Università Ca' Foscari di Venezia

Dipartimento di Informatica
Dottorato di Ricerca in Informatica

Ph.D. Thesis

Beyond Linear Chain: A Journey through Conditional Random Fields for Information Extraction from Text

Diego Marcheggiani

Supervisor
Dott. Fabrizio Sebastiani

Supervisor
Dott. Andrea Esuli

PhD Coordinator
Prof. Riccardo Focardi

2014

Author's e-mail: diego.marcheggiani@isti.cnr.it

Author's address:

Dipartimento di Informatica
Università Ca' Foscari di Venezia
Via Torino, 155
30172 Venezia Mestre – Italia
tel. +39 041 2348411
fax. +39 041 2348419
web: http://www.dsi.unive.it

To Laura, the reason why this work exists.

Abstract

Natural language, spoken and written, is the most important way for humans to communicate information to each other. In the last decades *natural language processing* (NLP) researchers have studied methods aimed at making computers "understand" the information enclosed in human language. *Information Extraction* (IE) is a field of NLP that studies methods aimed at extracting information from text so that it can be used to populate a structured information repository, such as a relational database. IE is divided into several subtasks, each of which aims to extract different structures from text, such as entities, relations, or more complex structures such as ontologies. In this thesis the term "information extraction" is (somehow arbitrarily) used to identify only the subtasks that are formulated as *sequence labeling* tasks.

Recently, the main approaches by means of which IE has been tackled rely on supervised machine learning, which needs human-labeled data examples in order to train the systems that extract information from yet unseen data. When IE is tackled as a sequence labeling task (as in e.g., *named-entity recognition*, *concept extraction*, and in some cases *opinion mining*), among the best-performing supervised machine learning methods are certainly *probabilistic graphical models*, and, specifically, *Conditional Random Fields* (CRFs). In this thesis we investigate two major aspects related to information extraction from text via CRFs: the creation of CRFs models that outperform the commonly adopted, state-of-the-art, "linear-chain" CRFs, and the impact of the quality of training data on the accuracy of CRFs system for IE. In the first part of the thesis we use the capabilities of the CRFs framework to create new kinds of CRFs (i.e., two-stage, ensemble, multi-label, hierarchical), that unlike the commonly adopted linear-chain CRFs have a customized structure that fits the task taken into consideration. We exemplify this approach on two different tasks, i.e., IE from medical documents and opinion mining from product reviews. CRFs, like any machine learning-based approach, may suffer if the quality of the training data is low. Therefore, the second part of the thesis is devoted to (1) the study of how the quality of the training data affects the accuracy of a CRFs system for IE; and (2) the production of human-annotated training data via semi-supervised *active learning* (AL).

We start in Chapter 2 by facing the task of IE from medical documents written in the Italian language; this consists in extracting chunks of text that instantiate

concepts of interest for medical practitioners, such as drug dosages, pathologies, treatments, etc.. We propose two novel approaches: a cascaded, two-stage method composed by two layers of CRFs, and a confidence-weighted ensemble method that combines standard linear-chain CRFs and the proposed two-stage method. Both the proposed models are shown to outperform a standard linear-chain CRFs IE system.

In Chapter 3 we investigate the problem of aspect-oriented sentence-level opinion mining from product reviews, that consists in predicting, for all sentences in the review, whether the sentence expresses a positive, neutral, or negative opinion (or no opinion at all) about a specific aspect of the product. We propose a set of increasingly powerful models based on CRFs, including a hierarchical multi-label CRFs scheme that jointly models the overall opinion expressed in a product review and the set of aspect-specific opinions expressed in each of its sentences. Also in this task the proposed CRF models obtain better results than linear-chain CRFs.

In Chapter 4 we move to studying the impact that the quality of training data has on the learning process and thus on the accuracy of a classifier. Low quality in training data sometimes derives from the fact that the person who has annotated the data is not an expert of the data domain. We test the impact of training data quality on the accuracy of IE systems oriented to the clinical domain.

Finally, in Chapter 5 we investigate the process of AL in order to obtain good-quality training data with minimum human annotation effort. We propose several AL strategies for a type of semi-supervised CRFs specifically devised for partially labeled sequences. We show that, with respect to the proposed strategies, margin-based strategies always obtain the best results on the four tasks we have tested them on.

Acknowledgments

After a journey more than three years long, there is an astonishing amount of people that I will forget to thank, so if you are not mentioned in the following and you think you deserve to be there, well, I owe you a beer. First things first, this thesis is a joint work of Fabrizio, Andrea and then me. Thank you guys to have got me here with professionalism, passion and patience.
I thank my colleagues and friends Fabrizio (il Farchi), Giacomo, Tiziano and Alejandro (in strict order of appearance), and all the colleagues of the NeMIS laboratory.

I want to thank Oscar that made me grow both professionally and personally.
I thank Thierry, under whom guidance I spent two great months in Paris.
I also thank *mamma e papà* for their unconditional love, help and support. I thank the rest of my family, first of all the little Andrea (Andreino) just for being what he is, Fabio and Monica for Andreino, *la Nonna* and *la Lella* for their endless affection.
I want to thank the new friends that I have met during this years, Luca (Pappalord), Lillo, Matteo, Gian-Luca, Luca (Rossi) and especially Daniele (Fulignu) that after almost 5 years of cohabitation is something closer to a relative than just a friend and Cristina for her lovely, hilarious puns. I thank the guys who made my stay in Paris as if I was at home, Antonio, Markus, Luc-Aurèlien, and all the guys of LIP6. A special mention to my old friends who helped me become what I am now (that could not be a positive thing) especially (my other family) Casapiddu (la Sguattera, Jorgio, Rambo, Mangi) and the friends from Todi that would be impossible to mention one by one without missing someone, except maybe *la Postina*.
Finally, I want to thank the person who, by standing me, supporting me and making me happy, has made the biggest effort in the production of this thesis, my significant other, Laura.

Contents

Introduction **ix**
I.1 Natural Language Processing Tasks . . . ix
I.2 Sequence Labeling . . . x
I.3 Structure of the Thesis . . . xi
I.4 Key Publications . . . xii

1 Graphical Models and Conditional Random Fields **1**
1.1 Graphical Models . . . 1
1.1.1 Undirected Models . . . 2
1.1.2 Directed Models . . . 3
1.2 Conditional Random Fields . . . 4
1.2.1 Linear-Chain Conditional Random Fields . . . 6
1.2.2 Inference in Linear-Chain CRFs . . . 9
1.2.3 Parameter Learning in Linear-Chain CRFs . . . 12
1.3 Inference in General CRFs . . . 13
1.4 Summary . . . 14

2 Information Extraction from Medical Reports **15**
2.1 Introduction . . . 15
2.2 CRF Models and Algorithms . . . 18
2.2.1 Preliminaries and Notation . . . 18
2.2.2 A Baseline CRFs System for IE . . . 18
2.2.3 A Two-Stage CRFs System for IE . . . 19
2.2.4 A Confidence-Weighted Ensemble of CRFs systems for IE . . . 21
2.2.5 Positional Features . . . 22
2.3 Experiments . . . 23
2.3.1 Dataset . . . 23
2.3.2 Evaluation Measures . . . 25
2.3.3 Experimental Setting . . . 26
2.3.4 Results . . . 28
2.3.5 A note on IE in Resource-Scarce Languages . . . 32
2.4 Related Work . . . 33
2.5 Summary . . . 36

3 Conditional Random Fields for Extracting Aspect-Related Opinions **37**
3.1 Introduction . . . 37

3.1.1 Problem Definition . . . 39
3.2 CRFs Models and Algorithms . . . 40
3.2.1 Models . . . 41
3.2.2 Training via Sampling . . . 45
3.2.3 Maximum-A-Posteriori Inference . . . 50
3.3 Experiments . . . 50
3.3.1 Evaluation Measures . . . 50
3.3.2 Dataset . . . 51
3.3.3 Features . . . 55
3.3.4 Experimental Setting . . . 56
3.3.5 Results . . . 56
3.4 Related Work . . . 58
3.5 Summary . . . 59

4 Training Data Quality in Information Extraction 61
4.1 Introduction . . . 61
4.1.1 Our Contribution . . . 63
4.2 Experiments . . . 64
4.2.1 Inter-coder agreement . . . 64
4.2.2 Experimental Protocol . . . 65
4.2.3 Results . . . 66
4.3 Related Work . . . 69
4.3.1 Inter-coder Agreement . . . 69
4.3.2 Effects of Low-Quality Training Data . . . 70
4.4 Summary . . . 71

5 Active Learning with Partially Labeled Sequential Data 73
5.1 Introduction . . . 73
5.2 Partially-labeled CRFs . . . 76
5.2.1 Forward-Backward Algorithm . . . 76
5.2.2 Marginal Probability . . . 77
5.2.3 Viterbi Algorithm . . . 77
5.2.4 Log-Likelihood . . . 78
5.3 Active Learning Strategies for CRFs with Partially Labeled Sequences 78
5.3.1 Greedy Strategies . . . 80
5.3.2 Viterbi Strategies . . . 81
5.3.3 Random Strategy . . . 83
5.4 Experiments . . . 83
5.4.1 Experimental Setting . . . 83
5.4.2 Results . . . 86
5.5 Related Work . . . 91
5.6 Summary . . . 92

6 Conclusions **93**
6.1 Summary 93
6.2 Future Directions 94

A Appendix **97**

Bibliography **101**

List of Figures

1.1 Example of a factor graph 3

1.2 Example of a Bayesian network 4

1.3 Linear-chain CRF 7

2.1 Baseline linear-chain CRF 20

2.2 Distribution of report lengths: the X axis represents ranges of lengths, while the Y axis represents the number of reports whose length falls in the given range. 24

2.3 Histogram representing, for each of the nine tags, the percentage of segments for the tag that occurs in a given quintile of the text. E.g., the leftmost group of bars indicates the percentages of segments for each tag that occurs in the first 20% of the text. 29

3.1 An example hotel review annotated with aspect-specific ratings at the sentence level. 38

3.2 LC CRF model. The plate around the graph stands for "replication" $\forall a \in \mathbb{A}$. 43

3.3 IML CRF model. The plate around the graph stands for "replication" $\forall t \in T$. 44

3.4 CML CRF model. 45

3.5 LCO CRF model. The plate around the graph stands for "replication" $\forall a \in \mathbb{A}$. 46

3.6 IMLO CRF model. The plate around the graph stands for "replication" $\forall t \in T$. 47

3.7 CMLO CRF model. 48

4.1 Microaveraged F_1 as a function of the fraction ω of the training set that is annotated by C_β instead of C_α ("corruption ratio"). The dashed line represents the experiments in Batch 1, the dotted line represents those in Batch 2, and the solid one represents the average between the two batches. The vertical bars indicate, for each $\omega \in \{10, 20, ..., 80, 90\}$, the standard deviation across the 10 runs deriving from the 10 random choices of the elements in Tr_β. 67

4.2 Macroaveraged F_1 as a function of the fraction ω of the training set that is annotated by C_β instead of C_α ("corruption ratio"). The dashed line represents the experiments in Batch 1, the dotted line represents those in Batch 2, and the solid one represents the average between the two batches. The vertical bars indicate, for each $\omega \in \{10, 20, ..., 80, 90\}$, the standard deviation across the 10 runs deriving from the 10 random choices of the elements in Tr_β. 68

5.1 F_1 results on the CoNLL2000 dataset. The annotation is done only on the selected token. The maximum number of annotated tokens used (2100) represents ∼1% of the entire training set. 87
5.2 F_1 results on the CoNLL2003 dataset. The annotation is done only on the selected token. 2100 annotated tokens represent the ∼0.8% of the CoNLL2003 training set. 88
5.3 F_1 results on the CoNLL2003 dataset. The annotation is done on the selected token plus the previous one and the next one. 6100 annotated tokens represent the ∼2.4% of the entire training set. 89
5.4 F_1 results on the CoNLL2000POS dataset. The annotation is done only on the selected token. 2100 annotated tokens represent ∼1% of the entire training set. 90
5.5 F_1 results on the NLPBA2004 dataset. The annotation is done only on the selected token. 2100 annotated tokens represent the ∼0.4% of the entire training set. 91

List of Tables

2.1 Per-tag statistics for the UmbertoI(RadRep) dataset. Column 2 and Column 3 indicate the number $NT(i)$ of tokens and the number $NS(i)$ of segments contained in the dataset for the given tag, Column 4 the number $ND(i)$ of documents with at least one segment for the given tag, Column 5 the average number $ANS(i)$ of segments per document for the given tag, Column 6 the average segment length $ASL(i)$ for the given tag (segment length is the number of tokens contained in it). 23

2.2 Results of the single-annotator experiments. Results are averages across two experiments, (i) train the system on `A1-only` and test it on `Both(1)`, and (ii) train the system on `A2-only` and test it on `Both(2)`. The first row reports the inter-annotator agreement values for A1 and A2 as measured on `Both`. 30

2.3 Results of the multiple-annotator experiment: train the system on the union of `A1-only` and `A2-only`, and test it on the union of `Both(1)` and `Both(2)`. The first row reports the inter-annotator agreement values for A1 and A2 as measured on `Both`. 32

3.1 Number of opinion expressions at the sentence level, broken down by aspect and opinion. Out of 5799 annotated sentences, 4810 sentences contain at least one opinion-laden expression. 52

3.2 Inter-annotator segment-level aspect agreement, expressed in terms of F_1 (higher is better). 53

3.3 Inter-annotator segment-level opinion agreement (restricted to the true positive aspects for each segment), expressed in terms of sMAE^M (lower is better). 53

3.4 The collection of model factors and their corresponding features. Feature vectors: $\mathbf{x}_t$: {words, bigrams, SWN/MPQA/GI bigrams, χ^2 lexicon matches} in the t-th segment in review $\mathbf{x}$; $\mathbf{x}_o$: {words, bigrams, SWN/MPQA/GI bigrams} in $\mathbf{x}$. 55

3.5 Results for different CRF models averaged on five experiments with five different random seeds. Segment-level aspect identification results in terms of F_1 (higher is better). 57

3.6 Results for different CRF models averaged on five experiments with five different random seeds. Segment-level opinion prediction results (restricted to the true positive aspects for each segment) in terms of MAE^M (lower is better). 57

3.7 F_1 comparison between the best-performing model (IMLO) and the human annotators with IMLO results averaged on five experiments with five different random seeds. 58
3.8 sMAE^M comparison between the best-performing model (IMLO) and the human annotators with IMLO results averaged on five experiments with five different random seeds. 58

4.1 Extraction accuracy for the homogeneous setting ($\omega = 0$) and heterogeneous setting ($\omega = 100$), for both batches of experiments (and for the average across the two batches), and along with the resulting inter-coder agreement values expressed as $\kappa(\omega)$. 69

5.1 Training Data Statistics. 84

A.1 Inter-annotator segment-level aspect agreement, expressed in terms of F_1 (higher is better). 0, 1 and 2 are the different annotators and 0-1 means pairwise agreement between annotator 0 and annotator 1. . . . 98
A.2 Inter-annotator segment-level opinion agreement (restricted to the true positive aspects for each segment), expressed in terms of sMAE^M (lower is better). 0, 1 and 2 are the different annotators and 0-1 means pairwise agreement between annotator 0 and annotator 1. 99

Introduction

Language is the main structured means with which humans, and non-humans as well, convey information. Natural language, spoken and written, can be considered the most important way for humans to communicate information with each other. During the past ages, different cultures have built their own languages, each of which is subject to well-defined rules that give some kind of structure to the abstract concept one may want to express. Basically, humans have found a set of common rules to understand each other and express concepts and information. In the modern era, with the rise of automation and the invention of computers, humans have wondered whether an automatic system could understand the natural language of a human being. *Natural Language Processing* (NLP) can be considered as the field that studies methods aimed at extracting some kind of "structure" from a text written in natural language so that a computer could manage the information enclosed in it.

I.1 Natural Language Processing Tasks

The concept of "structure" in a natural language text depends on what one is looking for: the grammatical structure of a sentence, the concepts represented by the entire text, the concepts instantiated by the words in the text, the sentiment conveyed by a word, and so on. The extraction from text of each of these types of information has been tackled in different tasks that we briefly summarize[1], in order of increasing difficulty, as follows:

- The most basic type of text analysis is *part-of-speech (POS) tagging*. This task aims to find, for each word in a text, its corresponding part of speech, i.e., understand if a word is a noun, an adjective, a verb, etc.

- Slightly more complex than POS tagging is *phrase chunking*. Phrase chunking aims to segment a piece of text, usually a sentence, into chunks, i.e., sequences of contiguous words that constitute the basic non-recursive phrases corresponding to the major part-of-speech categories.

- *Information extraction* aims to extract factual structured information from text. Depending on the type of structure one wants to extract, different subtasks are taken into consideration. The subtask that has to do with the extraction of instance of categories such as Location, Person, and Organization, is called *named-entity recognition*. Similar to named-entity recognition is the *concept*

[1]Note that this is not meant to be an exhaustive list of NLP tasks, but only a list of the tasks taken into consideration in this thesis.

extraction subtask, in which the elements to extract are instances of larger concepts (e.g., clinical concepts) that, differently from entities, can span several tokens and are usually domain-dependent. Both of these subtasks can be tackled as sequence *labeling problems*. The subtask aimed at the extraction of relations between entities, e.g., is-employee-of for person-organization entity pairs, is the *relation extraction* task. More complex subtasks of information extraction have to do with the extraction of more complex structure from text, e.g., trees, ontologies, etc.

It is worth to notice that in this dissertation we will use the term "information extraction" to refer to the sequence labeling subtasks (e.g., named-entity recognition, concept extraction) instead of its broader meaning.

- The problems tackled by NLP go beyond the recognition of the factual information expressed in a text. *Sentiment analysis*, in its most traditional setting, aims to recognize if in a text there is some kind of opinion, expressed by an opinion holder, toward a certain object.

I.2 Sequence Labeling

Given the sequential nature of written language, many NLP tasks are best formulated as sequence labeling problems. Part-of-speech tagging aims to label a sequence of words with their parts of speech; phrase chunking aims to assign to each word their membership in a certain kind of phrase; named-entity recognition aims to recognize if a word or a sequence of words are instances of some predefined concept. In the early years of NLP, sequence labeling problems have been solved with automatic systems based on hand-crafted rules. Hand-crafted rules systems, although efficient, are very expensive: the production of rules requires the joint work of a domain expert and a language engineer. An alternative approach that partially solved this issue is the adoption of machine learning systems. In the machine learning scenario a domain expert is certainly needed, but only in order to annotate training data necessary to the machine learning algorithm. For a domain expert the effort of annotating data is much lower than the effort of creating a set of rules. Nowadays the machine learning approach is the most frequently used and most efficient approach for dealing with sequence labeling tasks. Probabilistic graphical models such as Hidden Markov Models (HMMs) and, in particular, Conditional Random Fields (CRFs) are the state of the art in sequence labeling problems. One of the main features of probabilistic graphical models such as Conditional Random Fields is the capability to let the language engineer not only decide the features that model the problem, but also decide how to model the interaction and dependencies among such features and between the features and the elements to classify. These features are common to the algorithms that belong to the family of structured learning algorithms which comprise the structured perceptron, structured Support Vector Machines (SVMs) (a

structured version of the traditional SVMs) and all the maximum-margin approaches using structure prediction frameworks, etc.

Probabilistic graphical models, like any machine learning-based approach, may suffer if the quality of training data is low. Low quality in training data generally arise when the data are annotated by coders that are not expert in the domain, e.g., the medical domain. This lack of experience usually brings about annotation errors in the training data. The use of training data of bad quality will generate trained models of bad quality, that will produce annotation errors similar to those present in the training data. The quality of training data is thus a crucial aspect of the problem, and the creation of high-quality training data is an integral part of the learning process. For this reason research has explored methods, such as *active learning*, that, given a certain amount of effort spent for training data creation, allow to maximize the quality of the training data with respect to the effectiveness potentially produced by models learnt on them.

I.3 Structure of the Thesis

This dissertation faces the sequence labeling task in NLP in two contingent problems: (1) the modeling of the task using CRFs, and (2) the quality of data used to train such models. It is composed by four main chapters, plus an introductory chapter on probabilistic graphical models and their notation, and a conclusions chapter. Chapters 2 and 3 are devoted to the study of novel CRFs structures for the problems of information extraction from clinical text and aspect-related opinion extraction, pushing the capabilities of CRFs beyond those typical of the classic linear-chain structure. Chapters 4 and 5 are devoted to studying the impact of the quality of training data on the learning process. In particular, Chapter 4 shows how poorly annotated data can affect the performance of a CRF classifier on the task of information extraction from clinical text, while Chapter 5 proposes active learning techniques for creating high-quality training data in the context of semi-supervised CRFs.

The outline of this dissertation is thus the following:

- Chapter 1 is a preliminary introduction to probabilistic graphical models and CRFs. It is devoted to introducing the basic techniques of inference and learning on CRFs with the mathematical notation that will be adopted throughout the dissertation.

- In Chapter 2 we discuss the problem of performing information extraction from free-text medical reports via supervised learning. We present two novel approaches to IE for radiology reports: (i) a cascaded, two-stage method based on the well known linear-chain Conditional Random Fields (LC-CRFs), and (ii) a confidence-weighted ensemble method that combines standard LC-CRFs and the proposed two-stage method. We also report on the use of "positional

features", a novel type of feature intended to aid in the automatic annotation of texts in which the instances of a given concept may be hypothesized to occur in specific areas of the text.

- In Chapter 3 we study the finer-grained problem of *aspect-oriented opinion mining at the sentence level*, which consists of predicting, for all sentences in a product review, whether the sentence expresses a positive, neutral, or negative opinion (or no opinion at all) about a specific aspect of the product. For this task we propose a set of increasingly powerful models based on Conditional Random Fields, including a hierarchical multi-label CRFs scheme that jointly models the overall opinion expressed in the review and the set of aspect-specific opinions expressed in each of its sentences. In order to cope with the complex structure of the multi-label models, where exact inference is not possible, we revert to approximate inference via Gibbs sampling. We evaluate the proposed models against a newly created dataset of hotel reviews in which the set of aspects and the opinions expressed concerning them are manually annotated at the sentence level.

- In Chapter 4 we test the impact of training data quality on the accuracy of information extraction systems oriented to the clinical domain. Low quality in training data sometimes derives from the fact that the person who has annotated the data is different (e.g., more junior) from the one against whose judgment the automatically annotated data must be evaluated. We compare the accuracy deriving from training data annotated by the authoritative coder (i.e., the one who has annotated the test data), with the accuracy deriving from training data annotated by a non-authoritative coder.

- In Chapter 5 we study the behaviour of several active learning strategies in a semi-supervised learning scenario. We test the effectiveness of these strategies on different sequence labeling tasks, such as phrase chunking, part-of-speech tagging, named-entity recognition, and biomedical-entity recognition. We find that margin-based strategies constantly achieve good performance in all such tasks. The results also show that the choice of the single-token annotation strategy can bring to unpredictable results on sequence labeling tasks where the structure of the sequences is not regular, as in named-entity recognition.

- Finally, Chapter 6 sums up the topics embraced in this dissertation and proposes new directions for future work.

I.4 Key Publications

The material in this dissertation, and in particular the material in Chapters 2 to 4, is based on the following publications:

- Andrea Esuli, Diego Marcheggiani, and Fabrizio Sebastiani. An enhanced CRFs-based system for information extraction from radiology reports. *Journal of Biomedical Informatics*, 46(3):425–435, 2013

- Diego Marcheggiani, Oscar Täckström, Andrea Esuli, and Fabrizio Sebastiani. Hierarchical multi-label conditional random fields for aspect-oriented opinion mining. In *Proceedings of the 36th European Conference on Information Retrieval (ECIR 2014)*, Amsterdam, NL, 2014

- Diego Marcheggiani and Fabrizio Sebastiani. On the effects of low-quality training data on information extraction from clinical reports. Presented at the *5th Italian Information Retrieval Workshop (IIR 2014)*. Rome, IT, 2014

- Diego Marcheggiani and Thierry Artières. An experimental comparison of active learning strategies for partially labeled sequences. Technical report. 2014

1

Graphical Models and Conditional Random Fields

In this chapter we introduce the theoretical framework that we use throughout this dissertation: graphical models and, in particular, conditional random fields. We first briefly introduce graphical models, and then we deeply dig into conditional random fields in order to present the basis of the framework we adopt and extend in the rest of this thesis. In this chapter we explain the main techniques to perform the most important machine learning operations with conditional random fields, i.e., training the model and label prediction for unseen data.

1.1 Graphical Models

Graphical models (GMs), or probabilistic graphical models (PGMs) [61], are a formalism used to easily describe a probabilistic distribution by using graphs and allowing inference on them. The main advantage of GMs is that they can model a distribution over many variables, partitioning these variables and dealing with them in modeling a small, "local" subset of variables, instead that modeling everything together.

There are two main kinds of GMs, *undirected models* and *directed models*. The main difference between the two classes of models is that they are represented by different graphs, undirected the first and directed the second. This difference means that probability distributions are modeled differently. Before describing undirected and directed graphical models, we need to introduce some notation that is common to both types of model[1].

Let Y be a set of random variables whose values come from set $\mathcal{Y}$. An assignment of values from $\mathcal{Y}$ to Y is denoted by $\mathbf{y}$. We consider y_s as the value assigned to the s-th random variable Y_s.

We will use this notation throughout this chapter to define GMs and how they work.

[1] In this chapter and in the rest of the thesis we adopt the same standard notation used in [134].

1.1.1 Undirected Models

In order to define an undirected model (also called *Markov network* or *Markov random field*) we need to introduce other concepts, that can be seen as the bricks with which we can build an undirected model. We define a *factor* as a local function of the form $\Psi_c(\mathbf{y}_c)$ where the index c ranges from 1 to C, where C is the total number of factors necessary to model the probability distribution p over sets of random variables Y. The factor $\Psi_c(\mathbf{y}_c)$ is defined by a subset $\mathbf{y}_c$ of variables, and its outcome is a non-negative scalar that represents the compatibility of the variables in $\mathbf{y}_c$; the higher is the compatibility of the assignment, the higher is the probability of the assignments. An undirected graphical model can define a family of probability distributions that factorize according to a given set of factors $\mathcal{F} = \{\Psi_c\}$, that is:

$$p(\mathbf{y}) = \frac{1}{Z} \prod_{c=1}^{C} \Psi_c(\mathbf{y}_c) \tag{1.1}$$

Given that the factors are non-negative functions, the product of factors is positive but may not sum to 1. Thus the role of Z is to make sure that the probability p sums to 1:

$$Z = \sum_{\mathbf{y}} \prod_{c=1}^{C} \Psi_c(\mathbf{y}_c) \tag{1.2}$$

When Z is expressed as a function of the set of factors $\mathcal{F}$ is called *partition function*. Notice that the summation on $\mathbf{y}$ represents the summation over all the possible assignments of the sequence, and so, computing Z is in general intractable.

A *random field* is a particular distribution among those defined by an undirected graphical model formalized by Equation (1.1).

Graphical models are so called because the probability distributions they define are prone to be represented as graphs. In fact, the distribution defined in Equation (1.1) can be represented easily by a graph and, more exactly, by a *factor graph* [63].

A factor graph is a bipartite graph $G = (V, F, E)$ where the set $V = \{1, ..., |Y|\}$ represents the nodes that correspond to the random variables of the model, the set $F = \{1, ..., C\}$ represents the other kinds of nodes that corresponds to the factors, and E is the set of edges that link the random variable nodes to the factor nodes. If there is an edge between a random variable node corresponding to a random variable Y_s and a factor node corresponding to a factor Ψ_c, then Y_s is one of the arguments of Ψ_c.

The graph in Figure 1.1 is a factor graph that describes all the distributions over three random variables, y_1, y_2, y_3, that is

$$p(y_1, y_2, y_3) = \frac{1}{Z} \Psi_1(y_1, y_2) \Psi_2(y_2, y_3) \Psi_3(y_1, y_3) \tag{1.3}$$

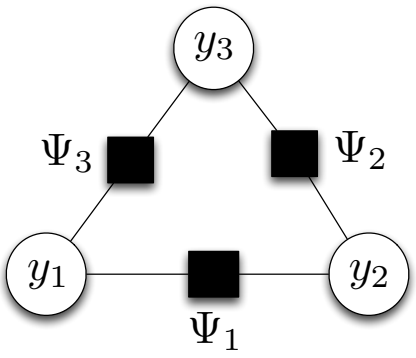

Figure 1.1: Example of a factor graph

As we can see the factor graph reflects precisely the factorization in Equation (1.3). In fact the round nodes represent the random variables, while the squared black nodes represent the factors; finally the edges that connect factor nodes and random variable nodes make explicit the association of variables as factor arguments.

Undirected models have been used in a large span of tasks, from image analysis to term dependencies modeling. Probably the most popular form of undirected model is the Conditional Random Field that, as we will see in Section 1.2, has been used to solve lots of natural language processing tasks and image classification tasks as well.

1.1.2 Directed Models

Differently from undirected models, where we cannot interpret the probabilistic interactions between single variables, in directed models this is possible. In fact, instead of using local functions, directed models factorize the probability distribution into local conditional probability distributions. In this section we define G as a directed acyclic graph (DAG), where $\pi(s)$ represents the indices of parents of Y_s in G. Formally, a directed graphical model is a family of distributions that factorize as:

$$p(\mathbf{y}) = \prod_{s=1}^{S} p(y_s|\mathbf{y}_{\pi(s)}) \tag{1.4}$$

The distribution $p(y_s|\mathbf{y}_{\pi(s)})$ is called *local conditional distribution* and is the locally normalized equivalent of a factor, the undirected local functions. In fact the local conditional distribution is an actual probability distribution, and there is no need of a normalization factor Z.

Figure 1.2 is an example of a directed graphical model (also called *Bayesian network*) that describes the family of distributions over variables y_1, y_2, y_3, namely:

$$p(y_1, y_2, y_3) = p(y_3|y_1, y_2)p(y_2|y_1)p(y_1) \tag{1.5}$$

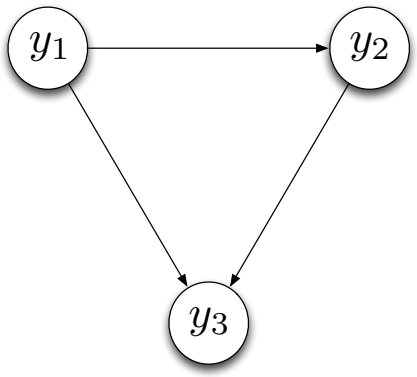

Figure 1.2: Example of a Bayesian network

As we can see from Equation (1.5), variable y_3 depends on both y_1 and y_2; this is reflected on the graph where the node that represents y_3 has two incoming edges, one from node y_1, and one from y_2.

Directed models have been used as a framework over the years to build several supervised, semi-supervised and unsupervised learning algorithms. The most famous such models are probably *Hidden Markov Models* (HMM) [105] used in many important works on information extraction such as [119, 33] but also in other fields such as speech recognition and *Maximum Entropy Markov Models* (MEMM) [79], used in many information extraction works [60] as well as in other fields [127]. Also the *Latent Dirichlet Allocation* (LDA) [9] algorithm for topic modeling belongs to the category of directed graphical models.

1.2 Conditional Random Fields

Conditional random fields (CRFs) [64] belong to a class of learning algorithm that are specifically devised for learning and predicting structures, such as sequences, trees, and also more complex structures namely the *structured learning* algorithms. This class of algorithms in addition to CRFs includes structured maximum margin methods (i.e., structured SVMs) [51, 141, 150, 151], structured perceptron [18], and methods based on reranking approaches [16, 24, 40, 52, 88, 89, 93, 118].

In this dissertation we will make extensive use of CRFs as the machine learning framework to learn and predict different labeled structures. In order to use CRFs as a machine learning tool for learning and predicting random variables, we need to introduce a distinction among random variables. From now on we consider X as the set of *input variables*, that we assume are always observed, while we consider Y as the set of *output variables* from which we learn the model during the learning phase and we wish to predict during the classification phase.

CRFs are a particular instance of undirected models, used for defining the conditional probability $p(\mathbf{y}|\mathbf{x})$, where $\mathbf{y}$ is an assignment for the output variables (also identified as the *classes*, or *tags*, or *labels*), and $\mathbf{x}$ is the assignment for the input variables, namely, the *features*. Let us take as an example the task of part-of-speech (POS) tagging, where to each word or *token* of a sentence we need to associate the correct POS label. In such scenario we will have that $\mathbf{y}$ is an assignment of POS labels for the associated vector of tokens $\mathbf{x}$.

Recalling the Equation (1.1), the general formula that defines a CRF is the following:

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{c=1}^{C} \Psi_c(\mathbf{y}_c, \mathbf{x}_c) \tag{1.6}$$

Equation (1.6), even if apparently very similar to Equation (1.1), has one important difference: now as normalization factor we have $Z(\mathbf{x})$, a function of the input variables $\mathbf{x}$:

$$Z(\mathbf{x}) = \sum_{\mathbf{y}} \prod_{c=1}^{C} \Psi_c(\mathbf{y}_c, \mathbf{x}_c) \tag{1.7}$$

Assuming to have two GMs with the same variables and the same interactions between them, and that in one GM we condition on part of the variables, the GM with conditioned variables will be "simpler" than the other one, given that conditioning on part of the variables is equivalent to constraining some variables to assume a specific value. Thus, in general, given that $\mathbf{x}$ are the variables on which we condition on, calculating $Z(\mathbf{x})$, as function of $\mathbf{x}$, is easier than calculating Z.

The factors Ψ_c adopted in CRFs are defined as:

$$\Psi_c(\mathbf{y}_c, \mathbf{x}_c) = \exp\left\{ \sum_{k=1}^{K(c)} \theta_{ck} f_{ck}(\mathbf{y}_c, \mathbf{x}_c) \right\} \tag{1.8}$$

where:

- θ is the *parameters* vector of the model and is typically learnt from the training data;
- $f(\mathbf{y}_c, \mathbf{x}_c)$ is a *feature function* i.e., functions that return 0 or 1 depending on the values of their arguments (see Section 1.2.1 for a more detailed explanation);
- k is an index that ranges on $[1, K(c)]$, the number of feature functions and associated parameters relative to factor Ψ_c. The quantity $K(c)$ is determined by the dimension of the vectors $\mathbf{y}_c$ and $\mathbf{x}_c$;
- the subscript ck, of both parameters and feature functions, tells us that each factor has its own set of feature functions and associated parameters.

The last observation is true in general, but it is often the case, that models rely on *parameter tying*. Using parameter tying means that we can define different subsets of factors that share common feature functions and the relative parameters. In order to do this, we define a subset of all the factors $\mathcal{D}$ in G such that $\mathcal{D} = \{D_1, D_2, ..., D_P\}$ where each D_p is a *factor template*, that is, a set of factors that share the same feature functions and parameters.

The CRFs that use parameter tying can be defined as:

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{D_p \in \mathcal{D}} \prod_{\Psi_c \in D_p} \Psi_c(\mathbf{y}_c, \mathbf{x}_c; \theta_p) \tag{1.9}$$

where the "tied" factors are defined as:

$$\Psi_c(\mathbf{y}_c, \mathbf{x}_c; \theta_p) = \exp\left\{ \sum_{k=1}^{K(p)} \theta_{pk} f_{pk}(\mathbf{y}_c, \mathbf{x}_c) \right\} \tag{1.10}$$

Equation (1.10) makes explicit how parameter tying works. For each factor template D_p, with associated feature functions and parameters, there are many factors Ψ_c with their associated random variables. The random variables of each factor will affect the parameters and feature functions that define factor template D_p.

Parameter tying is useful when we need to model structures with repeated elements; in this case it is better to have the same parameters describing every single element of the structure instead of having each element described by its own set of parameters.

All these concepts will be explained in more detail in the next section, where we will describe the most frequently used instance of CRFs, Linear-Chain CRFs.

1.2.1 Linear-Chain Conditional Random Fields

Linear-Chain (LC) CRFs are probably the most frequently used CRF in the field of sequence labeling tasks of information extraction [80, 86, 93, 94, 100, 102, 112, 115, 132]. The LC-CRF is tailored to deal with labeled sequences, so it is not surprising that it has been successfully used in tasks such as part-of-speech tagging [64], phrase chunking [120, 135], negation detection [19], and spoken language understanding along with reranking methods [24].

The graph structure of the LC-CRF (Figure 1.3) is the reason why this model works so well on sequences; in fact, it is composed of two factors that describe the interactions between input variables and output variables associated with an element of the sequence, and between output variables associated with contiguous elements of the sequence.

