) and to evaluate the impact of feature selection, 10-fold cross validation is performed on the training sets (Section 7.3.1). The results are shown in Figure 7.6. The curves depict the average F_1 measure (cf. Equation 2.6) of the 10 partitions. The different numbers of features are detected with different parameters specified for the respective selection method. The transparent band around the line depicts the standard deviation for the according number of features.⁶ The significance of the difference of the methods is tested regarding the area under the curves in Figure 7.6 via Welch’s t-test with a significance level of $\alpha = 0.05$.

Comparing the results on the two data sets, the methods lead to similar results whereas the differences are clearest on CoNLL data. All approaches outperform the random selection significantly. The approach of χ^2 OAA is worse than the conceptionally similar IG OAA and the more naive *Simple IG* on BC2 data.

⁴ Note that the performance of the IUPAC tagger in this chapter is not state-of-the-art, please compare the results with the ones presented in Chapter 4, especially the F_1 measure in Figure 4.6

⁵ http://www.nlm.nih.gov/databases/databases_medline.html

⁶ In iterative feature pruning, different numbers of features can occur at the same iteration of pruning. In that case, the closest detected number of features is used to compute average and standard deviation.

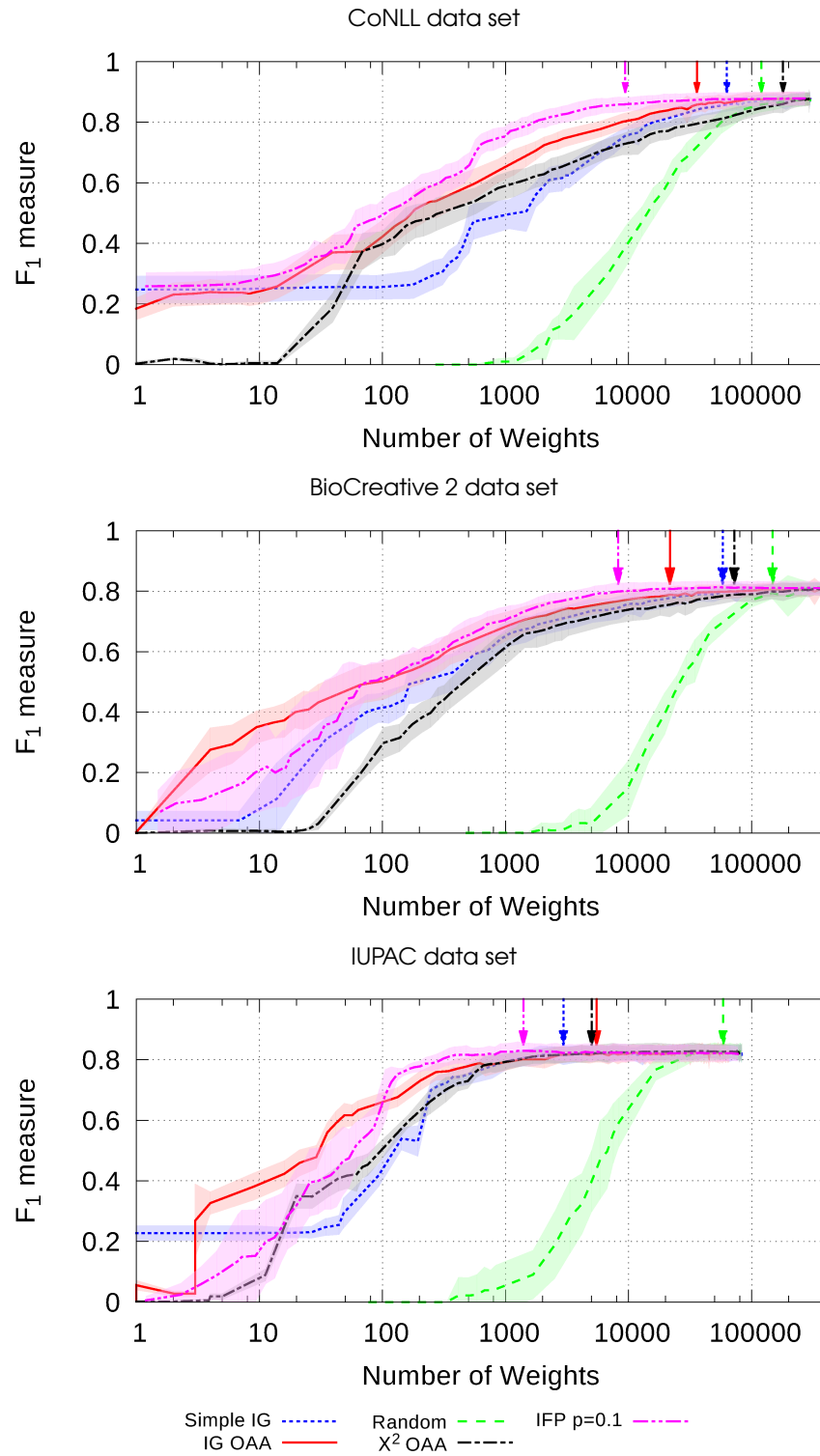


Figure 7.6: Comparison of the average F_1 measure using 10-fold cross-validation. Used features are determined with methods described in Section 7.2. The transparent band shows the standard deviation. The arrows show a possible selection of the model features ($g = 2 \cdot 10^{-6}$, $\Delta = 0.02$, cf. Section 7.3.3).

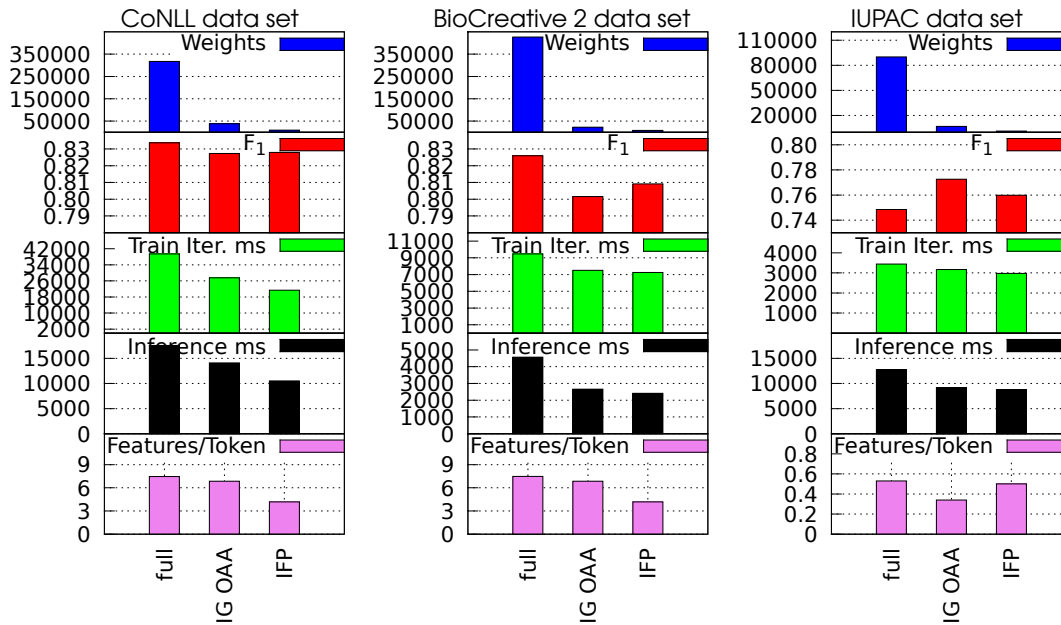


Figure 7.7: Results on independent test sets (note the different scales for some of the histograms).

IG OAA outperforms all other filtering approaches. Assuming the goal to reach the highest possible F_1 measure, *IFP* leads to better results than *IG OAA* on both data sets. Only if an extremely small number of features remains (fewer than about 50), *IG OAA* leads to better results. The superiority of *IFP* to χ^2 *OAA* is significant ($p = 0.02$) on the CoNLL data set.

7.3.3 Results on independent test sets

We need to find the parameter assignment to determine the feature subset in the final model. For the filter approaches, the parameter is p_{filter} . For *IFP*, the meaningful number of features at which the pruning is stopped has to be detected. Based on the smoothed⁷ values of F_1 measure in 10-fold cross-validation (see Figure 7.6), two measures to automatically detect these parameters are used: The maximally accepted loss in F_1 measure is denoted with Δ , the threshold for the gradient is g . The detection of the feature subset is performed via backward selection starting with high numbers of features. The first position on the curve for which the gradient is smaller than g or the F_1 measure is smaller than Δ is selected. The values $g = 2 \cdot 10^{-6}$ and $\Delta = 0.02$ lead to the positions denoted by arrows in Figure 7.6. The results for these values are evaluated in the following. The advantage of backward selection to forward selection is that it may capture interacting features more easily (Kohavi and John, 1997).

A model is built on the full training set applying *IFP* or filtering with the detected parameters. In Figure 7.7 the results on independent test sets mentioned in Section 7.3.1 are

⁷ Smoothing via computation of median in a running window.

depicted. The smallest numbers of features are achieved by *IFP* followed by *IG OAA* and therefore shown in comparison with the full model. The F_1 measures decrease up to the accepted $\Delta = 0.02$ on BC2 data and to a lower amount for the CoNLL data. For IUPAC, the F_1 measure even increases. The best trade-off between F_1 measure and the number of features is always achieved by *IFP*.

A smaller number of features should induce a faster model in training and inference. The time of evaluating $P_{\vec{\lambda}}(\vec{y}|\vec{x})$ (compare to Equation 3.43) on all training examples is shown in the third bar charts. This computation is crucial for training durations as it has to be performed many times. These numbers correspond roughly to the number of features, the durations are smaller than for the original model. The time complexity of training and inference in a CRF is dependent on the average number of features per token (cf. Section 7.1), shown in the fifth bar chart. Obviously, the speed up depends on the overall limitation in complexity due to a smaller overhead in feature handling and this value, which shows around 50 % decrease for CoNLL and BioCreative data. These numbers are different for the IUPAC data, the reason is that the training data was not sampled uniformly but selected via active learning. Hence, the complexity of the model is lower, for only every second token features are existing on average in the model built without feature selection. These numbers do not decrease as dramatically for IUPAC than for the other data sets.

The reduced feature numbers are beneficial for short training times. Training the full CoNLL model lasts 5788 seconds, 2397s for BC2 respectively. With the feature set detected by *IFP*, these numbers reduce to 2156s and 670s. This improvement is not helpful in practice, as the *IFP* procedure incorporates training the model. However, filtering via *IG OAA* improves overall training time as it is computationally inexpensive. It leads to 5230s and 1002s for training a model. For IUPAC, there is only a slight decrease.

Reducing the number of features also leads to a faster inference⁸: The fourth bar charts show durations for tagging 10000 sampled abstracts from Medline. Best results are achieved by *IFP* (CoNLL: 10.52s instead of 17.56s, BC2: 2.42s instead of 4.56s, IUPAC: 8.7s instead of 12.75), followed by the filtering methods (*IG OAA*: CoNLL: 14s, BC2: 2.65s, IUPAC: 9.15s).

Summarizing, a reduced training iteration time of 76 % of original time evaluating $P_{\vec{\lambda}}(\vec{y}|\vec{x})$ with a loss of only 1.7 % F_1 (absolute value) on the BC2 data is possible with *IFP*. The tagging time is reduced to 53 %. On the CoNLL data, a loss of only 0.57 % in F_1 occurs with savings of even 54 % of original computing time. Tagging time is reduced to 60 %. On the IUPAC data, an increase in F_1 of 2.4 % occurs with a speed up of 0.78 % for training and to 0.64 % of the tagging time.

Revisiting the distribution and counts of weights and features in Figures 7.1 and 7.2 with the proposed feature selection methods leads to the depictions in Figure 7.8 and 7.9. Noteworthy is the drop in the frequency of features with weights close to zero, especially for *IFP* where the result of the method can clearly be seen as the shape of the curve changed (in Figure 7.2). The numbers of features decreases as expected especially in classes of automatically generated features (see Figure 7.8). These figures are similar for the IUPAC

⁸ Measuring only the computation, not the time to read the data from hard disk and to extract the features.

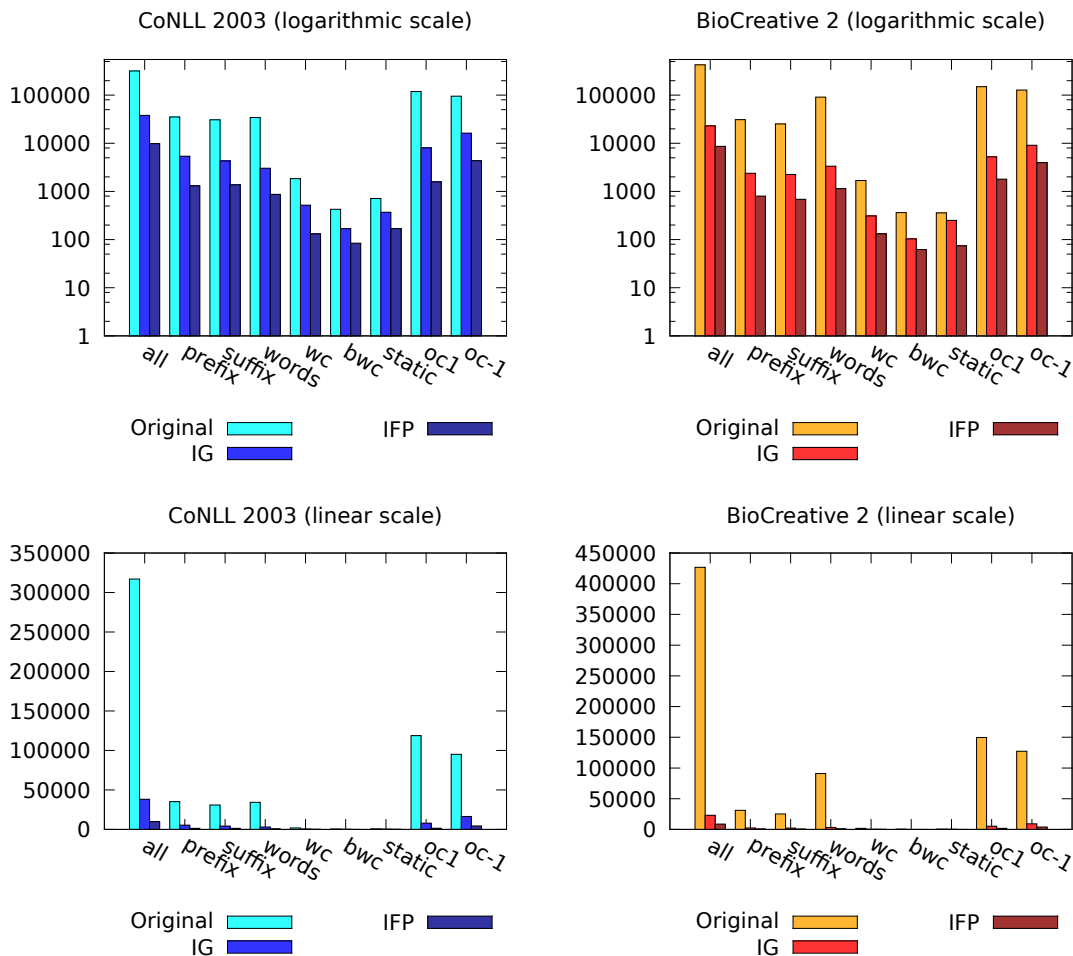


Figure 7.8: Feature distribution per class in CRF models with feature selection.

data set.

7.3.4 Discussion

Comparing the methods, the results are similar for the data sets. In 10-fold cross-validation, the random method is dominated by all other methods. *Simple IG* or χ^2 OAA are second worst, depending on the data set. *IFP* is the best method, closely followed by *IG OAA*; on IUPAC data, the latter leads to better results than *IFP*.

The *Simple IG* lacks the representation of different transitions in the features which is especially important for the CoNLL data with 4 entity classes of interest. No dictionaries with members of these classes have been used in the presented setting, so all classes are memorized with automatically generated features, hence, a large number of different features is needed for the different transitions in the CRF.

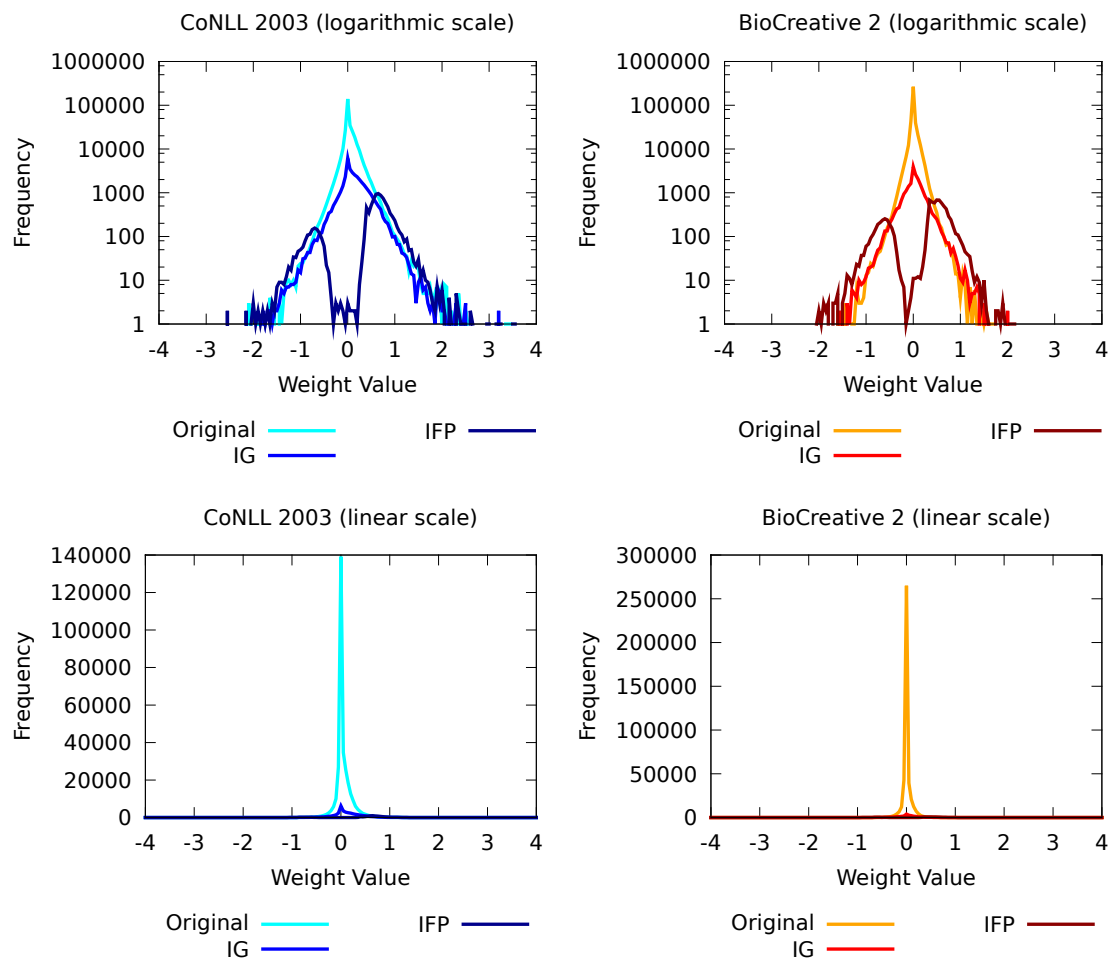


Figure 7.9: Distribution of weights in CRF models with feature selection.

The method χ^2 OAA always leads to worse results compared to IG OAA on the 10-fold cross-validation although it is a systematically similar approach. The reason is presumably the unbalancedness of the labels in the generated classification instances⁹ which is taken into account in the information remainder (Equation 7.3 and 7.4) but not in χ^2 (Equation 7.5).

IFP leads to better results than IG OAA for high F_1 measures. The reason is the limitation of IG OAA to use the same number of features (but not the same set) for each transition (specified by p_{filter}). This does not hold for IFP as it only relies on the model structure itself. The drawback is the higher computational cost due to the incorporated L-BFGS optimization.

The *random* method does not lead to good results, but it should be noted, that even this approach can remove 30%–40% without a dramatic decrease in F_1 measure. The reason are

⁹ Transitions of intermediate terms (like (O,O)) are for instance much more frequent than those of beginnings of entities (like (B,B)).

Data Set	Number of Features		
	Original	Remaining	%
CoNLL	269,506	17,377	6.45
BC 2	492,611	11,096	2.25
IUPAC	76,894	348	0.45

Table 7.2: Minimal numbers of original features needed to lose maximally 0.01 of F_1 measure applying IFP

unessential redundancies in the full feature set.