Formally, the equation that defines the LC-CRFs of Figure 1.3 is the following:

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{t=1}^{T} \Psi_u(y_t, \mathbf{x}_t, \theta_\tau) \prod_{t=2}^{T} \Psi_b(y_t, y_{t-1}, \theta_\xi) \tag{1.11}$$

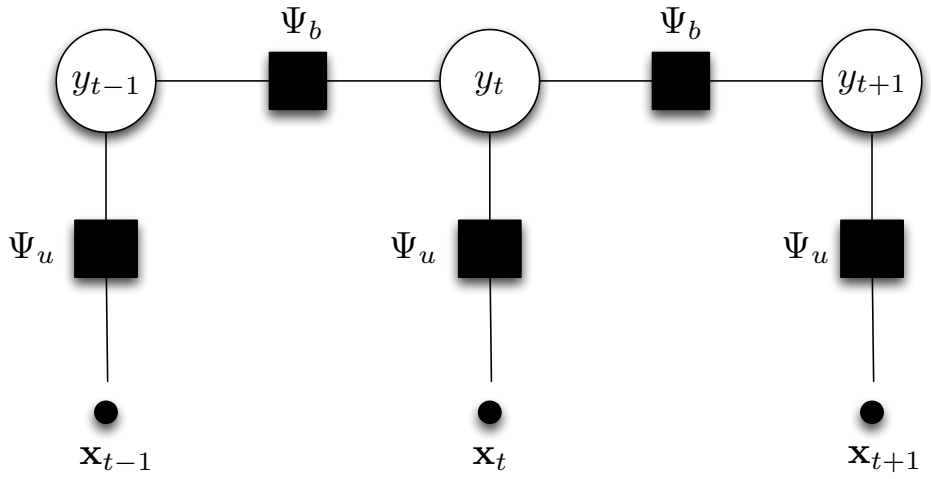

Figure 1.3: Linear-chain CRF

where we have two factor templates: the first one ranges over all the input variables that compose the sequence T (where $T = |\mathbf{x}|$), and the second one ranges over all the input variables minus one, given that it models pairs of contiguous output variables y_t and y_{t-1}.

It is worth noticing that the assignment vector of input variables $\mathbf{x}$ is a vector of vectors. In fact for each element of the sequence we have a vector of observations (features) $\mathbf{x}_t$ of length Ω. Differently from HMMs, CRFs allow to use rich sets of features associated with each element of the sequence. For example, in the POS tagging scenario introduced in Section 1.2, vector $\mathbf{y}$ would be composed by POS labels while each element $\mathbf{x}_t$ of the input vector $\mathbf{x}$ would be composed by several input variables representing the word at position t in the sequence, the lemma of the word, its prefixes, its suffixes, and so on.

The first factor template in Equation (1.11) models the co-occurrence between input variables and output variables. The factors belonging to the *unigram* factor template are defined as:

$$\Psi_u(y_t, \mathbf{x}_t; \theta_\tau) = \exp \sum_k^{K(\tau)} \theta_{\tau k} f_{\tau k}(y_t, \mathbf{x}_t) \tag{1.12}$$

The second factor template in Equation (1.11) models the co-occurrence between contiguous output variables. The factors belonging to the *bigram* factor template are defined as:

$$\Psi_b(y_t, y_{t-1}; \theta_\xi) = \exp \sum_k^{K(\xi)} \theta_{\xi k} f_{\xi k}(y_t, y_{t-1}) \tag{1.13}$$

As we can notice from Equation (1.12) and Equation (1.13), the sets of feature functions used and the parameters affected by the two factor templates are not the same. One factor template shares a set of feature functions and parameters and the other template factor another set. This is a design strategy that allows the designer of the CRFs to model different "aspects" of the distribution in different ways, given that the interaction/dependency between observed variables and output variables, and between contiguous output variables, has a different meaning, so that they cannot be mixed.

Feature Functions

Probably one of the hardest aspects to understand for someone who approaches for the first time CRFs are the feature functions. Feature functions, to put it in an easy and informal way, are the tools with which we encode input variables, output variables, and combinations of them, in order to create a binary vector used to fit the associated parameters and create the CRF model.

More formally, feature functions are binary functions that describe dependencies between input variables and output variables as detected in the training data. In LC-CRF the following two types of feature function are commonly used:

$$f_{idv}(y_t, \mathbf{x}_t) = \begin{cases} 1 & \text{if } y_t = i \text{ and } \mathbf{x}_{td} = v \\ 0 & \text{otherwise} \end{cases} \tag{1.14}$$

$$f_{ij}(y_t, y_{t+1}) = \begin{cases} 1 & \text{if } y_t = i \text{ and } y_{t-1} = j \\ 0 & \text{otherwise} \end{cases} \tag{1.15}$$

where there is a function of type $f_{idv}(y_t, \mathbf{x}_t)$ and a function of type $f_{ij}(y_t, y_{t-1})$ for each combination of $i \in \mathcal{Y}$, $j \in \mathcal{Y}$, $d \in \Omega$, and actual value v of the input variable, taken from the data. $d \in \Omega$ represents the type of input variable that feature functions take into consideration. Taking as example the POS scenario, the types of input variables are the different types of features adopted, e.g., $\Omega_{POS} = \{word, lemma, prefix, suffix\}$. For simplicity of notation, in Equation (1.12) and Equation (1.13) we have used indexes τk and ξk to range on all the possible values of the triple (i, d, v) and of the pair (i, j), respectively.

The following is an example of a feature function of the type described in Equation (1.14) in the context of POS tagging:

$$f_{NN,word,\texttt{guitar}}(y_t, \mathbf{x}_t) = \begin{cases} 1 & \text{if } y_t = NN \\ & \text{and } \mathbf{x}_{t,word} = \texttt{guitar} \\ 0 & \text{otherwise} \end{cases} \tag{1.16}$$

where NN is a POS label indicating a noun, *word* is a type of feature, and `guitar` an actual value that occurs for the type *word* in the training/test data. It can be paraphrased as saying "If the label of the current token is NN, and the current

token is an occurrence of the word `guitar`, then return 1, else return 0". An example feature function of the type described in Equation (1.15) is instead

$$f_{NN,JJ}(y_t, y_{t-1}) = \begin{cases} 1 & \text{if } y_t = NN \\ & \text{and } y_{t-1} = JJ \\ 0 & \text{otherwise} \end{cases} \tag{1.17}$$

where JJ is a POS tag indicating an adjective; the meaning of this feature function is straightforward.

Summing up, functions of the type formalized by Equation (1.14) describe the relation that occurs between an output variable and the input variable of type d at position t in the sequence, whereas functions of the type formalized by Equation (1.15) describe instead the relation that occurs between the output variable at position t in the sequence and the output variable at position $t-1$.

In the next subsections we will explain how to perform inference on a CRF in order to train a CRF model and use this model to make predictions over unobserved data.

1.2.2 Inference in Linear-Chain CRFs

In supervised machine learning there are two main tasks, parameter learning and prediction. In CRFs, in order to accomplish these two tasks we need to perform inference on the model. Inference is used in parameter learning as a subroutine, while it is used directly on unobserved data during prediction. In general, given a fixed parameter vector θ, inference in CRFs consists in predicting the values of a sequence $\mathbf{y}$ given an observed input $\mathbf{x}$. The inference used in parameter learning computes the marginal distribution $p(\mathbf{y}_c|\mathbf{x}, \theta)$ where $\mathbf{y}_c$ can be either a single variable or a set of variables that share a factor. On the other hand, the inference used in prediction computes the most likely assignment $\mathbf{y}^*$ for a new input $\mathbf{x}$. In a LC-CRF, in order to compute the marginal distributions and the most likely assignment it is possible to adopt the same dynamic-programming algorithms adopted in HMM [105], i.e., the *forward-backward* algorithm and the *Viterbi* algorithm. These two algorithms are special cases for chain-structured models of the *belief-propagation* algorithm [63].

Forward-Backward Algorithm

The forward-backward algorithm is used to calculate the marginal distributions needed in the parameter learning procedure; in the case of LC-CRFs these are, $p(y_t|\mathbf{x})$ and $p(y_t, y_{t-1}|\mathbf{x})$. In order to do so, the first thing that we need to calculate in order to compute the two marginal distributions is the normalization function $Z(\mathbf{x})$.
In order to simplify the notation, let

$$\Psi_t(y_t, y_{t-1}, \mathbf{x}_t) = \Psi_u(y_t, \mathbf{x}_t, \theta_\tau)\Psi_b(y_t, y_{t-1}, \theta_\xi) \tag{1.18}$$

Given Equation (1.18) we can re-define Equation (1.11) as:

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{t=1}^{T} \Psi_t(y_t, y_{t-1}, \mathbf{x}_t) \tag{1.19}$$

with the normalization factor of Equation (1.7) re-defined as:

$$Z(\mathbf{x}) = \sum_{\mathbf{y}} \prod_{t=1}^{T} \Psi_t(y_t, y_{t-1}, \mathbf{x}_t) \tag{1.20}$$

Naively, the function in Equation (1.20) would be calculated as:

$$\begin{aligned} Z(\mathbf{x}) = \sum_{y_T} \sum_{y_{T-1}} \Psi_T(y_T, y_{T-1}, \mathbf{x}_T) \sum_{y_{T-1}} \Psi_{T-1}(y_{T-1}, y_{T-2}, \mathbf{x}_{T-1}) \\ \sum_{y_{T-2}} \Psi_{T-2}(y_{T-2}, y_{T-3}, \mathbf{x}_{T-2}) ... \end{aligned} \tag{1.21}$$

As we can see, calculating $Z(\mathbf{x})$ in a naive way is exponentially expensive. However, it is worth noticing that in Equation (1.21) the intermediate sums are calculated several times during the computation of the outer sums. In this case a dynamic programming approach that keeps trace of the inner sums can save an exponential amount of computations. A way to exploit this situation is to introduce for each element of the sequence t, a forward vector α_t that keeps trace of the intermediate sums for each possible value of the output variables S. The forward vector α_t is calculated by a recursive function called *alpha recursion* that takes into consideration also the forward vector of the elements before t, i.e., $t-1, t-2, ..., 1$. The alpha recursion is defined as:

$$\alpha_t(j) = \sum_{i \in S} \Psi_t(j, i, \mathbf{x}_t) \alpha_{t-1}(i) \tag{1.22}$$

with the initialization vector equal to

$$\alpha_1(j) = \Psi_1(j, y_0, \mathbf{x}_1) \tag{1.23}$$

where y_0 is an initial variable with a fixed value.

In the same way, but starting the computation from the end of the sequence instead of the beginning, we define a backward vector β_t that, as the forward vector, keeps trace of the intermediate sums. Similarly to the forward vector, the backward vector is calculated with a recursive function called *beta recursion*. The beta recursion is defined as:

$$\beta_t(i) = \sum_{j \in S} \Psi_{t+1}(j, i, \mathbf{x}_{t+1}) \beta_{t+1}(j) \tag{1.24}$$

Analogously to the alpha recursion, the initialization vector is:

$$\beta_T(i) = 1 \tag{1.25}$$

The computation of $Z(\mathbf{x})$ adopting the alpha recursion is:

$$Z(\mathbf{x}) = \sum_{y_T} \alpha_T(y_T) \tag{1.26}$$

while adopting the beta recursion is:

$$Z(\mathbf{x}) = \sum_{y_1} \Psi_1(y_1, y_0, \mathbf{x}_1)\beta_1(y_1) = \beta_0(y_0) \tag{1.27}$$

The two recursions are also used to calculate the marginal distributions $p(y_t|\mathbf{x})$ and $p(y_t, y_{t-1}|\mathbf{x})$ that, as we will see in Section 1.2.3, are necessary in the parameter learning phase. Both the marginal distributions $p(y_t|\mathbf{x})$ and $p(y_t, y_{t-1}|\mathbf{x})$ are given by the combination of the results of the alpha and beta recursions as:

$$p(y_t|\mathbf{x}) = \frac{1}{Z(\mathbf{x})}\alpha_t(y_t)\beta_t(y_t) \tag{1.28}$$

$$p(y_t, y_{t-1}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})}\alpha_{t-1}(y_{t-1})\Psi_t(y_t, y_{t-1}, \mathbf{x})\beta_t(y_t) \tag{1.29}$$

Putting everything together, the forward-backward algorithm consists of calculating the forward vectors (Equations (1.22) and (1.23)), computing the backward vectors (Equations (1.24) and (1.25)) calculating $Z(\mathbf{x})$ and then applying Equation (1.28) and Equation (1.29) in order to obtain the marginals.

Viterbi Algorithm

The other important inference task in CRFs is what is called *maximum-a-posteriori (MAP) assignment*. This inference task is performed in the context of LC-CRF by the *max-product algorithm*, also called *Viterbi algorithm* [32].

The Viterbi algorithm substitutes the sum from the alpha recursion in Equation (1.22) with a maximization; this modification yields the Viterbi recursion:

$$\delta_t(j) = \max_{i \in S} \Psi_t(j, i, \mathbf{x}_t)\delta_{t-1}(i) \tag{1.30}$$

In order to obtain the maximum assignment, after the computation of the δ variables we need to compute the backward recursion given by:

$$y_T^* = \arg\max_{i \in S} \delta_T(i) \tag{1.31}$$

$$y_t^* = \arg\max_{i \in S} \Psi_t(y_{t+1}^*, i, \mathbf{x}_{t+1})\delta_t(i) \tag{1.32}$$

The two recursions form the Viterbi algorithm are used in prediction phase to obtain the actual predictions of unobserved data.

1.2.3 Parameter Learning in Linear-Chain CRFs

Parameter learning in CRFs is the task of estimating the vector parameter θ so that the resulting conditional distribution $p(\mathbf{y}|\mathbf{x}, \theta)$ fits, in the best possible way, the data examples $\mathcal{D} = \{\mathbf{x}^{(n)}, \mathbf{y}^{(n)}\}_{n=1}^{N}$, which have both input and output variables observed. What one would like to obtain with parameter learning is that, having observed the input variables of one of the examples in $\mathcal{D}$, the distribution $p(\mathbf{y}|\mathbf{x}^{(n)}, \theta)$ obtained after parameter learning is close to the real distribution $p(\mathbf{y}^{(n)}|\mathbf{x}^{(n)})$ with the true output $\mathbf{y}^{(n)}$.

The standard way to achieve this in CRFs is estimating the parameters θ that maximize the *conditional log-likelihood*:

$$\ell(\theta) = \sum_{n=1}^{N} \log p(\mathbf{y}^{(n)}|\mathbf{x}^{(n)}; \theta) \tag{1.33}$$

Expanding Equation (1.33), by adopting Equation (1.18) for the sake of simplicity, and thus using a feature function f_k that ranges over both the feature functions of Equations (1.14) and (1.15) with associated parameter θ_k, we obtain:

$$\ell(\theta) = \sum_{n=1}^{N} \sum_{t=1}^{T} \sum_{k=1}^{K} \theta_k f_k(y_t^{(n)}, y_{t-1}^{(n)}, \mathbf{x}_t^{(n)},) - \sum_{n=1}^{N} \log Z(\mathbf{x}^{(n)}) \tag{1.34}$$

It could happen that, in order to model a problem, a large number of parameters can be necessary. In order to avoid overfitting, (i.e., a model that is too specific on the training data and fails to generalize to other unseen data), *regularization* is often used. The regularized log-likelihood is:

$$\ell(\theta) = \sum_{n=1}^{N} \sum_{t=1}^{T} \sum_{k=1}^{K} \theta_k f_k(y_t^{(n)}, y_{t-1}^{(n)}, \mathbf{x}_t^{(n)}) - \sum_{n=1}^{N} \log Z(\mathbf{x}^{(n)}) - \sum_{k=1}^{K} \frac{\theta_k^2}{2\sigma^2} \tag{1.35}$$

where σ is a regularization parameter that determines how much θ is penalized in order to avoid overfitting.

In order to maximize $\ell(\theta)$ a numerical optimization must be adopted, given that it cannot be maximized in closed form. The partial derivative of Equation (1.35) to be given as input to a numerical optimization algorithm is:

$$\frac{\partial \ell}{\partial \theta_k} = \sum_{n=1}^{N} \sum_{t=1}^{T} f_k(y_t^{(n)}, y_{t-1}^{(n)}, \mathbf{x}_t^{(n)}) - \sum_{n=1}^{N} \sum_{t=1}^{T} \sum_{y,y'} f_k(y, y', \mathbf{x}_t^{(n)}) p(y, y'|\mathbf{x}^{(n)}) - \frac{\theta_k}{\sigma^2} \tag{1.36}$$

The meaning of Equation (1.36), that can be considered the loss function that we want to be as close as possible to zero, is straightforward. The first term represents the value of the feature functions f_k obtained from the training data. The second term represents the expected value obtained from the model. The expectation of

the feature function f_k given the model, is obtained by calculating the marginal probability $p(y, y'|\mathbf{x}^{(n)})$ with the forward-backward algorithm of Section 1.2.2. In an unregularized scenario the gradient would be zero when the two terms have the same value, that is, when the model perfectly fits the data. In this borderline situation the trained model would probably suffer from overfitting; the regularization term will help preventing such an undesirable situation.

Finally, the optimization of $\ell(\theta)$ can be done with several well known optimization algorithms, such as *steepest gradient ascent*, *Newton's method* [7], *conjugate gradient* [166] and *limited-memory BFGS* [14]. It is worth noticing that in the case of LC-CRF the function $\ell(\theta)$ is concave, which means that every local optimum is a global optimum.

1.3 Inference in General CRFs

Getting out from the comfort zone of the LC-CRFs, inference starts to be a little bit more problematic. In Section 1.2.2 we described inference algorithms such as the forward-backward and the Viterbi algorithm, that belong to the class of *exact inference* algorithms. The forward-backward algorithm and the Viterbi algorithm are a specialization, for chain-structured graphs, of two general algorithms of the family of *belief propagation* algorithms [63] for tree-structured graphs, the *sum-product* algorithm, and the *max-product* algorithm. In fact, exact inference is possible and relatively easy in GMs, and in particular in CRFs, only when the CRF graph structure is a tree, i.e., a chain is a particular type of tree. When the CRF graph structure is more complex than a tree, that is, a graph with cycles, then we have two choices: the first is using the junction tree algorithm, the second is relying on approximate inference algorithms. The *junction tree* algorithm [61] creates, from a graph with cycles, an equivalent tree-shaped graph. This transformation is done removing the cycles from the original graph and grouping variables into clusters. Once the structure of the original graph is transformed into a more manageable shape, exact inference algorithms are applied to get the marginal probabilities. The problem with the junction tree algorithm is that for complex graphs the creation of a tree is exponentially complex in the worst case, i.e. is often not applicable, leaving the approximate inference as the only suitable route.

The most important families of *approximate inference* algorithms for GMs are the *Markov-Chain Monte Carlo* (MCMC) algorithms [106], and the *variational* algorithms [157]. The MCMC algorithms belong to the broader family of *sampling* algorithms. These algorithms generate samples from the distribution of interest, that in the case of CRFs is the conditional distribution $p(\mathbf{y}|\mathbf{x})$, in order to approximate the marginal distributions of the type $p(y_t|\mathbf{x})$. The most popular sampling algorithms for inference in graphical models are *Gibbs sampling* and *Metropolis-Hasting sampling* [8]. They have been successfully adopted to perform approximate inference in CRFs [45, 124, 125].

Variational algorithms convert the inference problem into an optimization problem. This family includes the *loopy belief propagation* algorithms [90], that are the approximate counterpart of the belief propagation algorithms where the sum-product and max-product algorithms are applied regardless the presence of cycles in the graph. The algorithms are usually stopped when a stopping condition is reached or after a certain number of epochs [103, 135].

1.4 Summary

In this chapter we introduced the theoretical basis and the notation of the framework we will use in the rest of the thesis, namely, the CRFs. We first introduced the broader family of probabilistic graphical models, distinguishing between directed and undirected methods. As special case of undirected models we described CRFs in general and its most popular variation, linear-chain CRF, with the main inference techniques necessary to perform parameter learning and prediction. Finally, we briefly explained the main inference techniques in CRFs with a general graph structure.

2

Information Extraction from Medical Reports

In this chapter we adopt CRFs as the learning algorithm to extract information from texts written in natural language. In particular, we discuss the problem of performing information extraction from free-text radiology reports via several types of CRFs. In this task, segments of text (not necessarily coinciding with entire sentences, and possibly crossing sentence boundaries) need to be annotated with tags representing concepts of interest in the radiological domain. We present two novel approaches to IE for radiology reports: (i) a cascaded, two-stage method based on pipelining two taggers generated via the well known linear-chain conditional random fields (LC-CRFs) learner, and (ii) a confidence-weighted ensemble method that combines standard LC-CRFs and the proposed two-stage method. We compare these two novel approaches against traditional LC-CRFs. We also report on the use of "positional features", a novel type of feature intended to aid in the automatic annotation of texts in which the instances of a given concept may be hypothesized to systematically occur in specific areas of the text. We present experiments on a dataset of mammography reports in which the proposed ensemble is shown to outperform a traditional, single-stage CRFs system in two different, applicatively interesting scenarios.

2.1 Introduction

Information Extraction (IE – sometimes also referred to as *concept extraction* [10, 25, 55, 149]) is the discipline concerned with the extraction of natural language expressions (and relations between them)[1] from free text, where these expressions instantiate concepts of interest in a given domain (see e.g., [78, 111]).

For instance, given a corpus of job announcements, one may want to extract from (or: "identify in", "annotate in") each announcement the natural language expressions that describe the nature of the job, the promised annual salary, the

[1] As pointed out in Section I.1, in this dissertation we use the term information extraction (IE) referring to the sequence labeling subtasks of information extraction.

job location, etc. Another instance of IE is searching free text for named entities, i.e., names of persons, locations, geopolitical organizations, and the like [92]. Put another way, IE may be seen as the activity of populating a structured information repository (such as a relational database of job announcements, where individual announcements are tuples and "job", "annual salary", "job location" are attributes) from an unstructured information source such as a corpus of free text.

An application of great interest is extracting information from free-text clinical narratives, such as medical reports, admission / discharge summaries, and clinical notes in general. These narratives are unstructured in nature, since they are written in free text by medical personnel, and because of their informal nature are particularly difficult to handle automatically. Medical reports and clinical narratives are characterized by informal language, and are usually fraught with abbreviations (sometimes idiosyncratic to the specific hospital or department where they originated), ungrammatical language, acronyms (even non-standard ones), and typos; this is due to the fact that clinical narratives are often the result of hasty compilation, or dictation, or the application of speech recognition technology. As a result, the correct identification of expressions (e.g., named entities, phrases, sentences, or other) that instantiate concepts of interest (such as drugs, drug dosages, pathologies, treatments, and the like) is more difficult than, say, in the domain of medical literature (e.g., books, abstracts, scientific articles), where language is usually tighter.

Nonetheless, performing information extraction on these narratives would be extremely beneficial, since extracting key data and converting them into a structured (e.g., tabular) format would greatly contribute towards endowing patients with truly actionable electronic medical records. These records, aside from improving interoperability among different medical information systems, could be used in a variety of applications, from patient care, to decision support, to epidemiology and clinical studies. Of particular interest is the fact that automatically spotting personal information in narratives may be used for their automatic de-identification (i.e., anonymization) [154], an important task given the confidential nature of clinical narratives.

This chapter describes our efforts towards automatically extracting information from radiology reports.

Nowadays, the dominant machine learning method for IE from clinical narratives is *conditional random fields* [64, 133, 134], a method explicitly devised for *sequence learning*, i.e., for learning to classify items that naturally occur in sequences (the words that make up the clinical narratives obviously have this property). Most authors that have participated in the recent i2b2 challenges devoted to IE from clinical narratives (see [155, 156]) have indeed used CRFs [49, 53, 99, 149], often in their simple "linear chain" form (LC-CRFs). In this chapter we present two novel approaches to using LC-CRFs for clinical text IE: (i) a cascaded, two-stage method based on pipelining two LC-CRFs systems, one that analyses text at the clause level and a second one that analyses it at the token (word) level, and (ii) a confidence-weighted ensemble method that combines a standard, token-level LC-CRFs and the

proposed cascaded method.

When input to a CRFs learner, each token (i.e., word instance) occurring in the narrative must be represented as a vector of features. Typical features that are used in CRFs for clinical IE are the word itself, its morphological root (stem), its part of speech, its prefixes and suffixes, and other information that can directly be extracted from the text; the features of the surrounding tokens are also usually added to the vector representing a token. If specialized lexical resources are available (e.g., the UMLS metathesaurus), entries from these resources can also be usefully added to the representation of a token whenever the token, or a larger sequence of tokens that includes it, is recognized as standing in certain relationships with those entries. In this chapter we report on the additional use of "positional features", a novel type of features intended to aid in the automatic annotation of texts in which the instances of a given concept tend to systematically occur in specific areas of the text (e.g., towards the beginning of the text, towards the middle of the text, etc.).

We experimentally validate our newly introduced methods on a dataset of 500 mammography reports written in Italian. Two applicatively interesting scenarios are tested. The first scenario simulates the existence of a single human annotator who provides the training data and whose classification decisions are to be taken as the gold standard. The second scenario simulates the existence of multiple human annotators; a mix of training examples from each annotator are thus used, and a mix of the classification decisions of each of them is used as the gold standard. This also represents a novelty in the literature on clinical IE, since this literature has never (to the best of our knowledge) taken up the issue of distinguishing, and separately addressing, these two applicative scenarios. The only work that propose a similar scenario is [73] in which, in a text classification task, an automatic classification system is compared with two different human annotators in order to assess the system accuracy against the naturally varying performance of two independent human annotators.

The results of our experiments show that our two proposed LC-CRFs systems outperform, in both scenarios, a baseline system consisting of linear-chain CRFs, and show that in most cases the performance of our systems even exceeds human performance, as measured by inter-annotator agreement. The results obtained from the use of positional features are instead less conclusive.

The rest of the chapter is organized as follows. In Section 2.2.3 we describe the LC-CRFs system and the set of features that constitute our baseline. In Section 2.2.4 we turn to describing our proposed novel methods: the cascaded LC-CRFs system (Section 2.2.3), the ensemble of LC-CRFs learners (Section 2.2.4), and the positional features (Section 2.2.5). Section 2.3 is devoted to describing our experiments and commenting on their results. Section 2.4 discusses related work, while Section 2.5 concludes, pointing at avenues for future research.

2.2 CRF Models and Algorithms

2.2.1 Preliminaries and Notation

Let $\mathbf{X}$ be a set of texts. Let a text $\mathbf{x} \in \mathbf{X}$ consist of a vector $\mathbf{x} = \langle x_1, \ldots, x_{|\mathbf{x}|} \rangle$, where each x_t is a *token* (i.e., a word occurrence), $|\mathbf{x}|$ denotes the dimensionality of vector $\mathbf{x}$ (in this case: the length of the text), and x_{t_1} occurs before x_{t_2} in the text (noted $x_{t_1} \preceq x_{t_2}$) if and only if $t_1 \leq t_2$.

Let $C = \{c_1, \ldots, c_m\}$ be a predefined set of *tags* (a.k.a. *concepts*, or *markables*), or *tagset*. We will take *information extraction* (IE) to be the task of determining, for each $\mathbf{x} \in \mathbf{X}$ and for each $c_r \in C$, a vector $\mathbf{y}_r = \langle y_{r1}, \ldots, y_{r|\mathbf{x}|} \rangle$ of labels $y_{rt} \in \{c_r, \overline{c}_r\}$, which indicates which tokens in the text are labeled with tag c_r. Since each $c_r \in C$ is dealt independently of the other tags in C, we will hereafter drop the r subscript and treat IE as the *binary* task of determining, given text $\mathbf{x}$ and given tag c, a vector $\mathbf{y} = \langle y_1, \ldots, y_{|\mathbf{x}|} \rangle$ of labels $y_t \in \{c, \overline{c}\}$.

We assume each token x_t to be itself represented by a vector $\mathbf{x}_t$ of $\Omega = |\mathbf{x}_t|$ features.

Tokens labeled with a tag c usually come in coherent sequences, or "segments". Hereafter, a *segment* ζ of text $\mathbf{x}$ for tag c will be a pair (x_{t_1}, x_{t_2}) consisting of a start token x_{t_1} and an end token x_{t_2} such that (i) $x_{t_1} \preceq x_{t_2}$, (ii) all tokens $x_{t_1} \preceq x_t \preceq x_{t_2}$ are labeled with tag c, and (iii) the token that immediately precedes x_{t_1} and the one that immediately follows x_{t_2} are *not* tagged with tag c. For example, in the excerpt

> `Da ambo i lati si apprezzano` <PAE>`formazioni nodulari solide` </PAE>`, a margini sostanzialmente regolari.`

the markers <PAE>and </PAE>are meant to indicate that all the tokens between them are to be understood as tagged with the PresenzaAssenzaEnhancement (PAE) tag. In this case, (`formazioni`, `solide`) is a segment for the PAE tag. In general, a text $\mathbf{x}$ may contain zero, one, or several segments for tag c.

2.2.2 A Baseline CRFs System for IE

As a baseline learning algorithm we have used *linear-chain conditional random fields* (LC-CRFs - [64, 133, 134]), in Charles Sutton's GRMM implementation[2].

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{t=1}^{|\mathbf{x}|} \Psi_u(y_t, \mathbf{x}_t) \prod_{t=1}^{|\mathbf{x}|-1} \Psi_{rb}(y_t, y_{t+1}, \mathbf{x}_t) \tag{2.1}$$

Differently from the LC-CRF introduced in Equation (1.11) the structure of the model adopted in this work is slightly different. In fact, the CRFs in Equation (2.1) employ a slightly different factor template respect to the model in Equation (1.11). Here

[2] `http://mallet.cs.umass.edu/grmm/`

we use a bigram factor, that we called *enriched bigram*, that take into consideration not only the co-occurrence of two adjacent output variables y_t and y_{t+1}, but also the input variables $\mathbf{x}_t$ Formally, the factors belonging to the *enriched bigram* factor template are defined as:

$$\Psi_{rb}(y_t, y_{t+1}, \mathbf{x}_t) = \exp \sum_{hk} \theta_{hk} f_{hk}(y_t, y_{t+1}\mathbf{x}_t) \tag{2.2}$$

with a notation lighter than Equations (1.12) and (1.13) the index hk represents, all the possible value of the quadruple (i, j, d, v) as defined in Section 1.2.1.
In this work we have used a base set of features which includes:

1. one feature representing the word of which the token is an instance;
2. one feature representing its stem;
3. one feature representing its part of speech;
4. eight features representing its prefixes and suffixes (the first and the last n characters of the token, with $n = 1, 2, 3, 4$);
5. one feature representing information on token capitalization, i.e., whether the token is all uppercase, all lowercase, first letter uppercase, or mixed case.

Vector $\mathbf{x}_t$ includes the above information for each of x_{t-1}, x_t and x_{t+1}, for a total of 36 features.

All of these features are fairly standard in several instances of the information extraction task, including e.g., named-entity recognition. The IE system we will use as a baseline in the experiments of Section 2.3 will thus consists of the LC-CRFs learning system using as input the text represented via the features discussed in the present section.

Figure 2.1 is the factor graph that represent the CRF of Equation (2.1).

2.2.3 A Two-Stage CRFs System for IE

In medical reports it is often the case that the segments of interest are not short pieces of text scattered throughout the sentence, but span several clauses (or even several sentences). A given tag may be expected to be instantiated by an expression that spans several clauses, maybe cutting across sentence borders. This suggests the adoption of a cascaded, two-stage tagging scheme, in which

1. the first stage consists of tagging entire clauses; this acts as a coarse-grained filter, with the goal of removing from consideration clauses that are evidently irrelevant to the tag of interest;

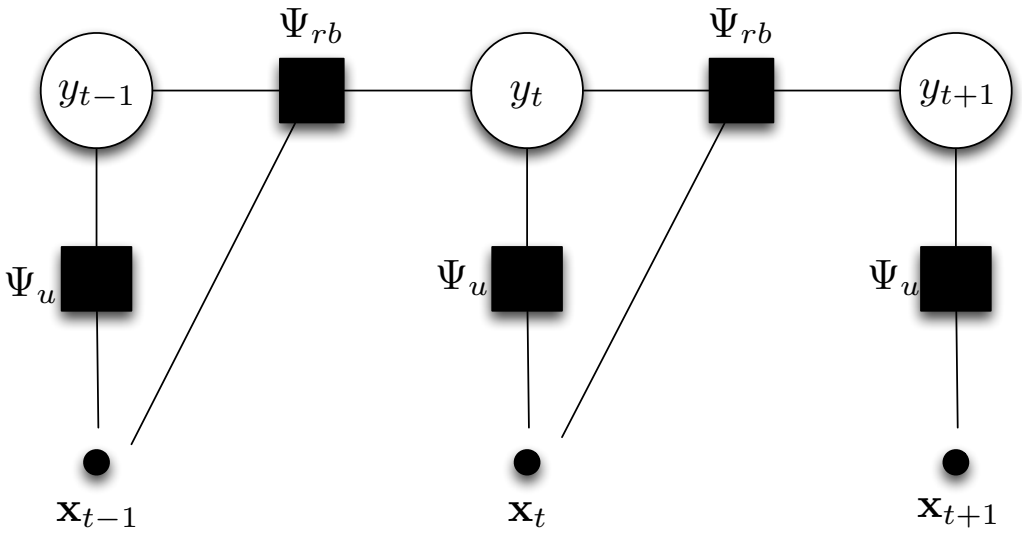

Figure 2.1: Baseline linear-chain CRF

2. the second stage consists of tagging individual tokens belonging to the clauses that have been attributed the tag in the first stage; this acts as a fine-grained tagger, with the goal of examining in detail the clauses that the previous phase has deemed of potential interest.

For the purpose of this chapter we heuristically define a *clause* as a set of tokens delimited both to the right and to the left by a punctuation symbol in the set {comma, period, colon, semicolon, question mark, exclamation mark}, and such that no punctuation symbol from this set appears inside it. For example, in the excerpt

> `Da ambo i lati si apprezzano` <PAE>`formazioni nodulari solide` </PAE>`, a margini sostanzialmente regolari.`

already quoted in Section 2.2.1, the clauses are `Da ambo i lati si apprezzano formazioni nodulari solide` and `a margini sostanzialmente regolari`.

As for the first stage, we implement it via a LC-CRFs tagger that, unlike the one described in Section 2.2.2, has the formulation

$$p(\mathbf{y}|\mathbf{s}) = \frac{1}{Z(\mathbf{s})} \prod_{d=1}^{D} \Psi_u(y_d, \mathbf{s}_d) \prod_{d=1}^{D-1} \Psi_b(y_d, y_{d+1}, \mathbf{s}_d) \tag{2.3}$$

Here, the sequence of tokens $\mathbf{x} = \langle x_1, \ldots, x_{|\mathbf{x}|} \rangle$ has been replaced with a sequence of clauses $\mathbf{s} = \langle s_1, \ldots, s_{|\mathbf{s}|} \rangle$, represented by feature vectors $\mathbf{s}_1, \ldots, \mathbf{s}_{|\mathbf{s}|}$, with all the equations of Section 2.2.2 rewritten accordingly.

We see the first stage as a sort of *text classification* phase, where entire texts (in this case: clauses) need to be classified, using the tags in our tagset as the classes. Accordingly, we opt for a "bag of words" (and word bigrams) representation, thus using as only features the tokens in the clause, and their bigrams. As a consequence, the $\mathbf{s}_t$ feature vector is the traditional sparse vector of text classification, whose dimensionality is the number of unique words and bigrams that occur at least once in the training set, and where 0 and 1 represent absence and presence of the word or bigram in the clause. However, note that we are not classifying a clause in isolation of the clauses that precede and follow it. The very fact that we are using a LC-CRFs approach, and that this involves the factors of the kind of Equation (2.2), means that the tag that will be attributed to a given clause will also depend on the tag that is probabilistically attributed to the clauses around it.

Quite obviously, we also implement the second stage via the standard LC-CRFs system described in Section 2.2.3, with the difference that clauses, instead of entire texts, are the object of tagging. Only the clauses that have passed through the filter of the first-stage system are tagged (at the token level). The evaluation of the cascaded system is performed by considering the token-level representation of the clauses discarded in the first stage as labeled with tag $\overline{c}$.

2.2.4 A Confidence-Weighted Ensemble of CRFs systems for IE

The two-stage CRFs described in Section 2.2.3 trades precision for recall. In fact, it can be expected to have better precision than the single-stage system, since all the clauses that are ruled out from consideration in the first stage may contain tokens that are erroneously attributed the tag by the single-stage system; by removing these clauses from consideration, a number of potential false positives become true negatives, thereby improving precision. However, by the same argument, the cascaded, two-stage system may be expected to have worse recall then the single-stage system, since the ruled-out clauses may contain tokens to which the single-stage system would have *correctly* attributed the tag; this generates false negatives that would otherwise have been true positives.

In sum, better precision but worse recall may be expected from going two-stage. Whether the two-stage system is better than the single-stage one thus comes down to understanding whether, by which amount, and for which tags, the improvement in precision outweigh the deterioration in recall.

One may want to try to obtain the best of both worlds by having a committee (or ensemble) of two taggers (a single-stage one and a two-stage one), where the final decision is also based on how confident each of the two taggers is in tagging a given token.