Mainly all these results are reflected on the independent test sets. *IFP* or *IG OAA* has the best trade-off between F_1 measure and the number of features. Good inference speed-ups can be achieved with both methods, corresponding to the low numbers of features. However, *IFP* cannot be used to speed up training as it incorporates the training procedure itself. Hence, *IG OAA* is proposed for this instance. For reducing tagging time as well as for understanding the model, *IFP* should be used as it leads to the smallest numbers of features.

In Table 7.2, the percentages and numbers of remaining features accepting maximally 0.01 loss in F_1 measure are depicted. Much more features can be ignored in the BC2 than in the CoNLL setting. This can be an indicator for the need for better generalizing features: as the classes have to be memorized by automatically generated features, less features can be eliminated. In BC2, features with better generalization characteristics are implemented. These values are notably low for IUPAC data, probably due to the use of active learning—nevertheless it needs to be kept in mind that the performance in F_1 measure in this experiment is not state of the art (cf. Chapter 4).

The results are concluded with an investigation of the hypotheses:

- A1** The explainability of models applying feature selection is improved by the lower complexity. Detected relevant features representing *e. g.* words or important pre- and suffixes help understand the different entity classes of interest. The fact that noisy features are removed allows for an investigation of the remaining features which can be assumed as meaningful.
 - A2** Training and tagging time are decreased by the lower complexity of the CRF. To improve training time, the use of a filtering approach like *IG OAA* is proposed as these methods are computationally inexpensive. To improve tagging time, *IFP* should be used as it leads to the lowest numbers of features.
 - A3** A small decrease in F_1 measure has to be accepted for the benefit of a model with a considerable lower number of features. An increase can occur depending on the data set.
- B** The recommendation for a method depends on the application: For improving training

time, a filtering method should be used, preferably *IG OAA* as it shows best results. For improving tagging time, the computationally more expensive *IFP* can be applied.

7.4 Conclusion and Future Work

A huge number of features is typically used to represent input text in CRFs. Different approaches for feature subset selection were presented, novel adaptations of filtering to the sequential structure of text as well as an iterative method. The methods have been evaluated on three domains, showing a decrease of computing time and complexity of the model. The F_1 measure varies slightly.

Summarizing, *IG OAA* is the best filter approach, a lower number of features can only be achieved with Iterative Feature Pruning (*IFP*) with a similar F_1 measure. *IFP* relies only on the CRF structure itself, so it is able to deal with different numbers of features per transition in contrast to the One-Against-All methods. Its main disadvantage is its higher computing cost due to the incorporated training process. It is notable that *IFP* and *IG OAA* are methods taking the sequential structure of the text into account, *IFP* via using the model itself, *IG OAA* via different feature sets for different transitions. The method *Simple IG*, which does not select features with respect to transitions, leads to worse results. The application of feature selection reduces runtime for inference as well as training time. This allows the implementation of a workflow with huge numbers of features from which the informative ones are automatically selected such that the adaption of a model to a new domain or entity class is simplified in everyday life.

The speed-up is not caused by the reduction of the overall number of features alone but in combination with the average number of holding features per token: It needs to be investigated if approaches can be developed to reduce especially this number.

Building a new named entity recognizer often includes annotation of a corpus. It has to be investigated, how the need for features changes during the process of enriching the training set with examples (*e. g.* via active learning). This is especially interesting as it has been shown here that the active learning based model has a lower complexity *ab initio*.

Another point is that the proposed methods allow for more complex feature generations (*e. g.* with more context information). It has to be studied if new features, which could not be implemented before due to an exhaustive memory consumption or run-time demand, could improve the state-of-the-art results.

Chapter 8

User's Choice of Precision and Recall¹

8.1 Introduction

In information retrieval, the F_β measure, the weighted harmonic mean between recall and precision, is established as evaluation measure. The corresponding β value to be chosen is application-dependent. For information retrieval, the user may prefer a high recall to decrease the chance to miss important documents. For information extraction, a higher precision is beneficial. Methods for selecting β at training time exist for support vector machines (Schölkopf and Smola, 2002), maximum entropy models as well as conditional random fields (CRF, Lafferty, McCallum, and Pereira (2001), cf. Chapter 3), all of which are classically optimized by means of accuracy-related measures (Jansche, 2005; Joachims, 2005; Suzuki, McDermott, and Isozaki, 2006). The approach for CRFs is introduced in Section 8.2.1. A similar goal is known from the AmilCare system (Ciravegna and Petrelli, 2001) with the main focus on user involvement. For a discussion of different evaluation measures and their characteristics on segmentation tasks, see Section 2.8.

At inference time, a parameter to select between higher precision or recall can be introduced by changing the decision threshold for an adequate decision function $d(\cdot) \in \mathbb{R}$. In sequential segmentation tasks like named entity recognition (NER), precision can be increased with this approach without retraining. Increasing recall is possible with the allowance of overlaps as demonstrated for gene and protein names (Carpenter, 2007). This requires the computation of reliable confidences, which additional runtime is a drawback especially during inference (Culotta and McCallum, 2004; Suzuki, McDermott, and Isozaki, 2006).

In contrast to optimizing one special value or selecting the set of output entities in prediction phase, in this work an evolutionary optimization scheme is applied to optimize recall and precision in a multi-objective way to yield different model configurations, which can be selected by an end-user depending on the respective task with higher recall or higher precision without retraining. Thereby, the non-intentional choice of precision and recall by optimization of accuracy (which is performed by maximizing the log-likelihood of the model given the training data in the case of CRFs) is avoided.

The main contribution of this chapter is therefore the presentation of multi-objective optimization for conditional random fields (MOCRf). The feasibility of evolutionary optimization in such models is demonstrated. The resulting possibility to choose a β for F_β evaluation is meaningful for a user to be able to choose depending on their current application.

¹ This chapter is based on Klinger and Friedrich (2009b).

A similar problem was addressed by Minkov, Wang, et al. (2006) with a complementary approach. They tweak the decoding Viterbi algorithm to prefer more or less entities by adding a feature

$$f(y_{j-1}, y_j, \vec{x}, j) = \begin{cases} 1 & \text{if } y_i = O \\ 0 & \end{cases} . \quad (8.1)$$

Increasing the associated weight of this feature increases the likelihood of recognizing a token as outside of an entity and vice versa (compare to the introduction of the IOB format described in Section 2.3). In comparison to this approach, the one presented in this chapter is more flexible in terms of features to be allowed to change. The results are superior as we will see.

8.2 Methods

8.2.1 Approximation of Evaluation Measures

The main idea of optimizing a CRF as proposed by Suzuki, McDermott, and Isozaki (2006) is an approximation of the evaluation measure of interest by smoothing. This allows for first order optimization approaches and may support the evolutionary scheme proposed in this chapter as well. Therefore, the approximated main parts of the evaluation function of interest are introduced here, namely true positives (TP), false positives (FP), and false negatives (FN):²

$$\widetilde{\text{TP}} = \sum_{k, s_{jk}^*} [1 - \text{sig}(d(\vec{y}^{*k}, \vec{x}^k, s^*, \vec{\lambda}))] \quad (8.2)$$

$$\widetilde{\text{FP}} = \sum_{k, s'_{jk}} [1 - \text{sig}(d(\vec{y}^{*k}, \vec{x}^k, s^*, \vec{\lambda}))] \quad (8.3)$$

$$\widetilde{\text{FN}} = \sum_{k, s_{jk}^*} [\text{sig}(d(\vec{y}^{*k}, \vec{x}^k, s^*, \vec{\lambda}))] \quad (8.4)$$

where s_{jk}^* (in Eq. 8.2) are correct segments of interest³ in the k th sequence from data \mathcal{D} . In Equation 8.3, s'_{jk} are all possible segments of interest in sequence k excluding the correct ones. The generalized sigmoid function is given by

$$\text{sig}_{\alpha, \beta}(x) = \frac{1}{1 + \exp(\alpha \cdot x + \beta)} . \quad (8.5)$$

The function $d(\cdot) \in \mathbb{R}$ decides if the segment is correctly predicted ($d(\cdot) < 0$) or not ($d(\cdot) \geq 0$).

The advantage of this approximation in comparison to the real values TP, FP and FN is that $\widetilde{\text{TP}}$, $\widetilde{\text{FP}}$ and $\widetilde{\text{FN}}$ depend on the model parameters $\vec{\lambda}$ numerically. For more details, refer Suzuki,

² Compare to Section 2.8 on page 24.

³ B and I forming the entities, not O , cf. Section 2.4 on page 18.

McDermott, and Isozaki (2006). Precision and recall based on these values are denoted with $\widetilde{\text{prec}}(\vec{\lambda}, \mathcal{D})$ and $\widetilde{\text{rec}}(\vec{\lambda}, \mathcal{D})$. The advantage of using these approximated measures instead of the original ones may be a better evolutionary optimization procedure, described in the following.

8.2.2 Non-dominated Sorting Genetic Algorithm II (NSGA-II)

The NSGA-II is an evolutionary optimization scheme for multi-objective optimization presented here briefly. More details are explained in the original work by Deb, Pratap, et al. (2002).

As usual in evolutionary computation (Bäck, Fogel, and Michalewicz, 1997), main aspects are recombination, mutation and selection of a population of individuals representing solutions of a problem. Each has one or more assigned objective values. For multi-objective optimization the population is maintained to consist of diverse solutions. The result of the process is a population of non-dominated individuals near the real Pareto-optimal front. Domination means that a solution has at least one better and no worse objective value than another solution.

In each iteration of the optimization procedure, sorting of the individuals is necessary with respect to the non-domination. The result is a partition of the population into domination fronts, *i. e.*, each individual I has an assigned rank $r(I) \in \mathbb{N}$.

As mentioned, the population needs to be diverse and cover the Pareto-front with a good spread. This is achieved by assigning a *crowding distance* $c(I) \in \mathbb{R}^+$ to each individual. This measure represents the average distance to the individuals with most similar objective values in the same front.

These two values are used to define the comparator \prec and sort the individuals of a population:

$$I_1 \prec I_2 \text{ if } \begin{cases} (r(I_1) < r(I_2)) \\ \text{or } (r(I_1) = r(I_2) \text{ and } c(I_1) > c(I_2)). \end{cases} \quad (8.6)$$

This operator is used to select the individuals to form the succeeding population; in the original work, a tournament selection (Miller and Goldberg, 1995) is proposed.

The general workflow depicted in Figure 8.1 is as follows: First, the initial parent and offspring population is generated. In the evaluation step, the individuals are sorted with respect to \prec . By selection of the q first individuals, the succeeding population is created. If the stopping criterion (*e. g.* based on iteration number or values of objective functions) is not satisfied, this population is used in the next iteration to generate offspring by recombination and mutation and so on. The final set of solutions is defined by the last population.

8.2.3 Multi-Objective Optimization of CRFs (MOCRF)

To apply NSGA-II to optimize precision and recall we need to define initialization, recombination and mutation operators manipulating the parameters $\vec{\lambda} = \{\lambda_1, \dots, \lambda_m\}$ of a CRF. Each

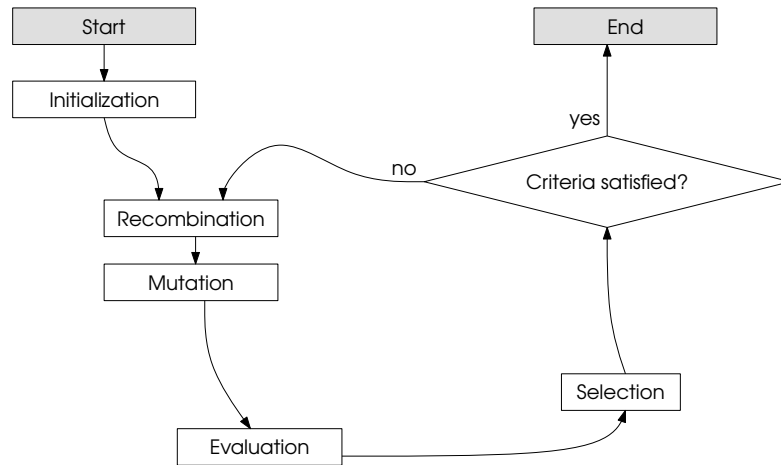


Figure 8.1: Workflow of an evolutionary algorithm.

individual in the following is represented by such a vector, therefore it is referred to them as $\vec{\lambda}_k$ ($1 \leq k \leq q$).

For initializing, a maximization of log-likelihood of an individual $\vec{\lambda}_1$ via L-BFGS is performed until convergence of the training algorithm. The initial population $P = \{\vec{\lambda}_1, \dots, \vec{\lambda}_q\}$ consists of this individual and $n - 1$ copies of the resulting parameters. The individuals $\vec{\lambda}_2, \dots, \vec{\lambda}_q$ are modified with the mutation operator $\text{mut}(\vec{\lambda})$: We add a normally distributed random value to each parameter:

$$\text{mut}(\lambda_k) = \lambda_k + \mathcal{N}(0, \sigma), \quad (8.7)$$

with $\mathcal{N}(\mu, \sigma)$ as a normally distributed random number with expectation value μ and standard deviation $\sigma \in \mathbb{R}$. This fixed step size is chosen because state-of-the-art adaptation methods like covariance matrix adaption (Hansen, 2006) needed the calculation of the covariance matrix. This is not feasible with the huge numbers of features (*cf.* Chapter 7). Additionally, exploring the Pareto-front starting at a point determined via log-likelihood optimization limits the need for exploration.

The recombination operator creates offspring from two parents (chosen by tournament selection). Two crossover variants are incorporated, in each application of recombination one is selected randomly: Intermediate recombination $\text{im}(\vec{\lambda}_1, \vec{\lambda}_2)$ or one-point crossover $\text{co}(\vec{\lambda}_1, \vec{\lambda}_2)$ (Bäck, Fogel, and Michalewicz (1997), $\lambda_{i,j}$ denotes component j of individual $\vec{\lambda}_i$; $r \in [1, n] \subset \mathbb{N}$ a uniformly distributed random variable):

$$\text{im}(\vec{\lambda}_1, \vec{\lambda}_2) = \left((\lambda_{1,1} + \lambda_{2,1})/2, \dots, (\lambda_{1,n} + \lambda_{2,n})/2 \right)^T, \quad (8.8)$$

$$\text{co}(\vec{\lambda}_1, \vec{\lambda}_2) = \left(\lambda_{1,1}, \dots, \lambda_{1,r}, \lambda_{2,r+1}, \dots, \lambda_{2,n} \right)^T. \quad (8.9)$$

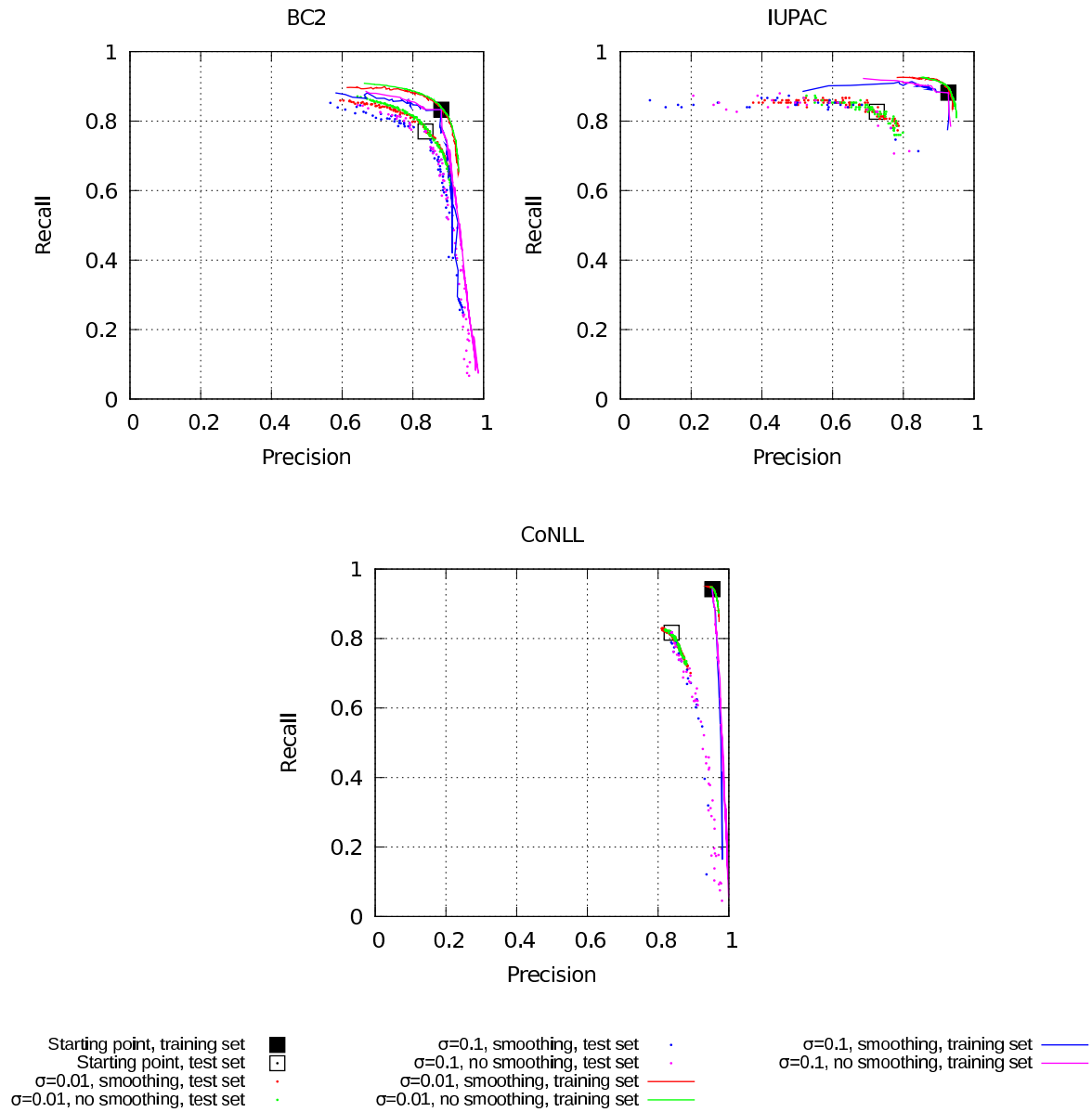


Figure 8.2: Comparison of different parameters: Step size σ with and without smoothing.

The objective functions are $\text{prec}(\vec{\lambda}, \mathcal{D})$ and $\text{rec}(\vec{\lambda}, \mathcal{D})$. Alternatively, $\widetilde{\text{prec}}(\vec{\lambda}, \mathcal{D})$ and $\widetilde{\text{rec}}(\vec{\lambda}, \mathcal{D})$ are used to evaluate if smoothing can support the optimization in the scenario presented here.

The implementation used in this work is based on Melcher (2007). It should be noted, that computing the objective functions can easily be done in parallel to decrease duration of the optimization process.

8.3 Experiments

8.3.1 Experimental Setting

The results for the proposed optimization approach are evaluated on three data sets from the field of named entity recognition, namely the BioCreative 2 (BC2) data set, as described in Chapter 5, the IUPAC data sets *IUPAC-Train-M* and *IUPAC-Test-M* as described in Chapter 4, and the CoNLL data as described in Section 7.3.1. For the BioCreative set, the configuration of the CRF using only the shortest possible annotation as exact true positive per entity is used. For CoNLL and IUPAC, an order 1 CRF with offset conjunction adding the features of one preceding and succeeding token is used. This follows the same setting as for the evaluation in Chapter 7. In all settings, a feature selection based on information gain is performed (namely *IG-OAA*, see Section 7.2.1).

The standard deviation σ (step size) used for mutating the individuals representing solutions is set to $\sigma = 0.01$. Greater step sizes would lead to a better exploration but a worse approximation of the real Pareto-front as it is shown in Section 8.3.2. All experiments are performed with a population size of $q = 100$ and 100 iterations of the multi-objective optimization. The parameters of the sigmoid function (cf. Section 8.2.1) are set to $\alpha = 1$ and $\beta = 0$ as proposed by Suzuki, McDermott, and Isozaki (2006).

8.3.2 Comparison of Smoothed and Non-Smoothed Objective Function

Smoothing the objective function as presented in Section 8.2.1 could improve the MOCRf procedure. Therefore, the comparison of the method with and without smoothing as well as with $\sigma = 0.1$ and $\sigma = 0.01$ is shown in Figure 8.2. The solid lines represent the detected Pareto front for the training sets, the dots of the same color the corresponding result on the test sets. The boxes show the results of the initial individual trained to maximize log-likelihood. Results for training from zero or random initialization are not shown as they produce significantly worse results due to local optima in the objective functions.