More specifically, let us note that the Viterbi algorithm used in both the single-stage and the cascaded, two-stage systems for maximizing $p(\mathbf{y}|\mathbf{x})$, also maximizes,

as an intermediate result, the conditional probabilities $p(y_t|\mathbf{x})$ of the individual tokens x_t. Let $p_{ss}(y_t|\mathbf{x})$ and $p_{ts}(y_t|\mathbf{x})$ denote these probabilities as maximized by the single-stage and the two-stage methods, respectively. We define the conditional probability $p_{en}(y_t|\mathbf{x})$ of the individual token x_t as maximized by our ensemble, as the average of the conditional probabilities computed by the single-stage and the two-stage methods, i.e.,

$$p_{en}(y_t|\mathbf{x}) = \frac{1}{2}(p_{ss}(y_t|\mathbf{x}) + p_{ts}(y_t|\mathbf{x})) \tag{2.4}$$

Since the conditional probability $p(y_t|\mathbf{x})$ may obviously be interpreted as the system's confidence in the fact that the label of $\mathbf{x}_t$ is y_t, Equation (2.4) formalizes a *confidence-weighted ensemble* [107]. Note that $p_{ts}(y_t|\mathbf{x})$ is obtained by the cascaded, two-stage tagger in the second stage whenever the clause that contains x_t has not been ruled out in the first stage. Otherwise, $p_{ts}(y_t|\mathbf{x})$ is in fact the probability that the first-stage tagger has attributed y_t to the entire clause which contains x_t, since all of the tokens in that clause obtain the same probability.

Therefore, our ensemble-based method consists of computing

$$\mathbf{y}_t^* = \arg\max_{y_t} p_{en}(y_t|\mathbf{x}) \tag{2.5}$$

for each token x_t.

2.2.5 Positional Features

As for representing tokens x_t via feature vectors $\mathbf{x}_t$, we think that there are margins of improvement over the choices discussed in Section 2.2.2. In particular, we observe that medical reports are often written according to a fairly standard pattern, according to which some tags are instantiated earlier on in the report, while some others are instantiated later. For instance, it is reasonable to suppose that clinical observations are presented at the beginning of the report, while a diagnosis and a prognosis are discussed later on.

In order to capture these recurring patterns, we introduce *positional features* that model the position of a token in the text. For instance, we might want to introduce a 4-ary feature that indicates whether token x_t occurs in the 1st quarter of the text, or in the 2nd, or in the 3rd, or in the 4th. In this way, if a given tag tends to be instantiated, say, in the 1st quarter of the report, this tendency will be detected during training, and will be brought to bear in the test phase, e.g., by using the fact that test token x_t occurs in the 1st quarter of the text as contributing evidence that x_t should be assigned the tag. In general, our positional features are k-ary features which, assuming that the text is divided into k consecutive, equal-sized parts, indicate in which of the k parts the token occurs.

In our experiments we use all positional features for $k = 2, 3, 4, 5$.

2.3 Experiments

2.3.1 Dataset

The dataset we have used to test the ideas presented in Section 2.2.4 (hereafter called the UmbertoI(RadRep) dataset) consists of a set of 500 free-text mammography reports written (in Italian) by medical personnel of the Istituto di Radiologia of Policlinico Umberto I, Roma, IT. The number of different radiologists who authored the reports is not known. The length of the reports ranges between 67 and 537 words; the mean and the variance of such length is 199 and 8786, respectively.

The reports have been subsequently annotated by two equally expert radiologists from the same institute; 191 reports have been annotated by annotator 1 (A1) only, 190 reports have been annotated by annotator 2 (A2) only, and 119 reports have been annotated independently by A1 and A2. From now on we will call these sets `A1-only`, `A2-only` and `Both`, respectively; `Both(1)` will identify the `Both` set as annotated by A1, and `Both(2)` will identify the `Both` set as annotated by A2. The annotation activity was preceded by an alignment phase, in which A1 and A2 jointly annotated 20 reports (not included in this dataset) in order to align their understanding of the meaning of the tags.

Table 2.1: Per-tag statistics for the UmbertoI(RadRep) dataset. Column 2 and Column 3 indicate the number $NT(i)$ of tokens and the number $NS(i)$ of segments contained in the dataset for the given tag, Column 4 the number $ND(i)$ of documents with at least one segment for the given tag, Column 5 the average number $ANS(i)$ of segments per document for the given tag, Column 6 the average segment length $ASL(i)$ for the given tag (segment length is the number of tokens contained in it).

	$NT(i)$	$NS(i)$	$ND(i)$	$ANS(i)$	$ASL(i)$
BI-RADS (BIR)	1544	400	293	0.65	3.86
InformazioniTecniche (ITE)	24301	615	614	0.99	39.51
IndicazioniEsame (IES)	5159	475	469	0.77	10.86
TerapieFollowup (TFU)	7137	523	458	0.84	13.65
DescrizioneEnhancement (DEE)	19795	803	440	1.30	24.65
PresenzaAssenzaEnhancement (PAE)	8461	588	439	1.94	7.04
EsitiChirurgici (ECH)	2068	230	203	0.37	8.99
DescrizioneProtesi (DEP)	2532	72	61	0.12	35.17
LinfonodiLocoregionali (LLO)	6602	537	509	0.87	12.29
Average	**8622**	**471**	**387**	**0.87**	**17.33**

The tagset is formed by 9 tags: BI-RADS (hereafter shortened as BIR)[3], In-

[3]BI-RADS indicates the mammography assessment categories from the Breast Imaging-Reporting

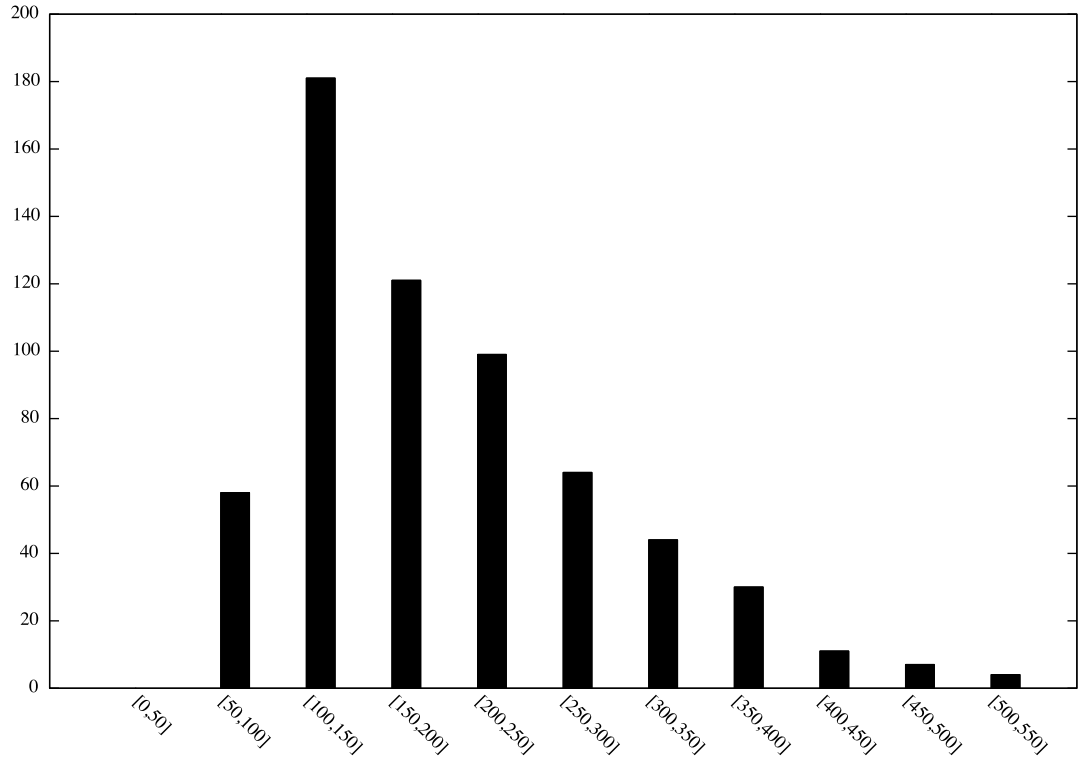

Figure 2.2: Distribution of report lengths: the X axis represents ranges of lengths, while the Y axis represents the number of reports whose length falls in the given range.

formazioniTecniche (ITE – "Technical Info"), IndicazioniEsame (IES – "Indications obtained from the Exam"), TerapieFollowup (TFU – "Followup Therapies"), DescrizioneEnhancement (DEE – "Description of Enhancement"), PresenzaAssenzaEnhancement (PAE – "Presence/Absence of Enhancements"), EsitiChirurgici (ECH - "Outcomes of Surgery"), DescrizioneProtesi (DEP – "Prosthesis Description"), and LinfonodiLocoregionali (LLO – "Locoregional Lymph Nodes"). On average, there are 0.87 segments for each tag in a given report, and the average segment length is 17.33 words. Table 2.1 reports additional statistics on the dataset; Figure 2.2 illustrates, in histogram form, the variability in report length across the dataset.

A closer inspection of this dataset reveals that

- it is not the case that for each tag $c_r \in C$ there is at least one segment ζ_{rj} in each report. For example, the only tags that are instantiated in Report 114

and Data System published by the American College of Radiology (ACR). These are standardized numerical codes typically assigned by a radiologist after interpreting a mammogram, which allow for concise and unambiguous understanding of patient records among doctors.

are BIR, PAE, ITE, and TFU;

- segments for different tags may overlap, i.e., the same token x_t may belong to two or more segments each pertaining to a different tag. For example, in Report 452 all the tokens in the sequence `in paziente con pregressa QUART sinistra mastoplastica additiva bilaterale` are tagged with both tags ECH and IES;

- two or more segments for the same tag c_r may be present in the same report. For instance, in Report 180 the two non-overlapping segments `un'area di potenziamento dopo contrasto` and `alcuni millimetrici foci di potenziamento` are both tagged with tag PAE;

- the "order of appearance" of the tags in a report is not fixed, i.e., given tags c' and c'' it is *not* the case that segments for tag c' always precede the segments for tag c''. For example, in Report 452 tag PAE appears after tag DEE, while in Report 180 it appears before it.

2.3.2 Evaluation Measures

As the evaluation measure we use the token-level variant (proposed in [28]) of the well-known F_1 measure, according to which a tagger is evaluated on an event space consisting of all tokens in the text. In other words, each token x_t (rather than each segment, as in the traditional "segmentation F-score" model [136]) counts as a true positive, true negative, false positive, or false negative for a given tag c_r, depending on whether x_t belongs to c_r or not in the predicted annotation and in the true annotation. As argued in [28], this model has the advantage that it credits a system for partial success (i.e., non-null overlap between a predicted segment and a true segment for the same tag), and that it penalizes both overtagging and undertagging.

As is well-known, F_1 combines the contributions of *precision* (π) and *recall* (ρ), and is defined as

$$F_1 = \frac{2\pi\rho}{\pi+\rho} = \frac{2TP}{2TP+FP+FN}$$

where TP, FP, and FN stand for the numbers of true positives, false positives, and false negatives, respectively. Note that F_1 is undefined when $TP = FP = FN = 0$; in this case we take F_1 to equal 1, since the tagger has correctly tagged all tokens as negative.

We compute F_1 across the entire test set, i.e., we generate a single contingency table by putting together all tokens in the test set, irrespectively of the text they belong to. We then compute both *microaveraged* F_1 (denoted by F_1^{μ}) and *macroaveraged* F_1 (F_1^{M}). F_1^{μ} is obtained by (i) computing the tag-specific values TP_r, FP_r and FN_r, (ii) obtaining TP as the sum of the TP_r's (same for FP and FN), and

then (iii) applying the $F_1 = \frac{2TP}{2TP + FP + FN}$ formula. F_1^M is obtained by first computing the tag-specific F_1 values and then averaging them across the c_r's.

An advantage of using F_1 as the evaluation measure is that it is symmetric, i.e., its values do not change if one switches the roles of the human annotator (i.e., the gold standard) and the automatic annotator (i.e., the system). This means that F_1 can also be used as a measure of agreement between any two annotators, regardless of whether they are human or machine, since it does not require one to specify who among the two is the gold standard against which the other needs to be checked. For this reason, in the following section we will use F_1 both (a) to measure the agreement between our system and a human annotator, *and* (b) to measure the agreement between two human annotators.

2.3.3 Experimental Setting

We have tested three different learning systems. The first is the LC-CRFs system described in Section 2.2.2, which we use as the baseline, while the second and the third are our cascaded, two-stage LC-CRFs system and our ensemble of taggers described in Section 2.2.3 and Section 2.2.4, respectively. In combination with each of the three learning systems we have also tested two different feature representations, one consisting of the baseline features described in Section 2.2.2, and one also including the positional features discussed in Section 2.2.5. Concerning these latter, note that in the first stage of the cascaded system (when used by itself and when used in the ensemble system) we have used positional features *for the clauses*, defined as the positional feature of their middle tokens.

We have optimized the regularization parameter σ of Equation (1.35) individually for both the baseline system of Section 2.2.2 and the two-stage system of Section 2.2.3, and individually for each experiment reported in this chapter. We have always carried out this optimization on a held-out validation set consisting of a randomly selected 10% of the original training set; after parameter optimization, the tagger has then been retrained on the entire training set. As a POS tagger we have used the one contained in the TextPro system [101].

We have run two main sets of experiments, each simulating a different operational scenario.

The first such set of experiments simulates a *single-annotator scenario*, i.e., an operational situation in which the organization has a single person in charge of annotating the reports. Simulating this scenario thus means using training and test documents annotated by the same person, since it is this person who would annotate the training data to be fed to a learning algorithm, and it is this person who would judge the accuracy of the automatically tagged data. We have thus run (i) an experiment in which we train the system on `A1-only` and test it on `Both(1)`, and (ii) an experiment in which we train the system on `A2-only` and test it on `Both(2)`. Since there is not much value in presenting the results of these two experiments

separately, in Table 2.2 we report accuracy figures obtained by averaging the results of the two experiments (or, equivalently, by merging, for each tag in the tagset, the two contingency tables obtained in the two experiments into a single contingency table and evaluating the results).

The second set of experiments simulates instead a *multiple-annotator scenario*, i.e., one in which there are several, equally trustworthy persons in the organization who are in charge of annotation. Simulating the latter scenario thus means using a mix of training documents from different annotators (since it is not a single person who would annotate the training data to be fed to the learning algorithm) *and* a mix of test documents from different annotators (since it is not according to the judgment of a single person that the accuracy of the automatically tagged data should be measured). We have thus run an experiment in which we train the system on the union of `A1-only` and `A2-only`, and test it on the union of `Both(1)` and `Both(2)`. Note that the union of `Both(1)` and `Both(2)` contains "contradictory information", since it contains two copies of each report in `Both`, and these two copies may contain annotations that contradict each other. This is very much in keeping with the fact that there are multiple, equally trustworthy annotators in the scenario we are simulating. The only practical consequence is that, in the presence of even the slight disagreement between `Both(1)` and `Both(2)`, a value of $F_1 = 1$ is unattainable even in theory. However, this is not problematic for our experiments, and the value of inter-annotator agreement of A1 and A2 as measured on `Both` via F_1 may be assumed as the "reasonable" accuracy upper bound that an automatic system cannot be expected to exceed. Table 2.3 reports the results obtained in this experiment.

Note that the two experiments are different in terms of both (i) training data *consistency* (since in the single-annotator experiments the annotations are all by the same person, hence are likely more consistent than in the multiple-annotator experiments), and (ii) training data *quantity* (since in the multiple-annotator experiment there are twice as many training data than in the single-annotator ones). This means that a comparison between the accuracy values obtained by a given system in these two sets of experiments will allow us to determine whether, for training data, consistency is a more important parameter than sheer quantity.

Note also that in both experiments the test documents are those in `Both`, i.e., those that have been annotated independently by both A1 and A2. In each such experiment we are thus in a position to compare the results of the experiments with the value of inter-annotator agreement obtained by A1 and A2 on the very same data. We are thus able to directly *compare human accuracy with machine accuracy* in a meaningful way, since, as argued in Section 2.3.2, F_1 can be used to measure the agreement between any two annotators, be them human or machine. More precisely,

- in the single-annotator experiment in which we train the system on `A1-only` and test it on `Both(1)`, comparing the results of the experiment with the value of inter-annotator agreement means comparing, using `Both(1)` as the common

testbed (i.e., using A1 as the judge), an automatic system trained on `A1-only` against human annotator A2 (analogously for the experiment in which we train on `A2-only` and test on `Both(2)`);

- in the multiple-annotator experiment in which we train the system on the union of `A1-only` and `A2-only`, and test it on the union of `Both(1)` and `Both(2)`, comparing the results of the experiment with the value of inter-annotator agreement means comparing, using the union of `Both(1)` and `Both(2)` as the common testbed (i.e., using a mix of A1 and A2 as the judges), an automatic system trained on the union of `A1-only` and `A2-only` against a mix of A1 and A2.

We have also tested the statistical significance of the improvements obtained by our proposed methods with respect to the baseline. Specifically, we have compared the F_1 values as measured on each report, for each tag and for each pair of methods we compare. Given that we can exactly pair the F_1 value for a given method to the corresponding F_1 value for the other method, we have used a paired t-test.

2.3.4 Results

Single-annotator results

The first observation we can make from Table 2.2 is that both the F_1^{μ} and F_1^{M} values obtained by all the tested machine-learned systems are clearly higher than the corresponding inter-annotator agreement values. This indicates that, in our single-annotator scenario experiments, all our systems have proven to be even more accurate than humans. The difference between human performance and system performance is actually quite varied across the nine tags. In general, it seems that the system has a more uniform performance across the tags than the humans, who instead perform very well on some tags (e.g., ITE and DEP) and very bad on others (e.g., BIR and ECH)[4].

The results of the systems that make use of positional features (indicated as "+ PFs" in Table 2.2) are not statistically significantly different from the analogous systems that make no use of such features. A more fine-grained analysis does not seem to reveal any discernible pattern concerning which tags benefit from positional features and which do not. In fact, it might seem plausible to hypothesize that the tags whose instances tend to be more heavily concentrated in a specific quantile of the

[4]The reason why there is such a large disagreement between A1 and A2 on BIR seems to be that A2, differently from A1, tends to always tag with BIR expressions relative to histological evaluation and expressions relative to benign diseases; this is likely an indication that the two annotators went through an insufficient alignment phase before starting the annotation. The high disagreement between the two annotators on the tag ECH can be instead due to the fact that it is a very infrequent tag, and it may be the case that too few instances of this tag were encountered in the alignment phase.

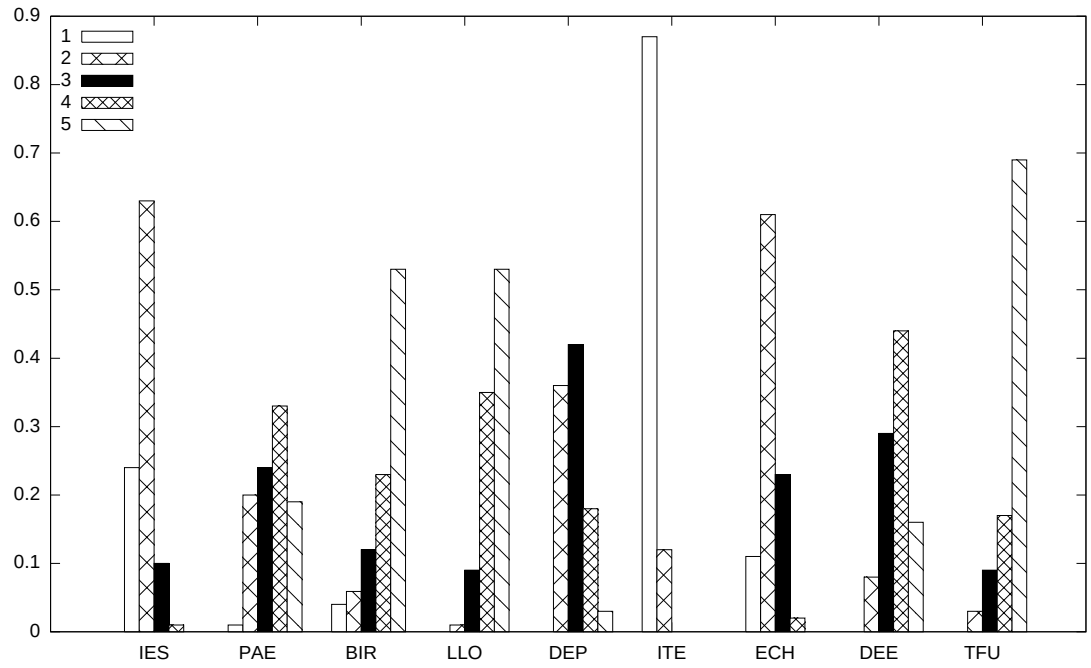

Figure 2.3: Histogram representing, for each of the nine tags, the percentage of segments for the tag that occurs in a given quintile of the text. E.g., the leftmost group of bars indicates the percentages of segments for each tag that occurs in the first 20% of the text.

text (such as, e.g., tag ITE, which is heavily concentrated in the 1st and 2nd quintiles – see Figure 2.3)) are the ones who benefit most from positional features. This hypothesis is not confirmed by the results. For instance, for the above-mentioned tag ITE, $F_1 = .987$ for both versions with and without positional features of the baseline system. Two tags such as IES and ECH which, as evident from the histogram, seem to be distributed in a similar way, obtain a very different contribution from positional features, with the former improving from $F_1 = .772$ to $F_1 = .810$ and the latter deteriorating from $F_1 = .667$ to $F_1 = .614$.

Whether positional features are used or not, the baseline system and the two-stage system bring about similar accuracy in terms of F_1^M, but the two-stage system clearly outperforms the baseline in terms of F_1^μ (with a statistical significance level of $p < 0.1$). This indicates that the two-stage configuration performs better than the baseline at annotating frequent tags, and worse than the baseline at annotating rare tags, since microaveraged metrics are heavily influenced by the performance of the system on frequent tags (macroaveraged effectiveness measures pay instead equal attention to each tag independently of frequency). A clear example is the tag DEP, which is the rarest of all tags in our dataset (see Table 2.1), and on which the two-stage system performs radically worse than the baseline system.

Table 2.2: Results of the single-annotator experiments. Results are averages across two experiments, (i) train the system on `A1-only` and test it on `Both(1)`, and (ii) train the system on `A2-only` and test it on `Both(2)`. The first row reports the inter-annotator agreement values for A1 and A2 as measured on `Both`.

	BIR	ITE	IES	TFU	DEE	PAE	ECH	DEP	LLO	F_1^μ	F_1^M
A1 vs. A2	0.492	0.993	0.778	0.815	0.660	0.694	0.574	0.912	0.884	0.812	0.756
Baseline	0.583	0.987	0.772	0.872	0.745	0.717	0.667	0.812	0.848	0.846	0.778
Two-Stage	0.582	0.993	0.792	0.845	0.746	0.694	0.666	0.772	0.852	0.873	0.771
Ensemble	0.601	0.991	0.814	0.873	0.772	0.730	0.679	0.823	0.853	0.858	0.793
Baseline + PFs	0.537	0.987	0.810	0.872	0.742	0.717	0.614	0.803	0.872	0.848	0.773
Two-Stage + PFs	0.480	0.991	0.812	0.825	0.736	0.691	0.664	0.890	0.853	0.870	0.771
Ensemble + PFs	0.551	0.991	0.821	0.865	0.775	0.719	0.673	0.881	0.861	0.859	0.793

Instead, the ensemble system obtains F_1^μ values close to the averages of the values obtained by the baseline and two-stage systems, and F_1^M values higher than those obtained by both systems. This indicates that the ensemble system is, on the whole, better than both its constituent systems, since it is better than the baseline when the tag is a frequent one, and it is much better than both systems on rare tags. In the configuration that does not use positional features, the ensemble system obtains $F_1^\mu = 0.858$ and $F_1^M = 0.793$, with an error reduction with respect to the baseline of 7.79% in terms of F_1^μ and of 6.75% on in terms of F_1^M. This improvement is confirmed to be significant by the statistical significance test with respect to both the baseline and the two-stage system (with $p < 0.005$ in both cases).

Concerning the difference among the performance obtained for the single tags, it is interesting to observe that the three tags for which the worst performance is obtained (BIR, ECH, and PAE) are also the ones (see Table 2.1) for which the segments have the smallest average length.(The same phenomenon will also be observed in the multiple-annotator experiments.) That long segments tend to be "easier" is explained by the fact that our evaluation measure is at the token level: since the difficult aspect in correctly tagging a segment is correctly spotting where the segment begins and ends, longer segments are evidently conducive to higher performance.

Multiple-annotator results

In the multiple-annotation scenario, the training set contains reports annotated by different annotators. Notwithstanding the alignment phase which the two annotators went through before starting their annotation work, each annotator may have a different annotation style, or a different understanding of the concepts which the tags represent. Therefore, similar segments may give rise to different tagging decisions by the two annotators. As a consequence, it is reasonable to assume that the resulting training set may possess a smaller internal consistency than in the single-annotator

scenario; other things (e.g., number of training examples) being equal, this yields a more difficult task for the learning system.

The results of the multiple-annotation experiments are displayed in Table 2.3. The first observation we can make is that values are generally lower (the relative deterioration being mostly around 2%) than the corresponding values in Table 2.2. This indicates that, in the tradeoff between training data consistency and training data quantity mentioned in the previous section, the former seems to be more important: notwithstanding the fact that twice as many training examples are used in the multiple-annotator experiments, performance decreases, due to the higher level of internal inconsistency of the training set. The topic of the quality of training data and the impact they have on the performance of an IE system will be thoroughly analyzed in the Chapter 4.

Concerning positional features, the multiple-annotator experiments confirm the results obtained in the single-annotator ones, since the use of positional features again does not bring about substantial differences. Significance tests confirms that the variations are not statistically significant. As in the single-annotator experiments, patterns are difficult to discern. Again, tags whose occurrence tends to be concentrated in a specific quantile of the texts do not seem to systematically benefit from the presence of positional features, and the very same tag (ECH) for which the positional features brought about a *deterioration* in F_1 from .667 to .614 in the single-annotation scenario, now even witnesses an *improvement* in F_1, from .618 to .645. The only conclusion one can draw is that more research needs to be done in order to assess how and whether positional features might benefit IE.

The improvement of the machine-learned systems with respect to the values of inter-annotator agreement are smaller in magnitude than those observed in the single-annotator scenario, but are still substantial. This indicates that the learning algorithms are good at mediating between the different, sometimes contradictory information received from the annotators at training time, thus producing automatic annotations that mediate between the annotation styles of the individual annotators.

Concerning the comparison among the three learning methods, the results confirm those of the single-annotator scenario, with the two-stage method obtaining higher F_1^μ and lower F_1^M than the baseline, and the ensemble method mediating between the F_1^μ values obtained by the other two systems and obtaining the best F_1^M value. The ensemble method obtained an F_1^μ value of .838 and an F_1^M value of .774, thus bringing about an error reduction with respect to the baseline of 9.50% for F_1^μ and of 6.61% for F_1^M.

Here too, we have tested the statistical significance of the observed improvements using the same method discussed for the single-annotator experiments. In this case the test shows that the improvement of the two-stage method over the baseline is statistically significant with $p < 0.05$. The observed improvement of the ensemble method is statistically significant with respect both to the baseline ($p < 0.01$) and the two-stage method ($p < 0.05$).

Table 2.3: Results of the multiple-annotator experiment: train the system on the union of `A1-only` and `A2-only`, and test it on the union of `Both(1)` and `Both(2)`. The first row reports the inter-annotator agreement values for A1 and A2 as measured on `Both`.

	BIR	ITE	IES	TFU	DEE	PAE	ECH	DEP	LLO	F_1^μ	F_1^M
A1 vs. A2	0.492	0.993	0.778	0.815	0.660	0.694	0.574	0.912	0.884	0.812	0.756
Baseline	0.537	0.994	0.768	0.866	0.710	0.688	0.618	0.775	0.863	0.821	0.758
Two-Stage	0.557	0.991	0.794	0.851	0.713	0.673	0.568	0.860	0.813	0.860	0.758
Ensemble	0.530	0.992	0.825	0.851	0.721	0.693	0.640	0.877	0.836	0.838	0.774
Baseline + PFs	0.516	0.983	0.794	0.873	0.698	0.679	0.645	0.843	0.861	0.828	0.766
Two-Stage + PFs	0.514	0.986	0.816	0.826	0.711	0.666	0.550	0.918	0.822	0.858	0.756
Ensemble + PFs	0.513	0.989	0.823	0.853	0.715	0.684	0.620	0.879	0.837	0.834	0.768

2.3.5 A note on IE in Resource-Scarce Languages

One of the main differences between the literature on information extraction from clinical texts (see Section 2.4) and our work, is that most of the former targets medical reports written in English, and as a consequence can leverage on the wide availability, for the English language, of lexical resources or "ontologies" specific to the domains of medicine or radiology. The present work may thus be taken as indicative of the level of extraction accuracy that can be attained for resource-scarce languages.

The availability of hierarchically organized domain-specific lexical resources is known to lead to higher accuracy in IE tasks. For instance, [49] reports obtaining a $F_1 = .800$ value on the i2b2 2010 challenge [156] by using CRFs with a standard set of features, and obtaining a $F_1 = .846$ value (thus reducing error by 23%) when features obtained by language-specific resources are brought into play. The reason is that, once such a resource is available, the vectorial representation of a token may be enhanced by using as features, additionally to the word the token is an instance of, its superordinate words in the hierarchy. This brings in higher statistical robustness (since the superordinate words, being more general, will tend to occur often), and reduces feature sparsity, which is beneficial to learning. Note that the same process of bringing in superordinate words cannot reliably be done via lexical resources that are not domain-specific, since non-technical terminology is more ambiguous, which may lead to selecting the *wrong* superordinates.

In additional experiments that we do not report here we have used "It(GT)-RadLex", a version of RadLex (a hierarchically organized controlled dictionary of radiology terms produced by the Radiological Society of North America[5]) that we have obtained by automatically translating RadLex into Italian via Google Translate.

[5] `http://rsna.org/RadLex.aspx`

For each lexical expression (i.e., word or word n-gram) found in a report which exactly matched an entry in It(GT)-RadLex, the depth-k ancestor of the entry in It(GT)-RadLex (with depth 1 indicating the root) was added to the feature vectors representing the tokens in the lexical expression, for all $k = 2, 3, 4$. No improvement was observed in our experiments with respect to not using It(GT)-RadLex, which may indicate that more careful translations of the technical terms in a specialized dictionary are needed in order to make an impact on this IE task.

To the best of our knowledge no system for IE from radiology reports for the Italian language has been reported yet in the literature.

2.4 Related Work

Information extraction from text. Gaizauskas et al. [36] give a good introduction to IE research carried out in the '90s, including work spawned by the MUC series of evaluation challenges, while Sarawagi presents a more up-to-date survey [111]. McCallum [78] gives an excellent introduction to IE aimed at the non-specialist.

Information extraction from clinical text. The literature on IE from texts of medical interest can be roughly subdivided into (a) works about extracting information from the (bio-)medical literature (books, scientific articles, and the like), and (b) works about extracting information from texts generated during clinical activity (such as admission summaries, discharge summaries, clinical notes, radiological reports, and the like). The former is an easier task than the latter, since clinical texts, unlike texts from the biomedical literature, are more informal, and thus abound in abbreviations (sometimes specific to the particular hospital or department where they originated), ungrammatical text, acronyms (even non-standard ones), and misspellings. Meystre et al. [85] review IE work specifically addressed to clinical narratives in the electronic health record[6], while McNaught and Black [84] do the same for IE work addressed to biomedical literature. Note that the term "concept extraction" is often used in the biomedical area to actually refer to information extraction (see e.g., [10, 25, 55, 149]); the latter is instead the standard name of the task, at least in the NLP community, where the task was first investigated.

Several works have been carried out in the field of information extraction from clinical narratives, and many of them use rule-based or dictionary-based systems in which the rules and dictionaries have been manually generated. Soderland et al. [129] extract instances of two concepts of medical interest, "diagnosis" and "symptoms", using a text analysis system and a dictionary induction system. Evans et al. [31] extract information about drug dosage, such as drug, dose level, and frequency, using a pattern-matching system. Harkema et al. [44] use an IE framework that selects relevant entities from a medical dictionary and then performs syntactic and semantic analyses over the text; the resulting information is then stored into a template that

[6] It has to be noted, however, that [85] takes a much wider view of what IE means, since it takes IE to also encompass other text-related tasks such as text classification.

represents domain-specific entities and their properties and relations between them. Sotelsek-Margalef and Villena-Román [130] present a framework that uses IE from clinical reports to suggest medical diagnoses. Mykowiecka et al. [91] use a rule-based system that extracts information from clinical records written in Polish. Grishman et al. [42] use a rule-based system to detect disease outbreaks from clinical narratives.

Some works use a hybrid approach that combines the rule-based and the machine learning approaches. For example, Zhou et al. [169] extract three different types of information from semistructured medical records, i.e., numerical values (e.g., age, blood pressure), medical terms (as from a patient's medical history), and categorical values (e.g., smoker/nonsmoker), for a total of twenty-four fields. They use two different approaches: for the first two types of information they use a combination of unsupervised pattern-matching methods, domain ontologies, and NLP techniques, while for the third type of information they use a machine learning method with input generated via NLP techniques. As another example, Taira et al. [140] use a hybrid rule-based / machine learning system to extract findings and their related properties using lexical resources and semantic parsing.

ML-based approaches to IE from clinical text. Only in recent years, also thanks to the interest generated by the i2b2 ("Informatics for Integrating Biology and the Bedside") challenges [155, 156], researchers involved in IE from clinical narratives have started to massively use machine learning techniques. The first task of the 2011 i2b2/VA challenge [156] is indeed similar to the task we have faced in the present work, but is a bit simpler since (a) the concepts involved are neither nested nor potentially overlapping, and (b) the narratives are in English, which is a resource-rich language.

There may be several reasons for the above-mentioned slow takeup of machine learning technology by the clinical NLP community. Torii et al. [149] conjecture that one of the reasons may be the fact that "phrases extracted from clinical text need to be normalized to fine-grained concepts, such as those defined in SNOMED CT and the Unified Medical Language." Certainly, one of the reasons is the fact that training data for clinical text has traditionally been scarce, often because of privacy and confidentiality issues; the datasets release within the context of the i2b2 challenge are probably the first publicly available datasets of their kind.

Most of the i2b2 participants that adopt a machine learning approach use CRFs of some sort. Torii et al. [149] use CRFs in a study of the portability of IE systems across multiple sources of clinical text; their CRFs system uses, aside from a standard set of text-derived features, features obtained via the use of the UMLS metathesaurus. Jiang et al. [49] use a CRFs system to perform named-entity recognition as a preliminary step towards extracting concepts of clinical interest from discharge summaries. Jonnalagadda et al. [53] use a CRFs system in which the feature representation of a token is augmented with words that are "similar" to the word the token is an instance of, and where "similarity" is measured via distributional semantics. Patrick and Li [99] were the best performers in the i2b2 2009 challenge on extracting medication-related concepts from discharge summaries, by using CRFs

to detect named entities and SVMs to further classify them. All of the above publications do not dwell on the details of the specific type of CRFs they actually use, which seems to indicate that they use the simple LC-CRFs type.

In terms of efforts not strictly related to i2b2, Li et al. [69] compare support vector machines (SVMs) and CRFs in the extraction of names of disorders from clinical text, and find that the latter are markedly superior to the former, obviously thanks to the fact that the former cannot encode dependencies among the labels of different tokens in the sequence. Wang and Patrick [160] use CRFs for named-entity recognition in admission summaries; further application of a classifier committee formed by an SVMs classifier and a maximum entropy classifier, classifies the recognized named entities into classes.

Cascaded information extraction systems. Cascaded, multi-stage systems have been proposed in the past for several NLP tasks as applied to clinical texts. For instance, Jonnalagadda et al. [54] describes a multi-stage algorithm for coreference resolution (i.e., detecting if two linguistic expressions actually refer to the same entity) as applied to analyzing medical discharge reports. In an attempt to automatically extract medication information from clinical records, Patrick and Li [99] use a CRFs learner and an SVMs learner arranged in a cascade, the CRFs learner being entrusted with recognizing named entities and the SVMs learner being entrusted with recognizing the relationship between two recognized entities. The above-mentioned work by Wang and Patrick [160] is in a similar vein. Outside the scope of the clinical IE, a classical two-stage approach is the reranking method [24, 40, 88, 89, 93]. In the first stage of the reranking approach a classifier is used to generate a list of candidate hypotheses for each unlabeled example. In the second stage each hypothesis is enriched with new features associated to the structure predicted in the first stage; then a second classifier i.e., the reranker, is used to rank the hypotheses list. Finally the best hypothesis is selected. This is different from what we do, since our system revolves around the notion of using CRFs in both phases: a first phase where clauses are the units of interest, and a second phase in which such units are the individual tokens.