Highest results are achieved with a step size of $\sigma = 0.01$ without a remarkable difference of using smoothed values instead of exact measures. Using a step size of $\sigma = 0.1$ leads to a better exploration, higher results for precision can be achieved than with the lower step size, but at the expense of a lower recall: As an example, for the BC2 data without smoothing, the highest precision of training data with $\sigma = 0.01$ is 0.93 with a recall of 0.65. The same precision with $\sigma = 0.1$ leads to 0.5 recall. But with the greater step size, the

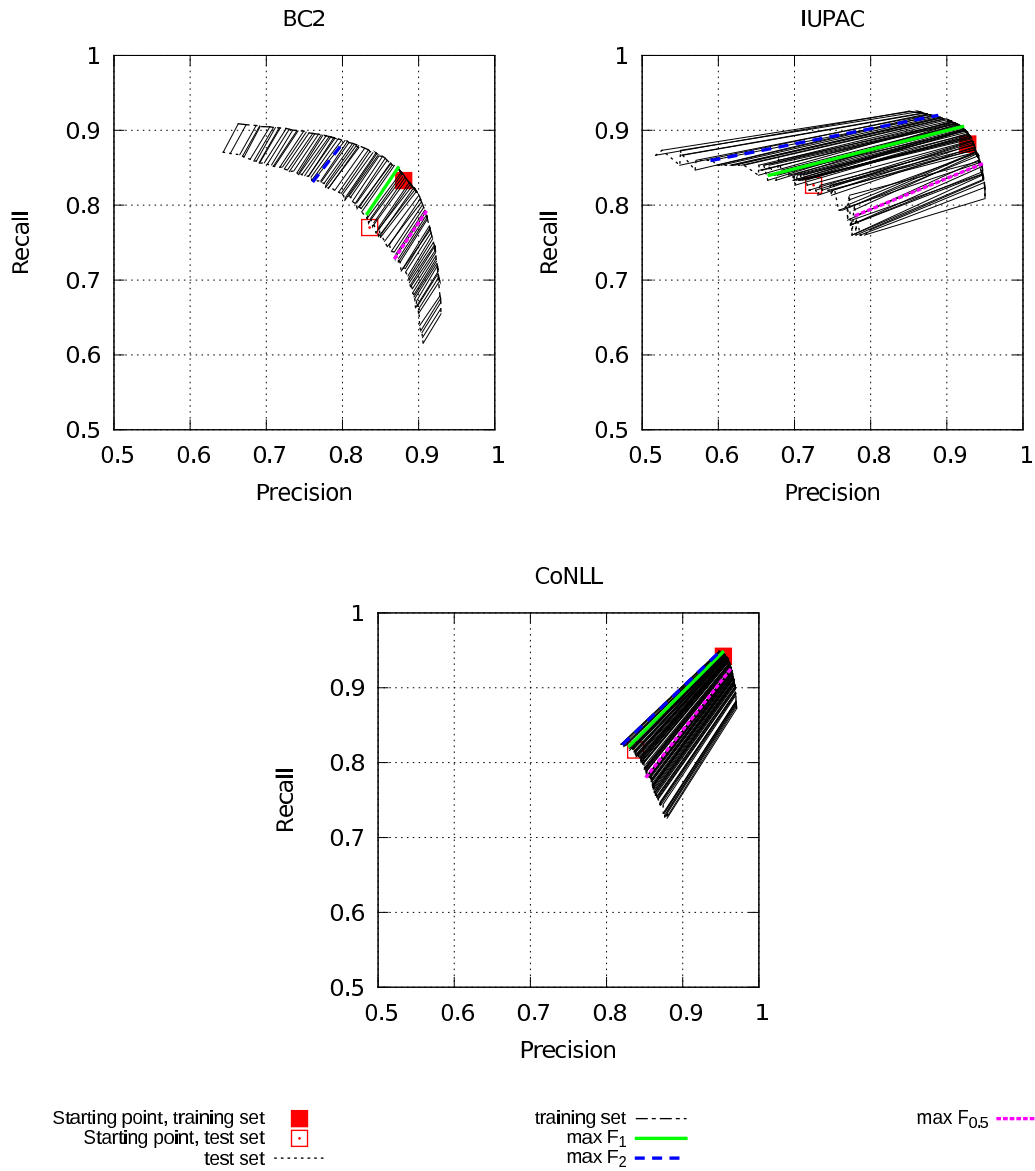


Figure 8.3: Results of the final population for $\sigma = 0.01$ without smoothing. Best F_1 , $F_{0.5}$ and F_2 values are shown in bold colored lines, selected on the training set with the according values on the test set.

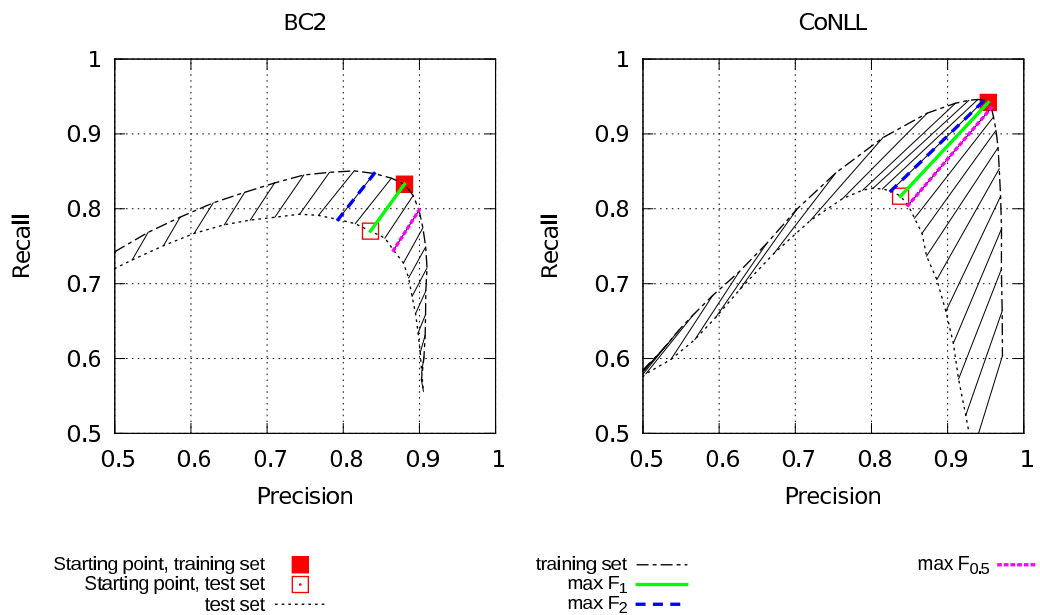


Figure 8.4: Results of the method proposed by Minkov, Wang, et al. (2006) for two data sets for comparison with MOCRE. Best F_1 , $F_{0.5}$ and F_2 values are shown in bold colored lines, selected on the training set with the according values on the test set.

highest achievable precision is 0.99. It can be concluded that the better trade-off is realized with the smaller step-size.

There is no recognizable difference in the results between the optimization with and without the smoothed objective function—the evolutionary algorithm can deal with the non-differentiable precision and recall sufficiently well.

Therefore, the results for $\sigma = 0.01$ without smoothing are examined further.

8.3.3 Results

Figure 8.3 depicts the final population for $\sigma = 0.01$ without smoothing for both data sets. The estimated Pareto-fronts for the training and test sets are shown, each individual forming one position in the plot on each front is connected with a line. The red boxes show the results of the initial individual trained to maximize log-likelihood. The blue, green and red line show the individual with highest F_2 , F_1 and $F_{0.5}$ measure respectively.

The Pareto-front on the training set is the one determined by MOCRE. The results shown as Pareto-front on the test set are the results of the same individuals connected by a line. The absence of crossings to a large extent shows that the generalization from the results on the training set to the results on the test set is feasible. To compare the results presented here with the previous approach by Minkov, Wang, et al. (2006), the results of their method are shown on the BC2 and CoNLL data sets in Figure 8.4.

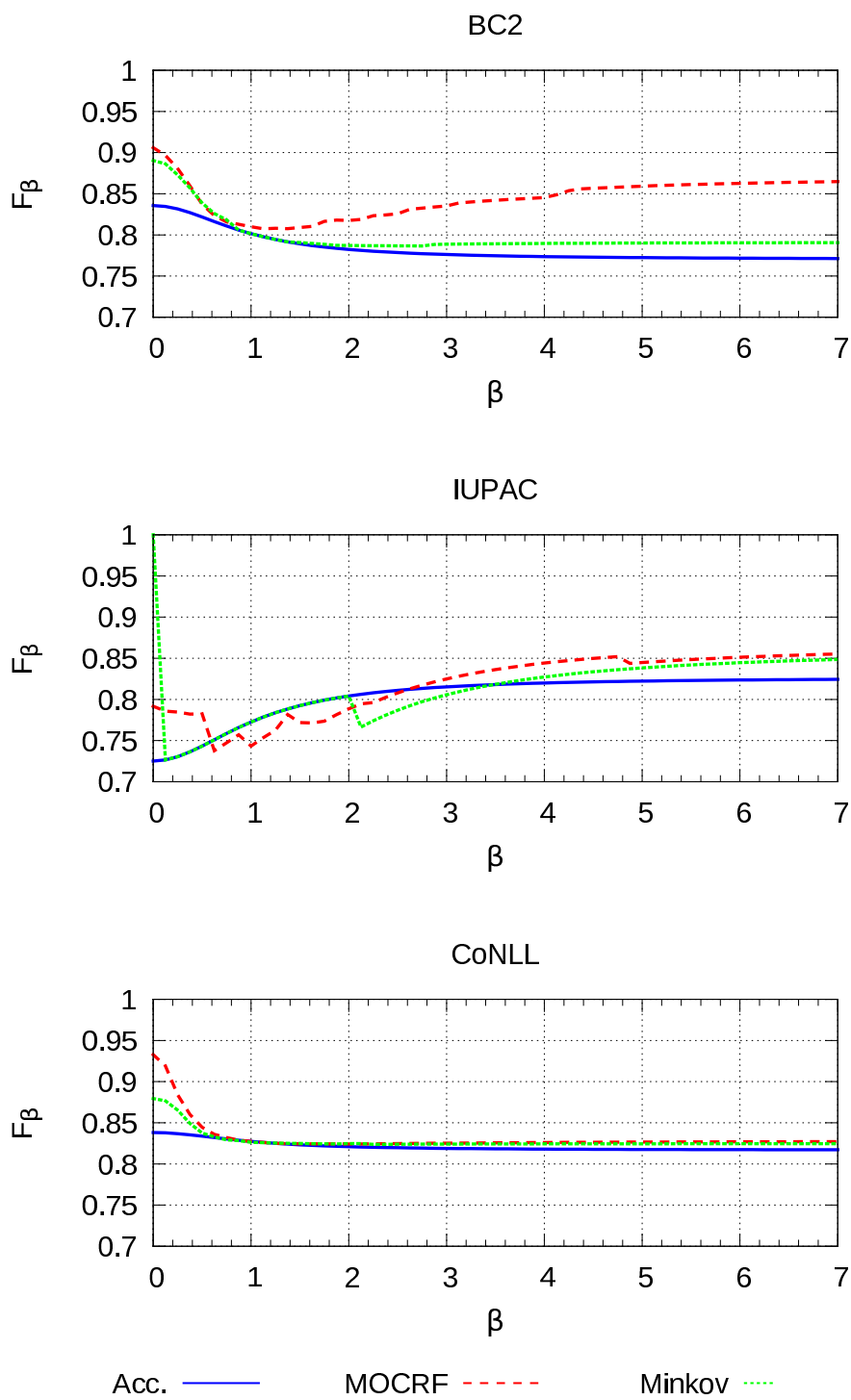


Figure 8.5: Results in F_β for different β on the result obtained via L-BFGS training w.r.t. log-likelihood and MOCRf.

Data Set	L-BFGS					MOCRf				
	$F_{0.25}$	$F_{0.5}$	F_1	F_2	F_4	$F_{0.25}$	$F_{0.5}$	F_1	F_2	F_4
BC2	83	82	80	78	78	88	84	81	82	85
IUPAC	73	74	77	80	82	78	78	74	79	85
CoNLL	84	83	83	82	82	87	84	83	83	83

Table 8.1: Results for classic L-BFGS training in comparison to MOCRf in %. Given are the best available F_β measures for $\beta = \{0.25, 0.5, 1, 2, 4\}$, as well as the result for L-BFGS for different data sets. All results are equal or better than for L-BFGS training for BC2 and CoNLL which does not optimize with respect to a special β value. These results are shown graphically in Figure 8.5.

It is noticeable, that the fronts in Figure 8.3 seem to be differently well explored in BC2, IUPAC and CoNLL data. On BC2 data, precision as well as recall can be increased at the expense of the other measure: The starting point is an F_1 measure of 86% with a precision of 88% and a recall of 83% on training data, highest possible precision is 93% (difference 5%), highest possible recall is 90% (difference to start: 7%). On CoNLL data, the starting point is an F_1 measure of 94% with a precision of 95% and a recall of 94% on training data, highest possible precision is 97% (difference 2%), highest possible recall is 95% (difference to start: 1%). On IUPAC data, the starting point is an F_1 measure of 90% with a precision of 93% and a recall of 88% on training data, highest possible precision is 95% (difference 5%), highest possible recall is 90% (difference to start: 2%).

This difference between the data sets is founded by the structure of the problem and the different dependencies of the objective functions on the data sets. In both cases, a spread set of solutions is made available by the proposed method. On the IUPAC data, the generalization to the test data is not as good as for CoNLL and BC2: The according training set is selected via active learning (*cf.* Section 2.2) and therefore not sampled in the same manner as the test set.

Assuming a user asking for a model characterized by an F_β measure with fixed β , the provided system multi-objectively trained exhibits better performance than the one trained to maximize log-likelihood for CoNLL and BC2. This is shown in Table 8.1 and Figure 8.5. On BC2 data, the results are better for all β values, for CoNLL data the results are the same for F_1 , but superior for all other values. For the IUPAC data, the results even slightly decrease, presumably due to the active learning assembled training data—during the process of active learning, log-likelihood-based training was applied.

On all data sets, the precision is higher than the recall for the model trained on log-likelihood. Therefore, F_β is monotonically decreasing for that method. For MOCRf, higher values of precision than for recall are achieved. On BC2 data, this even leads to a minimum of F_β for $\beta = 1$ as the same precision and recall are more difficult to achieve than other weightings. On CoNLL data, the exploration of recall is not as successful as on BC2 data.

The results for Minkov, Wang, et al. (2006) are shown additionally in Figure 8.5. Obviously,

their method provides an increase for the investigated values of β with a notably simple method. It has to deal with the same limitations as MOCRF, *i. e.* generalization problems for IUPAC and only a small exploration in the direction of a higher recall for CoNLL. MOCRF is superior or equal for all shown β on the three datasets.

8.4 Conclusion and Future Work

This chapter presents the application of multi-objective optimization via NSGA-II to maximize precision and recall in conditional random fields for named entity recognition. It is shown on all data sets that F_β measures for nearly all β could be increased in comparison to classical maximization of log-likelihood via L-BFGS and in comparison to the method of Minkov, Wang, et al. (2006). It has to be admitted that their method is simpler and produces a model with a different ratio of precision and recall faster. Nevertheless, after development and training via MOCRF, the complexity of applying a selected parameter set with an associated F_β measure is the same. This enables an end-user to choose a model with higher recall or precision without retraining or time-consuming computation of confidence measures. Possible applications include information retrieval with the need for a high recall to find most of the possible results, *e. g.* documents from a database as well as information extraction, where a high precision can help to detect correct relations between named entities. A time-consuming retraining can be avoided.

Additionally, the change of multiple features to spread solutions between precision and recall allows for an analysis of the problem: Comparing models reveals which features are responsible for confident solutions and unconfident solutions. This supports the understanding of a specific problem.

Main future work is to evaluate other multi-objective optimization heuristics to improve the result in terms of a higher spread of solutions and possibly a better approximation of the real Pareto-front. An integration of the initial training into the multi-objective optimization is also desirable.

Chapter 9

Incorporating Distant Information via Automatically Selected Skip Edges

9.1 Introduction

Many applications in the field of text segmentation, especially named entity recognition, have been addressed with linear chain conditional random fields. This structure is used in the applications presented in Part II and as a foundation for the improvements in Chapter 7 and 8 of Part III. Using a linear chain of variables to represent the labeling of text is straight forward, as processing text in a sequential manner suggests itself due to the way it is written and firstly perceived.

While language suggests this linear structure to represent written text, it does not necessarily model all dependencies: Co-referencing a prior entity is an example (while it could be seen as higher order linearity typically pointing back but not forward). Especially in non-scientific texts, information may as well be left out and filled in later to keep a story exciting. Another example is the use of filler words (or stop words or adjectives) as a trivial case where the meaning of words can be determined by distant tokens.

How are these thoughts related to the case of named entity recognition? The capabilities of a linear chain structure may be limited in at least two cases: Firstly, relations between distant tokens can have an impact on their meaning. This is a motivation which lead to the previous work presented in the following Section 9.1.1. Secondly, long entity classes cannot be captured as a whole, which is especially interesting because a characteristic of named entities in biology and chemistry is their high length with interdependencies between tokens of an entity. The distribution of the length of terms in the classes of gene names (BioCreative 2), IUPAC names and person names, organizations and places (CoNLL) is shown in Figure 9.1. Gene names and especially IUPAC names are much longer than entities like names, organizations and places. It needs to be investigated if a linear-chain conditional random field can capture this complexity or if another structure helps detecting such entities.

In the following, this challenge is approached as a search for meaningful skip chain templates (Sutton and McCallum, 2007).

9.1.1 Previous Work

The class of CRFs including skip chain edges (unrolled from skip chain templates) has been described by Sutton and McCallum (2007) and Galley (2006) in a named entity recognition scenario. In addition to the linear chain, a template is used to measure the dependencies

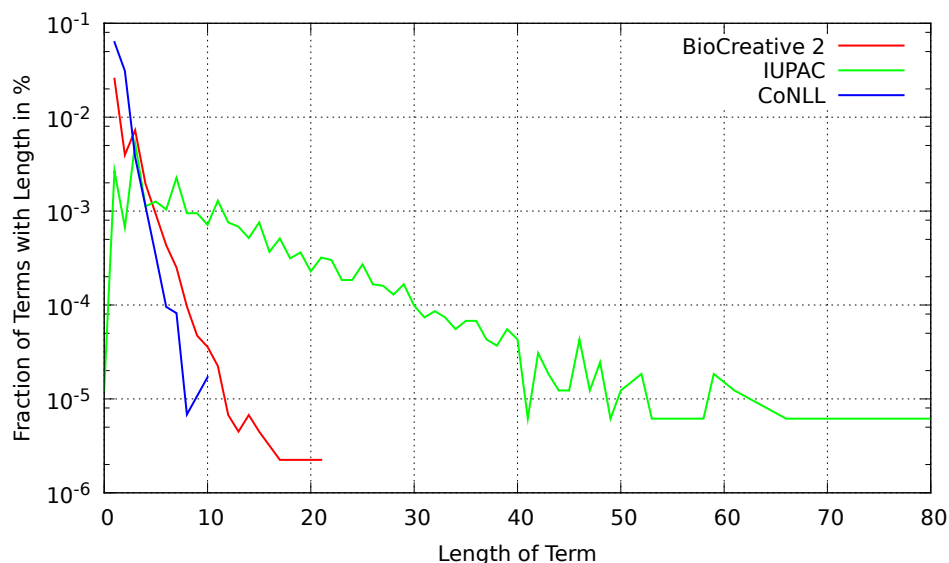


Figure 9.1: Distribution of the length of three entity classes.

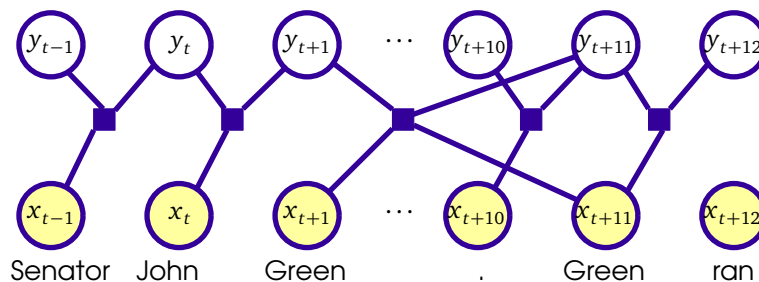


Figure 9.2: Example of a skip chain CRF structure as used by Sutton and McCallum (2007) (as factor graph depiction). Subsequent labels are connected as well as tokens representing the same string.

between same capitalized tokens. This is motivated by the assumption that same words in a sentence or document are likely to have the same label, despite of their token distance. An example for such skip chain CRF is shown in Figure 9.2. As stated by Sutton and McCallum (2007), each pair of nodes can be connected by a skip chain which the developer believes to be similar. They point out that the number of edges unrolled from a template may not be too high as the runtime and memory consumption increases prohibitively. Connecting only capitalized words allows to match most proper names (which is an entity class of interest in their test domain) while they are sparsely distributed.

The work by Liu, Huang, and Zhu (2010) enhances that approach by different classes of variables (as special keywords) to be connected. To adapt Sutton’s and McCallum’s approach to gene and protein names, they introduce three skip chain templates: Firstly, connecting the main parts of gene names (referred to as “keyword” in their work) defined by regular

expressions, secondly connecting only similar keywords, only differing to a certain extent, and thirdly connecting typed dependencies like prepositional modifiers or noun compound modifiers. On the BioCreative 2 NER data set presented in Chapter 5 they show an increase in F_1 measure from 71.73% with a linear chain to 73.14% with the best skip chain configuration for a strict evaluation not using the allowed alternatives in the gold data test set. Using the official evaluation with alternatives, they show an increase from 83.29% to 84.67% F_1 . They argue that the quality of the skip chains is essential for the improvement of the result compared to simple linear chain structures.

In contrast to previous work, this thesis addresses the question how to select meaningful skip edges automatically from a set of possibilities. This does not make the domain specific development unnecessary (as the application of feature selection still needs the development of features for a domain) but helps to find templates to improve the results. It can select a specific subset of automatically generated clique templates. This task can be understood as a combinatorial optimization problem: Finding a factor graph with a structure maximizing the performance of the model on a test set.

Several approaches have been published about optimizing the structure of Markov networks or more specifically conditional random fields. They can be divided into methods searching for such structure with a measure to judge the quality of a structure and filtering approaches to decide about the quality of an edge. Beside those, regularization is another way to find a good structure during training.

The work by Schmidt, Murphy, et al. (2008) states to be the first dealing with the structure learning task in discriminatively trained, undirected graphical models. Similarly to Lee, Ganapathi, and Koller (2006) (which is dealing with general Markov networks), L_1 regularization is the incorporated method. While this approach is very elegant due to the joint structure and parameter estimation, it has limitations to deal with large, dynamically generated factor graphs with a lot of features on each factor.

As long as the candidates for the optimal structure have a tractable size, a search in the space of graph structures is feasible. This approach, together with an approximation for the quality measure of each graph is adopted by Parise and Welling (2006). The advantage is that all dependencies in the graph are taken into account, the disadvantage is the complexity of the performed search.

A complementary and fast approach is to measure the quality of an edge with independence tests, as described by Bromberg, Margaritis, and Honavar (2009). The main contribution in their work is to minimize the needed independence tests to find the optimal graph structure.

9.1.2 Problem Description

The work described in Section 9.1.1 is focusing on graph structures of limited size or on non-conditional Markov graphs. In the following, the problem of finding skip edges is discussed in detail to understand the limitations of previous approaches.

A graph structure $G = (V, E)$ is defined via vertices V and edges $E = V \times V$. Optimizing the

structure corresponds to selecting a subset of edges which leads to the maximal performance. While this formulation is sufficient to understand the approaches described earlier, the formulation in terms of factor graphs is beneficial for the detection of skip chain edges.

Revisiting Section 3.4.3 on Page 53, a factor graph (Kschischang, Frey, and Loeliger, 2001) is a bipartite graph G between variables and factors defining a probability distribution of a set of output variables \vec{y} conditioned on input variables \vec{x} . Each factor Ψ_j computes the so-called score of variables which are neighbors in the graph. It is typically formulated as an exponential function of the weighted sum of features:

$$\Psi_j(\vec{x}, \vec{y}) = \exp \left(\sum_{i=0}^m \lambda_i f_i(\vec{x}_j, \vec{y}_j) \right). \quad (9.1)$$

A set of factor templates $\Theta = \{\theta_1, \dots, \theta_n\}$ consists of templates θ_k describing a set of tuples $\{(\vec{x}_k, \vec{y}_k)\}$ on which factors are instantiated for which the property $p_k(\vec{x}_k, \vec{y}_k)$ holds and shares $\vec{\lambda}_k$ and $f(\cdot)_k$ between all instantiated factors on the tuples. K_j is the number of parameters of the j th template. The probability distribution on a factor graph with templates Θ becomes

$$P(\vec{y}|\vec{x}) = \frac{1}{Z(\vec{x})} \prod_{\theta_j \in \Theta} \prod_{(\vec{x}_i, \vec{y}_i) \in \theta_j} \exp \left(\sum_{k=1}^{K_j} \lambda_{jk} f_{jk}(\vec{x}_i, \vec{y}_i) \right). \quad (9.2)$$

The task of finding meaningful skip chains corresponds to finding a set of templates $\hat{\Theta}$ describing tuples (y_u, y_v, \vec{x}) with a property $p(x_u, x_v)$. A linear chain template θ_{lc} with features $\vec{g}^{lc}(\vec{x}, j)$ for all possible combinations of label variables (as described in Section 3.4.2) is assumed to be present in all configurations. In the following, the set of templates $\hat{\Theta}$ to select from is defined by properties $p_k(x_u, x_v) := \text{holds iff } g_k^{lc}(\vec{x}, u) = 1 \text{ and } g_k^{lc}(\vec{x}, v) = 1$ with $k \in \{1, \dots, |\vec{g}^{lc}|\}$. Each template holds parameters for features $g_k^{lc}(\vec{x}, u) \vee g_k^{lc}(\vec{x}, v)$. That definition of $\hat{\Theta}$ is only for simplicity throughout this chapter. A more general formulation does not limit the methods described, though a small $|\hat{\Theta}|$ decreases runtime. Especially dependency properties as described by Liu, Huang, and Zhu (2010) may be included. Note that token similarity and key token templates are included via brief word class features in the prior definition (cf. feature definitions in Section 2.6).

An example of different skip chain factors to choose from is shown in Figure 9.3. The red property of matching `[.*ine]` seems to be a reasonable skip chain as it connects similar chemical names such that their class can influence the class of the others. The green matching stop words is an example how the size of the factor graph can exponentially increase what should be avoided. The orange matching of `[,]` could be able to capture enumerations because features which take preceding and succeeding tokens into account may have some importance.

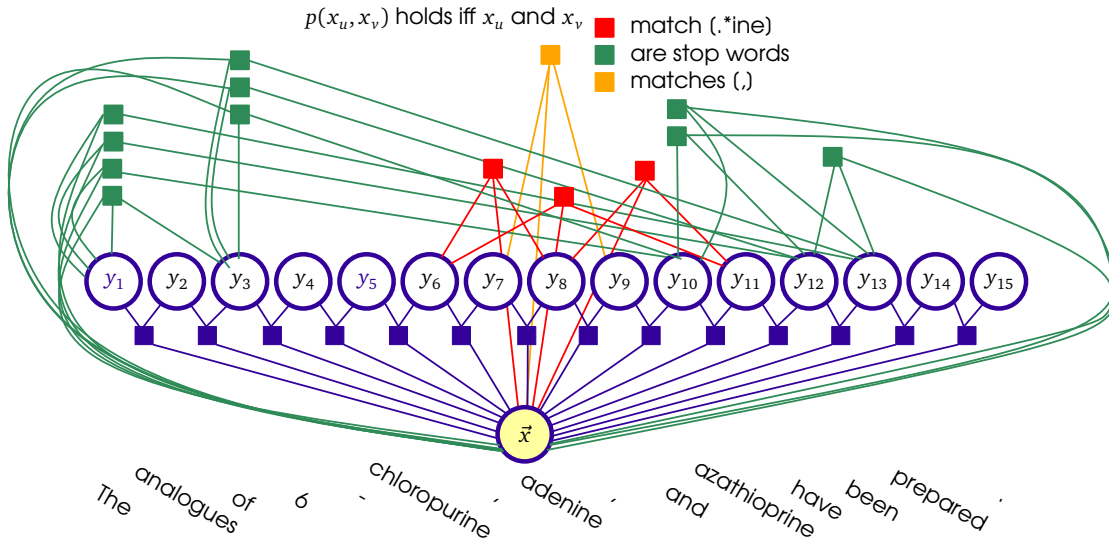


Figure 9.3: Different skip chain factor templates to choose additionally to the linear chain (example shortened from the abstract by Hasan and Srivastava (1992)).

9.2 Methods

As $\hat{\Theta}$, the set to select skip chain templates from can be very large, regularization or full searches cannot be applied.

Instead, a two step method is used. Firstly, obviously not beneficial templates are filtered. Secondly, a best-first-search is performed to find a meaningful combination. Two filter steps are tested, described in Sections 9.2.1 and 9.2.2. The best-first-search is described in Section 9.2.3.

9.2.1 Isolated filtering

Similarly to the application of information gain on feature selection, as described in Section 7.2.1, this measure can be used to decide about the quality of the features in a skip chain (while Chapter 7 deals with features of the linear chain template).

To apply the information gain measure for the quality of a feature on a skip chain template, instances as part of a classification problem are generated in compliance with the description in Section 7.2.1: For every generated factor $\Psi_j(y_u, y_v, \vec{x})$, an instance is built. The label is the combination of the values of both label variables, the boolean features $\vec{g}_j(\vec{x}_v, \vec{x}_u)$ are combined from $\vec{g}^{lc}(x_u)$ and $\vec{g}^{lc}(x_v)$, the features from the linear chain template on the corresponding positions representing only the input variables:

$$g_{jk}(\vec{x}, v, u) = \begin{cases} 1 & \text{if } g_k^{lc}(\vec{x}, v) \vee g_k^{lc}(\vec{x}, u) \\ 0 & \text{else} \end{cases} \quad (9.3)$$

The measure for a template can then be based on the features representing that instance. Note that the remainder (Equation 7.3) is sufficient to decide about the importance of one feature of a fixed template, but to distinguish between different templates based on these features, the full information gain needs to be taken into account (Equation 7.1 on page 108) as the number of instances is different.

In the following, the mean value $IG_{\text{mean}}(\theta_j)$ of the information gain of all features is used as a measure for the generating template θ_j :

$$IG_{\text{mean}}(\theta_j) = \frac{1}{|\vec{g}_j(\cdot)|} \sum_{k=1}^{|\vec{g}_j(\cdot)|} IG(g_{jk}(\cdot)). \quad (9.4)$$

The set $\hat{\Theta}'$ as a possible input for the filtering step described in Section 9.2.2 is

$$\hat{\Theta}' = \{ \theta_j | IG_{\text{mean}}(\theta_j) > t \} \quad (9.5)$$

with a specified threshold $t \in \mathbb{R}$.

9.2.2 Filtering via Markov Blanket based Graph Reduction

A second test to select templates which have a positive impact on the result, taking the linear chain into account is described in the following. It would be possible in principle to incorporate information theory based measures here as in the method described in Section 9.2.1. The drawback is the runtime in practice: The number of features is too high to compute the information gain given the features of the linear chain.

Instead, a more intuitive and feasible approach based on the Markov blanket of the variables touched by a skip chain template is proposed. In general, a Markov blanket of a node v in a graph are all its neighboring nodes, that means, given all these nodes, it is independent given all other nodes in the graph (compare to Figure 9.4). Therefore, to judge about a skip chain edge, it is sufficient to measure the importance on the variables of the Markov blanket of the variables touched by the skip chain templates. Examples for these variables given skip chain factors are shown in Figure 9.5. Therefore, the reduced graph $G_{MB}^j = (\vec{\Psi}_{MB}, \vec{x}_{MB}, \vec{y}_{MB})$ for a template T_j is derived from the fully unrolled original factor graph $G = (\vec{\Psi}, \vec{x}, \vec{y})$ by keeping all variables touched by factors instantiated by T_j , all factors instantiated by T^{lc} which include one of those variables as well as all variables in those.

Comparing this reduced graph to an unrolled graph with the same variables but without the skip chain factors can be performed by training and evaluation via exact algorithms as these graphs are sparse (cf. Section 3.4.3). The templates where this comparison shows better accuracy with skip chain factors than without on a hold-out set is included in the set of templates to participate in the best-first-search.

9.2.3 Best First Search on Templates

The most complete approach to find a suitable structure of the graph is a search through the space of all combinations of the skip chain templates. As the complexity is exponential in the

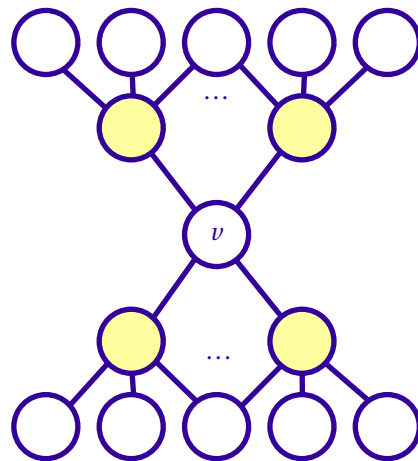
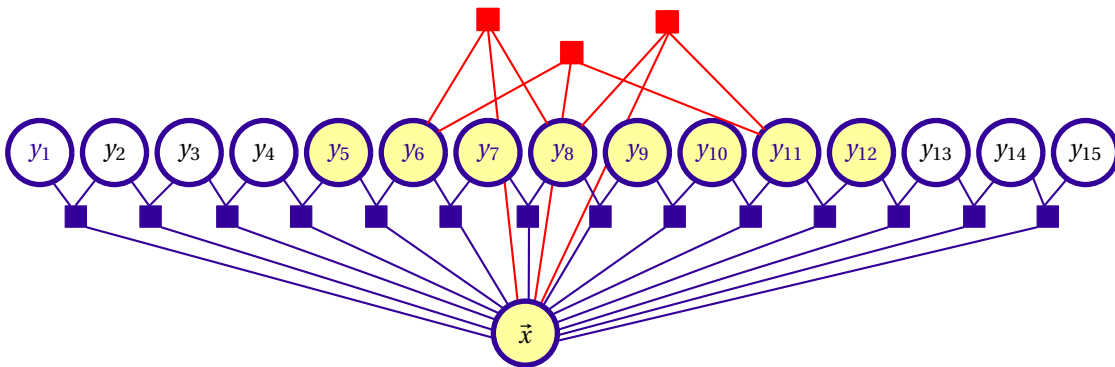
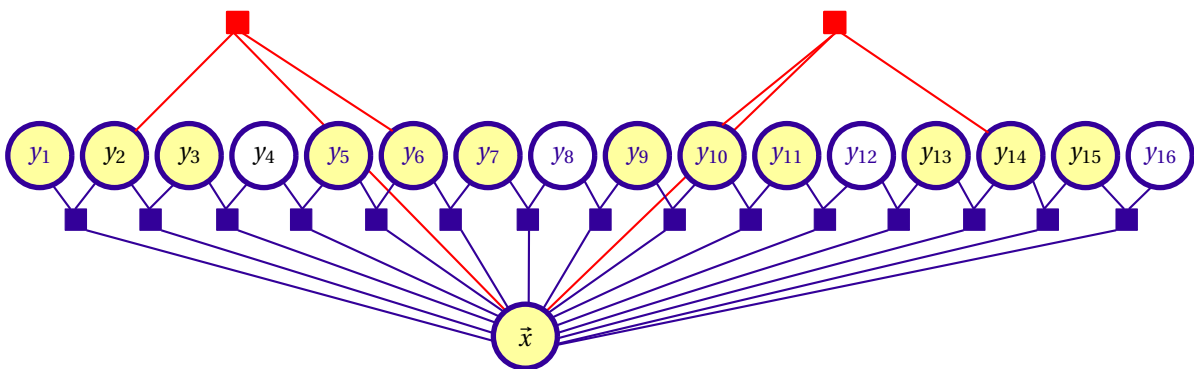


Figure 9.4: Markov blanket (shaded nodes) of a node v .



(a) Template “match *.ine” from Figure 9.3



(b) Examples with two distant factors

Figure 9.5: Examples for the Markov blankets of variables touched by the skip chain template. The skip chain factors are shown in red.

number of templates for specific properties and therefore prohibitive even for a small set of templates, the dependencies between possible templates are measured via best-first-search (BFS, Russell and Norvig (2003)) on all templates remaining from the filtering together with T^{lc} . Best-first-search is chosen here as an exemplary search strategy as it follows only the best alternative which limits the performance evaluations needed.

Starting with the linear chain, each template is added respectively, the model is trained and evaluated on a hold-out set. The best template is kept and the prior step is repeated. This process ends if none of the templates can improve the result.

Here, the same inference algorithm is used as for the final model: Loopy belief propagation with tree-based reparameterization for approximative inference (Wainwright, Jaakkola, and Willsky, 2001). During the steps of BFS, the weights for each template to be kept are adopted for the next search iteration. Thereby retraining the model can be performed in a smaller number of iterations than training from scratch.

9.3 Results

In the following, the two filter steps are analyzed (Section 9.3.1). Based on all proposed templates, the possible impact is discussed (Section 9.3.2).

9.3.1 Analysis of the Filtering Steps

The core of the proposed approach is the best-first-search, assuming that the set of templates to choose from is meaningful. To reduce the runtime of the best-first-search to a feasible period, these templates are filtered first. That presumes that no important templates are removed in this step. To analyze the two straight forward filtering approaches described in Section 9.2.1 and 9.2.2, 1000 templates are sampled from all proposed templates (as described in Section 9.1.2 for the IUPAC data set (described in Chapter 4)). The set *IUPAC-Train-M* is randomly split into a training and validation set of 417 and 46 instances respectively. Training a model for each of the 1000 templates and comparing it to the values for the filtering heuristics leads to the results depicted in Figure 9.6 and 9.7.

Both analyzes do not show a meaningful relation to the empirically determined important skip chains from the sampled set. The empirical evaluation shows that a subset of skip chain templates leads to an improved F_1 measure in comparison to the baseline with the linear chain only. But these successful templates are not all determined by the filtering heuristics.

To get an impression of assigned heuristic values for filtering and empirical values, the top 10 templates are depicted in Table 9.1. All three methods to determine important skip chain templates intuitively lead to reasonable results concerning the test domain of interest (chemical IUPAC names). The information gain based ranking mainly selects factors to connect words occurring close to IUPAC names or parts of them. Evident examples repeatedly used in a sentence showing the support to decide about a term are:

- D-Pen disulfide bridge

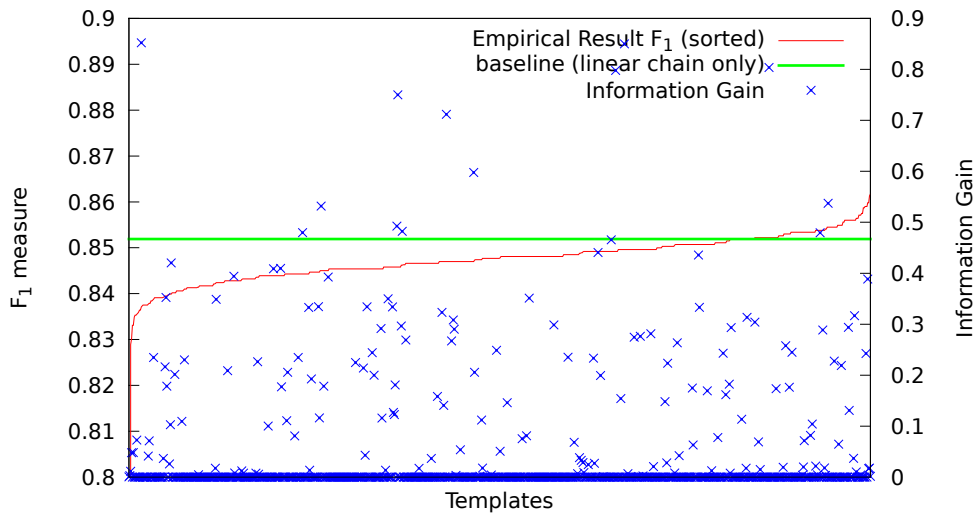


Figure 9.6: Information Gain-based filtering results compared to empirical results for a sampled subset of skip-chain templates.

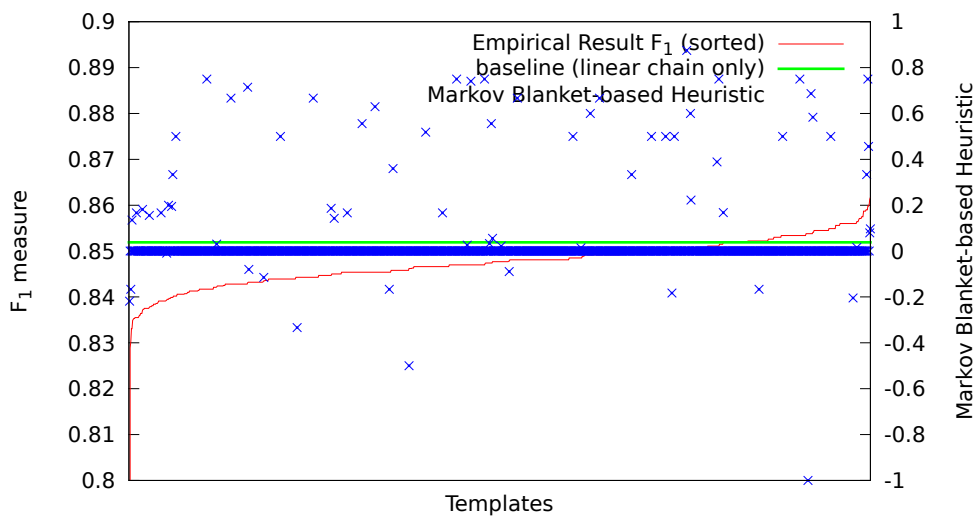


Figure 9.7: Markov blanket-based filtering results compared to empirical results for a sampled subset of skip-chain templates.