Positional features in information extraction. To the best of our knowledge, only another paper in the IE literature uses positional features: for a task of automatically extracting pros and cons of products from product reviews, Kim and Hovy [59] use positional features that indicate the first, the second, the last, and the second last sentence in a paragraph. Here the intuition is that, in a product review, pros and cons are important sentences that summarize the main point of the review. Our use is different, in the sense that we do not make any hypothesis of where important information for a given tag is located in a document, and simply record the relative position (as a percentage of the document length) in which the tagged segment is located. A fairly similar use of positional features is to be found in the work of Bramsen et al. [13]; however, the task addressed in [13] is not information extraction, but automatic segmentation of a document into temporally coherent

segments.

2.5 Summary

In this chapter we have presented novel solutions to the problem of extracting information from radiological reports (i.e., automatically annotating word sequences occurring in such reports according to a predefined set of concepts of interest) via supervised machine learning. Specifically, we have modified a standard linear-chain CRFs learning system in two novel ways, (i) cascading a clause-level LC-CRFs tagger and a token-level LC-CRFs tagger to obtain a two-stage system, and (ii) organizing the resulting two-stage system and a traditional token-level LC-CRFs system into a confidence-weighted ensemble. We have also reported novel work in representing text by introducing "positional" features, i.e., features aiming to represent the quantiles of the text in which the instances of a given concept mainly tend to occur.

The single-annotator and multiple-annotator experiments we have conducted have shown that the positional features do not bring any substantial advantage. While the intuition that underlies them seems correct and promising, further experimentation will be needed to check if they can be of real use in some application context.

The same experiments have also shown (as also confirmed by statistical significance tests) that (i) the two-stage method is clearly superior to the baseline method for the most frequent tags, but is slightly inferior to it on the less frequent ones, and that (ii) the ensemble method is superior to the baseline on both frequent and infrequent tags. Interestingly, the accuracy levels obtained via these technologies are higher than the inter-annotator agreement levels, as measured on the same test data and according to the same measure of agreement/accuracy. This is especially noteworthy since the feature set used in our work is fairly standard, and since it is widely believed that, as remarked by [149], "A tagger exploiting generic features can yield good performance, yet customized tokenization, features based on domain knowledge, hand-coded post-processing rules, and other fine-tuning can help improve performance". Our results, obtained via systems that use none of the above enhancements, show that machine-learned solutions nowadays reach effectiveness levels comparable to those of human experts. In particular, the experiments we have conducted do not use any specialized lexical resource or ontology, and are thus indicative of the level of accuracy that can be obtained in information extraction for resource-scarce languages.

3

Conditional Random Fields for Extracting Aspect-Related Opinions

From the factual IE of the previous chapter we move to a different applicative task, where the type of information to extract is not factual but subjective, namely, *opinion mining*. In this chapter we study the problem of *aspect-oriented* opinion mining from product reviews *at the sentence level*, which consists of predicting, for all sentences in the review, whether the sentence expresses a positive, neutral, or negative opinion (or no opinion at all) about a specific aspect of the product. For this task we propose a set of increasingly powerful models based on CRFs, including a hierarchical multi-label CRFs scheme that jointly models the overall opinion expressed in the review and the set of aspect-specific opinions expressed in each of its sentences. We evaluate the proposed models against a dataset of hotel reviews in which the set of aspects and the opinions expressed concerning them are manually annotated at the sentence level. We find that both hierarchical and multi-label factors lead to improved predictions of aspect-oriented opinions.

3.1 Introduction

Sharing textual reviews of products and services is a popular social activity on the Web. Some websites (e.g., Amazon, TripAdvisor[1]) act as hubs that gather reviews on competing products, thus allowing consumers to compare them. While an overall rating (e.g., a number of "stars") is commonly attached to each such review, only a few of these websites (e.g., TripAdvisor) allow reviewers to include *aspect-specific* ratings, such as distinct ratings for the Value and Service provided by a hotel.

The overall and the aspect-specific ratings may help the consumer to perform a first screening of the product, but they are of little use if she is interested in the *comments* about specific aspects of the product. For example, a low rating for the Rooms aspect of a hotel may be due to the small size of the room or to the quality

[1] `http://www.amazon.com/`, `http://www.tripadvisor.com/`

Overall rating: ★★★☆☆	*Aspect-specific opinions*	
Title: Good vlue [sic], terrible service	Value: Positive	Service: Negative
OK the value is good and the hotel is reasonably priced, but the service is terrible.	Value: Positive	Service: Negative
I was waiting 10 min at the erception [sic] desk for the guy to figure out whether there was a clean room available or not.	Checkin: Negative	Service: Negative
That place is a mess.	Service: Negative	
Rooms are clean and nice, but bear in mind you just pay for lodging, service does not seem to be included.	Cleanliness: Positive	Service: Negative

Figure 3.1: An example hotel review annotated with aspect-specific ratings at the sentence level.

of the furniture; different issues may be of different importance to different persons. In this case the user may have to read a large amount of text in order to retrieve the relevant information that is relevant to her.

Opinion mining research [72] has frequently considered the problem of predicting the overall rating of a review [97] or the ratings of its individual aspects [46]. While these are interesting research challenges, their practical utility is somewhat limited, since it is often the case that this information is already made explicit by the reviewers. Our goal is instead to build an automatic system that, given a sentence in a review and one of the predefined aspects of interest, (a) predicts if in the sentence an opinion concerning that aspect is expressed, and (b) if so, predicts the polarity of the opinion (i.e., if the opinion is positive, neutral/mixed, or negative). In the setup we consider a sentence may be relevant for (i.e., contain opinions concerning) zero, one, or several aspects at the same time, and the opinions contained in the same sentence and pertaining to different aspects may have different polarities. For example, *the room was spacious but the location was horrible* expresses a positive opinion for the Rooms aspect and a negative opinion for the Location aspect, while the remaining aspects are not touched upon.

The contribution of this study is twofold. First, inspired by the "coarse-to-fine" opinion model of [83] we develop an increasingly powerful set of multi-label conditional random field (CRF) schemes [64] that jointly model the overall, document-level opinion expressed by a review together with the aspect-specific opinions expressed at the sentence level. Our models are thus able to also predict the document-level ratings. However, as already pointed out, these ratings are of smaller practical interest, because they are often explicitly provided by the reviewers, whereas the aspect-level predictions are often not available and the sentence-level annotations (i.e., the indication of which sentences justify the aspect-level ratings) are never available. The use of a conditional model for this task is in contrast with previous work in this area, which has focused on generative models, mostly based on Latent Dirichlet Allocation, with strong independence assumptions [66, 87, 142, 158].

This problem has also been tackled via supervised learning methods in [70]. Like

us, this work relies on CRFs to model the structure of the reviews, but is however unable to cater for sentences that are relevant to more than one aspect at the same time, which is a strong limitation. Two works that are close in spirit to ours are [66, 158], and they may be considered the "generative counterparts" of our approach.

Second, we describe a dataset (that we make publicly available) of hotel reviews that we have annotated with aspect-specific opinions at the sentence level.[2] A previous dataset annotated by opinion at the sentence level exists [138], but the dataset introduced here also adds the aspect dimension and has a multi-label nature (i.e., allows sentences to be relevant to more than one aspect at the same time). Only very recently, and after we created our dataset, a dataset similar to ours has been presented [66], in which elementary discourse units (EDUs), which can be sub-sentence entities, are annotated using a *single-label* model. This dataset is composed of only 65 reviews, with a total of 1541 EDUs, while our dataset annotates 442 reviews, with a total of 5799 sentences.

The evaluation of generative models is often based on unannotated datasets [87, 158], and thus only on a qualitative analysis of the generated output. We believe that our dataset will be a valuable resource to fuel further research in the area by enabling instead a true quantitative evaluation, which enables a scientific comparison of different models.

3.1.1 Problem Definition

Before formally defining the task that we are going to tackle, we informally describe the principal components of the problem:

- **A review** is the text written by a human reviewer that describes (and comments on) the object of the review, e.g., a hotel, a restaurant, a smart-phone, and the like. It is composed by one ore more segments.
- **A segment** is a piece of text that is part of the review, and is typically bounded by two periods (e.g., a sentence).
- **Aspects** are the feature of the object of the review that the reviewer has written about, i.e., Room and Service are possible aspects of a hotel, while Screen and Battery are possible aspects of a smart-phone.
- **Aspect-related opinion** is the polarity of the opinion (e.g., positive, neutral, negative) expressed in a segment toward a specific aspect.
- **Overall rating** is the rating (e.g., 1 to 5 stars) expressed by the reviewer about the whole object of the review.

[2]The dataset can be freely downloaded at `http://nemis.isti.cnr.it/~marcheggiani/datasets/`

In order to formally model the problem, all the components listed above are formalized as different variables. These variables are then used to create the CRF structures we have adopted to solve this task.

- **A segment variable** x is an observed variable, that is, a feature of one of the segments that compose the review. The t-th segment of review $\mathbf{x}$ (which is composed by T consecutive segments) is modeled by a feature vector $\mathbf{x}_t$.
- **A segment opinion variable** y_t^a models the opinion related to a specific aspect $a \in \mathbb{A}$ of segment $t \in \{1, ..., T\}$, where $\mathbb{A}$ is the set of all possible aspects. The domain of the segment opinion variable is $\mathbb{Y} \cup \{\text{No-op}\}$. The No-op value (that stands for "no opinion") is used when none of the "real" opinion values of the domain $\mathbb{Y}$ hold. For each segment $\mathbf{x}_t$ we have as many of these variables as the number of aspects in $\mathbb{A}$.
- **An overall segment variable** x_o is an observed variable that models a single feature that describes the overall opinion. The overall review is modeled by an overall feature vector $\mathbf{x}_o$.
- **An overall opinion variable** y_o is the variable that represents the overall rating given by the reviewer. The domain of this variable is $\mathbb{Y}$.

Given a review $\mathbf{x}$, we seek to infer the values of the variables: $y_o \in \mathbb{Y}$ and the opinion y_t^a for each segment $t \in \{1, ..., T\}$ and each aspect $a \in \mathbb{A}$. This is a *multi-label* problem, since each segment t can be assigned up to $|\mathbb{A}|$ different opinions.

To model these variables we assume a feature vector $\mathbf{x}_t$ representing review segment t and a feature vector $\mathbf{x}_o$ representing the full review. For our experiments, that we report in Section 3.3, we adopt a dataset of hotel reviews. We define segments to correspond to sentences that compose a review. We define the sets of opinion labels $\mathbb{Y}$ and aspects $\mathbb{A}$ respectively as: $\mathbb{Y} = \{\text{Positive}, \text{Negative}, \text{Neutral}\}$ and $\mathbb{A} = \{\text{Rooms}, \text{Cleanliness}, \text{Value}, \text{Service}, \text{Location}, \text{Check-in}, \text{Business}, \text{Food}, \text{Building}, \text{Other}\}$. However, we want to stress that the proposed models are flexible enough to incorporate arbitrary sets of aspects and opinion labels, and to use a different type of segmentation.

3.2 CRFs Models and Algorithms

Previous work on aspect-oriented opinion mining has focused on generative probabilistic models [66, 142, 158]. Thanks to their generative nature, these models can be learnt without any explicit supervision. However, at the same time they make strong independence assumptions on the variables to be inferred, which is known to limit their performance in the supervised scenario considered in this study. Instead, we explore the use of CRFs. Specifically, we propose a hierarchical multi-label CRF model that jointly models the overall opinion of a review together with aspect-specific

opinions at the segment level. This model is inspired by the *fine-to-coarse* opinion model [83], which was recently extended to a partially supervised setting [138].[3] However, while previous work only takes opinion into account, we jointly model both sentence-level opinion and aspect, as well as overall review opinion. Below, we introduce a sequence of increasingly powerful CRF models for aspect-specific opinion mining, leading up to the full hierarchical sequential multi-label model.[4]

3.2.1 Models

Taking into consideration the CRFs of Equation (1.6), in this study we defined, $\mathbf{y} = \{y_o\} \cup \{y_t^a : t \in [1, T], a \in \mathbb{A}\}$. The models described in what follows differ in terms of their factorization of Equation (1.6), that is, in the structure of the graph underlying the distribution.

Linear-chain CRFs as a Baseline Model

As a baseline model, we take a simple first-order linear-chain CRFs (LC) in which a separate linear chain over opinions at the segment level is defined for each aspect. This model is able to take into account sequential dependencies between segment opinions [83, 95] specific to the same aspect, whereas opinions related to different aspects are assumed to be independent. Formally, the LC model is defined as:

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{a \in \mathbb{A}} \prod_{t=1}^{T} \Psi_s(y_t^a, \mathbf{x}_t) \prod_{t=2}^{T} \Psi_{\curvearrowright}(y_t^a, y_{t-1}^a) \tag{3.1}$$

The factors of the two factor templates in LC have this form:

$$\Psi_s(y_t^a, \mathbf{x}_t) = \exp \sum_{sk} \theta_{sk} f_{sk}(y_t^a, \mathbf{x}_t) \tag{3.2}$$

$$\Psi_{\curvearrowright}(y_t^a, y_{t-1}^a) = \exp \sum_{\curvearrowright k} \theta_{\curvearrowright k} f_{\curvearrowright k}(y_t^a, y_{t-1}^a) \tag{3.3}$$

Similarly to the factors in Section 1.2.1 this two factors respectively models the aspect-specific opinion of the segment at position t (Equation (3.2)), and the transition between the aspect-specific opinion variables at position $t-1$ and t in the linear chain corresponding to aspect a (Equation (3.3) [5]).

Differently from Section 1.2.1 the random variables here have the superscript a that means that we want to model not one, but several sequences.

[3] While we only consider the supervised scenario in this study, our model is readily extendable to the partially supervised setting by treating a subset of the fine-grained variables as latent.

[4] All models were implemented using Factorie [82].

[5] The symbol $\curvearrowright$ just identifies the "transition" factor.

Multi-label Models

The assumption of the LC model that the aspect-specific opinions expressed in each segment are independent of each other may be overly strong for two reasons. First, only a limited number of aspects are generally expressed in each segment. Second, when several aspects are mentioned, it is likely that there are dependencies between them based on discourse structure considerations. To address these shortcomings, we propose to model the dependencies between aspect-specific opinion variables within each segment, by adopting the multi-label pairwise CRFs formulation of [38, 135].

The CRFs *multi-label* models are characterized by the pairwise multi-label factor of Equation (3.4).

$$\Psi_m(y_t^a, y_t^b) = \exp \sum_{mk} \theta_{mk} f_{mk}(y_t^a, y_t^b) \tag{3.4}$$

This factor models the interdependence between opinion variables y_t^a and y_t^b of different aspects, a and b of the segment at step t with $b \in \mathbb{A}$ and $a \neq b$.

We first consider the *Independent Multi-Label* (IML) model, in which there are factors between the opinion variables within a segment, while each segment is independent from each other. In terms of Equation (1.6), the IML model factors as

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{a \in \mathbb{A}} \prod_{t=1}^{T} \Psi_s(y_t^a, \mathbf{x}_t) \prod_{b \in \mathbb{A} \setminus a} \Psi_m(y_t^a, y_t^b) \tag{3.5}$$

To allow for sequential dependencies between segments, the IML model can naturally be combined with the LC model (Figure 3.4). This yields the *Chain Multi-Label* (CML) model:

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{a \in \mathbb{A}} \prod_{t=1}^{T} \Psi_s(y_t^a, \mathbf{x}_t) \prod_{b \in \mathbb{A} \setminus a} \Psi_m(y_t^a, y_t^b) \prod_{t=2}^{T} \Psi_{\curvearrowright}(y_t^a, y_{t-1}^a, a) \tag{3.6}$$

Hierarchical (Multi-label) Models

Thus far, we have only modeled the aspect-specific opinions expressed at the segment level. However, many online review sites ask users to provide an overall opinion in the form of a numerical rating as part of their review. As shown by [83, 138], jointly modeling the overall opinion and the segment-level opinions in a hierarchical fashion can be beneficial to prediction at both levels.

The LC, IML and CML models can be adapted to include the overall rating variable in a hierarchical model structure analogous to that of the "coarse-to-fine" opinion model of [83]. This is accomplished by adding the following two factors to the three models above:

$$\Psi_o(y_o, \mathbf{x}_o) = \exp \sum_{ok} \theta_{ok} f_{ok}(y_o, \mathbf{x}_o) \tag{3.7}$$

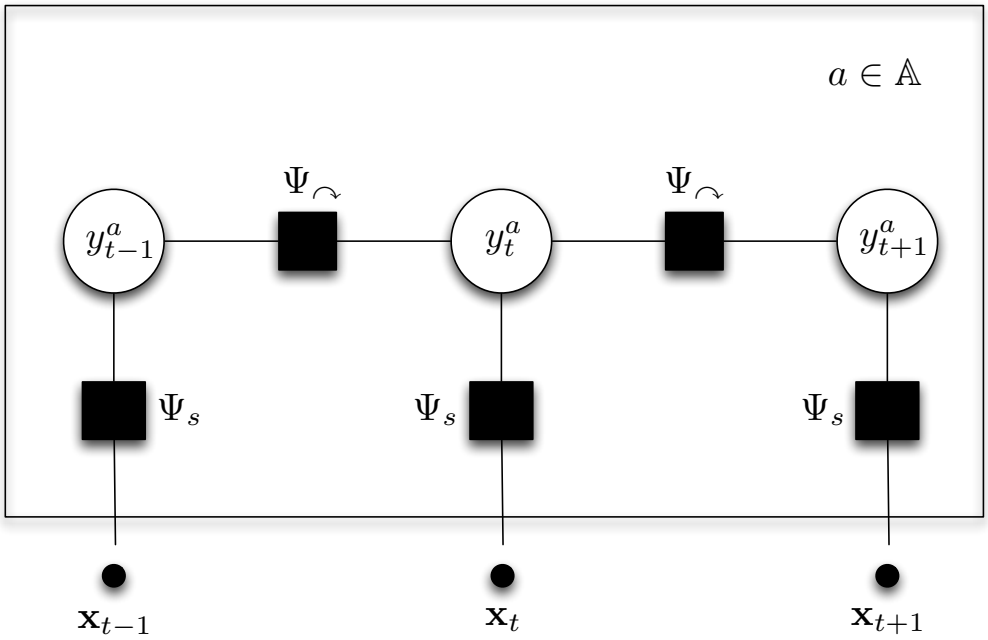

Figure 3.2: LC CRF model. The plate around the graph stands for "replication" $\forall a \in \mathbb{A}$.

$$\Psi_h(y_t^a, y_o) = \exp \sum_{hk} \theta_{hk} f_{hk}(y_t^a, y_o) \tag{3.8}$$

The factor template described in Equation (3.7) has the same use of the unigram template in Equation (3.2). The factor template of the form described in Equation (3.8) captures the dependence that exists between the overall opinion variable and the segment opinion variables, taking into account the aspects related to the segment opinion variables. The rationale behind the use of the overall opinion variable to calculate the segment opinion variable of each aspect is that having a hint of the overall opinion of a review can help (as in [83]) to better recognize the finer-grained sentiment, namely, the value of the segment opinion variable. In this work, differently from [83], we do not only connect the overall opinion variable with the segment opinion variables (Equation (3.8)), but we model the overall opinion variable through overall features (Equation (3.7)). In fact, given the generality of the overall opinion whose polarity is related to the object of a review as a whole (as opposed to the specificity of the aspect-related opinions whose polarities are related to specific aspect) the features that describe the two kinds of opinion are different. Therefore, the overall opinion variable cannot be "modeled" by the same features that model the segment opinion variables, but it needs a specific set of features. It is worth to notice, though, that even if we make use of the overall opinion, our goal is to recognize the aspect-related segment opinions. So, in general we are not

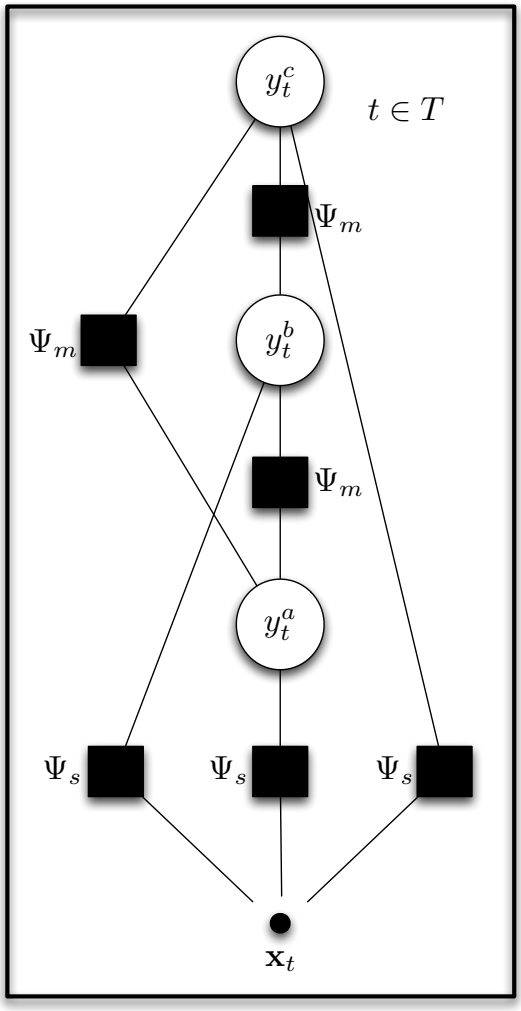

Figure 3.3: IML CRF model. The plate around the graph stands for "replication" $\forall t \in T$.

interested in obtaining the best possible performance on the overall opinion, as long as we obtain the best possible performance on the aspect-related segment opinions. Basically the overall opinion is just a way to obtain a more accurate model of the problem.

We joined the overall factor templates in Equations (3.7) and (3.8) to the three previous models, linear-chain, independent-loopy and loopy-chain to create three new models: *Linear-Chain-Overall* (LCO) (Equation (3.9), Figure 3.5), *Independent-Multi-Label-Overall* (IMLO) (Equation (3.10), Figure 3.6), and *Chain-Multi-Label-*

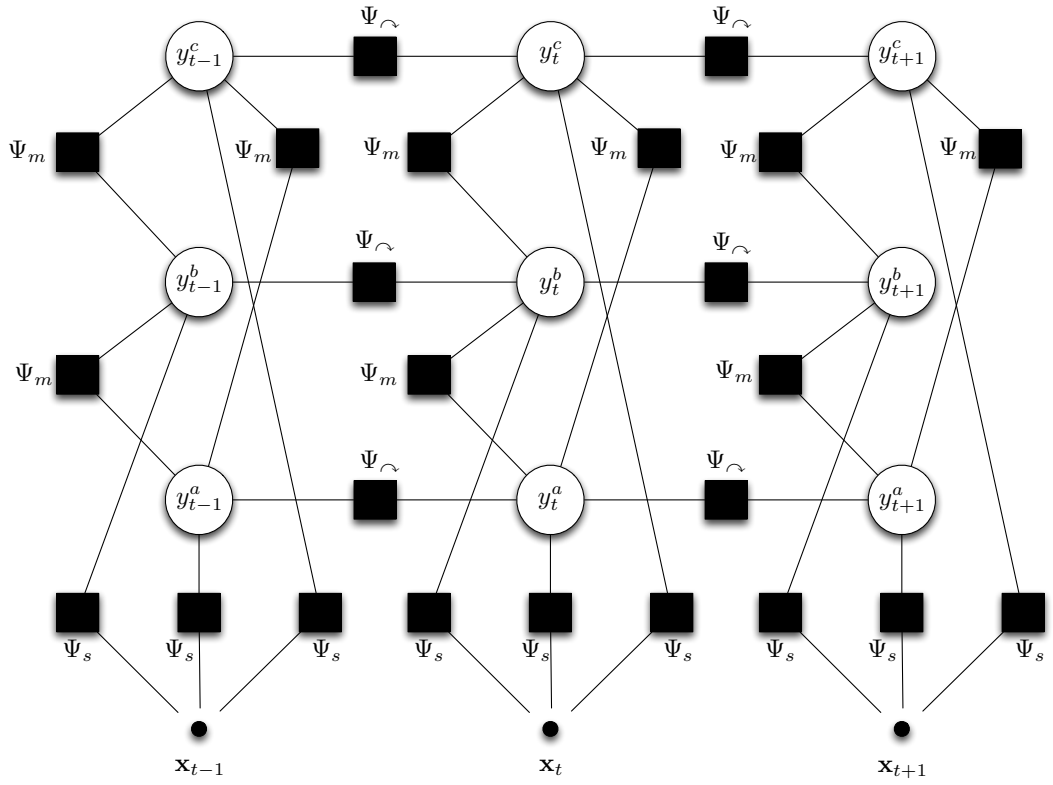

Figure 3.4: CML CRF model.

Overall (CMLO) (Equation (3.11), Figure 3.7).

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{a\in\mathbb{A}} \prod_{t=1}^{T} \Psi_s(y_t^a, \mathbf{x}_t)\Psi_h(y_t^a, y_o)\Psi_o(y_o, \mathbf{x}_o) \prod_{t=2}^{T} \Psi_\curvearrowright(y_t^a, y_{t-1}^a) \tag{3.9}$$

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{a\in\mathbb{A}} \prod_{t=1}^{T} \Psi_s(y_t^a, \mathbf{x}_t)\Psi_h(y_t^a, y_o)\Psi_o(y_o, \mathbf{x}_o) \prod_{b\in\mathbb{A}\setminus a} \Psi_m(y_t^a, y_t^b) \tag{3.10}$$

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{a\in\mathbb{A}} \prod_{t=1}^{T} \Psi_s(y_t^a, \mathbf{x}_t)\Psi_h(y_t^a, y_o)\Psi_o(y_o, \mathbf{x}_o) \prod_{b\in\mathbb{A}\setminus a} \Psi_m(y_t^a, y_t^b) \prod_{t=2}^{T} \Psi_\curvearrowright(y_t^a, y_{t-1}^a) \tag{3.11}$$

3.2.2 Training via Sampling

While the maximum-a-posteriori (MAP) assignment $\mathbf{y}^*$ and factor marginals can be inferred exactly in the LC and LCO models by means of variants of the Viterbi and forward-backward algorithms [83], exact inference is not tractable in the remaining

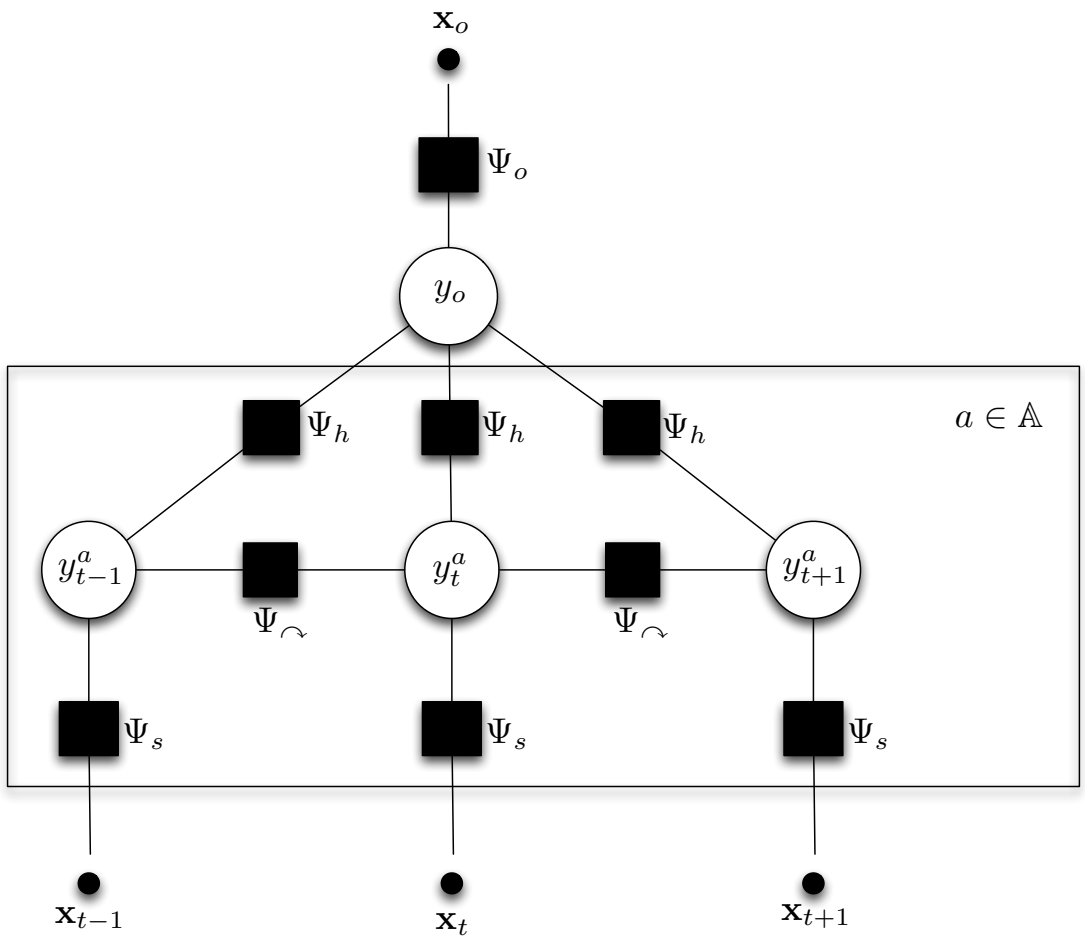

Figure 3.5: LCO CRF model. The plate around the graph stands for "replication" $\forall a \in \mathbb{A}$.

models. This is due to an exponential increase in the number of variable subsets to be enumerated (IML) and to the presence of loops in the graph structure (CML, IMLO and CMLO). We thus revert to approximate inference via Markov-Chain Monte Carlo (MCMC) sampling, specifically, via Gibbs sampling (see e.g., [8, 37]).

We train all models to approximately minimize the Hamming loss over the training set $\mathbf{D} = \{(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})\}_{i=1}^{N}$ by using the SampleRank algorithm [163, 164], which is a natural fit for sampling-based inference.[6]

[6]While inference and learning algorithms are likely to impact results, we view this decision as largely without loss of generality, since the focus of this study is on comparing model structures.

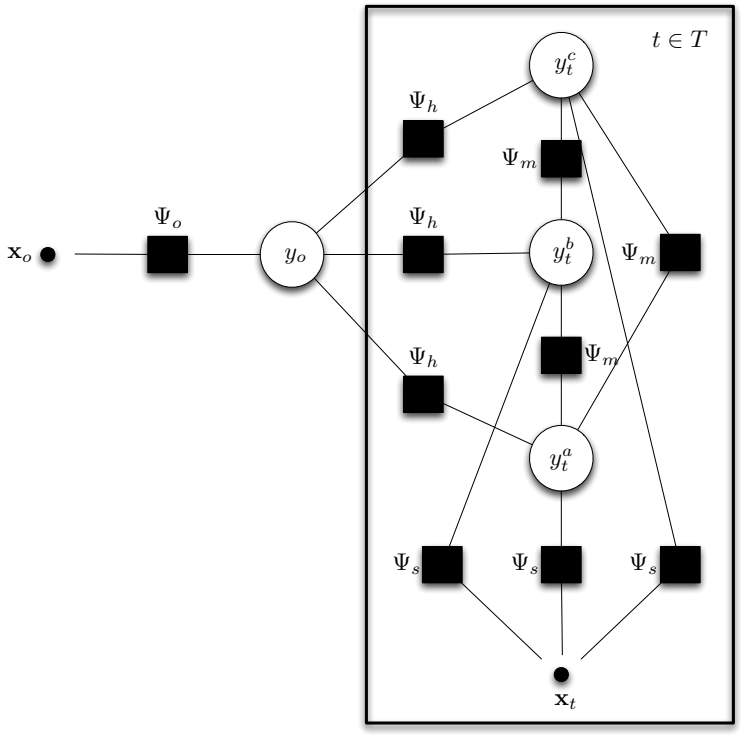

Figure 3.6: IMLO CRF model. The plate around the graph stands for "replication" $\forall t \in T$.

SampleRank

SampleRank [163, 164] is a rank-based learning algorithm that updates the model parameters θ after each MCMC sampling iteration. Each pair of consecutive samples in the MCMC chain is ranked according to two criteria, the *model score* and the *objective score*. If the two rankings disagree the model parameters are updated. The model score is the score returned by the current model $p(\mathbf{y}|\mathbf{x})$ having θ as parameter vector. The objective score is obtained by a *truth function* $\mathcal{F}(\mathbf{y})$, defined as

$$\mathcal{F}(\mathbf{y}) = -\sum_{t=1}^{T} \mathcal{L}(y_t, y_t^V) \tag{3.12}$$

which is the sequence-wise negative loss between the proposed assignment $\mathbf{y}$ and the true assignment $\mathbf{y}^V$. The element-wise loss function is defined as the Hamming loss, that is,

$$\mathcal{L}(y, y^V) = \begin{cases} 1 & \text{if } y \neq y^V \\ 0 & \text{otherwise} \end{cases} \tag{3.13}$$

The update of the model parameters θ is performed locally, that is, at each iteration the only parameters that are updated are the ones in the gradient:

$$\Delta = \mathbf{f}(\mathbf{y}^{(i)}, \mathbf{x}^{(i)}) - \mathbf{f}(\mathbf{y}'^{(i)}, \mathbf{x}^{(i)}) \tag{3.14}$$

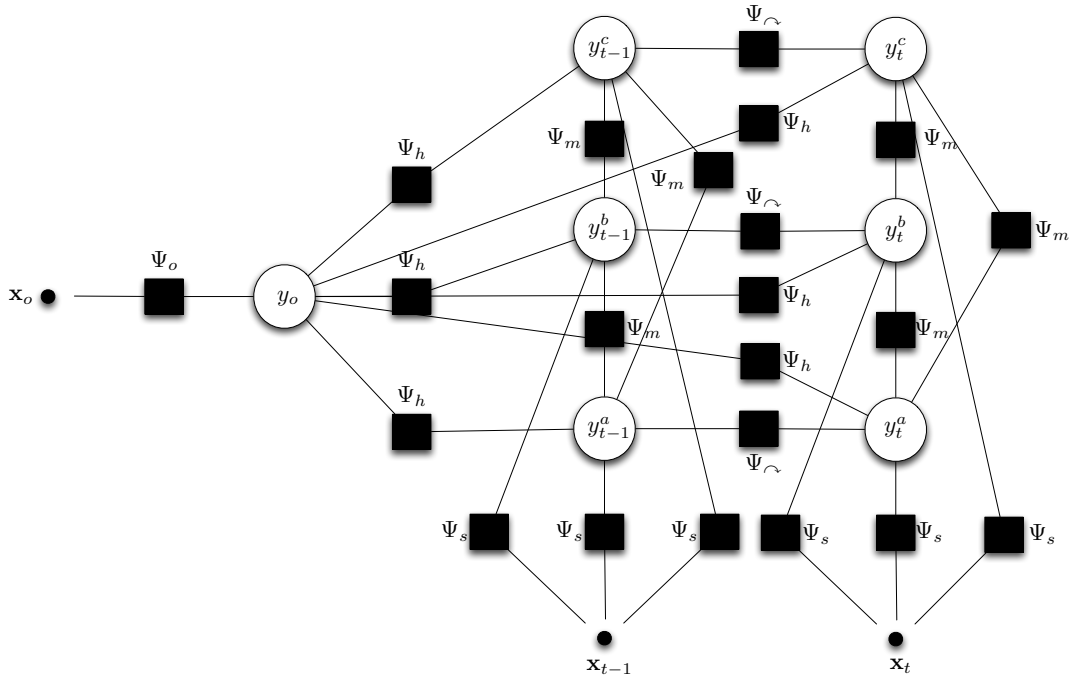

Figure 3.7: CMLO CRF model.

Here, $\mathbf{f}(\mathbf{y}^{(i)}, \mathbf{x}^{(i)})$ is the complete feature vector of the example $(\mathbf{y}^{(i)}, \mathbf{x}^{(i)})$. It takes into consideration all the feature functions of all the factor templates of a given model, i.e., $\sum_{sk} f_{sk} + \sum_{\frown k} f_{\frown k}$ of the LC factors of Equations (3.2) and (3.3). As an MCMC sampling algorithm to plug into SampleRank in order to draw the samples needed for the parameters update, we adopt Gibbs sampling.

Gibbs sampling

In general, MCMC sampling algorithms sample from a proposal distribution that maintains a record of the last sample. In this way each new distribution depends on samples drawn from the previous distribution. With *Gibbs sampling* [37], a particular instance of the class of MCMC sampling algorithms, the general steps to sample a new assignment $\mathbf{y}^{e+1}$ from an existing one $\mathbf{y}^e$, according to a CRF distribution such as the one in Equation (1.6) are the following ones:

1. Pick a variable at time t from the current sampled assignment $\mathbf{y}^e$ whose current value is y_{t_curr}; this step can be done randomly or with a particular strategy in mind.