	IG	MB	Empirical
1	W=5.6@2	W=1,3@-2	PREFIX2=in@1
2	W=disulfide@2	W=dimethyl	W=6@2
3	W=formation@-2	W=isoxazoly1@-1	PREFIX2=ph
4	W=dihydropyridinium@1	W=0	W=phenylethyl@2
5	W=phenylurea@-2	W=0@2	W=brainstem@-1
6	W=6H@2	W=tri	W=trifluoromethyl@2
7	W=6H@-2	W=6H@2	W=injection@-2
8	W=display@2	PREFIX2=3b@1	SUFFIX2=g2
9	W=TFPe@2	W=pyrido@-2	W=antidepressant@-2
10	W=respective@-2	PREFIX2=2b@-1	W=xenograft@1

Table 9.1: Top 10 skip chain templates determined by information gain-based filter, Markov blanket-based filter, and by training and comparison.

- N-(trifluoroacetyl)daunorubicin and disulfide
- the predicted formation of 4,4,4-trifluorocrotonaldehyde
- the formation of [^{99m}Tc]TRODAT-1 complex
- the formation of N-[2-(chloroethoxy)ethyl]norapocodeine
- the 1-methyl-4-phenyl-2,3-dihydropyridinium species 2,3-MPDP+

While these examples show that the selected skip chains are meaningful, they let us assume that these characteristics may be captured by the linear chain alone—a relation not measured by the proposed filtering method. Structurally similar templates are selected using the Markov blanket based filter. Empirically, templates having an impact on the result are more general words and prefixes. Examples are:

- the presence of a phenol group, irrespective of the nature of tertioalkyl group, imparted at least partial RAR gamma selectivity, whereas in series II, the presence of both adamantyl and phenol groups
- to be photochemically active and is a potential photoaffinity
- to the phenyl 4-position
- (7,8-dichloro-1-methyl-4-[4-(methylimidazo[4,5-c]pyrid-1-yl) phenyl]-2,3,4,5-tetrahydro-1H-1-benzazepin-2-one)
- following injection of citalopram
- after injection of [¹⁸F]4d

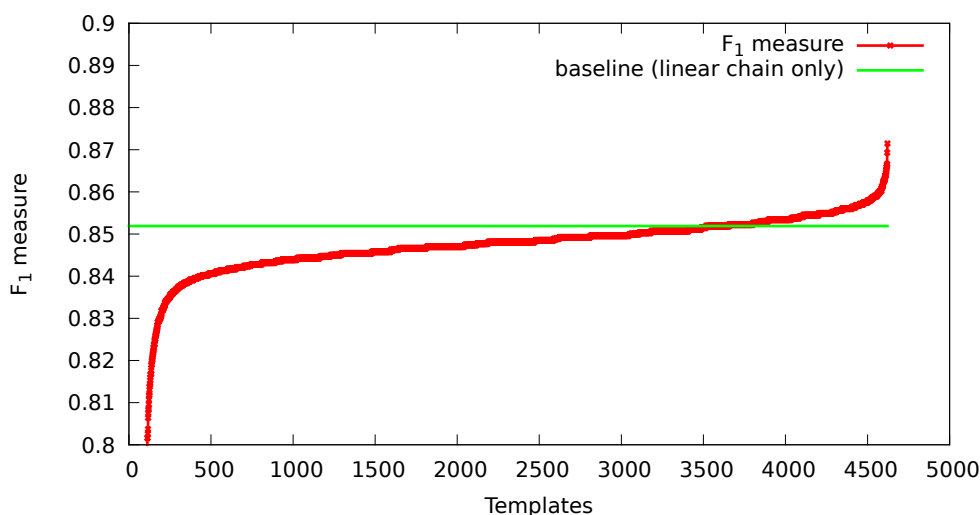


Figure 9.8: Results for all proposed templates measured empirically.

- exhibited antidepressant-like activity

Some of these examples are an indication that the skip factors are more important to decide about a token to be not part of an entity, instead of supporting their annotation.

9.3.2 Analysis of all Proposed Templates

Despite of the limited performance of the filters, the basic idea of searching for meaningful skip edges is analyzed. On the same set as described in Section 9.3.1 (IUPAC, split into training and validation), all templates which occur at least 10 times in the training data were taken into account. These are 5096 templates (from 16710 altogether). Analogous to the sampled subset used in the previous section, training with and without the skip factors leads to an empirically determined list of meaningful templates.

This leads to the results depicted in Figure 9.8. The inference algorithm did not converge for 474 templates. Most of the templates have no or negative impact on the result (3656 templates); 966 have positive impact. The top 10 templates are depicted in Table 9.2, together with their contribution. The most important template is to add a skip chain between tokens “alpha” with nearly 2% improvement in comparison to the linear chain only. This term occurs 319 times in the training data and is frequently part of IUPAC names (115) as well as outside of them (204). In a local, linear chain-based setting a feature based on this token can hardly contribute to a decision, but in a distant labeling setting it can. Similarly the second best feature to build skip chain factors, “PREFIX2=tr”: It occurs 698 times, 284 times in IUPAC names and 414 outside of them.

Most of the features forming the basis for templates with a positive impact are measuring words occurring close to or in chemical names or are typical chemical pre- or suffixes. These

	Template	F_1 measure	Diff. to baseline
1	W=alpha@2	87.15	1.96
2	PREFIX2=tr	86.93	1.74
3	W=liver@1	86.66	1.47
4	W=group@1	86.63	1.44
5	WC=AAA@1	86.59	1.40
6	PREFIX2=ox	86.59	1.40
7	PREFIX2=fu@1	86.54	1.35
8	W=rac@1	86.51	1.32
9	W=cis@-1	86.47	1.28
10	SUFFIX2=nt	86.44	1.25
11	SUFFIX2=ro@-2	86.39	1.20
12	W=was@-2	86.39	1.20
13	W=rac@-1	86.39	1.20
14	PREFIX2=hy@1	86.39	1.20
15	PREFIX2=am@1	86.36	1.17
16	SUFFIX2=31	86.32	1.13
17	SUFFIX2=,4	86.32	1.13
18	SUFFIX2=in@2	86.28	1.09
19	SUFFIX2=,4@1	86.28	1.09
20	W=3H@-1	86.28	1.09

Table 9.2: Top 20 skip chain templates from all proposed templates occurring at least 10 times.

features measure ambiguous characteristics of tokens where the probability of correctly identifying the surrounding terms can be increased by measuring the distant information of them. The context is taken into account by templates based on features of offset conjunction (like W=alpha@2 measuring the token alpha two tokens left of the skip chain connection). The reason is presumably that their common occurrence in an instance is not labeled differently.

For instance is alpha occurring as alpha-ribofuranosyl (which is labeled as IUPAC) or alpha1-adrenergic (not labeled as IUPAC). In both examples, alpha is occurring multiple times in the instance, but not with different labels. Similarly, tr can be a prefix of tributylstannyl (as IUPAC) or treatment (as non-IUPAC), but it is not probable that different labels occur in the same instance. The feature W=group@1 is a slightly different case as it does not occur as part of an IUPAC name itself but can occur in the context of chemical names which are difficult to distinguish between IUPAC and not. As an example, formamidino group would not be labeled as IUPAC, but p-methoxybenyl group is labeled as such. Annotation is quite difficult here, but it is likely that the annotator produced consistent data in one instance.

Noteable is the occurrence of very strict features like SUFFIX2=31—it is surprising that obviously some numbers are occurring frequently in IUPAC names and outside of them such

that a differentiation with distant information is beneficial.

This discussion illustrates that the found templates are meaningful in the context of the IUPAC example taken for evaluation here. The impact of the automated approach shows similar or better possible improvements as the manual approaches by Sutton and McCallum (2007, cf. Table 1.2) (0.4%) or Liu, Huang, and Zhu (2010) (1.2% on BioCreative II data) (although they tested on different data sets).

9.4 Conclusion and Future Work

This chapter presented the principle idea of building skip chain edges to capture distant information for named entity recognition in a similar manner as features to represent tokens are generated. Instead of hand-crafting domain and problem specific features, they are generated from the training data; analogously, potentially beneficial skip chain templates are taken into account. It has been shown that this approach is feasible and leads to an improved performance on an example domain.

To be able to apply this methodology in practice, the search complexity for meaningful structures needs to be reduced. Two filtering methods are proposed but they could not be proven to work perfectly. Nevertheless, the analysis shows that the idea of automatically selecting distant tokens as a basis for additional factors makes sense. The presented analysis can help in further work and be used as a training set for novel template filtering methods.

Future work includes the analysis how different templates work together for named entity recognition: What is the relation between the linear chain and a skip chain? What characteristics does the linear chain have where skip chains help?

Additionally due to the high complexity of empirically testing all templates, the interaction between different skip chains has not been analyzed.

Another interesting topic is to investigate the impact of specific factors on different evaluation measures. It can be assumed that some support accuracy, whereas some have a special impact on precision or recall.

Part IV
Recapitulation

Chapter 10

Summary and Conclusions

Conditional random fields are established as a state-of-the-art approach to many named entity recognition tasks. Nevertheless, especially the classes from biology and chemistry harbor challenges. These are, beyond others, high (or even infinite) numbers of disjunct entities to be described and therefore the lack of structured resources for all of them. Additionally, the variability of the use and context of them is high. This has been exemplified on the recognition of IUPAC and IUPAC-like names, variation mentions and gene/protein mentions in this thesis. The parameter analysis and the huge amount of typically incorporated features in these use-case studies shows the need for manual work to adapt models to specific domains.

Therefore, the generalizability of a CRF configuration is one main challenge addressed in this thesis. This involves to simplify the workflow to build domain specific models. Generating better models by means of performance and speed goes hand in hand with this goal.

Three main application areas are addressed in this thesis. By means of a generic workflow for named entity recognition (introduced in Chapter 2) conditional random fields (discussed in Chapter 3) adaptations for these real world scenarios are presented. Especially the development of domain specific features is beneficial for the performance of a model and reveals specific demands. IUPAC names (Chapter 4) are typically long entity mentions and need a lot of context information: Offset conjunction or white space locations are important next to prefixes and suffixes. Static morphological features like regular expressions nearly have no meaning—in contrast to gene/protein names where those are of utmost importance: Many of them do not follow a special structure, but especially the frequent use of acronyms leads to the possibility to learn dependencies between the specific morphology of tokens with offset conjunction. In the case of SNPs (Chapter 6) with multiple classes of interest to detect, offset conjunction is very important, too. Interestingly, static morphological features do not show such impact, probably because of the incorporation of more specific features, particularly for frequent mentions. Dictionaries of frequent entities are beneficial here, while they could not be proven to be that important for the other two case studies.

Main effort in these studies was in the context of feature design—next to the necessary corpus generation and domain analysis. Development and analysis of feature subset selection methods specialized for the sequential analysis of text (in Chapter 7) allow to use all features implemented for newly arising entity classes of interest. Without such automated fast methods to select the informative attributes, their exhaustive number would make manual selection necessary. Additionally, the proposed methods allow the application of the named entity recognizer with nearly half of the needed time for inference.

Next to parameters on sequentially structured CRFs, previous publications have shown

the value of incorporation of distant information with other structures of like skip-chains. In compliance with the idea of automatically selecting features, this thesis proposes the approach of automatically searching for meaningful skip-chains (in Chapter 9). This approach shows to be useful and to have a positive impact on the performance of a model. To speed up the search, two straight-forward filtering methods are implemented, unfortunately tested without showing a proper selection of skip-chain candidates. Nevertheless, the general idea of an automated structure selection is shown to be meaningful.

While these advancements are in the simplification of the development process of named entity recognition models, a generalizable application of a model needs to be ensured additionally. Depending on the use-case, a model specialized on a high recall or high precision may be needed (while other objectives can occur). Retraining a model is not practicable for each application, computing confidence scores increases runtime. The approach proposed in Chapter 8 allows the preparation of a set of solutions the user can choose from—without retraining and with better results than a previously published method addressing the same problem. Notable is that the incorporated evolutionary training procedure would not have been possible without the feature selection methods presented in Chapter 7 due to a too high number of parameters.

Altogether, the adaptations to novel domains show the feasibility of machine learning-based NER and highlights common pitfalls in designing such systems. The implemented applications are included in the Fraunhofer SCAI in-house-developed search engine SCAIView and practically help biologists and chemists in industry and academia. Furthermore, the novel enhancements to the fundamental methodology simplify the design and training of models.

Several works in the Department of Bioinformatics at Fraunhofer SCAI showed the value of the workflow incorporating the enhancements presented in this thesis to adapt models it to new classes (Kolářik, Klinger, and Hofmann-Apitius (2009); Klein (2010); Gurulingappa, Hofmann-Apitius, and Fluck (2010); unpublished work in context of Gurulingappa, Klinger, et al. (2010)).

Chapter 11

Future Work

The main contributions and innovations in this thesis are discussed in Chapter 10. From these topics, some open questions are raised and should be addressed in future work, summarized briefly in the following.

In Chapter 4, a workflow to detect chemical names, especially IUPAC-like names have been presented. The normalization results using dictionary mapping are convincing on MEDLINE data, but only known compounds can occur in the dictionary. For application on *e. g.* patent data, the approach by OSCAR3 and other tool sets which try to convert the name itself grammar-based to a structure needs to be followed further: Newly introduced chemical compounds cannot be normalized in a different way.

Chapter 5 shows how gene/protein names can be detected in text and how multiple annotations can be combined. Additionally, the most often claimed advantage of machine learning-based NER has been analyzed in this scenario: The stability over time with less effort of re-training and re-annotation than keeping a dictionary up-to-date. It was shown that the generalizability over time is limited. An important research topic is how a model can be made more stable. This may be seen as a special case of domain adaptation. In first experiments, a removal of differently distributed features in corpora from different years could not improve the results. Therefore, more elaborated methods need to be tested here.

Chapter 6 presents a workflow to detect SNP mentions. The challenges in normalizing those has been discussed. This is a very important research topic including different steps. In the approach presented in this thesis, one of the first questions which needs to be answered is how the different entities (STATE, LOCATION, TYPE, GENE) can be combined—an error analysis of the results of the system applied to MEDLINE shows limitations here. A possible solution can be a joined model finding named entities as well as their relation to one logical entity in one step, *e. g.* addressed with Imperatively Defined Factor Graphs (McCallum, Schultz, and Singh, 2009).

Different approaches, mainly fast filtering methods for feature selection for the problem of NER has been presented in Chapter 7. While the proposed method improves speed and reduces complexity, the iterative (slow) method IFP to select meaningful features is still better than the filtering methods—not a problem at inference time though. But the difference between IFP and filtering shows space for improvement of the filtering, which should be explored. An idea could be the combination of IFP with filtering or incorporating regularization methods additionally.

The method proposed in Chapter 8 allows a user to select between precision and recall at inference time, without retraining or time consuming confidence computation. While

this method works well, the evolutionary optimization is still slower than the L-BFGS-based training. The latter is currently used to get close to a good result followed by the evolutionary training—these two steps may be combined in an intelligent way to reduce training to only one method to be applied. An interesting additional question is to analyze which features are mainly responsible for high recall or high precision; a comparison of the different solutions allows that.

In Chapter 9 it is proven that searching for skip chains similarly to the use of huge feature sets can improve the performance of the NER system, measured in F_1 . Nevertheless, the proposed method lacks the presentation of well-working filter methods to make the structure optimization feasible without high computation costs. At the moment, the used best-first-search on a huge set of possible templates is not applicable practically. Selecting potential candidates and therefore limiting the set of templates would make the method relevant for applications.

This thesis dealt with simplifying the process to build models for the detection of names from new classes of interest, an important next task is to simplify the process to build models extracting relations between entities of the same or different classes. The road to go may be similar to this thesis: Analyzing the different parameter sets needed for different relational models, *e. g.* in a structured learning setting, followed by simplifications in the process of finding them.

Appendix A

Acknowledgements

I would like to thank those who made this thesis possible, first of all my supervisor Prof. Dr. Günter Rudolph at the University of Technology Dortmund, who motivated me to start with the Ph.D. and has always been open for all kinds of discussions. I want to thank Prof. Dr. Martin Hofmann-Apitius for giving me the chance to work in his Bioinformatics department at the Fraunhofer Institute for Algorithms and Scientific Computing (SCAI). Dr. Christoph M. Friedrich supervised the main part of my work at SCAI, he deserves my thanks for always being critical and for long and interesting discussions. It was a pleasure to share my office with him. I thank Dr. Juliane Fluck for keeping an eye on the applicability of my work and for fruitful and inspiring discussions. Thanks for motivating me in all kinds of difficult situations. Discussion with Dr. Katrin Tomanek from the University of Jena helped me a lot getting started as well as keeping a realistic self-understanding what a Ph.D. is about. Thanks to the whole Bioinformatics group at SCAI for the nice working environment which made the time there more fun than exhausting work—especially thanks to Corinna Klein and Antje Wolf.

Special thanks to Prof. Andrew McCallum for the stay in his group at the Computer Science Department at the University of Massachusetts Amherst. I had a great time there learning a lot, especially due to the great supervision by Sebastian Riedel and the excellent discussions with all group members of the Information Extraction and Synthesis Laboratory.

The different topics and areas in this Ph.D. and several conference trips were enabled by different funds and grants: These are the *Max-Planck/Fraunhofer Machine Learning collaboration project*¹ and *Bonn-Aachen Center for Information Technology*². Some work was performed in the framework of European integrated projects, namely *@neurIST*³ and *+Spaces*⁴. Additionally thanks appertain to *German Academic Exchange Service*⁵ (DAAD) for a Ph.D. student scholarship and the *U.S. Department of Energy*⁶ (DOE).

Special thanks to my family for believing in me and, of course, to Wiebke who often suffered from my up and downs in working for this thesis. Thanks for never complaining but always supporting me.

¹ <http://lip.fml.tuebingen.mpg.de/>

² <http://www.b-it-center.de/>

³ contract no. IST-027703, <http://www.aneurist.org/>

⁴ contract no. 248726, <http://www.positivspaces.eu/>

⁵ <http://www.daad.de/>

⁶ Grant Number DE-FG02-06ER64270, <http://www.energy.gov/>

Appendix B

Publications integrated in this Thesis

This section presents the publications of the author of this thesis which are used completely or partly in this thesis. Declarations of the contributions to publications with multiple authors are given.

The paper published in the BioCreative Workshop proceedings (Klinger, Friedrich, et al., 2007) and the subsequent journal paper (Smith, Tanabe, et al., 2008) are the basis for Chapter 5 and to a lower degree to Chapter 2. Main author's contribution was the implementation of the workflow for training and testing via bootstrapping and its application to search through space of features and feature groups for optimization of the configuration. Another important point was the development of the combination strategy of CRF trained on different annotations and the following combination. Christoph M. Friedrich supervised the work as a data mining expert and Juliane Fluck as an expert in biomedical text mining. Christoph M. Friedrich additionally developed the latent semantic analysis-based acronym disambiguation. Martin Hofmann-Apitius critically revised the manuscript and motivated participation in the BioCreative competition.

The publication "Identifying Gene Specific Variations in Biomedical Text" (Klinger, Furlong, et al., 2007) deals with the application of CRFs to detect variation mentions in text. It is the basis for Chapter 6. Main author's contribution is the optimization to the incorporated entity classes and the evaluations as well as the test corpus annotation together with Christoph M. Friedrich. Normalization was implemented and developed by Laura I. Furlong. She annotated the training data for the CRF. The rs number evaluation was developed by Christoph M. Friedrich together with Heinz Theo Mevissen. Juliane Fluck and Heinz Theo Mevissen performed the dictionary-based recognition of genes and proteins used in this work. Martin Hofmann-Apitius critically revised the manuscript.

The manuscript "Challenges in the Association of Human Variation Names with Unique Database Identifiers" (Thomas, Klinger, et al., 2011) is based on a Master's thesis the author of this thesis co-supervised. It is the basis for the discussions of normalization challenges in Chapter 6. Philippe Thomas developed these explanations in discussion with the author of this thesis as well as the other co-authors of the manuscript.

The tutorial "Classical Probabilistic Models and Conditional Random Fields" (Klinger and Tomanek, 2007) is the fundament for Chapter 3. Main contribution of the first author in this thesis is Section 3.4 describing CRFs as well as the introduction (now in Section 3.1). The Sections 3.2 and 3.3 were mainly written by Katrin Tomanek. Both authors discussed, planned and revised the full manuscript together.