2. Create the Gibbs sampling distribution:

$$p(y_t|\mathbf{y}^e \backslash y_{t\_curr}, \mathbf{x}) = \frac{1}{Z(\mathbf{x})} \frac{\prod_c^C \Psi_c(\mathbf{y}_c \backslash y_{t\_curr} \cup y_t, \mathbf{x}_c)}{\sum_{y_t} \prod_c^C \Psi_c(\mathbf{y}_c \backslash y_{t\_curr} \cup y_t, \mathbf{x}_c)} \tag{3.15}$$

Algorithm 1 SampleRank algorithm

Require: Training data $D = \{(\mathbf{x}^{(1)}, \mathbf{y}^{(1)})...(\mathbf{x}^{(n)}, \mathbf{y}^{(n)})\}$
Initial parameters θ_0

for $i \leftarrow 1$ until *convergence* **do**
  $\theta_{cur} = \theta_{i-1}$
  for training example $(\mathbf{y}^{(j)}, \mathbf{x}^{(j)})$ in D **do**
    $\mathbf{y}' \leftarrow \text{GibbsSampling}(\mathbf{y}, \mathbf{x}^{(j)}, \theta_{cur})$
    $\Delta \leftarrow \mathbf{f}(\mathbf{y}, \mathbf{x}) - \mathbf{f}(\mathbf{y}', \mathbf{x})$
    if $\frac{p(\mathbf{y}|\mathbf{x})}{p(\mathbf{y}'|\mathbf{x})} < 1 \wedge \mathcal{F}(\mathbf{y}) > \mathcal{F}(\mathbf{y}')$ **then**
      $\theta_{cur} = \theta_{cur} + \alpha\Delta$
    else if $\frac{p(\mathbf{y}|\mathbf{x})}{p(\mathbf{y}'|\mathbf{x})} > 1 \wedge \mathcal{F}(\mathbf{y}) < \mathcal{F}(\mathbf{y}')$ **then**
      $\theta_{cur} = \theta_{cur} - \alpha\Delta$
    else
      $\theta_{cur} = \theta_{cur} + 0$
    $\mathbf{y} \leftarrow \mathbf{y}'$
  $\theta_i = \theta_{cur}$
  if $\|\theta_{i-1} - \theta_i\|_2 < \epsilon$ **then**
    $convergence = true$

3. Sample a new assignment for y_t according to the Gibbs sampling distribution, $y_{t\_new} \sim p(y_t|\mathbf{y}^e \backslash y_{t\_curr}, \mathbf{x})$.

4. Create the new sample $\mathbf{y}^{(e+1)} = (\mathbf{y}^e \backslash y_{t\_curr}) \cup y_{t\_new}$

Stopping criterion

In its traditional formulation, SampleRank does not have a stopping criterion. A fixed number of iterations must be specified before the training process begins [82]. With this setting one could either perform too many training iterations, thus spending computational time in vain, or perform too few iterations, thus obtaining a parameter vector that is not adequately fitted to the training data. In this work, instead of using a fixed number of iterations, at the end of each iteration we check if the ℓ_2-norm of the sum of the atomic gradients is smaller than a fixed value ϵ. When the ℓ_2-norm is below the threshold ϵ, i.e., there have not been significant updates in the parameter vector θ, we consider the SampleRank algorithm has reached convergence.

Algorithm 1 puts everything together, showing the version of SampleRank algorithm adopted in this work.

3.2.3 Maximum-A-Posteriori Inference

In this scenario, in which exact inference is unfeasible, in order to obtain the most likely configuration $p(\mathbf{y}^*|\mathbf{x})$, instead of the well-known Viterbi algorithm we adopt a sampling approach. The *Maximum-A-Posteriori* (MAP) inference with sampling consists in a certain number of sampling (Gibbs sampling in this work) iterations in which, after each iteration, the posterior probability of the assignment given the learnt parameters $p(\mathbf{y}|\mathbf{x})$ is calculated. The assignment with maximum posterior probability is selected after a fixed number of sampling iterations. The outcome of this process is the most likely configuration $\mathbf{y}^*$ and its conditional probability $p(\mathbf{y}^*|\mathbf{x})$.

3.3 Experiments

In this section we study the proposed models empirically. After discussing our evaluation strategy, we describe and evaluate the creation of a new dataset of hotel reviews, which we have manually annotated with aspect-specific opinion at the sentence level. Finally, we compare the proposed models quantitatively by their performance on this dataset.

3.3.1 Evaluation Measures

When evaluating system output and comparing human annotations below, we first evaluate the capability of the system to correctly recognize if a segment carries an opinion for a certain aspect or not, and then the capability of the system to correctly recognize the opinion expressed towards each applicable aspect. In other words, we view the task as composed of the following two subtasks:

Aspect identification: for each segment and for each aspect, predict if there is any opinion expressed towards the aspect in the segment. Since each of these aspect-specific tasks is a binary problem, for this subtask we adopt the standard F_1 evaluation measure. The F_1 measure is the same as Section 2.3.2. The evaluation is performed for each aspect, and in order to have an overview on the overall accuracy of the system on all the aspects, we calculate the Macro F_1.

Opinion prediction: for each segment and each applicable (true positive) aspect for the segment, predict the opinion expressed towards the aspect in the segment. Since opinions are placed on an ordinal scale, as an evaluation measure we adopt *macro-averaged mean absolute error* (MAE^M) [3], a measure for evaluating ordinal classification that is also robust to imbalanced datasets.

Let $\mathbf{T}$ be the correct label assignments and let $\widehat{\mathbf{T}}$ be the corresponding model predictions. Let $\mathbf{T}_j = \{y_i : y_i \in \mathbf{T}, y_i = j\}$ and let n be the number of unique labels

in $\mathbf{T}$. The macro-averaged mean absolute error is defined as

$$\mathrm{MAE}^M(\mathbf{T}, \widehat{\mathbf{T}}) = \frac{1}{n} \sum_{j=1}^{n} \frac{1}{|\mathbf{T}_j|} \sum_{y_i \in \mathbf{T}_j} |y_i - \hat{y}_i| \tag{3.16}$$

This is suitable for evaluating the overall review-level opinion predictions. However, when evaluating the aspect-specific opinions at the segment level, we instead report $\mathrm{MAE}^M(\mathbf{T}_{\widehat{\mathbf{I}}_a}, \widehat{\mathbf{T}}_{\widehat{\mathbf{I}}_a})$, where $\widehat{\mathbf{I}}_a$ is the indices of segment opinion labels that were predicted as true positive for aspect a and $\mathbf{T}_{\widehat{\mathbf{I}}_a}$ is the set of true positive opinion labels for aspect a.

We calculated the Macro MAE_M as well, in order to have an overview of the performance of the system on all the aspects together.

Inter-annotator Agreement Measures

We also use the F_1 and MAE^M measures to assess inter-annotator agreement, by computing the average of these measures over all pairs of annotators. While the F_1 measure and the micro-averaged version of MAE are both symmetric, the use of macro-averaging makes MAE^M asymmetric, i.e., switching the predicted labels with the gold standard labels may change the outcome. This is problematic when used to measure inter-annotator agreement, since no annotator can be given precedence over the others. A reasonable strategy is to symmetrize the measure by treating each annotator in turn as the gold standard, and by averaging the two sets of results. This yields

$$\mathrm{sMAE}^M(\mathbf{T}, \widehat{\mathbf{T}}) = \frac{1}{2} \Big(\mathrm{MAE}^M(\mathbf{T}, \widehat{\mathbf{T}}) + \mathrm{MAE}^M(\widehat{\mathbf{T}}, \mathbf{T}) \Big) \tag{3.17}$$

3.3.2 Dataset

To facilitate a quantitative empirical evaluation of our proposed models, we have produced a new dedicated dataset of manually annotated hotel reviews. Three equally experienced annotators provided sentence-level annotations of a subset of 500 randomly selected reviews from the publicly available TripAdvisor dataset [158]. The full TripAdvisor dataset consists of 235,793 hotel reviews crawled over a period of one month. In addition to the review text, each review comes with a hotel identifier, an overall rating and optional aspect-specific ratings for the following seven aspects: Rooms, Cleanliness, Value, Service, Location, Check-in, and Business. All review-level ratings are on a discrete ordinal scale from 1 to 5 (with -1 indicating that an aspect-specific rating was not provided by the reviewer). Each review was split into sentences using the sentence splitter from [39] (plus a small number of regular expressions so as to deal with the unarticulated shape of some reviews).

Table 3.1: Number of opinion expressions at the sentence level, broken down by aspect and opinion. Out of 5799 annotated sentences, 4810 sentences contain at least one opinion-laden expression.

	Other	Service	Rooms	Clean.	Food	Location	Check-in	Value	Building	Business	NotRelated	*Total*
Pos	893	513	484	180	287	435	93	188	185	23	63	*3344*
Neg	353	248	287	66	127	51	56	87	62	3	40	*1377*
Neu	167	40	111	5	82	38	12	35	22	4	350	*866*
Total	1413	801	882	251	496	524	161	310	269	30	453	*5134*

Annotation

The annotations are related to nine aspects typically mentioned in hotel reviews. In addition to the seven aspects explicitly present (at the review level) in the TripAdvisor dataset, we decided to add two other aspects (Food and Building), since many comments in the reviews refer to them. Furthermore, the "catch-all" aspects Other and NotRelated were added, for a total of eleven aspects. Other captures those opinion-related aspects that cannot be assigned to any of the first nine aspects, but which are still about the hotel under review, for example, *Extremely friendly place.* The NotRelated aspect captures those opinion-related aspects that are not relevant to the hotel under review, as for example in *The crew on our flight were wonderful.* In what follows, segments marked as NotRelated are treated as non-opinionated.

The annotation distinguishes between Positive, Negative and Neutral/Mixed opinions. The Neutral/Mixed label is assigned to opinions that are about an aspect without expressing a polarized opinion, and to opinions of contrasting polarities, such as *The room was average size* (neutral) and *Pricey but worth it!* (mixed). The annotations also distinguish between explicit and implicit opinion expressions, that is, between expressions that refer directly to an aspect and expressions that refer indirectly to an aspect by referring to some other property/entity that is related to the aspect. For example, *Fine rooms* is an explicitly expressed positive opinion concerning the Rooms aspect, while *We had great views over the East River* is an implicitly expressed positive opinion concerning the Location aspect, and *All doors seem to have to slam to close* is an implicit negative opinion concerning the Rooms aspect.

Each of the three annotators was assigned 139 unique reviews, plus 83 reviews which all three annotators independently annotated, for a total of 500 annotated reviews. The 83 shared reviews were used to measure inter-annotator agreement, as described below. To minimize bias in these measurements, the reviews were ordered in such a way that every eighth reviews, one review was shared between all the annotators; this ensured that each annotator had the same amount of annotation experience when annotating the same shared review.

During the annotation process, the annotators identified a number of corrupted

Table 3.2: Inter-annotator segment-level aspect agreement, expressed in terms of F_1 (higher is better).

	Other	Service	Rooms	Clean.	Food	Location	Check-in	Value	Building	Business	*Avg*
Overall	.607	.719	.793	.733	.794	.795	.464	.575	.553	.631	*.675*
Implicit	.167	.123	.263	.111	.306	.286	.061	.131	.095	.333	*.188*
Explicit	.479	.684	.706	.739	.741	.710	.481	.560	.521	.624	*.625*

Table 3.3: Inter-annotator segment-level opinion agreement (restricted to the true positive aspects for each segment), expressed in terms of sMAE^M (lower is better).

	Other	Service	Rooms	Clean.	Food	Location	Check-in	Value	Building	Business	*Avg*
Overall	.308	.219	.191	.114	.234	.259	.003	.202	.150	.029	*.171*
Implicit	.167	.000	.000	.000	.074	.061	.000	.000	.000	.000	*.030*
Explicit	.262	.167	.147	.064	.190	.119	.000	.179	.092	.000	*.122*

reviews (primarily caused by site scraping errors in the original TripAdvisor data set). These reviews were subsequently removed from the final dataset, which as a resulted of 442 reviews, of which 73 shared among all three annotators. The remaining 369 unique reviews were partitioned into a training set (258 reviews, 70% of the total) and a test set (111 reviews, 30% of the total). The data was split by selecting reviews for each subset in an interleaved fashion, so that each subset constitutes a minimally biased sample both with respect to the full dataset and with respect to annotator experience.

Table 3.1 shows, for each aspect and for each opinion type, the number of segments annotated with that aspect and that opinion type (across the unique reviews and averaged across the shared reviews). Both opinions and aspects have a markedly imbalanced distribution in the dataset. As expected, the imbalance with respect to opinion is towards the Positive label. In terms of aspects, the Rooms, Service and Other aspects dominate.

Inter-annotator Agreement

We perform an inter-annotator agreement evaluation on the 73 shared reviews which all of the annotators annotated independently. We use the evaluation measures of Section 3.3.1 F_1 and sMAE^M.

For each aspect we have measured the agreement on implicit opinions only, on explicit opinions only, and on overall opinions, (i.e implicit and explicit opinions jointly). Table 3.2 shows the agreement on the task of annotating a segment as opinionated or not, computed by means of the F_1 measure. Table 3.2 shows two

interesting fact. The first is that, on average, there is a high disagreement if we take into consideration the agreement on implicit opinions; this result tells us that implicit opinions are very hard to spot. The second is that, by evaluating agreement jointly, we can see that it is higher than the single agreements separately. It is clear thus that it has been difficult for the human annotators to distinguish between implicit and explicit opinions, and that often the difference between them is too subtle to be recognized, even by a human annotator. For this reason, we have decided to take into consideration for this work the overall opinions without making any difference between implicit and explicit ones.

From Table 3.2 we can see that the worst level of agreement is obtained on the aspects Building, Value, Other, and Check-In. Other is a special aspect that the annotators used when they were not confident enough about using one of the nine main aspects. The aspects on which the annotators have been more uncertain are Value and Building. The disagreement on Building is due to the fact that has not been clear to an annotator what is and what is not part of a building, (for example, whether the pool is part of the building or not). The same happens for the Value aspect; here the disagreement evaluated by F_1 is mostly due to the confusion between Other and Value, since some annotators tended to annotate with the Value aspect sentences about the price of the Internet connection, while some annotators annotated them with the aspect Other, for instance, i.e., *Still ticks me off that the Ritz and other 4/5 star hotels charge an extra $10 for internet access* is annotated by one annotator as Other and by another annotator as Value. For the aspect Check-In, the reason of this low agreement is due to the misconception that some of the annotators have on the aspect Check-In and the aspect Service. Looking at the data, two annotators tended to annotate as Check-In all the sentences where a front-desk employee is mentioned, even when not engaged in a check-in activity, while the third annotator considered this expression as referring to the aspect Service, i.e., the sentence *the front desk manager is very helpful and always willing to go out of his way to get what you need* is annotated by two annotators as Check-In and by one as Service.

Table 3.3 shows the agreement on annotations that have been evaluated as true positive in the previous evaluation, that is, the sentences in which two annotators agree on the aspect. The disagreement in this section is generally quite low. The aspects with higher disagreement are Location, Food, and Other. The disagreement is mainly due to neutral-positive or neutral-negative errors, decisions about which the annotators had some small uncertainty. For example, regarding the aspect Location, in sentences in which just the walking distance from the hotel to some place of interest is mentioned, without any clue about the real opinion of the reviewer, the personal opinion of the annotator is used as the opinion label. So it can happen that `Close to shops, etc.` is a good location for an annotator and a bad location for another one. Detailed tables on inter-annotator agreement are included in Appendix A, in Tables A.1 and A.2.

Table 3.4: The collection of model factors and their corresponding features. Feature vectors: $\mathbf{x}_t$: {words, bigrams, SWN/MPQA/GI bigrams, χ^2 lexicon matches} in the t-th segment in review $\mathbf{x}$; $\mathbf{x}_o$: {words, bigrams, SWN/MPQA/GI bigrams} in $\mathbf{x}$.

Factor	Description	Features
$\Psi_s(y_t^a, \mathbf{x}_t)$	Segment aspect-opinion	$\mathbf{x}_t \otimes y_t^a \otimes a$
$\Psi_o(y_o, \mathbf{x}_o)$	Overall opinion	$\mathbf{x}_o \otimes y_o$
$\Psi_\frown(y_t^a, y_{t+1}^a)$	Segment aspect-opinion transition	$y_t^a \otimes y_{t+1}^a \otimes a$
$\Psi_m(y_t^a, y_t^b)$	Multi-label segment aspect-opinion	$y_t^a \otimes y_t^b \otimes a \otimes b$
$\Psi_h(y_t^a, y_o)$	Hierarchical overall / segment aspect-opinion	$y_t^a \otimes y_o \otimes a$

3.3.3 Features

The joint problem of aspect-oriented opinion prediction requires to model features that help to discriminate opinions and aspects, as well as opinions specific to a particular aspect. In the experiments that follow we use both word and word bigram identity features, as well as a set of polarity lexicon features based on the General Inquirer (GI) [131], the MPQA [165], and the SentiWordNet (SWN) [4] lexicons. The numerical polarity values of these lexicons are mapped into the set {Positive, Negative, Neutral}. The mapped lexicon values are used to generalize word bigram features by substituting the matching words of the bigram with the correspondent polarity. For example, the bigram *nice hotel* is generalized to the bigram *SWN:positive hotel*, from looking up *nice* in SentiWordNet. In order to move from the SWN synsets to real words, we applied a traditional transformation. For each <word, POS>pair in the SWN lexicon we summed the positive (negative) scores of each synset associated with it, divided by the ranking of the synset[7] in a global positive (negative) score. The polarity of the word for a given POS is obtained by the higher global score, i.e., positive score larger than negative score means Positive polarity, and two equal scores mean Neutral polarity. These features are used both with segment-level and review-level factors; see Table 3.4.

In order to better discriminate to which aspect an opinion is referred, we created a new aspect-specific lexicons, the χ^2 *lexicon*. The entries of this lexicons are the words obtained from the aspect segmentation algorithm proposed in [158]. This iterative algorithm uses χ^2 statistics to calculate the co-occurrence between some words that are used to describe certain aspects and the words that compose the reviews. The process start with a seed set of describing words for each aspect, then it calculates the dependence with χ^2 of each word in a review with the words in the seed sets, at the end of the iteration adds the words with the higher correlation to a specific aspect, in the seed set of that aspect. After that, the process starts again

[7]The ranking of the synset is computed by taking into consideration the frequency of use of the word relatively to each synset.

with bigger seed sets. The words obtained after this process are the words that are used in a review to describe a certain aspect. We use the output of this algorithm to create what we call the χ^2 *lexicon*, in which each word is associated with the (normalized) frequency with which the word is used to describe a certain aspect.

3.3.4 Experimental Setting

We use Factorie [82] as CRFs framework in order to easily design CRFs with arbitrary, non-standard, graph structures. Factorie[8] is general machine learning framework written in Scala. It offers programming tools to design probabilistic graphical models and to perform inference and parameters learning on them as the SampleRank algorithm and various sampling strategies.

We showed in Section 3.2.2 how SampleRank requires some hyper-parameters do be set. We fixed the learning rate to $\alpha = 1$, the gradient threshold $\epsilon = 0.1^{-5}$ and the Gibbs sampling iterations for the MAP inference at 100.

We trained our models on the training set described in Section 3.3.2, for a total of 258 unique reviews. We have run two sets of experiments. The first set of experiments evaluates the proposed models in order to compare them with the baseline, testing them on the test set composed by 111 reviews. The second set of experiments evaluates how the best model performs compared with human annotators on the three shared sets composed by 73 reviews.

The aim of these two sets of experiments is evident. In the first set of experiments we want to understand how good our proposed systems are with respect to the state of the art, and to check if adding information on the structure of the problem helps to solve it better. In the second set of experiments we want to check how close our best system is to the behavior of a human annotator.

3.3.5 Results

In this section we will present two evaluation scenarios, the first one evaluates the improvement of performance of the newly proposed methods with respect to the LC baseline, while the second evaluation is meant to compare the best automatic annotator with the human annotators.

System comparison

As shown in Tables 3.5 and 3.6, the multi-label and hierarchical models outperform the LC baseline in both aspect identification and opinion prediction. In particular, the multi-label models (IML, CML) significantly outperform the baseline on both subtasks, which shows the importance of modeling the interdependence of different aspects and their opinions within a segment. On the other hand, combining both multi-label and transition factors in the hierarchical model (CMLO) leads to worse

[8] http://factorie.cs.umass.edu/

Table 3.5: Results for different CRF models averaged on five experiments with five different random seeds. Segment-level aspect identification results in terms of F_1 (higher is better).

	Other	Service	Rooms	Clean.	Food	Location	Check-in	Value	Building	Business	*Avg*
LC	.499	.606	.662	.700	.579	.623	.329	.395	.298	.000	*.469*
IML	.542	.597	.664	.732	.605	.668	.371	.373	.363	.000	*.491*
CML	.489	.645	.655	.708	.605	.673	.327	.408	.358	.076	*.494*
LCO	.515	.586	.661	.697	.582	.611	.301	.384	.368	.173	*.488*
IMLO	.513	.621	.685	.702	.593	.614	.370	.363	.348	.040	*.485*
CMLO	.531	.629	.663	.706	.602	.618	.271	.393	.350	.081	*.485*

Table 3.6: Results for different CRF models averaged on five experiments with five different random seeds. Segment-level opinion prediction results (restricted to the true positive aspects for each segment) in terms of MAE^M (lower is better).

	Other	Service	Rooms	Clean.	Food	Location	Check-in	Value	Building	Business	*Avg*
LC	.526	.721	.572	1.00	.566	.932	.644	.616	.693	.000	*.627*
IML	.520	.659	.494	.956	.377	.939	.670	.700	.668	.000	*.598*
CML	.492	.681	.613	.978	.482	.906	.735	.691	.377	.000	*.595*
LCO	.482	.626	.398	1.00	.633	.903	.690	.490	.233	.000	*.546*
IMLO	.473	.615	.398	1.00	.457	.970	.343	.469	.269	.000	*.500*
CMLO	.499	.626	.428	1.00	.711	.906	.536	.552	.232	.000	*.549*

predictions compared to only including the multi-label factors (IMLO) or the transition factors. We hypothesize that this is due to inference errors, where the more complex graph structure causes the Gibbs sampler to converge more slowly. Furthermore, while the hierarchical models provide a significant improvement compared to their non-hierarchical counterparts in terms of opinion prediction, modeling both the overall and segment-level opinions is not helpful for aspect identification. This is not too surprising, given that the overall opinion contains no information about aspect-specific opinions.

The overall review-level opinion prediction results (not shown in Tables 3.5 and 3.6) are in line with the segment-level results. The IMLO model (.504) outperforms the LCO baseline (.518), as measured with MAE^M. However, as with the segment-level predictions, including both multi-label and transition factors in the hierarchical model (CMLO) hurts overall opinion prediction (.544).

Table 3.7: F_1 comparison between the best-performing model (IMLO) and the human annotators with IMLO results averaged on five experiments with five different random seeds.

	Other	Service	Rooms	Clean.	Food	Location	Check-in	Value	Building	Business	*Avg*
Human	.607	.719	.793	.795	.553	.575	.794	.464	.733	.631	*.675*
IMLO	.479	.585	.606	.614	.536	.673	.407	.429	.208	.190	*.473*

Table 3.8: sMAE^M comparison between the best-performing model (IMLO) and the human annotators with IMLO results averaged on five experiments with five different random seeds.

	Other	Service	Rooms	Clean.	Food	Location	Check-in	Value	Building	Business	*Avg*
Human	.308	.219	.191	.259	.150	.202	.234	.003	.114	.029	*.171*
IMLO	.676	.498	.445	.142	.451	.704	.212	.387	.025	.415	*.396*

Human comparison

We now turn to a comparison between the best-performing model (IMLO) and the human annotators, treating the model as a fourth annotator when computing inter-annotator agreement. This allows us to assess how far our model is from human-level performance. Tables 3.7 and 3.8 clearly show that much work remains to be done for both subtasks. The aspects Building and Business are difficult to detect for the automatic system, while a human identifies them with ease. We believe that the reasons for the poor performance may be different for the two aspects. For the Business aspect, the reason is likely the scarcity of training annotations, whereas for the Building aspect the reason may be lexical promiscuity (that is, a hotel building may be described by a multitude of features, such as interior, furniture, architecture, etc.).

Interestingly, the ability of the system in identifying the Value aspect is close to human level, on the other hand, it performs dramatically worse on opinion prediction on the same aspect. We suggest that this is because assessing the value of something coined in absolute terms (for example, that a $30 room is cheap) requires world knowledge (or feature engineering).

3.4 Related Work

From the seminal works of [153] and [97] the field of sentiment analysis has witnessed a continuing growth. Much of the work done in the past ten years is summarized in [72, 96]. Specifically, the first work to tackle the problem of extracting aspect-related opinions from reviews is [46]. The authors detect the aspects of an object in a review by looking for the most frequent noun phrases in the dataset. A "sentiment

orientation" is then associated with each detected aspect by employing a method that exploits a seed set of positive and negative adjectives, and the relations with other adjectives in WordNet.

This problem has been fertile soil for generative learning methods. Many works that approach this problem in fact make an extensive use of topic modeling methods, such as Latent Dirichlet Allocation (LDA). The most representative works using topic modeling are [159] (that represents the topic-modeling extension of [158]), [142] (that adopts an extension of LDA to discover aspects and then rate them), and [87] (where the authors extend LDA by introducing the interdependence between aspects and ratings). In [158] the authors first segment a review according to predefined set of keywords, and then adopt a generative model that, by using the overall rating, calculates the emphasis and the opinion polarity of a user toward a specific aspect. In [159] the authors extend their generative model with a topic modeling approach that allows them to segment the review into aspects in an unsupervised fashion. [142] uses an extended version of the Multi-Grain LDA introduced in [143] to discover aspects in online reviews. [87] introduces an extension of LDA called Interdependent LDA (ILDA). ILDA is a generative model that, in order to generate an opinion phrase, first generates the aspect and then generates the rating by conditioning on the generated aspect. The term that describes the aspect and the sentiment term are then generated by conditioning respectively on the aspect and the rating. The problem has also been faced with supervised learning methods. OpinionMiner [50] uses a variation of HMMs that incorporates POS labels and lexical patterns in order to extract opinions, distinguishing between explicit and implicit, from different class of aspects. In [70] the authors propose two CRFs structures, a variant of Skip-chain CRFs that links words connected by a conjunction, and a Tree CRFs that follows the shape of the semantic tree of the sentences. Unlike us they extract negative opinions, positive opinions, and their related aspects, without considering the neutral opinions.

3.5 Summary

In this chapter we have considered the problem of aspect-oriented opinion mining at the sentence level. Specifically, we have devised a sequence of increasingly powerful conditional random field models, culminating in a hierarchical multi-label model that jointly models both the overall opinion expressed in a review and the set of aspect-specific opinions expressed in each sentence of the review. Moreover, we have produced a manually annotated dataset of hotel reviews in which the set of relevant aspects and the opinions expressed concerning these aspects are annotated for each sentence; we make this dataset publicly available with the hope to spur further research in this area. We have evaluated the proposed models on this dataset; the empirical results show that the hierarchical multi-label model outperforms a strong comparable baseline. We also compared the performances of the best-performing model against the average inter-annotator agreement among the three annotators.

This comparison clearly shows that there is still much work to do to reach human-like performance in this task.

4

Training Data Quality in Information Extraction

Supervised machine learning-based systems such as the ones presented in Chapters 2 and 3 rely extensively on training data. While a lot of work has been devoted to devising learning methods that generate more and more accurate information extractors, on the other hand, little work (if any) has been devoted to investigating the effect of the quality of training data on the learning process. Low quality in training data sometimes derives from the fact that the person who has annotated the data is different (e.g., more junior) from the one against whose judgment the automatically annotated data must be evaluated. In this chapter we test the impact of such data quality issues on the accuracy of information extraction systems oriented to the clinical domain. We do this by comparing the accuracy deriving from training data annotated by the authoritative coder (i.e., the one who has also annotated the test data), with the accuracy deriving from training data annotated by a different coder. The results indicate that, although the disagreement between the two coders (as measured on the training set) is substantial, the difference in accuracy is not so. This suggests that current learning technology is robust to the use of training data of suboptimal quality.

4.1 Introduction

In the last five years there has been a flurry of work on information extraction from clinical documents, i.e., on algorithms capable of extracting, from the informal and unstructured texts that are generated during everyday clinical practice (e.g., admission reports, radiological reports, discharge summaries, clinical notes), mentions of concepts relevant to such practice [27, 49, 53, 69, 85, 99, 149, 156, 160]. Most of this literature is about methods based on supervised learning, i.e., methods for training an information extraction system from manually annotated examples. The use of learning algorithms based on support vector machines (SVMs – [49, 69, 123]), hidden Markov models (HMMs – [71]), and conditional random fields (CRFs – [27, 49, 53, 69, 99, 149, 160]) has been proposed; of these, CRFs have lately become the most popular, and have become the *de facto* standard in clinical information

extraction.

While a lot of work has been devoted to devising text representation methods and variants of the aforementioned supervised learning methods that generate more and more accurate information extractors, little work (if any) has been devoted to investigating the effects of the quality of training data on the learning process. Issues of quality in the training data may arise for different reasons:

1. In several organizations it is often the case that the original annotation is performed by coders (a.k.a. "annotators", or "assessors") as a part of a daily routine in which fast turnaround, rather than annotation quality, is the main goal of the coders and/or of the organization. An example is the (increasingly frequent) case in which annotation is performed via crowdsourcing (e.g., Mechanical Turk, CrowdFlower, etc.[1]) [41, 128].

2. In many organizations it is also the case that annotation work is usually carried out by junior staff (e.g., interns), since having it accomplished by senior employees would make costs soar.

3. It is often the case that the coders entrusted with the annotation work were not originally involved in designing the tagset (i.e., the set of concepts whose mentions are sought in the documents). As a result, the coders may have an imperfect understanding of the true meaning of these concepts, or of how their mentions are meant to look like, which may negatively affect the quality of their annotation.

4. The data used for training the system may sometimes be old or outdated, with the annotations not reflecting the current meaning of the concepts anymore. This is an example of a phenomenon, called *concept drift* [104, 110], which is well known in machine learning.

We may summarize all the cases mentioned above by saying that, should the training data be re-annotated by an *authoritative coder* (hereafter indicated as C_α), the resulting annotations would be, to a certain extent, different. We would also be able to precisely measure this difference, by measuring the *inter-coder agreement* (via measures such as Cohen's kappa – see e.g., [2, 30]) between the training data Tr as coded by C_α and the training data as coded by whoever else originally annotated them (whom we will call, for simplicity, the *alternative coder* – hereafter indicated as C_β). In the rest of this chapter we will take the authoritative coder C_α to be *the coder whose annotations are to be taken at face value*, i.e., considered as the "gold standard". As a consequence we may assume that C_α is the coder who, once the system is trained and applied, has also the authority to evaluate the accuracy of the automatic annotation (i.e., decide which annotations are correct and which are

[1] `https://www.mturk.com/`, `http://crowdflower.com/`

not)[2]. In this case, inter-coder disagreement measures the amount of noise that is introduced in the training data by having them annotated by a coder C_β different from the authoritative coder C_α.

It is natural to expect the accuracy of an information extraction system to be lower if the training data have been annotated by an alternative coder C_β, and higher if they have been annotated by C_α herself. However, note that this is *not* a consequence of the fact that C_α is more experienced, or senior, or reliable, than C_β. Rather, it is a consequence of the fact that standard supervised learning algorithms are based on the assumption that the training set and the test set are identically and independently distributed, i.e., that both sets are randomly drawn from the *same* distribution. As a result, these algorithms learn to replicate the subjective annotation style of their supervisors, i.e., of those who have annotated the training data. This means that we may expect accuracy to be higher simply when the coder of the training set and the coder of the test set are the *same* person, and to be lower when the two coders are different, irrespective of how experienced, or senior, or reliable, they are. In other words, the very fact that a coder is entrusted with the task of evaluating the automatic annotations (i.e., of annotating the test set) makes this coder *authoritative by definition*. For this reason, the authoritative coder C_α may equivalently be defined "the coder who has annotated the test set" (or: "the coder whose judgments we adhere to when evaluating the accuracy of the system"), and "alternative coder" to mean "a coder different from the authoritative coder".

The above arguments point to the fact that the impact of training data quality (under its many facets discussed in items (1)-(4) above) on the accuracy of information extraction systems may be measured by

1. evaluating the accuracy of the system in a *homogeneous* setting (i.e., both training and test sets annotated by the authoritative coder C_α), and then

2. evaluating the loss in accuracy, with respect to the homogeneous setting, that derives from working instead in a *heterogeneous* setting (i.e., test set annotated by C_α and training set annotated by an alternative coder C_β)[3].

4.1.1 Our Contribution

In this chapter we test the impact of training data quality on the accuracy of information extraction systems oriented to the clinical domain. We do this by testing

[2]In some organizations this authoritative coder may well be a fictional entity, e.g., several coders may be equally experienced and thus equally authoritative. However, without loss of generality we will hereafter assume that C_α exists and is unique.

[3]In the domain of classification the homogeneous and heterogeneous settings have also been called *self-classification* and *cross-classification*, respectively [162]. We depart from this terminology in order to avoid any confusion with *self-learning* (which refers to retraining a classifier by using, as additional training examples, examples the classifier itself has classified) and *cross-lingual classification* (which denotes a variant of text classification which exploits synergies between training data expressed in different languages).

the accuracy of a state-of the-art, CRFs-based system on a dataset of radiology reports (originally discussed in [27]) in which a portion of the data has independently been annotated by two different experts. In other words, we try to answer the question: "What is the consequence of the fact that my training data are not sterling quality? that the coders who produced them are not entirely dependable? How much am I going to lose in terms of accuracy of the trained system?"

In these experiments we not only test the "pure" homogeneous and heterogeneous settings described above, but we also test *partially heterogeneous* settings, in which increasingly large portions of the training data as annotated by C_α are replaced with the corresponding portions as annotated by C_β, thus simulating the presence of incrementally higher amounts of noise. For each setting we compute the inter-coder agreement between the two training sets; this allows us to study the relative loss in extraction accuracy as a function of the agreement between authoritative and alternative assessor as measured on the training set. Since in many practical situations it is easy to compute (or estimate) the inter-coder disagreement between (a) the coder to whom we would ideally entrust the annotation task (e.g., a senior expert in the organization), and (b) the coder to whom we can indeed entrust it given time and cost constraints (e.g., a junior member of staff), this will give the reader a sense of how much inter-coder disagreement generates how much loss in extraction accuracy.

The rest of the chapter is organized as follows. In Section 4.2 we describe experiments that attempt to quantify the degradation in extraction accuracy that derives from low-quality training data, with Section 4.2.2 devoted to spelling out the experimental setting and Section 4.2.3 devoted instead to presenting and discussing the results. Section 4.3 reviews related work in information extraction from clinical documents and on establishing the relations between training data quality and extraction accuracy. Finally, Section 4.4 concludes, discussing avenues for further research.

4.2 Experiments

In this chapter we adopt the same experimental setting of Chapter 2. We employ the LC-CRF model of Section 2.2.2 using the positional features of Section 2.2.5. We conducted our experiment on the UmbertoI(RadRep) dataset described in Section 2.3.1 and evaluated the performance with the token-level variant of the F_1 measure described in Section 2.3.2.

4.2.1 Inter-coder agreement

As a measure of inter-coder agreement we use Cohen's kappa (noted κ), defined as

$$\kappa = \frac{P(A) - P(E)}{1 - P(E)} \tag{4.1}$$

where $P(A)$ denotes the probability (i.e., relative frequency) of agreement calculated as:

$$P(A) = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.2}$$

and $P(E)$ denotes the probability of chance agreement calculated as:

$$P(E) = \frac{(TP + FP)(TP + FN) + (TN + FP)(TN + FN)}{TP + TN + FP + FN} \tag{4.3}$$

where, TP, TN, FP and FN stand for the numbers of true positives, true negatives false positives, and false negatives, respectively (see [2, 30] for more details).

We opt for kappa since it is the most widely known, and best understood, measure of inter-coder agreement. For Cohen's kappa too we work at the token level, i.e., for each token x_t we record whether the two coders agree on whether x_t is labeled or not with the concept c of interest.

Incidentally, note that (as observed in [28]) we can compute Cohen's kappa only thanks to the fact that we conduct our evaluation at the token level. Those who conduct their evaluation at the segment level (e.g., [15]) find that they are unable to do so, since in order to be defined kappa needs the notion of a true negative to be also defined, and this is undefined at the segment level. Evaluation at the segment level thus prevents the use of kappa and leaves F_1 as the only choice.

4.2.2 Experimental Protocol

In Chapter 2, experiments on the UmbertoI(RadRep) dataset were run using either `1-only` and/or `2-only` (i.e., the portions of the data that only one coder had annotated) as training data and `Both(1)` and/or `Both(2)` (i.e., the portion of the data that both coders had annotated, in both versions) as test data.