The paper "Detection of IUPAC and IUPAC-like Chemical Names" (Klinger, Kolářik, et al.,

2008) is integrated in Chapter 4. Main contribution of the first author is the definition of the entity classes in discussion with Corinna Kolářik and the optimization of the configuration of the CRF. This includes the development of specialized features as well as the application of higher order models. Juliane Fluck contributed to the introduction, Christoph M. Friedrich supervised the work.

The workshop contribution “Chemical Names: Terminological Resources and Corpora Annotation” (Kolářik, Klinger, et al., 2008) is closely related to that work as it addresses the problem of chemical named entity detection in general. The approach is not based on machine learning but on the dictionary and rule-based system ProMiner (Hanisch, Fundel, et al., 2005). This paper is not integrated as a whole into this thesis. Aspects occur in Chapter 4, namely Section 4.2. Main contribution of the author of this thesis is in the analysis of the dictionaries assembled by Corinna Kolářik and the application of the IUPAC system onto the presented test corpus which was annotated by Juliane Fluck and another annotator. The author of this thesis computed evaluations of the dictionaries on the corpus as well as the inter-annotator agreement.

The three applications described above (gene and protein names as described in “Named Entity Recognition with Combinations of Conditional Random Fields”, “Identifying Gene Specific Variations in Biomedical Text” and “Detection of IUPAC and IUPAC-like Chemical Names”) contributed to the overview article “Knowledge Environments Representing Molecular Entities for the Virtual Physiological Human” (Hofmann-Apitius, Fluck, et al., 2008). The aspects described in the former publications were written by the author. This article is not integrated into this thesis.

The paper “Feature Subset Selection in Conditional Random Fields for Named Entity Recognition” (Klinger and Friedrich, 2009a) is the basis for Chapter 7. All work was done by the main author. Christoph M. Friedrich revised the manuscript.

All work in the publication “User’s Choice of Precision and Recall in Named Entity Recognition” (Klinger and Friedrich, 2009b) was done by the first author.

Bibliography

- ACDLabs (2007). *ACDName*. Software. http://www.acdlabs.com/products/name_lab/name/ – last accessed: 2007-12-18. See p. 66.
- Ad Hoc Committee on Mutation Nomenclature (1996). “Update on nomenclature for human gene mutations”. In: *Human Mutation* 8.3, pp. 197–202. See p. 95.
- Adobe Systems Incorporated (2007). *PDF Reference*. online. http://www.adobe.com/devnet/acrobat/pdfs/pdf_reference.pdf. See p. 15.
- Agarwal, S. and Yu, H. (2009). “Automatically classifying sentences in full-text biomedical articles into Introduction, Methods, Results and Discussion.” In: *Bioinformatics* 25.23, pp. 3174–3180. See p. 16.
- Ando, R. K. (2006). “BioCreative II Gene Mention Tagging System at IBM Watson”. In: *Proceedings of the Second BioCreative Challenge Evaluation Workshop*, pp. 101–103. See pp. 20, 78.
- Androutsopoulos, I., Koutsias, J., Chandrinou, K. V., Paliouras, G., and Spyropoulos, C. D. (2000). “An evaluation of Naive Bayesian anti-spam filtering”. In: *Proceedings of the workshop on Machine Learning in the New Information Age, 11th European Conference on Machine Learning*. Ed. by G. Potamias, V. Moustakis, and M. van Someren. Barcelona, Spain, pp. 9–17. See p. 34.
- Androutsopoulos, I., Paliouras, G., Karkaletsis, V., Sakkis, G., Spyropoulos, C. D., and Stamatopoulos, P. (2000). “Learning to Filter Spam E-Mail: A Comparison of a Naive Bayesian and a Memory-Based Approach”. In: *Proceedings of the workshop "Machine Learning and Textual Information Access", 4th Conference on Principles and Practice of Knowledge Discovery in Databases*. Ed. by H. Zaragoza, P. Gallinari, and M. Rajman. Lyon, France, pp. 1–13. See p. 34.
- Anstein, S., Kremer, G., and Reyle, U. (2006). “Identifying and Classifying Terms in the Life Sciences: The Case of Chemical Terminology”. In: *Proceedings of the Fifth Language Resources and Evaluation Conference*. Ed. by N. Calzolari, K. Choukri, A. Gangemi, B. Maegaard, J. Mariani, J. Odijk, and D. Tapias. Genoa, Italy, pp. 1095–1098. See p. 60.
- Antonarakis, S. (1998). “Recommendations for a nomenclature system for human gene mutations. Nomenclature Working Group.” In: *Human Mutation* 11.1, pp. 1–3. See p. 95.
- Azov, A. (2005). “Stories Behind Gene Names”. In: *Caduceus Spring*, pp. 1–4. See p. 77.

- Bäck, T., Fogel, D., and Michalewicz, Z., eds. (1997). *Handbook of Evolutionary Computation*. Bristol, UK: Institute of Physics Publishing and Oxford University Press. See pp. 123, 124.
- Baeza-Yates, R. and Ribeiro-Neto, B. (1999). *Modern Information Retrieval*. New York, USA: ACM Press, Addison Wesley. See p. 3.
- Bañeres, B., Cesareni, G., Hirschman, L., Krallinger, M., Leitner, F., and Valencia, A., eds. (2009). *Proceedings of the BioCreative II.5 Workshop*. Madrid, Spain: CNIO. See pp. 6, 8.
- Baroni, M., Evert, S., Kilgarriff, A., and Sharoff, S., eds. (2007). *Proceedings of the 3rd Web as Corpus Workshop (WAC3)*. Association for Computational Linguistics, Special Interest Group on Web as Corpus. Belgium. See p. 15.
- Beaudet, A. L. and Tsui, L. C. (1993). “A suggested nomenclature for designating mutations.” In: *Human Mutation* 2.4, pp. 245–248. See p. 95.
- Beierle, C. and Kern-Isberner, G. (2003). *Methoden wissensbasierter Systeme*. 2nd. in german. Wiesbaden, Germany: Vieweg. See p. 39.
- Berger, A. L., Pietra, S. D., and Pietra, V. J. D. (1996). “A Maximum Entropy Approach to Natural Language Processing”. In: *Computational Linguistics* 22.1, pp. 39–71. See pp. 35, 39.
- Berry, M. W., ed. (2004). *Survey of Text Mining*. Springer. See p. 4.
- Beutler, E., McKusick, V. A., Motulsky, A. G., Sriver, C. R., and Hutchinson, F. (1996). “Mutation nomenclature: nicknames, systematic names, and unique identifiers.” In: *Human Mutation* 8.3, pp. 203–206. See p. 95.
- Bishop, C. M. (1995). *Neural Networks for Pattern Recognition*. Clarendon Press. See p. 105.
- (2006). *Pattern Recognition and Machine Learning*. Springer. See pp. 33, 35, 39, 51, 52, 56.
- Bjorne, J., Heimonen, J., Ginter, F., Airola, A., Pahikkala, T., and Salakoski, T. (2009). “Extracting complex biological events with rich graph-based feature sets”. In: *Proceedings of the Workshop on BioNLP: Shared Task*. Association for Computational Linguistics. Boulder, Colorado, pp. 10–18. See p. 6.
- Blei, D. M. and Lafferty, J. D. (2009). “Topic Models”. In: ed. by A. N. Srivastava and M. Sahami. Chapman & Hall/CRC. Chap. 4, pp. 71–94. See p. 3.
- Blei, D. M., Ng, A. Y., Jordan, M. I., and Lafferty, J. (2003). “Latent Dirichlet allocation”. In: *Journal of Machine Learning Research* 3, 993–1022. See p. 3.
- Bonis, J., Furlong, L. I., and Sanz, F. (2006). “OSIRIS: a tool for retrieving literature about sequence variants”. In: *Bioinformatics* (July 2006), pp. 2567–2569. See pp. 91, 101.

- Borgelt, C. and Kruse, R. (2002). *Graphical Models: Methods for Data Analysis and Mining*. Wiley & Sons. See p. 56.
- Borgman, C. L. and Siegfried, S. L. (1992). "Getty's synonyme and its cousins: A survey of applications of personal name-matching algorithms." In: *Journal of the American Society for Information Science* 43.7, 459–476. See p. 23.
- Breiman, L., Friedman, J., Stone, C. J., and Olshon, R. A. (1984). *Classification and Regression Trees*. Chapman & Hall/CRC. See p. 105.
- Bromberg, F., Margaritis, D., and Honavar, V. (2009). "Efficient Markov Network Structure Discovery using Independence Tests". In: *Journal of Artificial Intelligence Research* 35.1, pp. 449–484. See p. 135.
- Buyko, E., Tomanek, K., and Hahn, U. (2007). "Resolution of Coordination Ellipses in Named Entities in Scientific Texts". In: *Proceedings of the 10th Conference of the Pacific Association for Computational Linguistics (PACLING 2007)*. Melbourne, Australia, pp. 163–171. See p. 32.
- CambridgeSoft (2007). *Name=Struct*. Software. <http://www.cambridgesoft.com/databases/details/?db=16> – last accessed 2009-08-13. See p. 66.
- Caporaso, J. G., Baumgartner, W. A., Randolph, D. A., Cohen, K. B., and Hunter, L. (2007). "MutationFinder: a high-performance system for extracting point mutation mentions from text." In: *Bioinformatics* 23.14, pp. 1862–1865. See pp. 5, 90.
- Carletta, J. (1996). "Assessing Agreement on Classification Tasks: The Kappa Statistic". In: *Computational Linguistics* 22, pp. 249–254. See p. 17.
- Carpenter, B. (2007). "LingPipe for 99.99% Recall of Gene Mentions." In: *Proceedings of the 2nd BioCreative workshop*. Madrid, Spain. See p. 121.
- ChemAxon (2007). *Marvin*. Software. <http://www.chemaxon.com/marvin/> – last accessed 2008-01-11. See p. 73.
- Chen, S. F. and Rosenfeld, R. (2000). "A Survey of Smoothing Techniques for ME Models". In: *IEEE Transactions on Speech and Audio Processing* 8.1, pp. 37–50. See p. 49.
- Ciravegna, F. and Petrelli, D. (2001). "User involvement in customizing adaptive Information Extraction: position paper". In: *Proceedings of the IJCAI-2001 Workshop on Adaptive Text Extraction and Mining*. Seattle, US. See p. 121.
- Clark, A., Fox, C., and Lappin, S., eds. (2010). *The Handbook of Computational Linguistics and Natural Language Processing*. Hoboken, NJ, USA: Wiley-Blackwell. See p. 4.

- Cohen, A. M. (2005). “Unsupervised gene/protein named entity normalization using automatically extracted dictionaries”. In: *ACL-ISMB Workshop on Linking Biological Literature, Ontologies and Databases*, pp. 17–24. See p. 24.
- Cohen, K. B. (2008). *Mutation and Semantic representations in natural language*. Talk at Workshop on Annotation, Interpretation and Management of Mutations (AIMM) at European Conference on Computational Biology (ECCB). http://www.ebi.ac.uk/Rebholz-srv/aimm/presentations/K.Cohen_Keynote_Aimm_22Nov2008.pdf, last accessed 2010-09-21. See p. 89.
- Cohen, K. (2009). *Tutorial on Biomedical Natural Language Processing: BioNLP*. Talk at Conference on Recent Advances in Natural Language Processing. See p. 77.
- Cohn, D. A., Ghahramani, Z., and Jordan, M. I. (1996). “Active Learning with Statistical Models”. In: *Journal of Artificial Intelligence Research* 4, pp. 129–145. See p. 15.
- Cohn, D., Atlas, L., and Ladner, R. (1994). “Improving Generalization with Active Learning”. In: *Mach. Learn.* 15.2, pp. 201–221. ISSN: 0885-6125. See p. 15.
- Corbett, P. (2007). *Oscar3*. Software. Online: <http://oscar3-chem.sourceforge.net>, last accessed 2007-12-13. See p. 61.
- Corbett, P., Batchelor, C., and Teufel, S. (2007). “Annotation of Chemical Named Entities”. In: *BioNLP 2007: Biological, translational, and clinical language processing*. Prague, pp. 57–64. See pp. 61, 65.
- Corbett, P. and Murray-Rust, P. (2006). “High-Throughput Identification of Chemistry in Life Science Texts”. In: *2nd International Symposium on Computational Life Science (CompLife)*, pp. 107–118. See p. 61.
- Cowie, J. and Wilks, Y. (1996). *Information Extraction*. Online – <http://www.dcs.shef.ac.uk/~yorick/papers/infoext.pdf> – last accessed 07/22/2010. See p. 3.
- Culotta, A. and McCallum, A. (2004). “Confidence Estimation for Information Extraction”. In: *Proceedings of Human Language Technology and North American Association of Computational Linguistics Conference (HLT-NAACL)*, pp. 109–112. See p. 121.
- Cun, Y. L., Denker, J. S., and Solla, S. A. (1990). “Optimal Brain Damage”. In: *Advances in Neural Information Processing Systems*, pp. 598–605. See p. 107.
- Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002). “A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II”. In: *IEEE Transactions on Evolutionary Computation* 6.2, pp. 182–197. See p. 123.
- DeCaprio, D., Vinson, J. P., Pearson, M. D., Montgomery, P., Doherty, M., and Galagan, J. E. (2007). “Conrad: Gene Prediction using Conditional Random Fields”. In: *Genome Research* 17.9, pp. 1389–1396. See p. 32.

- Doms, A. and Schroeder, M. (2005). "GoPubMed: exploring PubMed with the Gene Ontology." In: *Nucleic Acids Research* 33.Web Server issue, W783–W786. See p. 6.
- Dowell, K. (2009). *ProMiner at MGI*. Talk at Fraunhofer Text Mining Symposium. available online <http://www.scai.fraunhofer.de/en/events/archiv/2009/tms09.html> – last accessed 07/26/2010. See p. 6.
- Dunnen, J. T. den and Antonarakis, S. E. (2000). "Mutation nomenclature extensions and suggestions to describe complex mutations: a discussion." In: *Human Mutation* 15.1, pp. 7–12. See p. 95.
- (2001). "Nomenclature for the description of human sequence variations." In: *Human Genetics* 109.1, pp. 121–124. See pp. 90, 95.
- Efron, B. and Tibshirani, R. J. (1993). *An Introduction to the Bootstrap*. Chapman & Hall/CRC. See pp. 26, 63.
- Eigner-Pitto, V., Eiblmaier, J., Kraut, H., Isenko, L., Saller, H., and Loew, P. (2007). *Mining, Storage, Retrieval: the Challenge of Integrating Chemoinformatics with Chemical Structure Recognition in Text and Images*. Talk on 5th Fraunhofer Symposium on Text Mining in the Life Sciences. <http://www.scai.fhg.de/tms07.html> – last accessed: 2007-12-18. See p. 66.
- Eller, G. A. (2006). "Improving the Quality of Published Chemical Names with Nomenclature Software". In: *Molecules* 11, pp. 915–928. See p. 59.
- Ellie, E., Camou, F., Vital, A., Rummens, C., Gâteau, G., Delpech, M., and Valleix, S. (2001). "Recurrent subarachnoid hemorrhage associated with a new transthyretin variant (Gly53Glu)." In: *Neurology* 57.1, pp. 135–137. See p. 99.
- Engelson, S. P. and Dagan, I. (1996). "Minimizing Manual Annotation Cost in Supervised Training from Corpora". In: *In Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*. Springer, pp. 319–326. See p. 15.
- Feldman, R. and Sanger, J. (2007). *The Text Mining Handbook: Advanced Approaches to Analyzing Unstructured Data*. Cambridge, England: Cambridge University Press. See p. 23.
- Finkel, J. R., Grenager, T., and Manning, C. (2005). "Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling". In: *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 363–370. See p. 53.
- Finkel, J. R. and Manning, C. D. (2009). "Joint Parsing and Named Entity Recognition". In: *Proceedings of Human Language Technology Conference and North American Chapter of the Association for Computational Linguistics (HLT-NAACL)*. See p. 23.

- Florian, R., Jing, A. I. and Hongyan, and Zhang, T. (2003). "Named Entity Recognition through Classifier Combination". In: *Proceedings of the 7th Conference on Natural Language Learning at HLT-NAACL 2003*. See p. 78.
- Fluck, J., Mevissen, H. T., Dach, H., Oster, M., and Hofmann-Apitius, M. (2007). "ProMiner: Recognition of Human Gene and Protein Names using regularly updated Dictionaries". In: *Proceedings of the Second BioCreative Challenge Evaluation Workshop*. Ed. by L. Hirschman, M. Krallinger, and A. Valencia. Centro Nacional de Investigaciones Oncologicas. CNIO, pp. 149–151. See p. 24.
- Fredj, R. B., Gross, E., Chouchen, L., B'Chir, F., Ahmed, S. B., Neubauer, S., Kiechle, M., and Saguem, S. (2007). "Mutational spectrum of dihydropyrimidine dehydrogenase gene (DPYD) in the Tunisian population." In: *Comptes rendus biologies* 330.10, pp. 764–769. See p. 19.
- Friedrich, C. M., Revillion, T., Hofmann, M., and Fluck, J. (2006). "Biomedical and Chemical Named Entity Recognition with Conditional Random Fields: The Advantage of Dictionary Features". In: *Proceedings of the Second International Symposium on Semantic Mining in Biomedicine (SMBM 2006)*. Ed. by S. Ananiadou and J. Fluck, pp. 85–89. See pp. 20, 64, 67.
- Frigui, H. and Nasraoui, O. (2004). "Simultaneous Clustering and Dynamic Keyword Weighting for Text Documents". In: ed. by M. W. Berry. Springer. Chap. 3, pp. 45–72. See p. 3.
- Galley, M. (2006). "A Skip-Chain Conditional Random Field for Ranking Meeting Utterances by Importance". In: *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*. Sydney, Australia: Association for Computational Linguistics, pp. 364–372. See p. 133.
- Goodman, J. (2004). "Exponential Priors for Maximum Entropy Models". In: *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL)*, pp. 305–312. See p. 107.
- Gupta, K. K., Nath, B., and Ramamohanarao, K. (2007). "Conditional Random Fields for Intrusion Detection". In: *Proceedings of the 21st International Conference on Advanced Information Networking and Applications Workshops*. Washington, DC, USA: IEEE Computer Society, pp. 203–208. See p. 32.
- Gurulingappa, H., Hofmann-Apitius, M., and Fluck, J. (2010). *Concept Identification and Assertion Classification in Patient Health Records*. in preparation. See p. 150.
- Gurulingappa, H., Klinger, R., Hofmann-Apitius, M., and Fluck, J. (2010). "An Empirical Evaluation of Resources for the Identification of Diseases and Adverse Effects in Biomedical Literature". In: *2nd Workshop on Building and evaluating resources for biomedical text*

- mining (7th edition of the Language Resources and Evaluation Conference). Valetta, Malta. See p. 150.
- Gurulingappa, H., Müller, B., Klinger, R., Mevissen, H.-T., Hofmann-Apitius, M., Fluck, J., and Friedrich, C. M. (2009). "Patent Retrieval in Chemistry based on semantically tagged Named Entities". In: *The Eighteenth Text REtrieval Conference (TREC 2009) Proceedings*. Ed. by E. M. Voorhees and L. P. Buckland. Gaithersburg, Maryland, USA. See p. 6.
- Guyon, I. and Elisseeff, A. (2003). "An Introduction to Variable and Feature Selection". In: *Journal of Machine Learning Research* 3, pp. 1157–1182. See p. 105.
- Guzikowski, A. P., Whittemore, E. R., Woodward, R. M., Weber, E., and Keana, J. F. W. (1997). "Synthesis of Racemic 6,7,8,9-Tetrahydro-3-hydroxy-1H-1-benzazepine-2,5-diones as Antagonists of N-Methyl-D-aspartate (NMDA) and α -Amino-3-hydroxy-5-methylisoxazole-4-propionic Acid (AMPA) Receptors". In: *Journal of Medicinal Chemistry* 40.15, pp. 2424–2429. See p. 64.
- Hand, D. J. and Yu, K. (2001). "Idiot's Bayes – not so stupid after all?" In: *International Statistical Review* 69.3, pp. 385–399. See p. 33.
- Hanisch, D., Fluck, J., Mevissen, H.-T., and Zimmer, R. (2003). "Playing Biology's Name Game: Identifying Protein Names in Scientific Text". In: *Pacific Symposium on Biocomputing*. Vol. 8, pp. 403–414. See p. 95.
- Hanisch, D., Fundel, K., Mevissen, H.-T., Zimmer, R., and Fluck, J. (2004). "ProMiner: Organism-specific protein name detection using approximate string matching". In: *Proceedings of the BioCreative Challenge Evaluation Workshop 2004*. See pp. 24, 27.
- Hanisch, D., Fundel, K., Mevissen, H.-T., Zimmer, R., and Fluck, J. (2005). "ProMiner: rule-based protein and gene entity recognition". In: *BMC Bioinformatics* 6 (Suppl 1).S14. See pp. 5, 7, 24, 60, 85, 91, 156.
- Hansen, N. (2006). "The CMA evolution strategy: a comparing review". In: *Towards a new evolutionary computation. Advances on estimation of distribution algorithms*. Ed. by J. Lozano, P. Larranaga, I. Inza, and E. Bengoetxea. Springer, pp. 75–102. See p. 124.
- Hasan, A. and Srivastava, P. C. (1992). "Synthesis and biological studies of unsaturated acyclonucleoside analogues of S-adenosyl-L-homocysteine hydrolase inhibitors." In: *J Med Chem* 35.8, pp. 1435–1439. See p. 137.
- He, X., Zemel, R. S., and Carreira-Perpinan, M. A. (2004). "Multiscale conditional random fields for image labeling". In: *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Vol. 2, pp. 695–702. See p. 32.
- Hettne, K. M., Stierum, R. H., Schuemie, M. J., Hendriksen, P. J. M., Schijvenaars, B. J. A., Mulligen, E. M. van, Kleinjans, J., and Kors, J. A. (2009). "A dictionary to identify small

- molecules and drugs in free text.” In: *Bioinformatics* 25.22, pp. 2983–2991. See pp. 5, 60, 76.
- Hettne, K. M., Williams, A. J., Mulligen, E. M. van, Kleinjans, J., Tkachenko, V., and Kors, J. A. (2010). “Automatic vs. manual curation of a multi-source chemical dictionary: the impact on text mining.” In: *Journal of Cheminformatics* 2.1, p. 4. See p. 5.
- Hirschman, L., Krallinger, M., and Valencia, A., eds. (2007). *Proceedings of the Second BioCreative Challenge Evaluation Workshop*. Centro Nacional de Investigaciones Oncologicas. CNIO. See pp. 6, 8, 64, 99.
- Hirschman, L., Yeh, A., Blaschke, C., and Valencia, A. (2005). “Overview of BioCreAtIvE: critical assessment of information extraction for biology.” In: *BMC Bioinformatics* 6 Suppl 1, S1. See pp. 6, 8, 77.
- Hofmann-Apitius, M., Fluck, J., Furlong, L., Fornes, O., Kolářik, C., Hanser, S., Boecker, M., Schultz, S., Sanz, F., Klinger, R., Mevissen, T., Gatterneyer, T., Oliva, B., and Friedrich, C. (2008). “Knowledge Environments Representing Molecular Entities for the Virtual Physiological Human”. In: *Philosophical Transactions of the Royal Society A* 366.1878, pp. 3091–3110. See pp. 6, 27, 156.
- Horn, F., Lau, A. L., and Cohen, F. E. (2004). “Automated extraction of mutation data from the literature: application of MuteXt to G protein-coupled receptors and nuclear hormone receptors.” In: *Bioinformatics* 20.4, pp. 557–568. See pp. 90, 101.
- Hsu, C.-N., Chang, Y.-M., Kuo, C.-J., Lin, Y.-S., Huang, H.-S., and Chung, I.-F. (2008). “Integrating high dimensional bi-directional parsing models for gene mention tagging”. In: *Bioinformatics* 24.13, pp. i286–i294. See pp. 78, 105.
- Huang, H.-S., Lin, Y.-S., lin, K.-T., Kuo, C.-J., Chang, Y.-M., Yang, B.-H., Chung, I.-F., and Hsu, C.-N. (2006). “High-Recall Gene Mention Recognition by Unification of Multiple Backward Parsing Models”. In: *Proceedings of the Second BioCreative Challenge Evaluation Workshop*, pp. 109–111. See p. 78.
- Humana, I. (2005). *Top 50 Drugs Brand-Name Prescribed*. Online. http://apps.humana.com/prescription_benefits_and_services/includes/Top50BrandDrugs.pdf – last accessed 2007-12-14. See p. 60.
- International HapMap Consortium (2005). “A haplotype map of the human genome”. In: *Nature* 437.7063, pp. 1241–2. See p. 89.
- Jansche, M. (2005). “Maximum Expected F-Measure Training of Logistic Regression Models”. In: *Proceedings of Human Language Technology Conference Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP)*. Vancouver. See p. 121.

- Jaynes, E. T. (1957). "Information Theory and Statistical Mechanics". In: *Physical Review* 106.4, pp. 620–630. See p. 35.
- Joachims, T. (2002). *Learning to Classify Text using Support Vector Machines*. Kluwer/Springer. See pp. 3, 20.
- (2005). "A Support Vector Method for Multivariate Performance Measures". In: *Proceedings of the International Conference on Machine Learning (ICML)*. Bonn, pp. 377–384. See p. 121.
- Jordan, M. I. and Weiss, Y. (2002). "Graphical models: Probabilistic inference". In: *The Handbook of Brain Theory and Neural Networks*. 2nd. Cambridge, MA, USA: MIT Press. See p. 56.
- Kemp, N. and Lynch, M. (1998). "The extraction of information from the text of chemical patents. 1. Identification of specific chemical names". In: *Journal of Chemical Information and Computer Sciences* 38.4, pp. 544–551. See p. 60.
- Kim, J.-D., Ohta, T., Pyysalo, S., Kano, Y., and Tsujii, J. (2009). "Overview of BioNLP-09 Shared Task on Event Extraction". In: *Proceedings of the Workshop on BioNLP: Shared Task*. Association for Computational Linguistics. Boulder, Colorado, pp. 41–49. See p. 6.
- Kiritchenko, S. and Matwin, S. (2001). "Email classification with co-training". In: *Proceedings of the conference of the Centre for Advances Studies on Collaborative research*. Ed. by D. A. Stewart and J. H. Johnson. Toronto, Ontario, Canada, pp. 192–201. See p. 34.
- Klein, C. (2010). "Information Extraction from Text for Improving Research on Small Molecules and Histone Modifications". PhD thesis. Bonn: Mathematisch Naturwissenschaftliche Fakultät der Rheinischen Friedrich-Wilhelms-Universität Bonn. See pp. 8, 61, 76, 150.
- Klinger, R. and Friedrich, C. M. (2009a). "Feature Subset Selection in Conditional Random Fields for Named Entity Recognition". In: *Proceedings of Recent Advances in Natural Language Processing (RANLP)*. Ed. by G. Angelova, K. Bontcheva, R. Mitkov, N. Nicolov, and N. Nikolov. Borovets, Bulgaria, pp. 185–191. See pp. 105, 156.
- (2009b). "User's Choice of Precision and Recall in Named Entity Recognition". In: *Proceedings of Recent Advances in Natural Language Processing (RANLP)*. Ed. by G. Angelova, K. Bontcheva, R. Mitkov, N. Nicolov, and N. Nikolov. Borovets, Bulgaria, pp. 192–196. See pp. 121, 156.
- Klinger, R., Friedrich, C. M., Fluck, J., and Hofmann-Apitius, M. (2007). "Named Entity Recognition with Combinations of Conditional Random Fields". In: *Proceedings of the Second BioCreative Challenge Evaluation Workshop*. Madrid, Spain, pp. 89–91. See pp. 20, 76, 77, 155.

- Klinger, R., Furlong, L. I., Friedrich, C. M., Mevissen, H. T., Fluck, J., Sanz, F., and Hofmann-Apitius, M. (2007). "Identifying Gene Specific Variations in Biomedical Text". In: *Journal of Bioinformatics and Computational Biology* 5.6. PMID 18172929, pp. 1277–1296. See pp. 89, 155.
- Klinger, R., Kolářik, C., Fluck, J., Hofmann-Apitius, M., and Friedrich, C. M. (2008). "Detection of IUPAC and IUPAC-like Chemical Names". In: *Bioinformatics* 24.13. Proceedings of the International Conference Intelligent Systems for Molecular Biology (ISMB)., pp. i268–i276. See pp. 20, 59, 155.
- Klinger, R., Pesch, R., Mevisen, H. T., and Fluck, J. (2009a). *Processing Full Text Publications in Portable Document Format*. Poster on BioCreative II.5 Workshop. See pp. 15, 26, 27.
- Klinger, R., Pesch, R., Mevissen, T., and Fluck, J. (2009b). *Text Mining in Full Text Articles – Methodological and Representation Issues*. Poster on 3rd International Biocuration Conference – Online in Nature Precedings <http://dx.doi.org/10.1038/npre.2009.3141.1>. See pp. 26, 27.
- Klinger, R. and Tomanek, K. (2007). *Classical Probabilistic Models and Conditional Random Fields*. Tech. rep. TR07-2-013. ISSN 1864-4503. Department of Computer Science, Dortmund University of Technology. URL: http://www.scai.fraunhofer.de/fileadmin/images/bio/data_mining/paper/crf_klinger_tomanek.pdf. See pp. 31, 155.
- Koh, K., Kim, S.-J., and Boyd, S. (2007). "An Interior-Point Method for Large-Scale l_1 -Regularized Logistic Regression". In: *Journal of Machine Learning Research* 8, pp. 1519–1555. See p. 107.
- Kohavi, R. (1995). "A study of cross-validation and bootstrap for accuracy estimation and model selection". In: *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*. Vol. 2. 12. Morgan Kaufmann, pp. 1137–1143. See p. 26.
- Kohavi, R. and John, G. H. (1997). "Wrappers for Feature Subset Selection". In: *Artificial Intelligence – Special Issue on relevance* 97.1-2, pp. 273–324. See pp. 105, 115.
- Kolářik, C., Hofmann-Apitius, M., Zimmermann, M., and Fluck, J. (2007). "Identification of new drug classification terms in textual resources". In: *Bioinformatics* 23.13, pp. i264–i272. See p. 60.
- Kolářik, C., Klinger, R., and Hofmann-Apitius, M. (2009). "Identification of Histone Modifications in Biomedical Text for Supporting Epigenomic Research". In: *BMC Bioinformatics* 10.S28. Proceedings of the Asia Pacific Bioinformatics Conference (APBC). See pp. 20, 150.
- Kolářik, C., Klinger, R., Friedrich, C. M., Hofmann-Apitius, M., and Fluck, J. (2008). "Chemical Names: Terminological Resources and Corpora Annotation". In: *Workshop on Building and evaluating resources for biomedical text mining (6th edition of the Language Resources and Evaluation Conference)*. Marrakech, Morocco, pp. 51–58. See pp. 59, 61, 62, 75, 156.

- Korn, G. A. and Korn, T. M. (2000). *Mathematical Handbook for Scientists and Engineers: Definitions, Theorems, and Formulas for Reference and Review*. 2 Revised. New York: Dover Publications Inc. See p. 35.
- Kschischang, F., Frey, B. J., and Loeliger, H.-A. (2001). "Factor Graphs and the Sum-Product Algorithm". In: *IEEE Transactions on Information Theory* 47.2, pp. 498–519. See pp. 40, 44, 55, 136.
- Kuo, C.-J., Chang, Y.-M., Huang, H.-S., Lin, K.-T., Yang, B.-H., Lin, Y.-S., Hsu, C.-N., and Chung, I.-F. (2006). "Rich Feature Set, Unification of Bidirectional Parsing and Dictionary Filtering for high F-Score Gene Mention Tagging". In: *Proceedings of the Second BioCreative Challenge Evaluation Workshop*, pp. 105–107. See p. 78.
- Lafferty, J. D., McCallum, A., and Pereira, F. C. N. (2001). "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data". In: *Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001)*. Morgan Kaufmann Publishers, pp. 282–289. See pp. 5, 31, 32, 43, 52, 53, 60, 105, 121.
- Lau, R., Rosenfeld, R., and Roukos, S. (1993). "Adaptive language modeling using the maximum entropy principle". In: *HLT '93: Proceedings of the workshop on Human Language Technology*. Princeton, New Jersey: Association for Computational Linguistics, pp. 108–113. See p. 36.
- Leach, S. M., Tipney, H., Feng, W., Baumgartner, W. A., Kasliwal, P., Schuyler, R. P., Williams, T., Spritz, R. A., and Hunter, L. (2009). "Biomedical discovery acceleration, with applications to craniofacial development." In: *PLoS Computational Biology* 5.3, e1000215. See p. 6.
- Leaman, R. and Gonzalez, G. (2008). "BANNER: An executable survey of advances in biomedical named entity recognition." In: *Pacific Symposium on Biocomputing*. Vol. 13, pp. 652–663. See p. 20.
- Leaman, R., Wojtulewicz, L., Sullivan, R., Skariah, A., Yang, J., and Gonzalez, G. (2010). "Towards Internet-Age Pharmacovigilance: Extracting Adverse Drug Reactions from User Posts to Health-Related Social Networks". In: *Proceedings of the Workshop on Biomedical Natural Language Processing*. See p. 5.
- Lee, L. C., Horn, F., and Cohen, F. E. (2007). "Automatic Extraction of Protein Point Mutations Using a Graph Bigram Association". In: *PLoS Computational Biology* 3.2, pp. 184–198. See pp. 90, 95, 100, 101.
- Lee, S.-I., Ganapathi, V., and Koller, D. (2006). "Efficient Structure Learning of Markov Networks using L1-Regularization". In: *Advances in Neural Information Processing Systems*. See p. 135.
- Lepar, V. and Shenoy, P. P. (1999). "A Comparison of Lauritzen-Spiegelhalter, Hugin, and Shenoy-Shafer Architectures for Computing Marginals of Probability Distributions." In:

- Uncertainty in Artificial Intelligence*. Ed. by G. F. Cooper and S. Moral. Vol. 14. San Francisco, CA: Morgan Kaufmann, pp. 328–337. See p. 56.
- Leser, U. and Hakenberg, J. (2005). “What makes a gene name? Named entity recognition in the biomedical literature.” In: *Brief Bioinformatics* 6.4, pp. 357–369. See pp. 77, 78.
- Lewis, D. D., Yang, Y., Rose, T., and Li, F. (2004). “A New Benchmark Collection for Text Categorization Research”. In: *Journal of Machine Learning Research* 5, pp. 361–397. See p. 113.
- Liu, H. and Motoda, H. (2008). *Computational Methods of Feature Selection*. Data Mining and Knowledge Discovery Series. Chapman & Hall/CRC. See p. 105.
- Liu, J., Huang, M., and Zhu, X. (2010). “Recognizing Biomedical Named Entities using Skip-chain Conditional Random Fields”. In: *Proceedings of the Workshop on Biomedical Natural Language Processing*. See pp. 20, 134, 136, 145.
- Lu, J. K., Chen, T. T., Chrisman, C. L., Andrisani, O. M., and Dixon, J. E. (1992). “Integration, expression and germ-line transmission of foreign growth hormone genes in medaka (*Oryzias latipes*).” In: *Journal of Molecular Marine Biology and Biotechnology* 1.4-5, pp. 366–375. See p. 26.
- Lupu, M., Piroi, F., Huang, X. J., Zhu, J., and Tait, J. (2009). “Overview of the TREC 2009 Chemical IR Track”. In: *The Eighteenth Text REtrieval Conference (TREC) Proceedings*. See p. 15.
- Maglott, D., Ostell, J., Pruitt, K. D., and Tatusova, T. (2005). “Entrez Gene: gene-centered information at NCBI”. In: *Nucleic Acids Research* 0, pp. D1–D6. See p. 96.
- Mann, G. S. and Yarowsky, D. (2003). “Unsupervised personal name disambiguation”. In: *Proceedings of the 7th Conference on Natural Language Learning at HLT-NAACL 2003*, pp. 33–40. See p. 24.
- Manning, C. D. and Schütze, H. (2003). *Foundations of Natural Language Processing*. 6th. MIT Press. See p. 4.
- Marcus, M. P., Santorini, B., and Marcinkiewicz, M. A. (1994). “Building a Large Annotated Corpus of English: The Penn Treebank”. In: *Computational Linguistics* 19.2, pp. 313–330. See p. 23.
- McCallum, A. (2003). “Efficiently Inducing Features of Conditional Random Fields”. In: *Proceedings of the 19th Conference in Uncertainty in Artificial Intelligence (UAI-2003)*, pp. 403–410. See pp. 20, 107.
- McCallum, A., Rohanimanesh, K., and Sutton, C. (2003). “Dynamic Conditional Random Fields for Jointly Labeling Multiple Sequences”. In: *Proceedings of the Conference on Neural Information Processing Systems*. See p. 23.

- McCallum, A., Schultz, K., and Singh, S. (2009). "FACTORIE: Probabilistic Programming via Imperatively Defined Factor Graphs". In: *Advances on Neural Information Processing Systems (NIPS)*. See p. 151.
- McDonald, R. T., Winters, R. S., Mandel, M., Jin, Y., White, P. S., and Pereira, F. (2004). "An entity tagger for recognizing acquired genomic variations in cancer literature". In: *Bioinformatics* 20, pp. 3249–3251. See pp. 32, 90, 91, 94.
- McDonald, R. and Pereira, F. (2005). "Identifying Gene and Protein Mentions in Text Using Conditional Random Fields". In: *BMC Bioinformatics* 6 Suppl 1, S6. See pp. 20, 32, 48, 49, 51, 52.
- McDonald, R., Winters, R. S., Ankuda, C. K., Murphy, J. A., Rogers, A. E., Pereira, F., Greenblatt, M. S., and White, P. S. (2006). "An automated procedure to identify biomedical articles that contain cancer-associated gene variants". In: *Human Mutation* 27.9, pp. 957–964. See pp. 20, 89.
- McNaught, A. D. and Wilkinson, A. (1997). *Compendium of Chemical Terminology – The Gold Book*. Blackwell Science. See pp. 8, 59.
- Melcher, J. (2007). *Java Non-Dominated Sorting Genetic Algorithm II (JNSGA II) – Implementation*. Software. <http://sourceforge.net/projects/jnsga2> (visited on 03/25/2009). See p. 126.
- Meyer, V. (2009). "Extraction, Classification, and empirical Comparison of Zones in Biomedical Texts". Magisterarbeit. Rheinische Friedrich-Wilhelms-Universität Bonn. See p. 15.
- Miller, B. L. and Goldberg, D. E. (1995). "Genetic algorithms, tournament selection, and the effects of noise". In: *Complex Systems* 9, pp. 193–212. See p. 123.
- Minkov, E., Wang, R. C., Tomasic, A., and Cohen, W. W. (2006). "NER Systems that Suit User's Preferences: Adjusting the Recall-Precision Trade-off for Entity Extraction". In: *Proceedings of the Human Technology Conference of the North American Chapter of the ACL*, pp. 93–96. See pp. viii, 122, 128, 130, 131.
- Mooij, J. M. and Kappen, H. J. (2007). "Sufficient conditions for convergence of the Sum-Product Algorithm". In: *IEEE Transactions on Information Theory* 53.21, pp. 4422–4437. See p. 56.
- Morton, T. and LaCivita, J. (2003). "WordFreak: an Open Tool for Linguistic Annotation". In: *HLT/NAACL 2003: demonstrations*, pp. 17–18. See p. 16.
- Murray-Rust, P. (1997). "Chemical Markup Language: A Simple Introduction to Structured Documents". In: *World Wide Web Journal* 2.4, pp. 135–147. See p. 67.

- Nadeau, D. and Sekine, S. (2007). *A Survey of Named Entity Recognition and Classification*. Online – <http://nlp.cs.nyu.edu/sekine/papers/li07.pdf> – last accessed 07/22/2010. See p. 4.
- Narayanaswamy, M., Ravikumar, K. E., and Vijay-Shanker, K. (2003). “A Biological Named Entity Recognizer”. In: *Proceedings of the Pacific Symposium on Biocomputing*, pp. 427–438. See pp. 20, 60.
- NCBI (2007). *PubChem Data*. Online. <ftp://ftp.ncbi.nlm.nih.gov/pubchem/Compound/CURRENT-Full/XML/> – last accessed 2007-09-05. See p. 64.
- Ogino, S., Gulley, M. L., Dunnen, J. T. den, Wilson, R. B., Molecular Pathology Training, T. A. for, and Committee, E. (2007). “Standard mutation nomenclature in molecular diagnostics: practical and educational challenges.” In: *The Journal of Molecular Diagnostics* 9.1, pp. 1–6. See p. 95.
- Ogren, P. V. (2006). “Knowtator: a protégé plug-in for annotated corpus construction”. In: *Proceedings of the 2006 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*. New York, New York: Association for Computational Linguistics, pp. 273–275. See p. 16.
- OpenEye (2007). *LexiChem*. Software. <http://www.eyesopen.com/products/toolkits/lexichem.html> – last accessed: 2007-12-18. See p. 66.
- Pafilis, E., O’Donoghue, S. I., Jensen, L. J., Horn, H., Kuhn, M., Brown, N. P., and Schneider, R. (2009). “Reflect: augmented browsing for the life scientist.” In: *Nature Biotechnology* 27.6, pp. 508–510. See p. 26.
- Parise, S. and Welling, M. (2006). “Structure Learning in Markov Random Fields”. In: *Advances in Neural Information Processing Systems*. See p. 135.
- Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems – Networks of Plausible Inference*. Revised Second Printing. Morgan Kaufmann Publishers, Inc. See p. 39.
- Pearson, K. (1900). “On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling”. In: *Philosophical Magazine Series 5* 50.302, pp. 157–175. See p. 109.
- Peng, F. and McCallum, A. (2004). “Accurate Information Extraction from Research Papers using Conditional Random Fields”. In: *Proceedings of Human Language Technology Conference and North American Chapter of the Association for Computational Linguistics (HLT-NAACL)*, pp. 329–336. See pp. 49, 107.
- (2006). “Information Extraction from Research Papers using Conditional Random Fields”. In: *Information Processing and Management* 42.4, pp. 963–979. See p. 3.