In this chapter we switch the roles of training set and test set, i.e., use `Both(1)` or `Both(2)` as training set (since for the purpose of this work we need training data with multiple, alternative annotations) and `1-only` or `2-only` as test set. Specifically, we run two batches of experiments: in Batch 1 Coder1 plays the role of the authoritative coder (C_α) and Coder2 plays the role of the alternative coder (C_β), while in Batch 2 Coder2 plays the role of C_α and Coder1 plays the role of C_β. Each of the two batches of experiments is composed of:

1. An experiment using the homogeneous setting, i.e., both training and test data are annotated by C_α. This means training on `Both(1)` and testing on `1-only` (Batch 1) and training on `Both(2)` and testing on `2-only` (Batch 2).

2. An experiment using the heterogeneous setting, i.e., training data annotated by C_β and test data annotated by C_α. This means training on `Both(2)` and testing on `1-only` (Batch 1) and training on `Both(1)` and testing on `2-only` (Batch 2).

3. Experiments using the partially heterogeneous setting, i.e., test data annotated by C_α, and training data annotated in part by C_β (ω% of the training documents, chosen at random) and in part by C_α (the remaining $(100-\omega)$% of the training documents). We call ω the *corruption ratio* of the training set; $\omega = 0$ obviously corresponds to the fully homogeneous setting while $\omega = 100$ corresponds to the fully heterogeneous setting.

   We run experiments for each $\omega \in \{10, 20, ..., 80, 90\}$ by monotonically adding, for increasing values of ω, new randomly chosen elements (10% at a time) to the set of training documents annotated by C_β. Since the choice of training data annotated by C_β is random, we repeat the experiment 10 times for each value of $\omega \in \{10, 20, ..., 80, 90\}$, each time with a different random such choice.

For each of the above train-and test experiment we compute the inter-coder agreement $\kappa(Tr, \widetilde{Tr})$ between the non-corrupted version of the training set Tr and the corrupted version $\widetilde{Tr}$. We then take the average among the 10 values of $\kappa(Tr_\alpha, \widetilde{Tr}_\alpha)$ deriving from the 10 different experiments run for a given value of ω and denote it as $\kappa(\omega)$; this value indicates the average inter-coder agreement that derives by "corrupting" ω% of the documents in the training set, i.e., by using for them the annotations performed by the alternative coder.

For each of the above train-and test experiment we also compute the extraction accuracy (via both F_1^μ and F_1^M) and the relative loss in extraction accuracy that results from the given corruption ratio.

4.2.3 Results

Figures 4.1 and 4.2 illustrate the results of our experiments by plotting F_1 as a function of the corruption ratio ω. For each value of ω, the corresponding level of inter-coder agreement $\kappa(\omega)$ (as averaged across the two batches) is also indicated. Table 4.1 reports precise extraction accuracy figures for the homogeneous and heterogeneous settings, for both batches of experiments and along with the resulting inter-coder agreement values.

A first fact that emerges is that macroaveraged (F_1^M) results are lower than the corresponding microaveraged (F_1^μ) results. This is unsurprising, and conforms to a well-known pattern. In fact, microaveraged effectiveness scores are heavily influenced by the accuracy obtained on the most frequent concepts (i.e., on the ones that label many tokens), which tends to be higher, since for frequent concepts training data abound. Conversely, in macroaveraged effectiveness measures each concept counts the same, which means that the low-frequency concepts (which tend to be the low-performing ones too) have as much of an impact as the high-frequency ones. See [23, pp. 591–593] for a thorough discussion of this point in a text classification context.

A second fact that may be noted is that there is a substantive difference in accuracy values between the two batches, with Batch 1 generating much higher

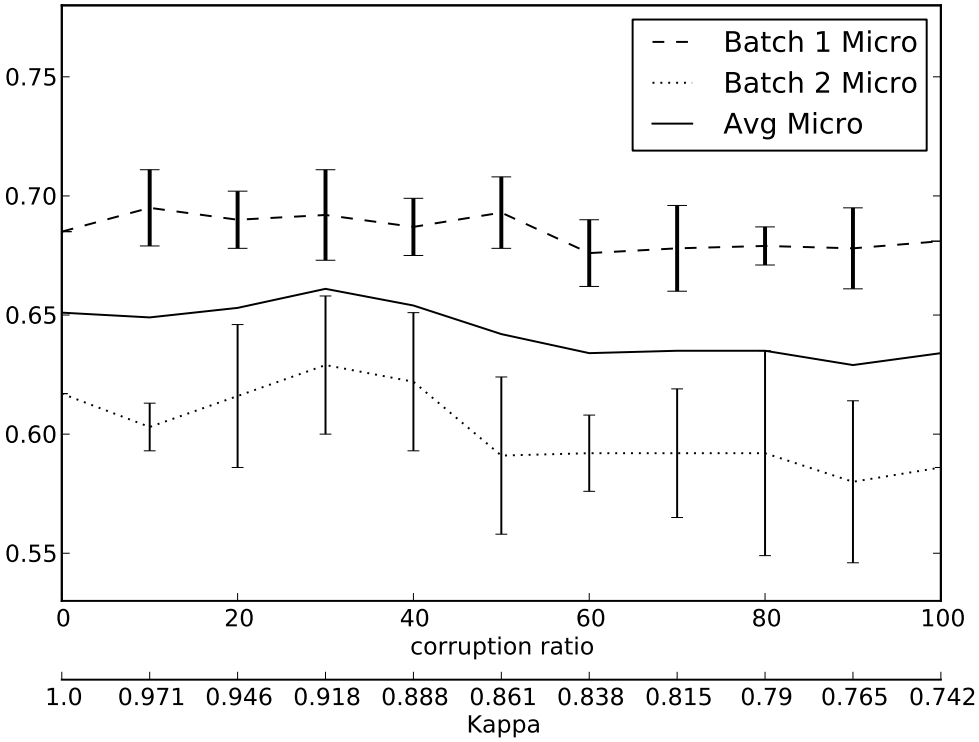

Figure 4.1: Microaveraged F_1 as a function of the fraction ω of the training set that is annotated by C_β instead of C_α ("corruption ratio"). The dashed line represents the experiments in Batch 1, the dotted line represents those in Batch 2, and the solid one represents the average between the two batches. The vertical bars indicate, for each $\omega \in \{10, 20, ..., 80, 90\}$, the standard deviation across the 10 runs deriving from the 10 random choices of the elements in Tr_β.

accuracy than Batch 2. This fact can be especially appreciated at the microaveraged level, which indicates that the most affected codes are the high-frequency ones. However, it is not possible to attribute this fact to the different behaviour of the two alternative coders, since this substantive difference is already present for $\omega = 0$, i.e., in the fully homogeneous case. It is thus likely that the difference is more simply due to Coder1 being more self-consistent in her annotation style than Coder2, or to the fact that `1-only` contains richer information than `2-only`.

A third fact that emerges is that, for both F_1^μ and F_1^M, there is (as could be expected, and notwithstanding some oscillations) a clear decreasing pattern for accuracy as a function of ω.

However, the most important (and perhaps surprising) fact is that this decrease

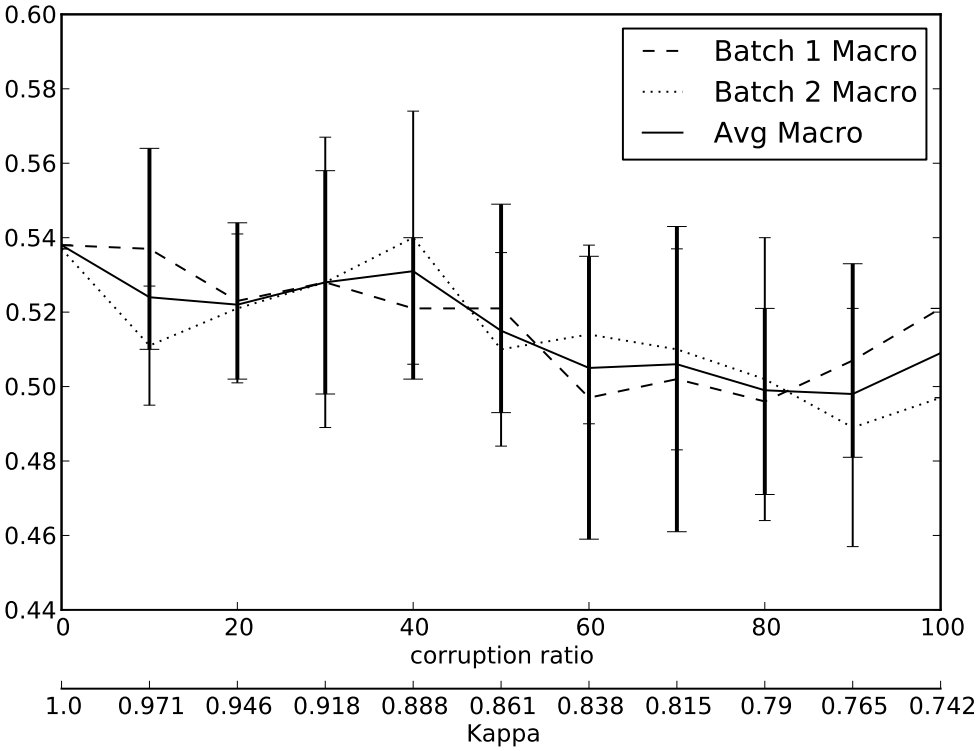

Figure 4.2: Macroaveraged F_1 as a function of the fraction ω of the training set that is annotated by C_β instead of C_α ("corruption ratio"). The dashed line represents the experiments in Batch 1, the dotted line represents those in Batch 2, and the solid one represents the average between the two batches. The vertical bars indicate, for each $\omega \in \{10, 20, ..., 80, 90\}$, the standard deviation across the 10 runs deriving from the 10 random choices of the elements in Tr_β.

is not dramatic. As specified in Table 4.1, a decrease in kappa values from 1.000 to 0.742 (a quite substantive decrease) determines a decrease in F_1^M (average across the two batches) from 0.538 to 0.501 (-6.87%) and from 0.651 to 0.634 (-2.61%).

This seems to suggest that current learning technology is robust to the use of training data of suboptimal quality. The fact that the decrease is more marked for F_1^M than for F_1^μ suggests that the most frequent codes are less affected than the most infrequent ones; this could be expected, since the most frequent codes have more training data, which somehow compensates for the suboptimal quality of these data.

This comparatively small decrease in accuracy might seem somehow surprising in the light of the results of [29], who show that, in a text classification context,

Table 4.1: Extraction accuracy for the homogeneous setting ($\omega = 0$) and heterogeneous setting ($\omega = 100$), for both batches of experiments (and for the average across the two batches), and along with the resulting inter-coder agreement values expressed as $\kappa(\omega)$.

	x	$\kappa(\omega)$	F_1^μ	F_1^M
Batch 1	0	1.00	0.685	0.538
	100	0.742	0.681	0.521
Batch 2	0	1.00	0.617	0.537
	100	0.742	0.586	0.497
Average	0	1.00	0.651	0.538
	100	0.742	0.634	0.501

minimal quantities of training noise can give rise to extremely substantial losses in classification accuracy. However, the key difference is that [29] uses artificially generated noise (since the datasets the authors use do not contain multiply annotated documents), and it is not clear that the noise model the authors use well represents the kind of noise we want to deal with here. Unfortunately the results obtained in Chapter 2 cannot be directly compared with the ones obtained in this chapter. The reason is the fact that for the training data adopted in Chapter 2 we do not have an assessment of inter-coder agreement. Thus, the decrease in accuracy that we observed in the multiple-annotator scenario (Section 2.3.4) with respect to the results obtained in the single-annotator scenario (Section 2.3.4), cannot be thoroughly evaluated as a function of the disagreement between coders.

4.3 Related Work

4.3.1 Inter-coder Agreement

Inter-coder agreement, or the lack thereof, has been widely studied for over a century (see e.g., [62] for an introduction). As a phenomenon, disagreement among coders naturally occurs when units of content need to be annotated by humans according to their semantics (i.e., when the occurrences of specific concepts need to be recognized within these units of content). Such disagreement derives from the fact that semantic content is a highly subjective notion: different coders might disagree with each other as to what the semantics of, say, a given piece of text is, and it is even the case that the same coder might at times disagree with herself (i.e., return different codes when coding the same unit of content at different times).

Inter-coder agreement may be measured by the relative frequency of the units of content on which coders agree, usually normalized by the probability of chance agreement. Many metrics for inter-coder agreement have been proposed over the years, "Cohen's kappa" probably being the most famous and widely used ("Scott's pi" and "Krippendorff's alpha" are others); sometimes (see e.g., [15, 27]) functions that were not explicitly developed for measuring inter-coder agreement (such as F_1, that was developed for measuring binary classification accuracy) are used. Recently, the authors of [98] adopted a probabilistic model that assess the agreement between coders by measuring the uncertainty associated with each annotation, and the bias toward specific categories. The levels of inter-coder agreement that are experimentally recorded in actual experiments vary a lot across experiments, types of content, and types of concepts that are to be recognized in the units of content under investigation. This extreme variance depends on factors such as "annotation domain, number of categories in a coding scheme, number of coders in a project, whether coders received training, the intensity of coder training, the annotation purpose, and the method used for the calculation of percentage agreements." [6]. The actual meaning of the concepts the coders are asked to recognize is a factor of special importance, to the extent that a concept on which very low levels of inter-coder agreement are reached may be deemed, because of this very fact, ill-defined.

4.3.2 Effects of Low-Quality Training Data

The literature on the effects of imperfect training data quality on prediction accuracy is extremely scarce, even within the machine learning literature at large. An early such study is [108], who look at these issues in the context of learning to predict prices of mutual funds from economic indicators. Differently from us, the authors work with noise artificially inserted in the training set, and not with naturally occurring noise. From experiments run with a linear regression model they reach the bizarre conclusion that "the predictive accuracy (...) is better when errors exist in training data than when training data are free of errors.", while the opposite conclusion is (somehow more expectedly) reached from experiments run with a neural networks model. A similar study, where the context is predicting the average air temperature in distributed heating systems, was carried out in [48]; yet another, where the goal was predicting the production levels of palm oil via a neural network, is [56].

In the context of a biomedical information extraction task[4] [43] examine the situation in which training data annotated by two different coders are available, and they found that higher accuracy is obtained by using both versions at the same time than by attempting to reconcile them or using just one of them. Their use case is different from ours, since in the case we discuss only one set of annotations, those of

[4]Biomedical IE is different from clinical IE, in that the latter (unlike the former) is usually characterized by idiosyncratic abbreviations, ungrammatical sentences, and sloppy language in general. See [29, Section 5] for a discussion of this point.

the alternative coder, are available as training data. Note also that training data annotated by more than one coder are rarely available in practice.

Closer to our application context, [29] have thoroughly studied the effect of imperfect training data quality in text classification. However, in their case the degradation in the quality of the training data is obtained, for mere experimental purposes, via the insertion of artificial noise, due to the fact that their datasets did not contain data annotated by more than one coder. As a result, it is not clear how well the type of noise they introduce models naturally occurring noise. [162] also address the text classification task (in the context of e-discovery from legal texts), bur differently from [29] they work with naturally occurring noise. Differently from the present work, the multiply-coded training data they use were coded by one coder known to be an expert coder and another coder known to be a junior coder; our work instead (a) focuses on information extraction, and (b) does not make any assumption on the relative level of expertise of the two coders.

4.4 Summary

Few researchers, if any, have investigated the loss in accuracy that occurs when a supervised learning algorithm is fed with training data of suboptimal quality. We do this in the case of information extraction systems (trained via supervised learning) as applied to the detection of mentions of concepts of interest in medical notes. Specifically, we test to what extent extraction accuracy suffers from the fact that the person who has annotated the test data (the "authoritative coder"), who is by definition the person to whose judgment we conform irrespectively of her level of expertise, is different from the person who has labeled the training data (the "alternative coder"). Our experimental results, that we have obtained on a dataset of 500 mammography reports annotated according to 9 concepts of interest, are somehow surprising, in the sense that they indicate that only a marginal decrease in extraction accuracy follows from a substantial decrease in training data quality. This seems to indicate that current supervised learning technology (and, in particular, the conditional random fields technology that we have used here) is robust to the use of training data of suboptimal quality. Since labeling cost is an important issue in the generation of training data (with senior coders costing much more than junior ones, and with internal coders costing much more than "mechanical turkers"), this result may give important indications as to the cost-effectiveness of low-cost annotation work.

5

Active Learning with Partially Labeled Sequential Data

This chapter is devoted to the study of how to improve the process of human annotation of text in order to obtain good-quality training data with minimum effort. In fact, the annotation of training data is a tedious and expensive work that requires the effort of expert human annotators. The process of *active learning* (AL) starts with a very small training set of annotated data and a large set of unannotated data. A learner is trained on the training set and the learnt model is used to select a sample of the unannotated data, via an AL strategy; the sample is then presented to a human annotator for annotation and subsequent inclusion in the training set. This process is iterated until some stopping criteria are met, e.g., size of training set, amount of human effort, effectiveness of the learnt model on a validation set. AL proved to be effective in optimizing the available human annotation effort with respect to sampling data to be annotated in random mode. In this chapter we consider the problem of designing AL strategies for sequence labeling tasks. While there seems to be agreement on the efficiency of AL strategies for learning classifiers on vectorial data (e.g., the uncertainty sampling strategy is quite popular), the case of sequence labeling is more complicated. In particular, one generally does not want to ask the labeling of full training sequences but instead ask the annotator to focus on a small part of a sequence, namely, the one with highest ambiguity. We investigate in this chapter the behavior of some AL strategies for sequence labeling tasks in a semi-supervised scenario. We compare experimentally the effectiveness of these strategies and we explore their behavior on different sequence labeling tasks: phrase chunking, part-of-speech tagging, named-entity recognition, and bio-entity recognition.

5.1 Introduction

It is a well-consolidated fact that the state-of-the-art methods in natural language processing tasks are machine learning approaches. The main problem of this kind

of methods is the cost of the production of human-annotated data to train such systems. Annotating data is a tedious and expensive work that requires the effort of one or, better, more than one human annotator, expert in the domain of the task to solve. This activity is even more expensive when we need training data for structured learning problems, in which an instance is composed by several annotations, such as in sequence labeling problems. In general, in order to create a dataset, a certain amount of data is asked to be labeled by human annotators. In order to minimize the human effort for annotating such data, several methods have been studied. The most popular methods, that we jointly use in this chapter, are *semi-supervised learning* and *active learning*. Each of them acts in a different way, both with the aim to reduce the need of large amounts of human annotated data.

Semi-supervised Learning Semi-supervised learning [170] is a learning model which starts with an initial training set of human annotated samples that has a much smaller size than those typically used in traditional supervised learning. The training set is enriched before been submitted to the learner by fetching new samples from a large set of unlabeled data and automatically labeling them by using various strategies that exploit the similarities and relations (when available) between the labeled and unlabeled data. Different approaches for learning classifiers from labeled and unlabeled data have been studied. The co-training approach, popularized by [12], adopts two classifiers that incrementally learn from each other's most confident predictions. The self-training approach [167], similarly to the previous method, uses a small amount of labeled data to create a classifier that will predict labels for the unlabeled data; the most confident labeled examples are then added to the training data. Graph-based methods [11, 168] generate graphs in which the nodes represent the examples, both labeled and unlabeled, and the edges represent the similarity between examples; then, similarity algorithms that work on graphs are adopted to select unlabeled examples that can be reasonably labeled as their "similar" labeled neighbors. For what concerns semi-supervised learning on sequence labeling tasks, some work has been recently done, especially adopting CRFs [74, 126, 139]. In particular, [152] introduced what we call in this dissertation the *partially-labeled conditional random fields* (PLCRFs), a semi-supervised type of CRFs capable to train a classifier even with partially-labeled sequences.

Active Learning The process of *active learning* (AL) asks a human annotator to annotate not random data, but only the most informative examples, in order to save human annotation effort. The informativeness of an example is based on the concept of how much new information the new example would give in learning a new classifier, once annotated. AL aims to create an annotated dataset composed by extremely informative examples. AL methods can be coarsely classified according to two orthogonal dimensions: (1) the AL scenario, and (2) the sampling approach.

The AL scenarios are mainly three, and they depend on how the examples that

have to be labeled are presented to the human annotator. The less common scenario is the *query synthesis* scenario [1], where the learner itself proposes to annotate examples in the input space, even synthetic ones. This scenario can lead to some annotation problems when the synthetic examples produced are not intelligible by a human being. The second most-popular AL scenario is the *stream-based* scenario [17], where new examples are proposed to the learner one at time and the learner decides whether to propose the example to the human annotator or to discard it. A typical example of this scenario is spam filtering [114], where spam mails are flagged as spam and then reused to re-train the spam filter. The third and most-common AL scenario is the *pool-based* scenario [68], and it is the one we have adopted in this work. The pool-based scenario reflects a real-world scenario where a large amount of unlabeled data can be collected at once, but the available resources for human annotation allow to annotate only a relatively small subset of those data. In this scenario only a small set of data is labeled and a pool of unannotated data is available. The examples are sampled from the unannotated set typically by their amount of informativeness, that can be measured in several ways.

The measure of informativeness that will be associated with unlabeled examples will determine the sampling approach that one adopts in the AL process. Among the most popular sampling approaches are:

- *Uncertainty sampling* [67], which measures the informativeness of an example by the uncertainty that a classifier has on it.

- *Query by committee sampling* [34], which selects the examples on which two different classifiers disagree the most.

- *Expected error reduction sampling* [109], whose goal is to find examples that once annotated will reduce the prediction error.

- *Density-weighted sampling* [35, 65], which exploits the structure of the data, sampling not just the examples with the higher information content, but also the ones that are somehow representative of the data distribution.

AL has proved to be useful in sequence labeling tasks [22, 26, 117, 137], in which researchers have mostly adopted as basic annotation unit the entire sequence. [161] exploited PLCRFs, so to adopt as basic annotation unit the single token. Furthermore they make use of the concept of *field confidence estimation*, introduced in [20], to re-estimate the confidence of the classifier on unlabeled examples after each token labeling. AL with partially annotated sequences has proven to be effective in reducing the amount of annotated data with respect to common AL approaches [161].

In this chapter we focus on AL strategies for partially-labeled sequences adopting the single token as annotation unit and PLCRFs as learning algorithm given its nature in dealing with partially labeled sequences. We propose several AL strategies

based on measures of *uncertainty* adapted for the AL with partially labeled sequences scenario and tailored on PLCRFs. In addition, we propose two novel AL strategies that take advantage of the finer granularity given by the partially-labeled AL scenario. We test the effectiveness of these strategies on four traditional sequence labeling tasks, showing that the choice of single-token annotation can bring to unpredictable results on sequence labeling tasks in which the structure of the sequences is not regular, e.g., named-entity recognition. We propose also a first solution to the problem of unpredictability.

5.2 Partially-labeled CRFs

Traditional CRFs (see Section 1.2.1) cannot learn a classifier from partially labeled sequences. [152] introduced the PLCRFs, an approach that allows to learn a CRF model using partially labeled sequences, marginalizing on those tokens that do not have an assigned label. In this chapter we propose several AL strategies tailored on PLCRFs.

In PLCRFs, $\mathbf{L}$ is a partially labeled sequence. It consists of a sequence of sets L_t where $L_t = \mathcal{Y}$ (where $\mathcal{Y}$ is the set of all the possible labels, as in Section 1.1) if there is no label information for the token at time t. L_t is a singleton containing y_t if the label of the token at time t is known. With this setup, the probability distribution $p(\mathbf{Y_L}|\mathbf{x})$ is calculated as:

$$p(\mathbf{Y_L}|\mathbf{x}) = \sum_{\mathbf{y} \in \mathbf{Y_L}} p(\mathbf{y}|\mathbf{x}) \tag{5.1}$$

where $p(\mathbf{y}|\mathbf{x})$ is calculated as in Equation (1.19).

In order to perform inference and parameter learning on PLCRFs, some modifications to traditional inference algorithms are required.

5.2.1 Forward-Backward Algorithm

Differently from traditional CRFs (Equations (1.22) and (1.24)), the forward and backward score (respectively α and β) are calculated as follows:

$$\alpha_{t,\mathbf{L}}(j) = \begin{cases} 0 & \text{if } j \notin L_t \\ \Psi_1(j, y_0, \mathbf{x}_1) & \text{if } t = 1 \\ \sum_{i \in L_{t-1}} \alpha_{t-1,\mathbf{L}}(i) \Psi_t(j, i, \mathbf{x}_t) & \text{otherwise} \end{cases} \tag{5.2}$$

$$\beta_{t,\mathbf{L}}(i) = \begin{cases} 0 & \text{if } j \notin L_t \\ 1 & \text{if } t = T \\ \sum_{j \in L_{t+1}} \beta_{t+1,\mathbf{L}}(j) \Psi_{t+1}(j, i, \mathbf{x}_{t+1}) & \text{otherwise} \end{cases} \tag{5.3}$$

where y_0 is a special label that encodes the beginning of a sequence and L_t is the set of the applicable labels at time t in the sequence $\mathbf{L}$. If the label is not observed, that

is, is hidden, then L_t is the entire set of possible labels $\mathcal{Y}$, otherwise it is composed just of the observed label y_t.

The forward and backward recursion of Equations (5.2) and (5.3) allow the computation of the α and β scores not necessarily on a complete lattice, but on a constrained lattice with blocked paths on observed labels.

5.2.2 Marginal Probability

The forward and backward scores, calculated in Equations (5.2) and (5.3), allow the computation of the marginal probability $p(y_t = j|\mathbf{x}, \mathbf{L})$, that is, the marginal probability to have label j at time t in the sequence, in a partially labeled scenario. Differently from the traditional CRFs, here we potentially have a partial annotated sequence $\mathbf{L}$, which means that the forward and backward scores, and thus the partition function Z are calculated on a potentially constrained lattice.

Formally:

$$p(y_t = j|\mathbf{x}, \mathbf{L}) = \frac{\alpha_{t,\mathbf{L}}(j) \cdot \beta_{t,\mathbf{L}}(j)}{Z_{\mathbf{L}}(\mathbf{x})} \tag{5.4}$$

where

$$Z_{\mathbf{L}}(\mathbf{x}) = \sum_{j \in L_t} \alpha_{t,\mathbf{L}}(j) \cdot \beta_{t,\mathbf{L}}(j) \tag{5.5}$$

for any value t.

The constrained marginal probability in CRFs has been introduced by [20] with the name of *field confidence estimation*. In our case, unlike [20], a "field" can be a non-contiguous set of annotated tokens of the sequence. In case sequence $\mathbf{L}$ does not have partial annotations, the forward and backward scores and the marginal probabilities are calculated as in the traditional CRFs.

5.2.3 Viterbi Algorithm

As in the traditional CRFs, the most probable sequence assignment is calculated with the Viterbi algorithm. The Viterbi recursion is adopted to calculate the posterior probability of the most probable sequence assignment. From this probability the backward recursion of the Viterbi algorithm is used to trace the most probable sequence assignment.

Similarly to the forward score (Equation (5.2)) the Viterbi score for partially labeled sequences is calculated as follows:

$$\delta_{t,\mathbf{L}}(j) = \begin{cases} 0 & \text{if } j \notin L_t \\ \Psi_1(j, y_0, \mathbf{x}_1) & \text{if } t = 1 \\ \max_{i \in L_{t-1}} \delta_{t-1,\mathbf{L}}(i) \Psi_t(j, i, \mathbf{x}_t) & \text{otherwise} \end{cases} \tag{5.6}$$

Given Equation (5.6) the probability of the most probable assignment is calculated as:

$$p(\mathbf{y}^*|\mathbf{x},\mathbf{L}) = \max_{\mathbf{y}} p(\mathbf{y}|\mathbf{x},\mathbf{L}) = \frac{\max_{j\in L_T} \delta_{T,\mathbf{L}}(j)}{Z_{\mathbf{L}}(\mathbf{x})} \tag{5.7}$$

5.2.4 Log-Likelihood

As in traditional CRFs, in order to learn the model parameters θ, the maximum log-likelihood approach is adopted. Given the definitions of PLCRFs in Equation (5.1), the log-likelihood function $LL(\theta)$, which has to be maximized in order to estimate the parameters θ of the model, is calculated as follows:

$$\begin{aligned}
LL(\theta) &= \sum_{n=1}^{N} \log p(\mathbf{Y}_{\mathbf{L}^{(n)}}|\mathbf{x}^{(n)}) \\
&= \sum_{n=1}^{N} \log \sum_{\mathbf{y}\in\mathbf{Y}_{\mathbf{L}^{(n)}}} \frac{\prod_{t=1}^{T} \Psi_t(y_t^{(n)}, y_{t-1}^{(n)}, \mathbf{x}_t^{(n)})}{\sum_{\mathbf{y}'\in\mathbf{Y}} \prod_{t=1}^{T} \Psi_t(y_t'^{(n)}, y_{t-1}'^{(n)}, \mathbf{x}_t^{(n)})} \\
&= \sum_{n=1}^{N} \log \sum_{\mathbf{y}\in\mathbf{Y}_{L^{(n)}}} \prod_{t=1}^{T} \Psi_t(y_t^{(n)}, y_{t-1}^{(n)}, \mathbf{x}_t^{(n)}) - \log \sum_{\mathbf{y}'\in\mathbf{Y}} \prod_{t=1}^{T} \Psi_t(y_t'^{(n)}, y_{t-1}'^{(n)}, \mathbf{x}_t^{(n)}) \\
&= \sum_{n=1}^{N} \log Z_{\mathbf{Y}_{L^{(n)}}}(\mathbf{x}^{(n)}) - \log Z_{\mathbf{Y}}(\mathbf{x}^{(n)})
\end{aligned} \tag{5.8}$$

Even though the log-likelihood in traditional linear-chain CRFs is a concave function, due to the incomplete labeling, this function is not concave and thus can lead to local optima during the optimization. However, it is still possible to use optimization algorithms such as gradient ascent, LBFGS, and the like, in order to maximize the log-likelihood.

In the following section we present several AL strategies that exploit the properties of the PLCRFs presented in this section.

5.3 Active Learning Strategies for CRFs with Partially Labeled Sequences

Pool-based AL ([67]) is probably the most common scenario in AL, where one has a large amount (pool) of unlabeled examples $\mathcal{U}_1$ and a small amount of labeled examples $\mathcal{T}_1$. In this scenario, the process of AL consists in a series of n iterations where a classifier Φ_i is trained with labeled examples $\mathcal{T}_i$, and then is used to classify the unlabeled examples $\mathcal{U}_i$. At this point an AL strategy $\mathcal{S}$ will select a number of examples B that once labeled will hopefully improve the performance of the next classifier Φ_{i+1}.

Algorithm 2 shows the pool-based AL framework for partially annotated sequences. Differently from AL for fully labeled sequences [26], thanks to the finer granularity of the partially labeled model, we use the token as basic annotation unit, instead of the entire sequence.

The point of using the partial labeling is in saving the request for human annotations on tokens whose labels are already known (inferred) by the classifier and concentrate on those tokens that the classifier finds hard to label. Using the semi-supervised approach of the PLCRFs we can take advantage of single-labeled tokens instead of an entire labeled sequence.

Algorithm 2 Pool-based active learning framework

Require: $\mathcal{T}_1$, the initial training set
$\mathcal{U}_1$, the initial unlabeled set
$\mathcal{S}$, the selected active learning strategy
n, the number of iterations
B, the dimension of the update batch

1: **for** $i \leftarrow 1$ to n **do**
2: $\Phi_i \leftarrow \text{train}(\mathcal{T}_i)$
3: $\mathcal{L}_i \leftarrow \Phi_i(\mathcal{U}_i)$
4: **for** $b \leftarrow 1$ to B **do**
5: $\mathbf{x}_*^{(b)} \leftarrow \arg\min_{\mathbf{x}_t \in \mathbf{x}, \mathbf{x} \in \mathcal{L}_i} \mathcal{S}(t, \mathbf{x})$
6: $\mathcal{L}_i \leftarrow \mathcal{L}_i - \mathbf{x}^{(b)} \cup \Phi_i(\mathbf{x}^{(b)}, y_*)$
7: $\mathcal{U}_i \leftarrow \mathcal{U}_i - \mathbf{x}_*^{(b)} \cup (\mathbf{x}_*^{(b)}, y_*)$
8: $\mathcal{T}_i \leftarrow \mathcal{T}_i - \mathbf{x}_*^{(b)} \cup (\mathbf{x}_*^{(b)}, y_*)$
9: $\mathcal{U}_{i+1} \leftarrow \mathcal{U}_i$
10: $\mathcal{T}_{i+1} \leftarrow \mathcal{T}_i$

The entire pool-based AL process with partially labeled sequences is summarized in Algorithm 2. The function $\mathcal{S}(t, \mathbf{x})$ is what, hereafter, we call an AL *strategy*. $\mathcal{S}(t, \mathbf{x})$ takes as input an automatically annotated sequence $\mathbf{x}$ and an element t of this sequence from the set of sequences $\mathcal{L}_i$ annotated by the PLCRF classifier Φ_i, and returns a measure of informativeness as a function of the classifier decision.

For each iteration through the update batch B, the most informative element $\mathbf{x}_*^{(b)}$, according to the AL strategy, is chosen. The subscript $*$, in this case, represents the most informative token, while the superscript (b) represents the sequence where the token appears. After the choice of the most informative token the sets $\mathcal{L}_i$, $\mathcal{U}_i$ and $\mathcal{T}_i$ are updated. $\mathcal{L}_i$ is updated by removing the annotated sequence $\mathbf{x}^{(b)}$ and all the information given by the classifier, and by adding the same sequence with the new manually labeled token (y_*) and all the re-estimated annotation given by the classifier $\Phi_i(\mathbf{x}^{(b)}, y_*)$. In the unlabeled set $\mathcal{U}_i$ and the training set $\mathcal{T}_i$ the most

informative token $\mathbf{x}_*^{(b)}$ is updated with its manually labeled version $(\mathbf{x}_*^{(b)}, y_*)$[1]. After B token annotations, the unlabeled set and the training set for the next iteration, respectively $\mathcal{U}_{i+1}$ and $\mathcal{T}_{i+1}$, are updated.

The inference methods of Section 5.2 allow not only to train a CRF model with partially labeled sequences, but give the possibility of classifying partially labeled sequences, using the known labels as support for the prediction of the other ones. Thus, in the AL scenario, each time a token is chosen it is immediately labeled, and this new information, as we can see from line 6 of Algorithm 2, is promptly used to re-estimate the informativeness of the other tokens in the sequence where the chosen token appears.

One may argue that, for a human annotator, annotating only one or few tokens, instead of the entire sequence, is a difficult task. This would be correct in the scenario in which the text is presented to the human annotator without any visual clue about the annotations. However, in [21] it is shown that presenting to the human annotator the highlighted sequence to be annotated along with the associated sequence of labels obtained by the classifier requires much less effort from the annotator than performing the annotation without any visual and contextual clue.

5.3.1 Greedy Strategies

In this section we present three AL strategies that select the most informative tokens, regardless of the assignment performed by the Viterbi algorithm, thus maximizing the informativeness at token level and not at sequence level as in Section 5.3.2.

The rationale behind all these strategies is that, even though we are looking for the most probable sequence assignment, we also want to annotate the most informative tokens individually, and these strategies offer us the opportunity to annotate the tokens, about which the model, regardless of the Viterbi algorithm, is most uncertain about.

Minimum Token Probability

The Minimum Token Probability (MTP) strategy employs as measure of informativeness the probability of the most probable assignment at time t regardless of the most probable sequence assignment chosen by the Viterbi algorithm, i.e.,

$$\mathcal{S}^{MTP}(t, \mathbf{x}) = \max_{j \in \mathcal{Y}} p(y_t = j | \mathbf{x}, \mathbf{L}) \tag{5.9}$$

This strategy, as the others of this section, greedily samples the tokens whose highest probability among the labels is the lowest.

[1] In order to have a light notation we omit the fact that when the most informative token is the first annotated token of a sentence, the whole sentence, with just one annotated token, is added to the training set $\mathcal{T}_i$

Maximum Token Entropy

Entropy [121] is a traditional measure for measuring uncertainty [5, 47]. The rationale of the Maximum Token Entropy (MTE) strategy is that, if more than one label have the same assigned marginal probability, the entropy will be high, that is,

$$\mathcal{S}^{MTE}(t, \mathbf{x}) = \sum_{j \in \mathcal{Y}} p(y_t = j|\mathbf{x}, \mathbf{L}) \cdot \log p(y_t = j|\mathbf{x}, \mathbf{L}) \tag{5.10}$$

In order to directly plug the $\mathcal{S}^{TE}$ function into the AL framework of Algorithm 2, we removed the minus sign at the beginning of the entropy formula. This allow us to use the min operator with a maximum entropy approach.

Minimum Token Margin

The Minimum Token Margin (MTM) strategy is a variant of the margin sampling strategy introduced in [113]. It calculates the informativeness by considering the two most probable assignments and by subtracting the highest probability by the lowest, regardless of the most probable sequence assignment; i.e.,

$$\mathcal{S}^{MTM}(t, \mathbf{x}) = \max_{j \in \mathcal{Y}} p(y_t = j|\mathbf{x}, \mathbf{L}) - \max_{j \in \mathcal{Y}}{}' p(y_t = j|\mathbf{x}, \mathbf{L}) \tag{5.11}$$

where $\max'$ calculates the second maximum value.