- Pesch, R. (2010). "Logische Strukturerkennung in biomedizinischen Volltext-Publikationen". MA thesis. University of Applied Sciences Bonn-Rhein-Sieg. See p. 16.
- Peterson, J. L. (2010). "The Challenges of Seeking and Receiving Support for Women Living With HIV." In: *Health Communication* 25.5, pp. 470–479. See pp. 4, 18.
- Phua, C., Lee, V., and Smith, K. (2006). "The personal name problem and a recommended data mining solution." In: *Encyclopedia of Data Warehousing and Mining*. Idea Group Publishing. See p. 24.
- Plackett, R. L. (1983). "Karl Pearson and the Chi-Squared Test". In: *International Statistical Review* 51.1, pp. 59–72. See p. 109.
- Plake, C., Schiemann, T., Pankalla, M., Hakenberg, J., and Leser, U. (2006). "AliBaba: PubMed as a graph." In: *Bioinformatics* 22.19, pp. 2444–2445. See p. 6.
- Quattoni, A., Collins, M., and Darrell, T. (2005). "Conditional Random Fields for Object Recognition". In: *Advances in Neural Information Processing Systems 17*. Ed. by L. K. Saul, Y. Weiss, and L. Bottou. Cambridge, MA: MIT Press, pp. 1097–1104. See p. 32.
- Quinlan, J. R. (1986). "Induction of Decision Trees". In: *Machine Learning* 1, pp. 81–106. See p. 105.
- Rabiner, L. R. (1989). "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition". In: *Proceedings of the IEEE* 77.2, pp. 257–286. See pp. 31, 34, 48, 51, 52, 61.
- Rebholz-Schuhmann, D., Kirsch, H., Arregui, M., Gaudan, S., Riethoven, M., and Stoehr, P. (2007). "EBIMed – text crunching to gather facts for proteins from Medline". In: *Bioinformatics* 23, pp. 237–244. See p. 60.
- Rebholz-Schuhmann, D., Marcel, S., Albert, S., Tolle, R., Casari, G., and Kirsch, H. (2004). "Automatic extraction of mutations from Medline and cross-validation with OMIM." In: *Nucleic Acids Research* 32.1, pp. 135–142. See pp. 90, 101.
- Reyle, U. (2006). "Understanding chemical terminology". In: *Terminology* 12.1, pp. 111–136. See p. 60.
- Rhodes, J., Boyer, S., Kreulen, J., Chen, Y., and Ordonez, P. (2007). "Mining Patents using Molecular Similarity Search". In: *Proceedings of the Pacific Symposium on Biocomputing*. Vol. 12, pp. 304–315. See p. 60.
- Rijsbergen, C. J. van (1979). *Information Retrieval*. Butterworth. See p. 24.
- Russell, S. and Norvig, P. (2003). *Artificial Intelligence – A Modern Approach*. Prentice Hall. See pp. 31, 33, 109, 140.

- Sang, E. F. T. K. and De Meulder, F. (2003). “Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition”. In: *Proceedings of Computational Natural Language Learning (CoNLL)*. Ed. by W. Daelemans and M. Osborne. Edmonton, Canada, pp. 142–147. See pp. 5, 113.
- Savary, A. and Jacquemin, C. (2003). “Reducing Information Variation in Text”. In: vol. 2705. Springer, pp. 145–181. See p. 24.
- Schmidt, M., Murphy, K., Fung, G., and Rosales, R. (2008). “Structure Learning in Random Fields for Heart Motion Abnormality Detection”. In: *Computer Vision and Pattern Recognition*. IEEE. Anchorage, AK, USA. See p. 135.
- Schölkopf, B. and Smola, A. J. (2002). *Learning with Kernels*. The MIT Press. See pp. 60, 121.
- Settles, B. (2004). “Biomedical Named Entity Recognition Using Conditional Random Fields and Rich Feature Sets”. In: *Proceedings of the COLING 2004 International Joint Workshop on Natural Language Processing in Biomedicine and its Applications (NLPBA)*. Geneva, Switzerland. See p. 32.
- (2005). “ABNER: an open source tool for automatically tagging genes, proteins and other entity names in text.” In: *Bioinformatics* 21.14, pp. 3191–3192. See pp. 20, 21, 79.
- Seung, H. S., Opper, M., and Sompolinsky, H. (1992). “Query by committee”. In: *Proceedings of the fifth annual workshop on Computational learning theory*. Pittsburgh, Pennsylvania, United States: ACM, pp. 287–294. See p. 15.
- Sha, F. and Pereira, F. (2003). “Shallow parsing with Conditional Random Fields”. In: *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology (NAACL '03)*. Edmonton, Canada: Association for Computational Linguistics, pp. 134–141. See p. 32.
- Shatkay, H., Höglund, A., Brady, S., Blum, T., Dönnies, P., and Kohlbacher, O. (2007). “SherLoc: high-accuracy prediction of protein subcellular localization by integrating text and protein sequence data.” In: *Bioinformatics* 23.11, pp. 1410–1417. See p. 3.
- Siegel, S. and Castellan, N. J. (1988). *Nonparametric Statistics for the Behavioral Sciences*. McGraw-Hill. See p. 17.
- Singh, S., Schultz, K., and McCallum, A. (2009). “Bi-directional Join Inference for Entity Resolution and Segmentation Using Imperatively-Defined Factor Graphs”. In: *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*. See p. 44.
- Smigielski, E. M., Sirotkin, K., Ward, M., and Sherry, S. T. (2000). “dbSNP: a database of single nucleotide polymorphisms”. In: *Nucleic Acids Research* 28.1, pp. 352–355. See p. 89.

- Smith, L., Tanabe, L. K., Ando, R. J. nee, Juo, C.-J., Chung, I.-F., Hsu, C.-N., Lin, Y.-S., Klinger, R., Friedrich, C. M., Ganchev, K., Torii, M., Liu, H., Haddow, B., Struble, C. A., Povinelli, R. J., Vlachos, A., Baumgartner Jr., W. A., Hunter, L., Carpenter, B., Tsai, R. T.-H., Dai, H. jie, Liu, F., Chen, Y., Sun, C., Katrenko, S., Adriaans, P., Blaschke, C., Perez, R. T., Neves, M., Nakov, P., Divoli, A., Mana, M., Mata-Vazquez, J., and Wilbur, W. J. (2008). “Overview of BioCreative II Gene Mention Recognition”. In: *Genome Biology* 9.Suppl 2, S2.2–S2.18. ISSN: 1465-6906. See pp. 77, 155.
- Song, Y., Huang, J., Councill, I. G., Li, J., and Giles, C. L. (2007). “Efficient topic-based unsupervised name disambiguation”. In: *In Proceedings of the 2007 Joint Conference on Digital libraries*, pp. 342–351. See p. 24.
- Srivastava, A. N. and Sahami, M., eds. (2009). *Text Mining – Classification, Clustering, and Applications*. Chapman & Hall/CRC. See p. 4.
- Steinbeck, C., Han, Y., Kuhn, S., Horlacher, O., Luttmann, E., and Willighagen, E. (2003). “The Chemistry Development Kit (CDK): An Open-Source Java Library for Chemo- and Bioinformatics”. In: *Journal of Chemical Information and Computer Sciences* 43, pp. 493–500. See p. 67.
- Sun, B., Tan, Q., Mitra, P., and Giles, C. L. (2007). “Extraction and Search of Chemical Formulae in Text Documents on the Web”. In: *Proceedings of the International World Wide Web Conference*, pp. 251–260. See p. 60.
- Sutton, C. and McCallum, A. (2004). *Collective Segmentation and Labeling of Distant Entities in Information Extraction*. Tech. rep. TR 04-49. Department of Computer Science, University of Massachusetts. See p. 53.
- (2005). “Joint Parsing and Semantic Role Labeling”. In: *Proceedings of the Conference on Computational Natural Language Learning*. See p. 23.
- (2007). “An Introduction to Conditional Random Fields for Relational Learning”. In: *Introduction to Statistical Relational Learning*. Ed. by L. Getoor and B. Taskar. MIT Press. Chap. 4, pp. 93–127. See pp. viii, 32, 44, 53, 133, 134, 145.
- Suzuki, J., McDermott, E., and Isozaki, H. (2006). “Training Conditional Random Fields with Multivariate Evaluation Measures”. In: *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the ACL*. Association for Computational Linguistics, pp. 217–224. See pp. 121, 122, 126.
- The UniProt Consortium (2006). “The Universal Protein Resource (UniProt)”. In: *Nucleic Acids Research* 35, pp. D193–D197. See p. 96.
- Thomas, P. (2008). “Automated extraction of variation mentions from literature sources and mapping to a unique database identifier”. MA thesis. Eberhard Karls Universität Tübingen. See p. 93.

- Thomas, P., Klinger, R., Furlong, L., Hofmann-Apitius, M., and Friedrich, C. (2011). “Challenges in the Association of Human Variation Names with Unique Database Identifiers”. In: *AIMM Special Issue of BMC Bioinformatics*. accepted. See pp. 89, 94, 95, 155.
- Tomanek, K. (2010). “Resource-Aware Annotation through Active Learning”. PhD thesis. Dortmund University of Technology. See p. 15.
- Tomanek, K., Wermter, J., and Hahn, U. (2007a). “A Reappraisal of Sentence and Token Splitting for Life Science Documents”. In: *Proceedings of the 12th World Congress on Medical Informatics (MEDINFO 2007)*. Brisbane, Australia, pp. 524–528. See p. 32.
- (2007b). “A reappraisal of sentence and token splitting for life science documents”. In: *Proceedings of the 12th World Congress on Medical Informatics*. See p. 65.
- (2007c). “Sentence and token splitting based on conditional random fields”. In: *Proceedings of the 10th Conference of the Pacific Association for Computational Linguistics*. Pacific Association for Computational Linguistics. Melbourne, Australia, pp. 49–57. See p. 18.
- Tsai, R. T.-H., Sung, C.-L., Dai, H.-J., Hung, H.-C., Sung, T.-Y., and Hsu, W.-L. (2006). “NERBio: using selected word conjunctions, term normalization, and global patterns to improve biomedical named entity recognition”. In: *BMC Bioinformatics* 7(Suppl 5).S11. See p. 53.
- Tsuruoka, Y., Tateishi, Y., Kim, J., Ohta, T., McNaught, J., Ananiadou, S., and Tsujii, J. (2005). “Developing a Robust Part-of-Speech Tagger for Biomedical Text”. In: *Advances in Informatics - 10th Panhellenic Conference on Informatics, LNCS 3746*, pp. 382–392. See p. 79.
- U.S. National Library of Medicine (2007). *MedlinePlus*. Online. <http://www.nlm.nih.gov/medlineplus/druginformation.html> – last accessed 2008-09-01. See p. 60.
- Vail, D. L. (2008). “Conditional Random Fields for Activity Recognition”. PhD thesis. Pittsburgh, USA: Computer Science Department, Carnegie Mellon University. See pp. 49, 107.
- Vail, D. L., Lafferty, J. D., and Veloso, M. M. (2007). “Feature Selection in Conditional Random Fields for Activity Recognition”. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3379–3384. See pp. 49, 107.
- Vlachos, A. (2008). “A stopping criterion for active learning”. In: *Comput. Speech Lang.* 22.3, pp. 295–312. ISSN: 0885-2308. See p. 15.
- Wainwright, M. J., Jaakkola, T., and Willsky, A. S. (2001). “Tree-based reparameterization for approximate inference on loopy graphs”. In: *Proceedings of the Conference on Neural Information Processing Systems*. See p. 140.
- Wallach, H. M. (2004). *Conditional Random Fields: An Introduction*. Tech. rep. MS-CIS-04-21. Department of Computer and Information Science, University of Pennsylvania. See p. 32.

- Weininger, D. (1988). "SMILES, a Chemical Language and Information System. 1. Introduction to Methodology and Encoding Rules". In: *Journal of Chemical Information and Computer Sciences* 28, pp. 31–36. See p. 59.
- Wilbur, J., Smith, L., and Tanabe, L. (2007). "BioCreative 2. Gene Mention Task". In: *Proceedings of the Second BioCreative Challenge Evaluation Workshop*, pp. 7–9. See pp. 25, 26, 77.
- Wishart, D. S., Knox, C., Guo, A. C., Shrivastava, S., Hassanali, M., Stothard, P., Chang, Z., and Woolsey, J. (2006). "DrugBank: a comprehensive resource for in silico drug discovery and exploration". In: *Nucleic Acids Research* 34, pp. D668–D672. See p. 60.
- World Wide Web Consortium (W3C) (1999). *HTML 4.01 Specification – W3C Recommendation*. online. <http://www.w3.org/TR/1999/REC-html401-19991224/>. See p. 15.
- Yang, Y. and Pederson, J. O. (1997). "A Comparative Study on Feature Selection in Text Categorization". In: *Proceedings of the International Conference on Machine Learning*. See p. 105.
- Yeh, A., Morgan, A., Colosimo, M., and Hirschman, L. (2005). "BioCreAtIvE task 1A: gene mention finding evaluation." In: *BMC Bioinformatics* 6 Suppl 1, S2. See p. 77.
- Yoneyama, T., Kasuya, H., Onda, H., Akagawa, H., Hashiguchi, K., Nakajima, T., Hori, T., and Inoue, I. (2004). "Collagen type I alpha2 (COL1A2) is the susceptible gene for intracranial aneurysms." In: *Stroke* 35.2, pp. 443–448. See p. 92.
- Younesi, E. (2008). "Analysis of Molecular Interaction Networks associated with Breast Cancer Tumour Susceptibility". MA thesis. Rheinische Friedrich-Wilhelms-Universität Bonn. See p. 6.
- Zamir, O., Etzioni, O., Madani, O., and Karp, R. M. (1997). "Fast and Intuitive Clustering of Web Documents". In: *Proceedings of the Conference on Knowledge Detection in Databases*, pp. 287–290. See p. 3.
- Zhang, X., Aberdeen, D., and Vishwanathan, S. V. N. (2007). "Conditional random fields for multi-agent reinforcement learning". In: *ICML '07: Proceedings of the 24th international conference on Machine learning*. Corvalis, Oregon: ACM, pp. 1143–1150. See p. 32.

Index

A

Abner 79, 82
 Accuracy 25, 121
 ACDName 66
 Acronyms 79
 Active Learning 15, 93, 116
 AliBaba 6
 AmilCare 121
 Amino Acids 79
 Annotation 64, 78, 93
 Annotation Guideline 64, 91
 Annotator 80

B

Bag of Words 6, 21
 Bayes' law 33
 BioCreative 113
 BioTagger 94
 Bootstrapping 26, 63, 81
 Brackets 79
 Brief Word Class 21

C

Canonical Form 23
 Cell Lines 79
 ChEBI 62
 Chemical Entities of Biological Interest 62
 Chemical Markup Language 67
 Chemical Names 79
 Chemistry Development Kit 67
 χ^2 -Statistics 109
 Classification 3, 31
 Clique Template 44, 54

Clustering 3
 Concave 50
 Conditional independence 39
 Conditional Model 35
 Conditional Probability 32
 Conditional Random Field 7, 31 f, 39, 43 f,
 91, 105, 121
 Basic Problems 48
 Inference 52
 Training 48
 Confidence 121
 CoNLL 113
 Context 79
 Contingency Table 24
 Coreference 3
 Cross Validation 74
 Crowding Distance 123

D

dbSNP 89
 Decision Threshold 121
 Device Independent File 15
 Directed Graphical Model 40
 Disambiguation 4, 23
 Discriminative Approach 33, 43
 Domain Adaptation 151
 Domination 123
 DrugBank 60, 62
 DVI 15
 Dynamic Programming 51

E

Empirical Distribution 105
 EntrezGene 79

- Enzymes 79
Evaluation 24, 81
Evolutionary Optimization 123
Exponential Prior 49
eXtensible Markup Language 15
- F**
- F_1 measure 18, 25
Factor Graph 40
Factorization 40
 F_β measure 25, 121
 Approximation 122
Feature Conjunction 107
Feature Extraction 20, 93
Feature Selection 105, 107, 151
Features 20, 66, 82, 108
Filter 105
Filtering 23
Forward-Backward Algorithm 51, 55
Fraunhofer Institute for Algorithms and
 Scientific Computing .. 7, 79, 150
- G**
- Gaussian Prior 49
Generalization 26, 151
Generative Approach 32
GeniaTagger 79
GoPubMed 6
- H**
- HapMap 89
Hidden Markov Model 31 f, 34, 43, 48, 51
HMDB 62
HTML 15
Hugo 79
Human metabolome Database 62
Hyperbolic Prior 49
HyperText Markup Language 15
- I**
- Imperatively Defined Factor Graphs .. 151
- InChI 59
Independency Graph 39
Information Extraction 3, 121
Information Gain 108
Information Retrieval 3, 121
Inter-Annotator Agreement 17 f, 61, 63, 65
IOB format 19, 31
Iterative Feature Pruning 110
IUPAC 8, 59, 113
- J**
- JLex 19
Joined Model 151
Joint Probability 32, 35
Junction Tree 55
- K**
- k -fold Cross-Validation 26
 κ statistic 17
KEGG 62
Knowtator 16
Kyoto Encyclopedia of Genes and Genomes
 62
- L**
- L-BFGS 110, 152
 L_1 Prior 49
Label Sequence 19
Lagrange function 37
Leave-One-Out 26
Lemmatization 20
LexiChem 66
Linear Chain CRF 47
Log-likelihood 48, 50
Log-linear model 39, 42
- M**
- Max-Sum Algorithm 55
Maximum Entropy Model . 32, 35, 39, 43,
 121

- Medical Subject Headings 62
 Medline 13, 60, 78, 85
 MEMA 90
 MeSH 62
 Message Passing 51, 55
 Multi-Objective Optimization 121
 Mutation 123
 MuteXt 90
- N**
- Naïve Bayes Model 32, 34, 43
 Name-to-Structure 59, 63
 Name=Struct 66
 Named Entity Recognition .. 4, 7, 31, 121, 133
 Dictionary-based 4, 20, 95
 Machine Learning-based 5, 151
 Rule-based 5, 20, 95
 Network Analysis 6
 Non-dominated Sorting Genetic Algorithm II 123
 Normalization 6, 23, 42, 63, 66, 89, 91, 94, 151
 Normalization Factor 44
- O**
- Observation 43
 Offset Conjunction 23, 66
 OpenEye 66
 Opsin 66
 Optimal Brain Damage 107
 Organisms 79
 OSCAR3 61, 66
 OSCAR3 151
 OSIRIS 91
- P**
- Parameter Estimation 48
 Part-of-Speech 23, 79
 Portable Document Format 15, 26
 Postprocessing 79
- PostScript 15
 Potential Function 42
 Precision 25, 121, 151
 Probabilistic Graphical Model 39
 Probabilistic Model 43
 ProMiner 7, 79, 85, 91
 Protégé 17
 PubChem 62
 PubMed 93
 PubMed Central 13
- R**
- Recall 25, 121, 151
 Recombination 123
 Regularization 107, 151
 Relation Extraction 4, 6
 Reuters 113
- S**
- SCAIView 6, 27, 150
 Seed Corpus 93
 Selection 123
 Semantic Search 6
 Sentence 18
 Sentence Splitting 18
 Sequence variations 89
 Shallow Parsing 23
 Single Nucleotide Polymorphism . 89, 151
 Skip Chain 152
 Skip Chain Template 133
 SMILES 59, 67
 Species 79
 Stemming 20, 67
 Stochastic Finite State Automaton 47
 Stopwords 79
 Structured Learning 31
 Sum-Product Algorithm 55
 Support Vector Machines 121
 Synonym 6
- T**
- Test Set 24

Text Mining	1
Text Segmentation	31, 133
Token	18
Token Sequence	19
Tokenization	18, 65, 82, 93
Topic Models	3
U	
Undirected Graphical Model	41
Units	79
Unrolling	54
V	
Validation Set	24
Viterbi Algorithm	52, 55
Vtag	90
W	
Welch's t-test	113
Word Class	21
WorkFreak	16
Wrapper	105
X	
XML	15