5.3.2 Viterbi Strategies

Differently from the strategy of Section 5.3.2, the AL strategies described in this section take into consideration the most probable sequence assignments obtained via the Viterbi algorithm. The rationale is that, with these strategies, the measure of uncertainty is chosen according to the outcome of the Viterbi algorithm.

Minimum Viterbi Probability

Minimum Viterbi Probability (MVP) takes as measure of informativeness the probability of the label chosen by the Viterbi algorithm in order to obtain the most probable assignment $\mathbf{y}^*$.

Formally:

$$\mathcal{S}^{MVP}(t, \mathbf{x}) = p(y_t^*|\mathbf{x}, \mathbf{L}) \tag{5.12}$$

It can be considered as the traditional minimum confidence strategy of [26], and it is the base of the strategies adopted in [161].

Minimum Viterbi Margin

The Minimum Viterbi Margin (MVM) strategy calculates the difference of the sequence probabilities of the two most probable sequence assignments at the variation of the label at time t. When the difference at time t is low, the Viterbi algorithm, in that time, chooses between two almost equiprobable sequence assignments.

Formally:

$$\mathcal{S}^{MVM}(t, \mathbf{x}) = p(\mathbf{y}^*_{y^*_t}|\mathbf{x}, \mathbf{L}) - p(\mathbf{y}'^*_{y'^*_t}|\mathbf{x}, \mathbf{L}) \tag{5.13}$$

where $\mathbf{y}'^*$ is the second most probable assignment when we change the label at time t, and y^*_t and y'^*_t are respectively the assignment chosen at time t by the Viterbi algorithm and the token assignments that returns the second most probable sequence assignment.

Maximum Viterbi Pseudo Entropy

The Maximum Viterbi Pseudo Entropy (MVPE), as its greedy counterpart, adopts entropy as the measure of informativeness. This strategy calculates for each token the "pseudo" entropy of the most probable sequences at the variation of the label associated with the token at position t. The prefix "pseudo" is used because, even though it is calculated with the same formula of entropy, the summation is over all possible labels that can be associated with a token $\mathcal{Y}$, and not all the possible sequence assignments.

$$\mathcal{S}^{MVPE}(t, \mathbf{x}) = \sum_{j \in \mathcal{Y}} p(\mathbf{y}^*_{y_t=j}|\mathbf{x}, \mathbf{L}) \cdot \log p(\mathbf{y}^*_{y_t=j}|\mathbf{x}, \mathbf{L}) \tag{5.14}$$

where $\mathbf{y}^*_{y_t=j}$ represents the most probable assignment with the label at time t constrained to the value j. As in the MTE strategy (Section 5.3.1) the minus sign is removed in order to plug the functions directly into the AL framework of Algorithm 2.

Minimum Expectation

PLCRFs allow us to inspect one token at a time in order to decide if it is worth to annotate. This fact gives us the possibility of exploiting two quantities in order to estimate the informativeness of a token: the sequence probability, usually adopted in the traditional AL for sequence labeling, and the marginal probabilities of the single tokens as in Section 5.3.1. The Minimum Expectation (ME) strategy combines the marginal probability $p(y_t = j|\mathbf{x}, \mathbf{L})$, i.e., the probability that a particular label j is associated with the token at time t, and the maximum sequence probability $p(\mathbf{y}^*_{y_t=j}|\mathbf{x}, \mathbf{L})$:

$$\mathcal{S}^{ME}(t, \mathbf{x}) = \sum_{j \in \mathcal{Y}} p(y_t = j|\mathbf{x}, \mathbf{L}) \cdot p(\mathbf{y}^*_{y_t=j}|\mathbf{x}, \mathbf{L}) \tag{5.15}$$

Here the maximum sequence probability is seen as a function, and what we calculate is its expected value. The rationale of this strategy is adding to the training set

those tokens where both the sequence probability returned by the Viterbi algorithm and the marginal probability of the considered labels is low.

Minimum Expectation Difference

Given that the ME strategy gives a high weight to the sequence probability, one might expect that tokens that belong to longer sequences are selected more frequently, given that the sequence probability of longer sequences is usually lower than that of shorter ones. One way to normalize this difference is subtracting the current maximum sequence probability, that is, the maximum sequence probability calculated without any new label estimation, from the expected value obtained from the estimation of the label assignment of the token.

$$\mathcal{S}^{MED}(t, \mathbf{x}) = \mathcal{S}^{ME}(t, \mathbf{x}) - p(\mathbf{y}^*|\mathbf{x}, \mathbf{L}) \tag{5.16}$$

The rationale of this strategy is that when the expected value is far from the maximum value (that is, the value returned by the Viterbi algorithm) it means that we have uncertainty on the token taken into consideration.

5.3.3 Random Strategy

The random strategy represents the naive strategy, that is, sampling random tokens without any external information. It is used as a baseline to compare the real effectiveness of the more sophisticated strategies.

5.4 Experiments

5.4.1 Experimental Setting

For the initial training set $\mathcal{T}_1$ we chose to annotate the necessary number of sentences to reach the 100 tokens mark. These first sequences are fully annotated, i.e., each token in the sequence is annotated with its true label. The dimension of the batch update B has been chosen with the idea that the work done by the AL process (retraining, estimation, and the like), with the exception of the annotation phase, is not comparable with the effort of the human annotators. These assumptions allow us to adopt a small update batch size, that we fixed at $B = 50$. A smaller update batch size would have been impractical and not realistic in a real AL scenario where actual annotators have to annotate several tokens and then wait, for each annotated token, the retraining of the classifier. We fixed the number of AL iterations n at 40; a study on stopping criteria is out of the scope of this work.

For each strategy and for each dataset, we report averaged results of three runs with a different randomly sampled initial training set $\mathcal{T}_1$.

Table 5.1: Training Data Statistics.

Dataset	# sequences	# tokens	# labels
CoNLL2000	8936	211727	11
CoNLL2000POS	8936	211727	35
CoNLL2003	17290	254979	4
NLPBA2004	18546	492551	5

Dataset	avg. ann. len.	avg. pos. tok.	avg. seq. len.
CoNLL2000	1.6	20.6	24
CoNLL2000POS	1	20.8	24
CoNLL2003	1.4	2.5	15
NLPBA2004	2.5	5.9	27

Dataset	avg. pos. ann.	% ann. co-occ.	% diff. ann. co-occ.
CoNLL2000	12	98%	98%
CoNLL2000POS	20.8	100%	99%
CoNLL2003	2.2	45%	32%
NLPBA2004	3.1	72%	47%

Dataset(s)

We have experimented and evaluated the eight AL strategies of Section 5.3 on four different sequence labeling tasks: part-of-speech tagging, phrase chunking, named-entity recognition, and bio-entity recognition. In order to do so we employed four sequence labeling datasets:

- the CoNLL2000 dataset, created for the CoNLL all-phrase chunking shared task [144];
- the CoNLL2000POS dataset, i.e., the CoNLL2000 dataset annotated with part-of-speech labels instead of chunking labels;
- the CoNLL2003 dataset, created for the CoNLL named-entity recognition shared task [145];
- the NLPBA2004 dataset, created for the biomedical entity recognition shared task [57].

We have processed the CoNLL2000 dataset in order to use it as dataset for the POS labeling task, setting the POS label already present in the CoNLL2000 dataset as token label instead of the chunk label.

All the datasets are publicly available and are standard benchmarks in sequence labeling tasks.

Table 5.1 shows several statistics that characterize each dataset:

- # sequences is the number of total sequences in the dataset;
- # tokens is the number of tokens in the dataset;
- # labels is the number of positive labels (labels different from the negative label O) with which the dataset is annotated;
- avg. ann. len. is the average length, in tokens, of the annotations (sequences of tokens that refer to the same label);
- avg. pos. tok. is the average number of tokens in a sequence annotated with a positive label;
- avg. seq. len. is the average length of a sequence;
- avg. pos. ann. is the average number of positive annotation in a sequence;
- % ann. co-occ. is the percentage of sequences that have more than one positive annotation;
- % diff. ann. co-occ. is the percentage of sequences that have two or more annotation with different labels.

As we can see from the statistics in Table 5.1 the datasets are very different from each other, in terms of dimensions, number of labels, distribution of the labels, sequence length, etc. This heterogeneity allowed us to test the AL strategies on different "data conditions", thus yielding more robust empirical evaluation.

The test sets we used to evaluate our proposed strategies are composed by 2012 sequences and 47377 tokens for the CoNLL2000 and CoNLL2000POS datasets, by 3452 sequences and 46394 tokens for the CoNLL2003 dataset, and by 3856 sequences and 101039 tokens for the NLPBA2004 dataset.

Features

The set of features we have used as representation of the tokens that compose the sequences are different for each dataset, with the exception of the CoNLL2003 dataset and the NLPBA2004 dataset, that share the same set of features. The features adopted in every dataset are quite standard for the task they are meant for. In the following we list the features adopted for each dataset.

CoNLL2000 The words in the [-2, +2] window, word bigrams in the [-1, +1] window, part-of-speech unigrams, bigrams and trigrams in the [-2, +2] window.

CoNLL2000POS The words in the [-2, +2] window, the character prefixes of length 1, 2, 3, 4, and the character suffixes of length 1, 2, 3, 4 of the current word, and a Boolean feature that states if the current word contains a hyphen.

CoNLL2003 and NLPBA2004 The words in the [-2, +2] window, word 4-grams before and after the current word, lowercase words and lowercase word bigrams in the [-2, +2] window, lowercase word trigrams in the [-1, +1] window, POS tag of the words, word bigrams, and word trigrams in the [-2, +2] window, phrase chunk of the words, word bigrams, and word trigrams in the [-2, +2] window, the character prefixes of length 2 and 3 and the character suffixes of length 2 and 3 of the current word. A feature the encode the word type, e.g., "starts with a capital letter", "contains only digits", "contains punctuation", etc., is also used.

Evaluation Measures

As evaluation measure we have used the token-level variant of the F_1 measure of Section 2.3.2.

5.4.2 Results

Learning Curves

From the learning curves of Figures 5.1, 5.2, 5.4 and 5.5 it is clear that most of the strategies have the same trend throughout the different datasets. This results is somewhat different from the results obtained in [117] in which there is not a clear winner among the strategies they proposed in a fully-labeled scenario. The strategies that perform particularly bad (worse than the RAND strategy in CoNLL2000POS and in CoNLL2003 dataset) in all the datasets are the MTE and MTP. This is expected, because the choice of the measure of informativeness related to the token without taking in consideration the Viterbi path is suboptimal in this task. Surprisingly, the MTM strategy even though based on the same principle of MTE and MTP, is very effective in most of the datasets. The most effective strategies, that is, the ones that are the faster to help the classifier to reach a better accuracy are the MTM, MVM, and MVP, in particular the margin-based strategies perform very good in all the datasets. The MVPE strategy performs particularly bad in the CoNLL2003 dataset but it performs better than the RAND strategy in the other datasets. The performance of the ME strategy is always above the average, in particular it is the best performing strategy in the NLPBA2004 dataset. However, in the CoNLL2003 dataset its performance is similar to the RAND's performance. Looking at the data, as expected, ME tend to choose tokens belonging to the longest sequences, regardless if the sequence is already partially annotated, that is, it tends to choose tokens from the same sequences. This behavior is not particularly relevant on the CoNLL2003 dataset given that the average number of positive tokens per sentence is not high (2.5, see Table 5.1). For the other datasets, the average number of positive tokens per sentence is high, and so the ME strategy is particularly effective. The MED strategy has the most heterogeneous behavior among the datasets. It shows average performances in the CoNLL2000 dataset and NLPBA2004 dataset, but is slower

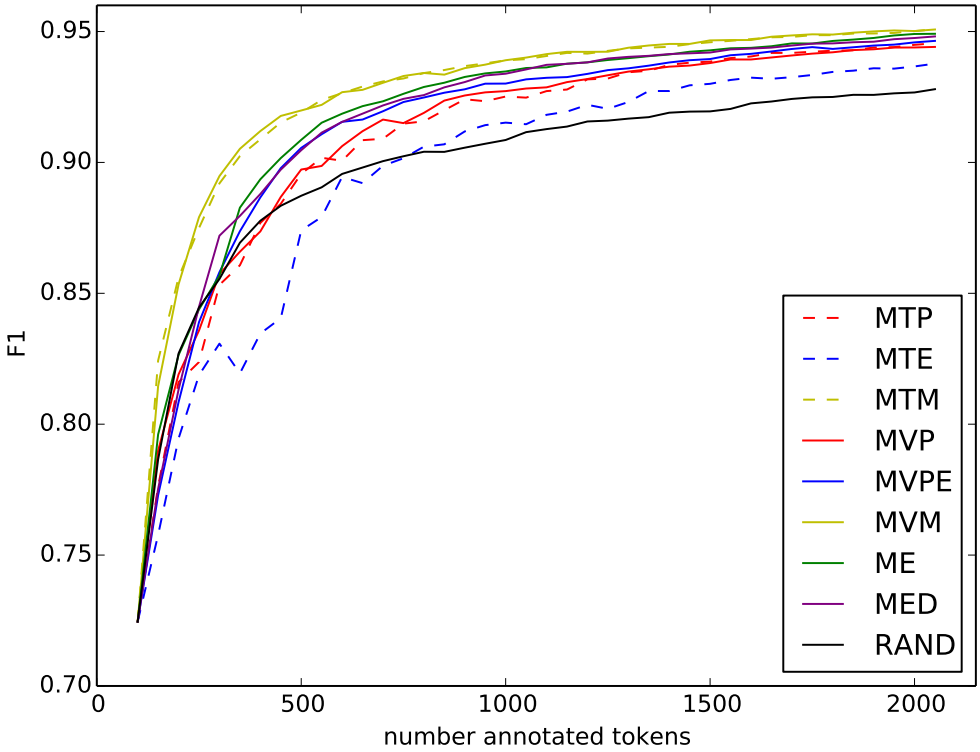

Figure 5.1: F_1 results on the CoNLL2000 dataset. The annotation is done only on the selected token. The maximum number of annotated tokens used (2100) represents $\sim$1% of the entire training set.

than the RAND strategy in the CoNLL2003 and CoNLL2000POS datasets.

Performance Drop Phenomenon

The learning curves we obtained by applying the proposed AL strategies on the CoNLL2003 dataset (Figure 5.2) do not show a monotonic or quasi-monotonic behavior as in Figures 5.1, 5.4 and 5.5, but instead there are some "random" drops of performance. In order to explain the phenomenon in Figure 5.2, that we called *performance drop phenomenon*, we need to introduce two AL concepts: *exploitation* and *exploration*. In AL a strategy is called *exploitative* if it inspects regions of the sample space around the decision boundaries, while an *explorative* strategy inspects also regions of the sample space far from the decision boundaries.

The strategies that we have tested so far are highly exploitative, given that the informativeness measure is given by the classifier decisions on the examples close to the decision boundaries. We argue that the behavior of the AL strategies on

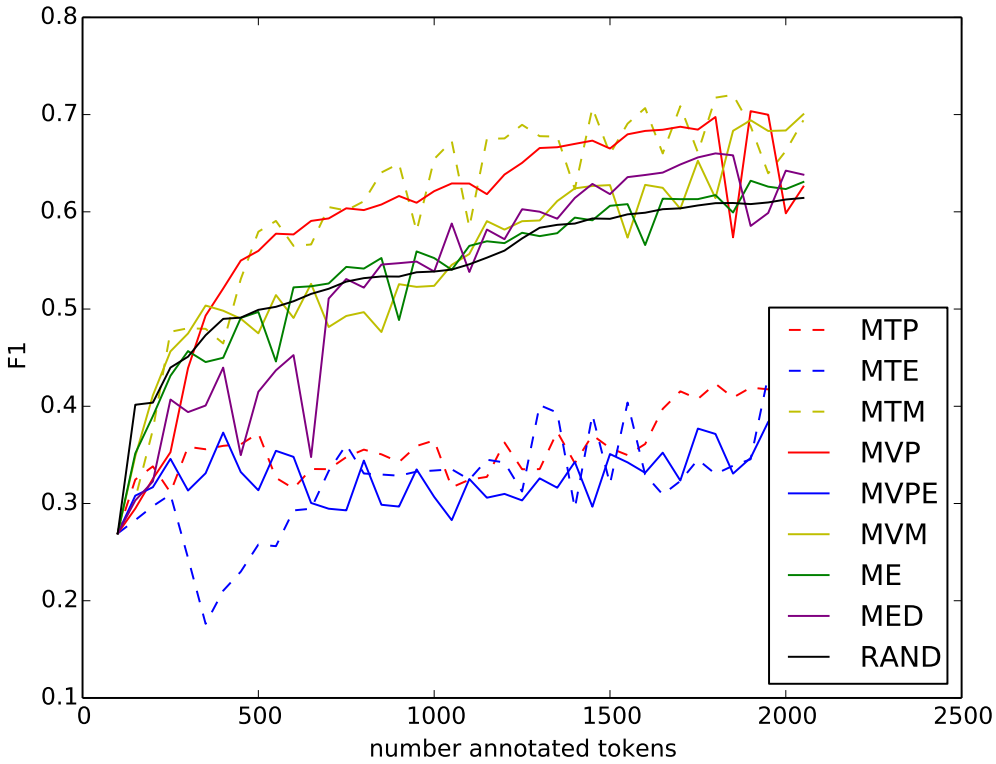

Figure 5.2: F_1 results on the CoNLL2003 dataset. The annotation is done only on the selected token. 2100 annotated tokens represent the ∼0.8% of the CoNLL2003 training set.

the CoNLL2003 dataset (Figure 5.2) is mostly due to two reasons. The first reason is that named entities are quite sparse, that is, they often appear just once in a sentence (55% of the CoNLL dataset has only one named entity per sentence), and have a heterogeneous structure, as opposed to the dense and homogeneous structures to be found in the chunk and POS datasets, where the co-occurrence of different annotations in a sentence is around 100% (see Table 5.1). The second is the lack of exploration around the boundaries, given that the strategies choose only a single token to annotate and to add to the training set. It happens, as one can see from Figure 5.2, that after some retraining of the classifier the obtained classifier is actually less accurate than the previous one. For example, given that a human annotator does not know the accuracy of a classifier at a given time, she may decide to stop the annotation at iteration 10. Iteration 10 may be, by chance, a point where the classifier has a loss of accuracy. Thus, with this particular side effect, AL can be considered useless.

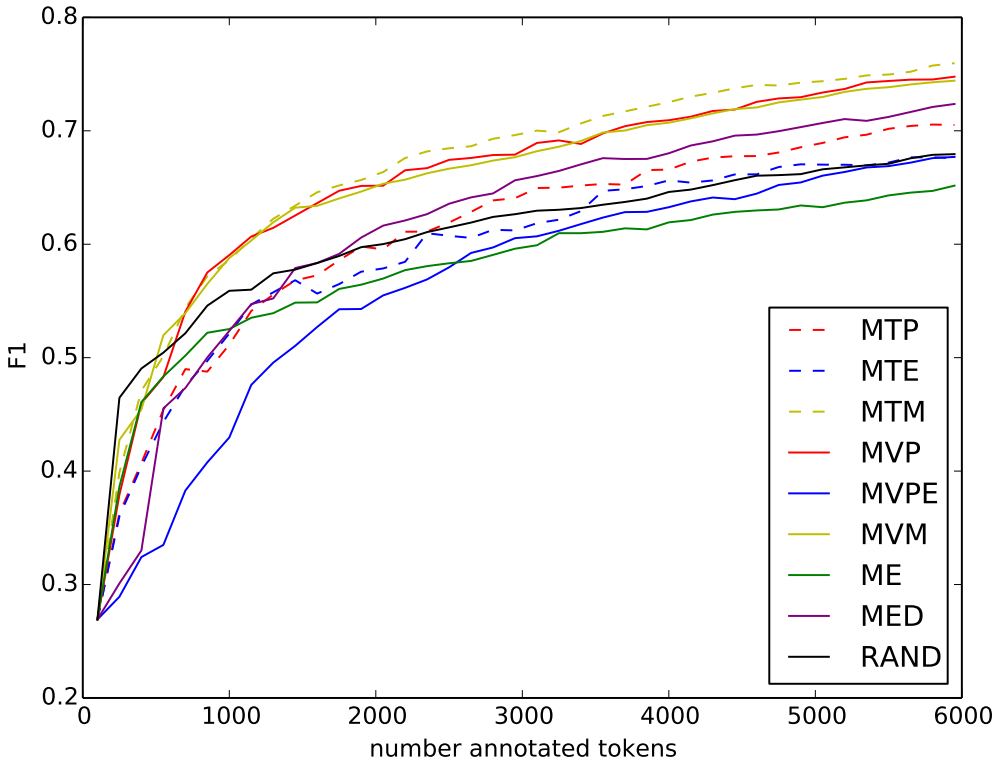

Figure 5.3: F_1 results on the CoNLL2003 dataset. The annotation is done on the selected token plus the previous one and the next one. 6100 annotated tokens represent the $\sim$2.4% of the entire training set.

A similar phenomenon, called *missed class effect* [147], happens in AL when the exploration phase is overtaken by the exploitation phase, which means that regions far from the decision boundaries are never explored. This leads to a classifier that is agnostic about a region of the sample space, which thus performs poorly on new samples that happen to lie in that region. In [147] the missed class effect is solved by adding an exploratory phase to the AL strategies, that is, choosing an entire sequence instead of a single token. This solution is not suitable in our context, given that we will loose all the advantages we have in the partially labeled scenario. However, a way to damp this effect without renouncing to the partial labeling is annotating, for each chosen token, the previous token and the next token. This will give an exploratory phase to the strategies without sensibly increasing the effort of the human annotators. We argue that, by adopting the annotating system proposed in [21] as explained in Section 5.3, annotating the tokens around the most informative token can be only slightly more difficult than annotating only the most informative

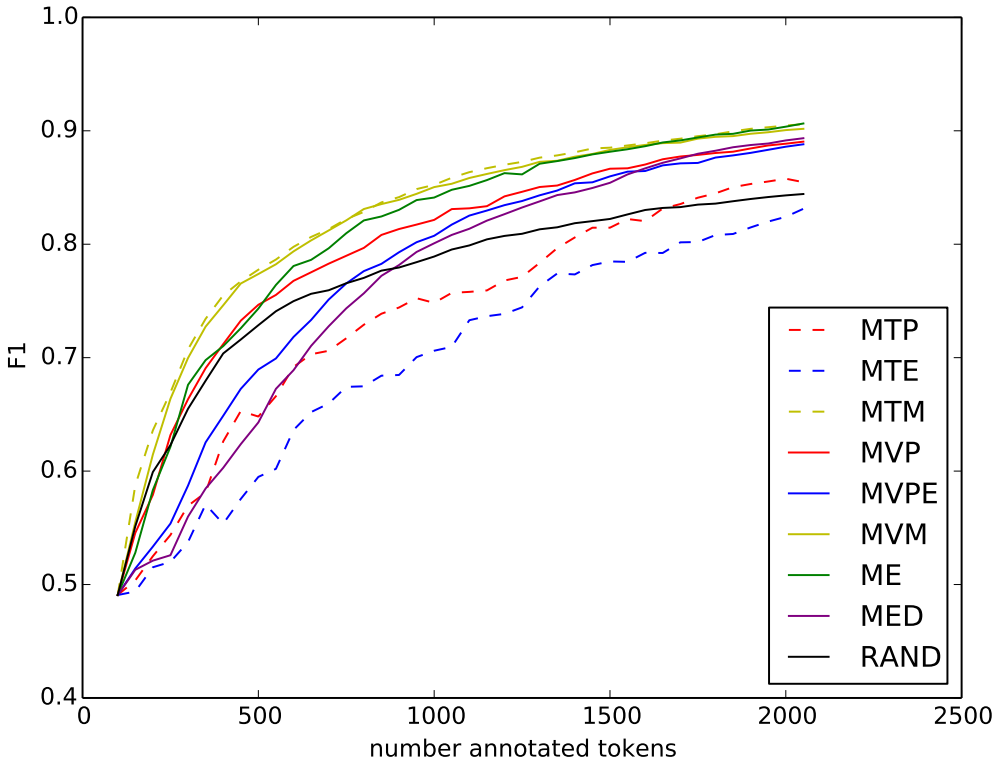

Figure 5.4: F_1 results on the CoNLL2000POS dataset. The annotation is done only on the selected token. 2100 annotated tokens represent ∼1% of the entire training set.

token. The learning curves of the AL strategies adopting the three tokens annotation methods are shown in Figure 5.3; as we can see the performance drop phenomenon has completely disappeared. Anyway, in the curves shown in Figure 5.3 it is possible to observe the same trend observed in the rest of the datasets. The margin-based strategies are the best performer along with the MVP strategy. Surprisingly, the ME strategy, that is among the best-performing strategy in the rest of the datasets, performs particularly bad in the CoNLL2003 dataset with a three-tokens annotation.

Given the strong similarity between the *performance drop phenomenon* and the *missed class effect*, further studies are necessary to understand if they are indeed the same phenomenon.

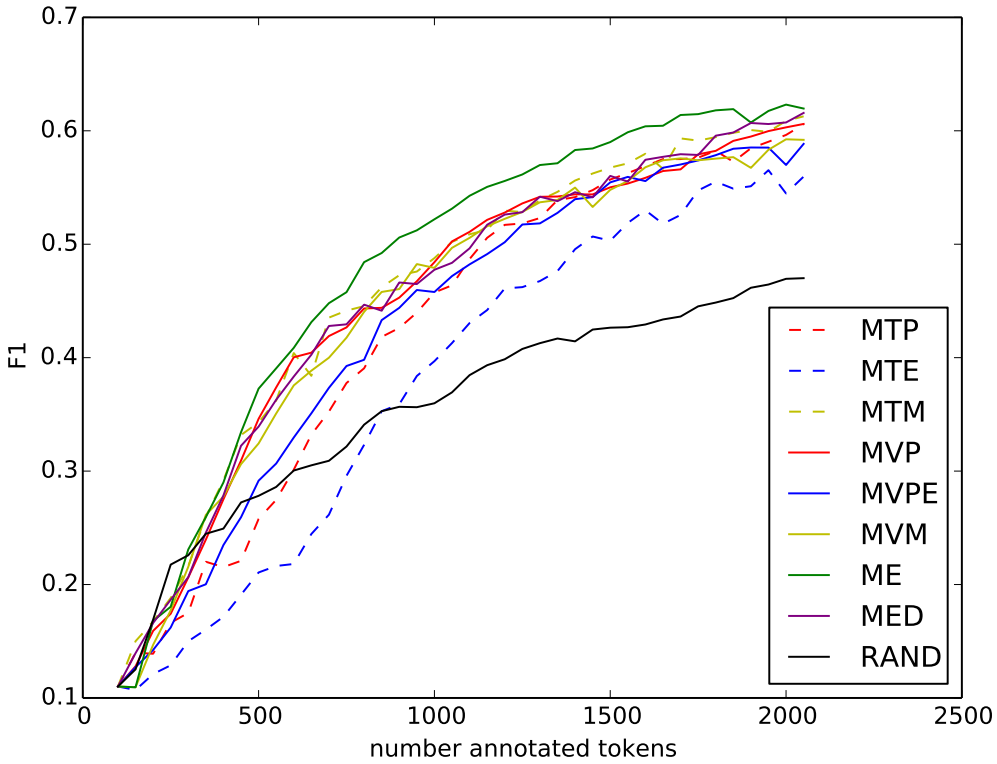

Figure 5.5: F_1 results on the NLPBA2004 dataset. The annotation is done only on the selected token. 2100 annotated tokens represent the ∼0.4% of the entire training set.

5.5 Related Work

In this section we summarize the most important works in pool-based AL; for a more exhaustive survey on other AL scenarios, see [116]. The first works on pool-based AL have been proposed in the field of text classification. [68] opened the way for pool-based AL with uncertainty sampling, followed by [81], which adopted a query-by-committee strategy with an expectation maximization technique in order to exploit also unlabeled documents. [148] proposed instead an AL technique specific for support vector machines (SVMs).

AL has also been adopted to help the construction of datasets in sequence labeling tasks. [22] employed a query-by-committee approach in order to select the most informative sentence for the part-of-speech tagging task. [58] used entropy as the measure of informativeness, combined with a coefficient of representativeness so as to choose the most representative and informative examples for each batch. [113] adopted as the uncertainty measure the so-called "margin", that consists in the

difference between the two highest posterior probabilities.

The works in sequence labeling in which a selection unit smaller than a sequence is successfully employed are [122], [146] and [161]. In [122], the authors adopt SVMs as the learning algorithm, and propose two strategies that combine three criteria: informativeness, representativeness, and diversity. SVMs allow them to use as selection unit a subset of the token in a sequence, without annotating, in any way, the other tokens in the sequence. In [146] the most uncertain tokens of the sequence are individually annotated, but the labels of non-annotated tokens in the sequence are then chosen by the classifier. The rationale of this is that humans do not need to annotate tokens which the classifier is certain about. [161] is the work that inspired our work: the authors adopt a minimum-confidence selection strategy with re-estimation using PLCRFs. In their most effective method they leave unannotated the tokens of the sequence not annotated by the human annotator. Differently from our work, they show that by adopting AL with partially labeled sequences with re-estimation, the cost of annotation can be dramatically reduced, thus obtaining the same level of performance of the classifier trained on the entire, fully-labeled dataset.

5.6 Summary

In this chapter we have presented several AL strategies tailored for the PLCRF in a pool-based scenario. We have tested the proposed strategies on four different datasets for four different sequence labeling tasks. We have found that on datasets with certain characteristics a particular phenomenon that makes the entire AL process useless shows up. This phenomenon consists in random drops of accuracy of the classifiers learnt during the AL process. We have proposed a first solution for this problem that does not have a relevant impact on the human annotation effort. Differently from other similar work in the field of AL, in this study we have shown that margin-based strategies constantly achieve good performance on four tasks with very different data characteristics.

6
Conclusions

6.1 Summary

In this thesis we have started from the study and the application of supervised machine learning systems for sequence labeling, we have passed through the study of the quality of training data, including an analysis of the creation of training data via AL for sequence labeling tasks. The first task we have faced has been the problem of IE from text in the medical domain, then we have moved to the more complex problem of aspect-oriented opinion mining from product reviews. After that, we have shifted to the study of the quality of training data, that in turn brought us to the study of the creation of training data with techniques of AL applied in a semi-supervised learning scenario.

In Chapter 2 we have shown that the proposed two-stage method, which extracts information from medical documents using two levels of granularity, outperforms the baseline method for the most frequent tags but is slightly inferior on infrequent tags, while the ensemble method, which merges the outcomes of the proposed two-stage method and the baseline method, outperforms the baseline on both frequent and infrequent tags. We also showed that the accuracy levels obtained by the proposed methods are higher than the inter-coder agreement levels between the two human coders.

In Chapter 3 we have considered the problem of aspect-oriented opinion mining at the sentence level, devising a sequence of increasingly powerful CRF models. We started from the simple linear-chain CRF model and finished with a hierarchical multi-label model that jointly models both the overall opinion expressed in a review and the set of aspect-specific opinions expressed in each sentence of the review. We evaluated the proposed models against a dataset of hotel reviews in which the set of aspects and the opinions expressed concerning them are manually annotated at the sentence level. We empirically demonstrated that both hierarchical and multi-label factors lead to improved predictions of aspect-oriented opinions.

In Chapter 4 we have investigated the impact of the quality of training data on the accuracy of an IE system in the medical domain. Our experimental results indicate that only a marginal decrease in extraction accuracy derives from a substantial decrease in training data quality, which indicates the robustness of the CRFs to the

use of training data of suboptimal quality.

Finally, in Chapter 5 we empirically studied the behaviour of several AL strategies for sequence labeling tasks in a semi-supervised scenario. In this semi-supervised scenario, margin-based strategies proved to be, constantly and considerably, faster to learn than the other proposed strategies on four common sequence labeling tasks. We also discovered a particular phenomenon, that consists in random drops of accuracy of the classifiers learnt during the AL process, that occurs in tasks in which the elements to extract are sparse and their structure is heterogeneous i.e., named-entity recognition. This phenomenon makes the AL process in the semi-supervised scenario very unpredictable and thus useless. We proposed a solution for such problem based on the annotation of tokens close to the token chosen by the AL strategies, that does not have a relevant impact on the effort of human annotators.

6.2 Future Directions

There are several avenues for future research that this thesis leaves open.

In Chapter 2, the findings presented would be strengthened if the same results could be replicated on further clinical datasets, possibly composed by clinical narratives different from mammography reports. The current limited public availability of datasets of clinical narratives is still an obstacle to progress in this direction, but initiatives like the i2b2 series of challenges already mentioned in Section 2.4 are a valiant step towards removing it.

The works that can be undertaken starting from the study presented in Chapter 3 are several. One of them is testing the proposed models on domains different from hotel reviews, possibly with a higher number of aspects. Another one is improving the feature engineering phase by taking into consideration features that co-occur in co-occurring aspects, in order to better recognize the presence of co-occurring aspects . A third one is studying a method that removes factors between aspects that never occur together, in order to lighten the multi-label model so as to make it more scalable on problems with a richer aspect set. However, the most interesting direction is probably the extension of the hierarchical and multi-label models to a semi-supervised scenario, thus exploiting also unlabeled data along with the labeled ones. This extension can be tackled in two main ways. The first, is the adoption of Hidden Conditional Random Fields (HCRFs), which are capable of learning a CRF model using unlabeled data and labeled data as well as training data. The second is the use of a self-training algorithm in order to learn from automatically annotated training data, i.e., annotating unlabeled data with a classifier trained on a small amount of labeled data, and then training a new classifier with the original labeled data plus the automatically annotated data. Both methods allow using a large amount of unlabeled data, that are obviously cheaper and easier to get than human-annotated data, to improve the accuracy of the trained CRF model.

Possible future works concerning the theme tackled in Chapter 4 are tightly

dependent on the availability of IE datasets with assessment of agreement between coders, which are very hard to find. Moreover it would be interesting to assess whether the use of artificial noise is plausible, studying methods to produce such noise and comparing it with the natural noise produced by human coders. It would be also useful, in this case, to investigate if there is some kind of common pattern or relation between different types of natural noise.

An interesting extension of the studies conducted in Chapter 5 would be to support the AL - semi-supervised learning pair with a *transfer learning* module in order to further reduce the human annotation effort. The field of transfer learning studies methods to adapt the data annotated to train a classifier for a source task A, to a different but similar target task B. Adding a transfer learning layer to the AL and semi-supervised layers would allow the creation of AL strategies that, taking into consideration a transformation of the data from the source domain to the target domain, bring information not only to the classifier meant for the source task but also to the classifier meant for the target task. In such a scenario the human effort for annotating training data would be significantly reduced.

A

Appendix

Table A.1: Inter-annotator segment-level aspect agreement, expressed in terms of F_1 (higher is better). 0, 1 and 2 are the different annotators and 0-1 means pairwise agreement between annotator 0 and annotator 1.

	F_1			
Aspect	0-1	0-2	1-2	Average
Overall : Other	0.649	0.625	0.547	0.607
Implicit : Other	0.201	0.154	0.147	0.167
Explicit : Other	0.529	0.552	0.356	0.479
Overall : Service	0.709	0.701	0.746	0.719
Implicit : Service	0.167	0.100	0.103	0.123
Explicit : Service	0.704	0.682	0.667	0.684
Overall : Room	0.802	0.789	0.788	0.793
Implicit : Room	0.356	0.222	0.211	0.263
Explicit : Room	0.730	0.722	0.667	0.706
Overall : Location	0.844	0.735	0.805	0.795
Implicit : Location	0.415	0.176	0.267	0.286
Explicit : Location	0.829	0.694	0.606	0.710
Overall : Building	0.533	0.527	0.598	0.553
Implicit : Building	0.000	0.286	0.000	0.095
Explicit : Building	0.526	0.524	0.514	0.521
Overall : Value	0.552	0.618	0.556	0.575
Implicit : Value	0.211	0.182	0.000	0.131
Explicit : Value	0.581	0.554	0.545	0.560
Overall : Food	0.807	0.811	0.764	0.794
Implicit : Food	0.324	0.333	0.261	0.306
Explicit : Food	0.806	0.725	0.693	0.741
Overall : Check-in	0.383	0.400	0.609	0.464
Implicit : Check-in	0.000	0.182	0.000	0.061
Explicit : Check-in	0.432	0.410	0.600	0.481
Overall : Cleanliness	0.684	0.766	0.750	0.733
Implicit : Cleanliness	0.000	0.333	0.000	0.111
Explicit : Cleanliness	0.714	0.773	0.730	0.739
Overall : Business	0.727	0.667	0.500	0.631
Implicit : Business	0.000	0.000	1.000	0.333
Explicit : Business	0.800	0.571	0.500	0.624
Overall : Macro	0.669	0.664	0.693	0.675
Implicit : Macro	0.167	0.197	0.199	0.188
Explicit : Macro	0.665	0.621	0.588	0.625

Table A.2: Inter-annotator segment-level opinion agreement (restricted to the true positive aspects for each segment), expressed in terms of sMAE^M (lower is better). 0, 1 and 2 are the different annotators and 0-1 means pairwise agreement between annotator 0 and annotator 1.

	MAE^M			
Aspect	0-1	0-2	1-2	Average
Overall : Other	0.200	0.347	0.378	0.308
Implicit : Other	0.379	0.000	0.122	0.167
Explicit : Other	0.140	0.316	0.331	0.262
Overall : Service	0.222	0.189	0.246	0.219
Implicit : Service	0.000	0.000	0.000	0.000
Explicit : Service	0.226	0.094	0.179	0.167
Overall : Room	0.184	0.200	0.189	0.191
Implicit : Room	0.000	0.000	0.000	0.000
Explicit : Room	0.147	0.128	0.166	0.147
Overall : Location	0.210	0.327	0.242	0.259
Implicit : Location	0.183	0.000	0.000	0.061
Explicit : Location	0.172	0.184	0.000	0.119
Overall : Building	0.150	0.219	0.082	0.150
Implicit : Building	0.000	0.000	0.000	0.000
Explicit : Building	0.154	0.123	0.000	0.092
Overall : Value	0.102	0.449	0.054	0.202
Implicit : Value	0.000	0.000	0.000	0.000
Explicit : Value	0.000	0.472	0.065	0.179
Overall : Food	0.079	0.264	0.358	0.234
Implicit : Food	0.000	0.222	0.000	0.074
Explicit : Food	0.098	0.186	0.287	0.190
Overall : Check-in	0.009	0.000	0.000	0.003
Implicit : Check-in	0.000	0.000	0.000	0.000
Explicit : Check-in	0.000	0.000	0.000	0.000
Overall : Cleanliness	0.167	0.175	0.000	0.114
Implicit : Cleanliness	0.000	0.000	0.000	0.000
Explicit : Cleanliness	0.175	0.016	0.000	0.064
Overall : Business	0.087	0.000	0.000	0.029
Implicit : Business	0.000	0.000	0.000	0.000
Explicit : Business	0.000	0.000	0.000	0.000
Overall : Macro	0.141	0.217	0.155	0.171
Implicit : Macro	0.056	0.022	0.012	0.030
Explicit : Macro	0.111	0.152	0.103	0.122

Bibliography

[1] Dana Angluin. Queries revisited. In *Proceedings of the 12th International Conference on Algorithmic Learning Theory (ALT 2001)*, pages 12–31, Washington, US, 2001.

[2] Ron Artstein and Massimo Poesio. Inter-coder agreement for computational linguistics. *Computational Linguistics*, 34(4):555–596, 2008.

[3] Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. Evaluation measures for ordinal regression. In *Proceedings of the 9th IEEE International Conference on Intelligent Systems Design and Applications (ISDA 2009)*, pages 283–287, Pisa, IT, 2009.

[4] Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. SentiWordNet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In *Proceedings of the 7th Conference on Language Resources and Evaluation (LREC 2010)*, Valletta, MT, 2010.

[5] Jason Baldridge and Miles Osborne. Active learning and the total cost of annotation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP 2004)*, pages 9–16, 2004.

[6] Petra S. Bayerl and Karsten I. Paul. What determines inter-coder agreement in manual annotations? a meta-analytic investigation. *Computational Linguistics*, 37(4):699–725, 2011.

[7] Dimitri P. Bertsekas. *Nonlinear programming*. Athena Scientific, 1999.

[8] Christopher M. Bishop. *Pattern recognition and machine learning*. Springer, Heidelberg, DE, 2006.

[9] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *The Journal of Machine Learning Research*, 3:993–1022, 2003.

[10] Said Bleik, Wei Xiong, Yiran Wang, and Min Song. Biomedical concept extraction using concept graphs and ontology-based mapping. In *Proceedings of the 4th IEEE International Conference on Bioinformatics and Biomedicine (BIBM 2010)*, pages 553–556, Hong Kong, CN, 2010.

[11] Avrim Blum and Shuchi Chawla. Learning from labeled and unlabeled data using graph mincuts. In *Proceedings of the 18th International Conference on Machine Learning (ICML 2001)*, pages 19–26, Williamstown, US, 2001.

[12] Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the 11th Annual Conference on Computational Learning Theory (COLT 1998)*, pages 92–100, Madison, US, 1998.

[13] Philip Bramsen, Pawan Deshpande, Yoong K. Lee, and Regina Barzilay. Finding temporal order in discharge summaries. In *Proceedings of the 30th AMIA Annual Symposium (AMIA 2006)*, pages 81–85, Washington, US, 2006.

[14] Richard H. Byrd, Jorge Nocedal, and Robert B. Schnabel. Representations of quasi-Newton matrices and their use in limited memory methods. *Mathemathical Programming*, 63:129–156, 1994.

[15] Wendy W. Chapman and John N. Dowling. Inductive creation of an annotation schema for manually indexing clinical conditions from emergency department reports. *Journal of Biomedical Informatics*, 39(2):196–208, 2006.

[16] Eugene Charniak and Mark Johnson. Coarse-to-fine n-best parsing and maxent discriminative reranking. In *The 43rd Annual Meeting of the Association for Computational Linguistics (ACL 2005)*, Ann Arbor,US, 2005.

[17] David Cohn, Les Atlas, and Richard Ladner. Improving generalization with active learning. *Machine Learning*, 15(2):201–221, 1994.

[18] Michael Collins. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *The 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, pages 1–8, Philadelphia, US, 2002.

[19] Isaac G. Councill, Ryan T. McDonald, and Leonid Velikovich. What's great and what's not: learning to classify the scope of negation for improved sentiment analysis. In *Proceedings of the Workshop on Negation and Speculation in Natural Language Processing (NeSp-NLP 2010)*, pages 51–59, Uppsala, SE, 2010.

[20] Aron Culotta and Andrew McCallum. Confidence estimation for information extraction. In *Proceedings of Human Language Technology Conference and North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2004)*, pages 109–112, Boston, US, 2004.

[21] Aron Culotta and Andrew McCallum. Reducing labeling effort for structured prediction tasks. In *Proceedings of the Twentieth National Conference on Artificial Intelligence and the Seventeenth Innovative Applications of Artificial Intelligence Conference, (AAAI 2005)*, pages 746–751, Pittsburgh, US, 2005.

[22] Ido Dagan and Sean Engelson. Committee-based sampling for training probabilistic classifiers. In *Proceeding of the 12th International Conference on Machine Learning (ICML 1995)*, pages 150–157, Lake Tahoe, US, 1995.

[23] Franca Debole and Fabrizio Sebastiani. An analysis of the relative hardness of Reuters-21578 subsets. *Journal of the American Society for Information Science and Technology*, 56(6):584–596, 2005.

[24] Marco Dinarelli, Alessandro Moschitti, and Giuseppe Riccardi. Discriminative reranking for spoken language understanding. *IEEE Transactions on Audio, Speech & Language Processing*, 20(2):526–539, 2012.

[25] Duy Dinh and Lynda Tamine. Biomedical concept extraction based on combining the content-based and word order similarities. In *Proceedings of the 26th ACM Symposium on Applied Computing*, pages 1159–1163, TaiChung, TW, 2011.

[26] Andrea Esuli, Diego Marcheggiani, and Fabrizio Sebastiani. Sentence-based active learning strategies for information extraction. In *Proceedings of the 1st Italian Information Retrieval Workshop (IIR 2010)*, pages 41–45, Padua, IT, 2010.

[27] Andrea Esuli, Diego Marcheggiani, and Fabrizio Sebastiani. An enhanced CRFs-based system for information extraction from radiology reports. *Journal of Biomedical Informatics*, 46(3):425–435, 2013.

[28] Andrea Esuli and Fabrizio Sebastiani. Evaluating information extraction. In *Proceedings of the Conference on Multilingual and Multimodal Information Access Evaluation (CLEF 2010)*, pages 100–111, Padua, IT, 2010.

[29] Andrea Esuli and Fabrizio Sebastiani. Training data cleaning for text classification. *ACM Transactions on Information Systems*, 31(4), 2013.

[30] Barbara Di Eugenio and Michael Glass. The kappa statistic: A second look. *Computational Linguistics*, 30(1):95–101, 2004.

[31] David A. Evans, Nicholas D. Brownlow, William R. Hersh, and Emily M. Campbell. Automating concept identification in the electronic medical record: An experiment in extracting dosage information. In *Proceedings of the Annual Fall Symposium of the American Medical Informatics Association*, pages 388–392, Washington, US, 1996.

[32] G. David Forney Jr. The Viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278, 1973.

[33] Dayne Freitag and Andrew McCallum. Information extraction with HMMs and shrinkage. In *Proceedings of the AAAI-99 Workshop on Machine Learning for Information Extraction*, pages 31–36, Orlando, US, 1999.

[34] Yoav Freund, H. Sebastian Seung, Eli Shamir, and Naftali Tishby. Selective sampling using the query by committee algorithm. *Machine learning*, 28(2-3):133–168, 1997.

[35] Atsushi Fujii, Takenobu Tokunaga, Kentaro Inui, and Hozumi Tanaka. Selective sampling for example-based word sense disambiguation. *Computational Linguistics*, 24(4):573–597, 1998.

[36] Robert Gaizauskas and Yorick Wilks. Information extraction: Beyond document retrieval. *Journal of Documentation*, 54(1):70–105, 1998.

[37] Stuart Geman and Donald Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Trans. Pattern Anal. Mach. Intell.*, 6(6):721–741, 1984.

[38] Nadia Ghamrawi and Andrew McCallum. Collective multi-label classification. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management (CIKM 2005)*, pages 195–200, Bremen, DE, 2005.

[39] Dan Gillick. Sentence boundary detection and the problem with the U.S. In *Proceedings of the Conference of the North American Chapter of the Association of Computational Linguistics (NAACL 2009)*, pages 241–244, Boulder, US, 2009.

[40] Alessandra Giordani and Alessandro Moschitti. Translating questions to sql queries with generative parsers discriminatively reranked. In *The 24th International Conference on Computational Linguistics (COLING 2012)*, pages 401–410, Mumbai, IN, 2012.

[41] Catherine Grady and Matthew Lease. Crowdsourcing document relevance assessment with Mechanical Turk. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 172—179, Los Angeles, US, 2010.

[42] Ralph Grishman, Silja Huttunen, and Roman Yangarber. Information extraction for enhanced access to disease outbreak reports. *Journal of Biomedical Informatics*, 35(4):236—246, 2002.

[43] Barry Haddow and Beatrice Alex. Exploiting multiply annotated corpora in biomedical information extraction tasks. In *Proceedings of the 6th Conference on Language Resources and Evaluation (LREC 2008)*, Marrakech, MA, 2008.

[44] Henk Harkema, Ian Roberts, Robert Gaizauskas, and Mark Hepple. Information extraction from clinical records. In *Proceedings of the 4th UK e-Science All Hands Meeting (AHM 2005)*, pages 39–43, Nottingham, UK, 2005.

[45] Xuming He, Richard S. Zemel, and Miguel A. Carreira-Perpinán. Multiscale conditional random fields for image labeling. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2004)*, pages 695–702, Washington, US, 2004.

[46] Minqing Hu and Bing Liu. Mining and summarizing customer reviews. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2004)*, pages 168–177, Seattle, US, 2004.

[47] Rebecca Hwa. Sample selection for statistical parsing. *Computational Linguistics*, 30(3):253–276, 2004.

[48] Surinder Jassar, Zaiyi Liao, and Lian Zhao. Impact of data quality on predictive accuracy of ANFIS-based soft sensor models. In *Proceedings of the 2009 IEEE World Congress on Engineering and Computer Science (WCECS 2009)*, volume II, San Francisco, US, 2009.

[49] Min Jiang, Yukun Chen, Mei Liu, S. Trent Rosenbloom, Subramani Mani, Joshua C. Denny, and Hua Xu. A study of machine-learning-based approaches to extract clinical entities and their assertions from discharge summaries. *Journal of the American Medical Informatics Association*, 18(5):601–606, 2011.

[50] Wei Jin, Hung Hay Ho, and Rohini K. Srihari. Opinionminer: a novel machine learning system for web opinion mining and extraction. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2009)*, pages 1195–1204, Paris, FR, 2009.

[51] Thorsten Joachims, Thomas Hofmann, Yisong Yue, and Chun-Nam Yu. Predicting structured objects with support vector machines. *Communications of the ACM*, 52(11):97–104, 2009.

[52] Richard Johansson and Alessandro Moschitti. Relational features in fine-grained opinion analysis. *Computational Linguistics*, 39(3):473–509, 2013.

[53] Siddhartha Jonnalagadda, Trevor Cohen, Stephen Wu, and Graciela Gonzalez. Enhancing clinical concept extraction with distributional semantics. *Journal of Biomedical Informatics*, 45(1):129—140, 2012.

[54] Siddhartha R. Jonnalagadda, Dingcheng Li, Sunghwan Sohn, Stephen T. Wu, Kavishwar Wagholikar, Manabu Torii, and Hongfang Liu. Coreference analysis in clinical notes: A multi-pass sieve with alternate anaphora resolution modules. *Journal of the American Medical Informatics Association*, 19(5):867–874, 2012.

[55] Ning Kang, Zubair Afzal, Bharat Singh, Erik M. van Mulligen, and Jan A. Kors. Using an ensemble system to improve concept extraction from clinical records. *Journal of Biomedical Informatics*, 45(3):423–428, 2012.

[56] Azme Khamis, Zuhaimy Ismail, Khalid Haron, and Ahmad T. Mohammed. The effects of outliers data on neural network performance. *Journal of Applied Sciences*, 5(8):1394–1398, 2005.

[57] Jin-Dong Kim, Tomoko Ohta, Yoshimasa Tsuruoka, Yuka Tateisi, and Nigel Collier. Introduction to the bio-entity recognition task at JNLPBA, year = 2004, address = Geneva, CH. In *Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications*, pages 70–75.

[58] Seokhwan Kim, Yu Song, Kyungduk Kim, Jeong-Won Cha, and Gary Geunbae Lee. MMR-based active machine learning for bio named entity recognition. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics (HLT-NAACL 2006)*, pages 69–72, New York, US, 2006.

[59] Soo-Min Kim and Eduard Hovy. Automatic identification of pro and con reasons in online reviews. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (COLING/ACL 2006)*, pages 483—490, Sydney, AU, 2006.

[60] Dan Klein, Joseph Smarr, Huy Nguyen, and Christopher D. Manning. Named entity recognition with character-level models. In *Proceedings of the 7th Conference on Natural Language Learning (CoNLL 2003)*, pages 180–183, Edmonton, CA, 2003.

[61] Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. The MIT Press, 2009.

[62] Klaus Krippendorff. *Content analysis: An introduction to its methodology*. Sage, Thousand Oaks, US, 2004.

[63] Frank R. Kschischang, Brendan J. Frey, and H-A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, 2001.

[64] John Lafferty, Andrew McCallum, and Fernando Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning (ICML 2001)*, pages 282–289, Williamstown, US, 2001.

[65] Ken Lang. Newsweeder: Learning to filter netnews. In *Proceedings of the 12th International Conference on Machine Learning (ICML 1995)*, pages 331–339, Tahoe City, US, 1995.

[66] Angeliki Lazaridou, Ivan Titov, and Caroline Sporleder. A Bayesian model for joint unsupervised induction of sentiment, aspect and discourse representations. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL 2013)*, pages 1630–1639, Sofia, BL, 2013.

[67] David D. Lewis and Jason Catlett. Heterogeneous uncertainty sampling for supervised learning. In *Proceedings of 11th International Conference on Machine Learning (ICML 1994)*, pages 148–156, New Brunswick, US, 1994.

[68] David D. Lewis and William A. Gale. A sequential algorithm for training text classifiers. In *Proceedings of the 17th ACM International Conference on Research and Development in Information Retrieval (SIGIR 1994)*, pages 3–12, Dublin, IE, 1994.

[69] Dingcheng Li, Karin Kipper-Schuler, and Guergana Savova. Conditional random fields and support vector machines for disorder named entity recognition in clinical texts. In *Proceedings of the ACL Workshop on Current Trends in Biomedical Natural Language Processing (BioNLP 2008)*, pages 94–95, Columbus, US, 2008.

[70] Fangtao Li, Chao Han, Minlie Huang, Xiaoyan Zhu, Ying-Ju Xia, Shu Zhang, and Hao Yu. Structure-aware review mining and summarization. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING 2010)*, pages 653–661, Bejing, CN, 2010.

[71] Ying Li, Sharon Lipsky Gorman, and Noémie Elhadad. Section classification in clinical notes using supervised hidden Markov model. In *Proceedings of the 2nd ACM International Health Informatics Symposium*, pages 744–750, Arlington, US, 2010.

[72] Bing Liu. *Sentiment analysis and opinion mining.* Morgan & Claypool Publishers, San Rafael, US, 2012.

[73] Tim Macer, Mark Pearson, and Fabrizio Sebastiani. Cracking the code: What customers say, in their own words. In *Proceedings of the 50th Annual Conference of the Market Research Society (MRS 2007)*, Brighton, UK, 2007.

[74] Gideon S. Mann and Andrew McCallum. Generalized expectation criteria for semi-supervised learning with weakly labeled data. *The Journal of Machine Learning Research*, 11:955–984, 2010.

[75] Diego Marcheggiani and Thierry Artières. An experimental comparison of active learning strategies for partially labeled sequences. Technical report. 2014.

[76] Diego Marcheggiani and Fabrizio Sebastiani. On the effects of low-quality training data on information extraction from clinical reports. Presented at the *5th Italian Information Retrieval Workshop (IIR 2014)*. Rome, IT, 2014.

[77] Diego Marcheggiani, Oscar Täckström, Andrea Esuli, and Fabrizio Sebastiani. Hierarchical multi-label conditional random fields for aspect-oriented opinion mining. In *Proceedings of the 36th European Conference on Information Retrieval (ECIR 2014)*, Amsterdam, NL, 2014.

[78] Andrew McCallum. Information extraction: Distilling structured data from unstructured text. *Queue*, 3(9):48–57, 2005.

[79] Andrew McCallum, Dayne Freitag, and Fernando C.N. Pereira. Maximum entropy Markov models for information extraction and segmentation. In *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)*, pages 591–598, Stanford, US, 2000.

[80] Andrew McCallum and Wei Li. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *Proceedings of the 7th Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology (HLT-NAACL 2003)*, pages 188–191, Edmonton, CA, 2003.

[81] Andrew McCallum and Kamal Nigam. Employing em and pool-based active learning for text classification. In *Proceedings of the Fifteenth International Conference on Machine Learning (ICML 1998)*, pages 350–358, Madison, US, 1998.

[82] Andrew McCallum, Karl Schultz, and Sameer Singh. Factorie: Probabilistic programming via imperatively defined factor graphs. In *Proceedings of the 23rd Annual Conference on Neural Information Processing Systems (NIPS 2009)*, pages 1249–1257, Vancouver, CA, 2009.

[83] Ryan McDonald, Kerry Hannan, Tyler Neylon, Mike Wells, and Jeff Reynar. Structured models for fine-to-coarse sentiment analysis. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL 2007)*, pages 432–439, Prague, CZ, 2007.

[84] John McNaught and William J. Black. Information extraction: The task. In Sophia Ananiadou and John McNaught, editors, *Text Mining for Biology and Biomedicine*, pages 143–176. Artech House Books, London, UK, 2006.

[85] Stephane M. Meystre, Guerguana K. Savova, Karin C. Kipper-Schuler, and John F. Hurdle. Extracting information from textual documents in the electronic health record: A review of recent research. In A. Geissbuhler and

C. Kulikowski, editors, *IMIA Yearbook of Medical Informatics*, pages 128–144. Schattauer Publishers, Stuttgart, DE, 2008.

[86] Einat Minkov, Richard C. Wang, and William W. Cohen. Extracting personal names from email: Applying named entity recognition to informal text. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing (HLT-EMNLP 2005)*, pages 443–450, Vancouver, CA, 2005.

[87] Samaneh Moghaddam and Martin Ester. ILDA: Interdependent LDA model for learning latent aspects and their ratings from online product reviews. In *Proceedings of the 34th ACM SIGIR International Conference on Research and Development in Information Retrieval (SIGIR 2011)*, pages 665–674, Bejing, CN, 2011.

[88] Alessandro Moschitti, Qi Ju, and Richard Johansson. Modeling topic dependencies in hierarchical text categorization. In *The 50th Annual Meeting of the Association for Computational Linguistics (ACL 2012)*, pages 759–767, Jeju Island, KR, 2012.

[89] Alessandro Moschitti, Daniele Pighin, and Roberto Basili. Semantic role labeling via tree kernel joint inference. In *The 10th Conference on Computational Natural Language Learning (CoNLL 2006)*, pages 61–68, New York, US, 2006.

[90] Kevin P. Murphy, Yair Weiss, and Michael I. Jordan. Loopy belief propagation for approximate inference: An empirical study. In *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence (UAI 99)*, pages 467–476, Stockholm, SE, 1999.

[91] Agnieszka Mykowiecka, Małgorzata Marciniak, and Anna Kupść. Rule-based information extraction from patients' clinical data. *Journal of Biomedical Informatics*, 42(5):923–936, 2009.

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

[93] Truc-Vien T. Nguyen and Alessandro Moschitti. Structural reranking models for named entity recognition. *Intelligenza Artificiale*, 6(2):177–190, 2012.

[94] Truc-Vien T. Nguyen, Alessandro Moschitti, and Giuseppe Riccardi. Conditional random fields: Discriminative training over statistical features for named entity recognition. In *EVALITA 2009 workshop, the 11st International Conference of the Italian Association for Artificial Intelligence (AI*IA 2009)*, Reggio Emilia, IT, 2009.

[95] Bo Pang and Lillian Lee. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL 2004)*, pages 271–278, Barcelona, ES, 2004.

[96] Bo Pang and Lillian Lee. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1/2):1–135, 2008.

[97] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up? Sentiment classification using machine learning techniques. In *Proceedings of the 7th Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, pages 79–86, Philadelphia, US, 2002.

[98] Rebecca J. Passonneau and Bob Carpenter. The benefits of a model of annotation. In *The 7th Linguistic Annotation Workshop & Interoperability with Discourse (LAW VII & ID)*, pages 187–195, Sofia, BG, 2013.

[99] Jon Patrick and Ming Li. High accuracy information extraction of medication information from clinical notes: 2009 i2b2 medication extraction challenge. *Journal of the American Medical Informatics Association*, 17:524–527, 2010.

[100] Fuchun Peng and Andrew McCallum. 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 2004)*, pages 329–336, Boston, US, 2004.

[101] Emanuele Pianta, Christian Girardi, and Roberto Zanoli. The TextPro tool suite. In *Proceedings of the 6th Language Resources and Evaluation Conference (LREC 2008)*, Marrakech, MA, 2008.

[102] David Pinto, Andrew McCallum, Xing Wei, and W. Bruce Croft. Table extraction using conditional random fields. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval (SIGIR 2003)*, pages 235–242, Toronto, CA, 2003.

[103] Ariadna Quattoni, Sybor Wang, L-P. Morency, Michael Collins, and Trevor Darrell. Hidden conditional random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(10):1848–1852, 2007.

[104] Joaquin Quiñonero-Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D. Lawrence, editors. *Dataset shift in machine learning*. The MIT Press, Cambridge, US, 2009.

[105] Lawrence R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. pages 267–296. Morgan Kaufmann Publishers Inc., San Francisco, US, 1990.

[106] Christian P. Robert and George Casella. *Monte Carlo statistical methods*. Springer, 2004.

[107] Fabio Roli, Giorgio Giacinto, and Gianni Vernazza. Methods for designing multiple classifier systems. In *Proceedings of the 2nd International Workshop on Multiple Classifier Systems (MCS 2001)*, pages 78–87, Cambridge, UK, 2001.

[108] Donald F. Rossin and Barbara D. Klein. Data errors in neural network and linear regression models: An experimental comparison. *Data Quality Journal*, 5(1), 1999.

[109] Nicholas Roy and Andrew McCallum. Toward optimal active learning through sampling estimation of error reduction. In *Proceedings of the 18th International Conference on Machine Learning (ICML 2001)*, pages 441–448, Williamstown, US, 2001.

[110] Claude Sammut and Michael Harries. Concept drift. In Claude Sammut and Geoffrey I. Webb, editors, *Encyclopedia of Machine Learning*, pages 202–205. Springer, Heidelberg, DE, 2011.

[111] Sunita Sarawagi. Information extraction. *Foundations and Trends in Databases*, 1(3):261—377, 2008.

[112] Sunita Sarawagi and William W Cohen. Semi-Markov conditional random fields for information extraction. In *Proceedings of Advances in Neural Information Processing Systems (NIPS 2004)*, pages 1185–1192, Vancouver, CA, 2004.

[113] Tobias Scheffer, Christian Decomain, and Stefan Wrobel. Active hidden Markov models for information extraction. In *Proceedings of the 4th International Conference on Advances in Intelligent Data Analysis (IDA 2001)*, pages 309–318, Cascais, PT, 2001.

[114] Greg Schohn and David Cohn. Less is more: Active learning with support vector machines. In *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)*, pages 839–846, Stanford, US, 2000.

[115] Burr Settles. Biomedical named entity recognition using conditional random fields and rich feature sets. In *Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and Its Applications (JNLPBA 2004)*, pages 104–107, Geneva, CH, 2004.

[116] Burr Settles. *Active learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, 2012.

[117] Burr Settles and Mark Craven. An analysis of active learning strategies for sequence labeling tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2008)*, pages 1070–1079, Honolulu, US, 2008.

[118] Aliaksei Severyn and Alessandro Moschitti. Structural relationships for large-scale learning of answer re-ranking. In *The 35th International ACM SIGIR conference on research and development in Information Retrieval (SIGIR 2012)*, pages 741–750, Portland, US, 2012.

[119] Kristie Seymore, Andrew McCallum, and Roni Rosenfeld. Learning hidden Markov model structure for information extraction. In *Proceedings of the AAAI-99 Workshop on Machine Learning for Information Extraction*, pages 37–42, Orlando, US, 1999.

[120] Fei Sha and Fernando Pereira. Shallow parsing with conditional random fields. In *Proceedings of the 7th Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology (HLT-NAACL 2003)*, pages 134–141, Edmonton, CA, 2003.

[121] Claude E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423, 623–656, 1948.

[122] Dan Shen, Jie Zhang, Jian Su, Guodong Zhou, and Chew-Lim Tan. Multi-criteria-based active learning for named entity recognition. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL 2004)*, pages 589–596, Barcelona, ES, 2004.

[123] Tawanda Sibanda, Tian He, Peter Szolovits, and Özlem Uzuner. Syntactically-informed semantic category recognition in discharge summaries. In *Proceedings of the Annual Symposium of the American Medical Informatics Association (AMIA 2006)*, pages 714–718, Washington, US, 2006.

[124] Sameer Singh, Karl Schultz, and Andrew McCallum. Bi-directional joint inference for entity resolution and segmentation using imperatively-defined factor graphs. In *Proceedins of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD 2009)*, pages 414–429, Bled, SI, 2009.

[125] Sameer Singh, Michael Wick, and Andrew McCallum. Distantly labeling data for large scale cross-document coreference. *Computing Research Repositories, (CoRR)*, 2010.

[126] Sameer Singh, Limin Yao, Sebastian Riedel, and Andrew McCallum. Constraint-driven rank-based learning for information extraction. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter*

of the Association for Computational Linguistics (HLT-NAACL 2010), pages 729–732, Los Angeles, US, 2010.

[127] Cristian Sminchisescu, Atul Kanaujia, and Dimitris N. Metaxas. Conditional models for contextual human motion recognition. *Computer Vision and Image Understanding*, 104(2-3):210–220, 2006.

[128] Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Y. Ng. Cheap and fast - but is it good? Evaluating non-expert annotations for natural language tasks. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing (EMNLP 2008)*, pages 254–263, Honolulu, US, 2008.

[129] Stephen Soderland, David Aronow, David Fisher, Jonathan Aseltine, and Wendy Lehnert. Machine learning of text analysis rules for clinical records. Technical Report TE-39, Center for Intelligent Information Retrieval, University of Massachusetts, Amherst, US, 1995.

[130] Anastasia Sotelsek-Margalef and Julio Villena-Román. MIDAS: An Information-Extraction Approach to Medical Text Classification. *Procesamiento del Lenguaje Natural*, 41:97–104, 2008.

[131] Philip J. Stone, Dexter C. Dunphy, Marshall S. Smith, and Daniel M. Ogilvie. *The General Inquirer: A Computer Approach to Content Analysis*. The MIT Press, Cambridge, US, 1966.

[132] Charles Sutton and Andrew McCallum. Collective segmentation and labeling of distant entities in information extraction. In *ICML Workshop on Statistical Relational Learning and Its Connections to Other Fields*, 2004.

[133] Charles Sutton and Andrew McCallum. An introduction to conditional random fields for relational learning. In Lise Getoor and Ben Taskar, editors, *Introduction to Statistical Relational Learning*, pages 93–127. The MIT Press, Cambridge, US, 2007.

[134] Charles Sutton and Andrew McCallum. An introduction to conditional random fields. *Foundations and Trends in Machine Learning*, 4(4):267–373, 2012.

[135] Charles Sutton, Andrew McCallum, and Khashayar Rohanimanesh. Dynamic conditional random fields: Factorized probabilistic models for labeling and segmenting sequence data. *Journal of Machine Learning Research*, 8:693–723, 2007.

[136] Jun Suzuki, Erik McDermott, and Hideki Isozaki. 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 (ACL/COLING 2006)*, pages 217—224, Sydney, AU, 2006.

[137] Christopher T. Symons, Nagiza F. Samatova, Ramya Krishnamurthy, Byung H. Park, Tarik Umar, David Buttler, Terence Critchlow, and David Hysom. Multi-criterion active learning in conditional random fields. In *Proceedings of the 18th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2006)*, pages 323–331, Washington, US, 2006.

[138] Oscar Täckström and Ryan T. McDonald. Discovering fine-grained sentiment with latent variable structured prediction models. In *Proceedings of the 33rd European Conference on Information Retrieval (ECIR 2011)*, pages 368–374, Dublin, IE, 2011.

[139] Oscar Täckström and Ryan T. McDonald. Semi-supervised latent variable models for sentence-level sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL 2011)*, pages 569–574, Portland, US, 2011.

[140] Ricky K. Taira, Stephen G. Soderland, and Rex M. Jakobovits. Automatic structuring of radiology free-text reports. *RadioGraphics*, 21(1):237–245, 2001.

[141] Benjamin Taskar, Vassil Chatalbashev, Daphne Koller, and Carlos Guestrin. Learning structured prediction models: a large margin approach. In *The 22nd International Conference on Machine Learning (ICML 2005)*, pages 896–903, Bonn, DE, 2005.

[142] Ivan Titov and Ryan T. McDonald. A joint model of text and aspect ratings for sentiment summarization. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics (ACL 2008)*, pages 308–316, Columbus, US, 2008.

[143] Ivan Titov and Ryan T. McDonald. Modeling online reviews with multi-grain topic models. In *Proceedings of the 17th International Conference on World Wide Web (WWW 2008)*, pages 111–120, Beijing, CN, 2008.

[144] Erik F. Tjong Kim Sang and Sabine Buchholz. Introduction to the CoNLL-2000 shared task: Chunking. In *Proceedings of the 2nd Workshop on Learning Language in Logic and 4th Conference on Computational Natural Language Learning (LLL/CoNLL 2000)*, pages 127–132. Lisbon, PT, 2000.

[145] Erik F. Tjong Kim Sang and Fien De Meulder. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the 7th Conference on Natural Language Learning (CONLL 2003)*, pages 142–147, Edmonton, CA, 2003.

[146] Katrin Tomanek and Udo Hahn. Semi-supervised active learning for sequence labeling. In *Proceedings of the 47th Annual Meeting of the Association for*

Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the AFNLP (ACL-IJCNLP 2009), pages 1039–1047, Singapore, 2009.

[147] Katrin Tomanek, Florian Laws, Udo Hahn, and Hinrich Schütze. On proper unit selection in active learning: co-selection effects for named entity recognition. In *Proceedings of the NAACL HLT 2009 Workshop on Active Learning for Natural Language Processing*, pages 9–17, Boulder, US, 2009.

[148] Simon Tong and Daphne Koller. Support vector machine active learning with applications to text classification. *Journal of Machine Learning Research*, 2:45–66, 2001.

[149] Manabu Torii, Kavishwar Wagholikar, and Hongfang Liu. Using machine learning for concept extraction on clinical documents from multiple data sources. *Journal of the American Medical Informatics Association*, 18(5):580–587, 2011.

[150] Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. Support vector machine learning for interdependent and structured output spaces. In *Proceedings of the 21 st International Conference on Machine Learning (ICML 2004)*, Banff, CA, 2004.

[151] Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453—1484, 2005.

[152] Yuta Tsuboi, Hisashi Kashima, Hiroki Oda, Shinsuke Mori, and Yuji Matsumoto. Training conditional random fields using incomplete annotations. In *Proceedings of the 22nd International Conference on Computational Linguistics (COLING 2008)*, pages 897–904, Manchester, UK, 2008.

[153] Peter D. Turney. Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL 2002)*, pages 417–424, Philadelphia, US, 2002.

[154] Özlem Uzuner, Yuan Luo, and Peter Szolovits. Evaluating the state of the art in automatic de-identification. *Journal of the American Medical Informatics Association*, 14(5):550–563, 2007.

[155] Özlem Uzuner, Imre Solti, and Eithon Cadag. Extracting medication information from clinical text. *Journal of the American Medical Informatics Association*, 17(5):514–518, 2010.

[156] Özlem Uzuner, Brett R. South, Shuying Shen, and Scott L. DuVall. 2010 i2b2/VA challenge on concepts, assertions, and relations in clinical text. *Journal of the American Medical Informatics Association*, 18(5):552–556, 2011.

[157] Martin J. Wainwright and Michael I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1/2):1–305, 2008.

[158] Hongning Wang, Yue Lu, and Chengxiang Zhai. Latent aspect rating analysis on review text data: A rating regression approach. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2010)*, pages 783–792, Washington, US, 2010.

[159] Hua Wang, Heng Huang, Feiping Nie, and Chris H. Ding. Cross-language web page classification via dual knowledge transfer using nonnegative matrix tri-factorization. In *Proceedings of the 34th ACM International Conference on Research and Development in Information Retrieval (SIGIR 2011)*, pages 933–942, Bejing, CN, 2011.

[160] Yefeng Wang and Jon Patrick. Cascading classifiers for named entity recognition in clinical notes. In *Proceedings of the RANLP 2009 Workshop on Biomedical Information Extraction*, pages 42–49, Borovets, BG, 2009.

[161] Dittaya Wanvarie, Hiroya Takamura, and Manabu Okumura. Active learning with subsequence sampling strategy for sequence labeling tasks. *Journal of the Natural Language Processing*, pages 153–173, 2011.

[162] William Webber and Jeremy Pickens. Assessor disagreement and text classifier accuracy. In *Proceedings of the 36th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2013)*, pages 929–932. Dublin, IE, 2013.

[163] Michael Wick, Khashayar Rohanimanesh, Kedar Bellare, Aron Culotta, and Andrew McCallum. SampleRank: Training factor graphs with atomic gradients. In *Proceedings of the 28th International Conference on Machine Learning (ICML 2011)*, pages 777–784, Bellevue, US, 2011.

[164] Michael Wick, Khashayar Rohanimanesh, Aron Culotta, and Andrew McCallum. Samplerank: Learning preferences from atomic gradients. In *The 24th Annual Conference on Neural Information Processing Systems Workshop on Advances in Ranking (NIPS 2009)*, Vancouver, CA, 2009.

[165] Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing (HLT/EMNLP 2005)*, pages 347–354, Vancouver, CA, 2005.

[166] Stephen J. Wright and Jorge Nocedal. *Numerical optimization*. Springer Verlag, 1999.

[167] David Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, pages 189–196, Cambridge, US, 1995.

[168] Dengyong Zhou, Jiayuan Huang, and Bernhard Schölkopf. Learning from labeled and unlabeled data on a directed graph. In *Proceedings of the 22nd International Conference on Machine Learning (ICML 2005)*, pages 1036–1043, Bonn, DE, 2005.

[169] Xiaohua Zhou, Hyoil Han, Isaac Chankai, Ann A. Prestrud, and Ari D. Brooks. Converting semi-structured clinical medical records into information and knowledge. In *Proceedings of the 21st International Conference on Data Engineering (ICDE 2005)*, pages 1162–1169, Tokyo, JP, 2005.

[170] Xiaojin Zhu and Andrew B. Goldberg. *Introduction to semi-supervised learning.* Synthesis lectures on artificial intelligence and machine learning. Morgan & Claypool Publishers, 2009.