Natural Language Processing for the Semantic Web

Diana Maynard
University of Sheffield

Kalina Bontcheva
University of Sheffield

Isabelle Augenstein
University College London

SYNTHESIS LECTURES ON THE SEMANTIC WEB: THEORY AND TECHNOLOGY #15

MORGAN & CLAYPOOL PUBLISHERS

Copyright © 2017 by Morgan & Claypool

Natural Language Processing for the Semantic Web

Diana Maynard, Kalina Bontcheva, and Isabelle Augenstein

www.morganclaypool.com

ISBN: 9781627059091 paperback
ISBN: 9781627056328 ebook

DOI 10.2200/S00741ED1V01Y201611WBE015

A Publication in the Morgan & Claypool Publishers series
SYNTHESIS LECTURES ON THE SEMANTIC WEB: THEORY AND TECHNOLOGY

Lecture #15
Series Editors: Ying Ding, *Indiana University*
Paul Groth, *Elsevier Labs*
Founding Editor Emeritus: James Hendler, *Rensselaer Polytechnic Institute*
Series ISSN
Print 2160-4711 Electronic 2160-472X

ABSTRACT

This book introduces core natural language processing (NLP) technologies to non-experts in an easily accessible way, as a series of building blocks that lead the user to understand key technologies, why they are required, and how to integrate them into Semantic Web applications. Natural language processing and Semantic Web technologies have different, but complementary roles in data management. Combining these two technologies enables structured and unstructured data to merge seamlessly. Semantic Web technologies aim to convert unstructured data to meaningful representations, which benefit enormously from the use of NLP technologies, thereby enabling applications such as connecting text to Linked Open Data, connecting texts to each other, semantic searching, information visualization, and modeling of user behavior in online networks.

The first half of this book describes the basic NLP processing tools: tokenization, part-of-speech tagging, and morphological analysis, in addition to the main tools required for an information extraction system (named entity recognition and relation extraction) which build on these components. The second half of the book explains how Semantic Web and NLP technologies can enhance each other, for example via semantic annotation, ontology linking, and population. These chapters also discuss sentiment analysis, a key component in making sense of textual data, and the difficulties of performing NLP on social media, as well as some proposed solutions. The book finishes by investigating some applications of these tools, focusing on semantic search and visualization, modeling user behavior, and an outlook on the future.

KEYWORDS

natural language processing, semantic web, semantic search, social media analysis, text mining, linked data, entity linking, information extraction, sentiment analysis

This book is a timely exposition of natural language processing and its role and importance for those seeking to apply semantic technologies. Clearly written with good coverage of the key topics and a comprehensive bibliography, the text will be invaluable for semantic web practitioners and more widely.

Prof John Davies
BT Research & Technology
Adastral Park UK
November 2016

Contents

1 Introduction . . . 1

1.1 Information Extraction . . . 2
1.2 Ambiguity . . . 4
1.3 Performance . . . 5
1.4 Structure of the Book . . . 7

2 Linguistic Processing . . . 9

2.1 Introduction . . . 9
2.2 Approaches to Linguistic Processing . . . 9
2.3 NLP Pipelines . . . 10
2.4 Tokenization . . . 12
2.5 Sentence Splitting . . . 14
2.6 POS Tagging . . . 15
2.7 Morphological Analysis and Stemming . . . 16
2.7.1 Stemming . . . 17
2.8 Syntactic Parsing . . . 19
2.9 Chunking . . . 21
2.10 Summary . . . 23

3 Named Entity Recognition and Classification . . . 25

3.1 Introduction . . . 25
3.2 Types of Named Entities . . . 26
3.3 Named Entity Evaluations and Corpora . . . 27
3.4 Challenges in NERC . . . 27
3.5 Related Tasks . . . 29
3.6 Approaches to NERC . . . 30
3.6.1 Rule-based Approaches to NERC . . . 30
3.6.2 Supervised Learning Methods for NERC . . . 31
3.7 Tools for NERC . . . 33

3.8 NERC on Social Media . . . 34
3.9 Performance . . . 34
3.10 Summary . . . 35

4 Relation Extraction . . . 37
4.1 Introduction . . . 37
4.2 Relation Extraction Pipeline . . . 37
4.3 Relationship between Relation Extraction and other IE Tasks . . . 39
4.4 The Role of Knowledge Bases in Relation Extraction . . . 40
4.5 Relation Schemas . . . 41
4.6 Relation Extraction Methods . . . 42
4.6.1 Bootstrapping Approaches . . . 42
4.7 Rule-based Approaches . . . 44
4.8 Supervised Approaches . . . 45
4.9 Unsupervised Approaches . . . 46
4.10 Distant Supervision Approaches . . . 47
4.10.1 Universal Schemas . . . 48
4.10.2 Hybrid Approaches . . . 49
4.11 Performance . . . 49
4.12 Summary . . . 50

5 Entity Linking . . . 53
5.1 Named Entity Linking and Semantic Linking . . . 54
5.2 NEL Datasets . . . 54
5.3 LOD-based Approaches . . . 55
5.3.1 DBpedia Spotlight . . . 55
5.3.2 YODIE: A LOD-based Entity Disambiguation Framework . . . 56
5.3.3 Other Key LOD-based Approaches . . . 57
5.4 Commercial Entity Linking Services . . . 58
5.5 NEL for Social Media Content . . . 59
5.6 Discussion . . . 60

6 Automated Ontology Development . . . 61
6.1 Introduction . . . 61
6.2 Basic Principles . . . 61
6.3 Term Extraction . . . 63

6.3.1 Approaches Using Distributional Knowledge 64
6.3.2 Approaches Using Contextual Knowledge 65
6.4 Relation Extraction 66
6.4.1 Clustering Methods 66
6.4.2 Semantic Relations 66
6.4.3 Lexico-syntactic Patterns 68
6.4.4 Statistical Techniques 68
6.5 Enriching Ontologies 69
6.6 Ontology Development Tools 70
6.6.1 Text2Onto 70
6.6.2 SPRAT 70
6.6.3 FRED 70
6.6.4 Semi-automatic Ontology Creation 71
6.7 Summary 71

7 Sentiment Analysis 73

7.1 Introduction 73
7.2 Issues in Opinion Mining 75
7.3 Opinion-Mining Subtasks 76
7.3.1 Polarity Recognition 76
7.3.2 Opinion Target Detection 76
7.3.3 Opinion Holder Detection 77
7.3.4 Sentiment Aggregation 77
7.3.5 Further Linguistic Subcomponents 78
7.4 Emotion Detection 79
7.5 Methods for Opinion Mining 81
7.6 Opinion Mining and Ontologies 83
7.7 Opinion-Mining Tools 85
7.8 Summary 86

8 NLP for Social Media 87

8.1 Social Media Streams: Characteristics, Challenges, and Opportunities 88
8.2 Ontologies for Representing Social Media Semantics 90
8.3 Semantic Annotation of Social Media 92
8.3.1 Keyphrase Extraction 92
8.3.2 Ontology-based Entity Recognition in Social Media 93

8.3.3 Event Detection . . . 99
8.3.4 Sentiment Detection and Opinion Mining . . . 100
8.3.5 Cross-media Linking . . . 101
8.3.6 Rumor Analysis . . . 102
8.3.7 Discussion . . . 103

9 Applications . . . 107

9.1 Semantic Search . . . 107
9.1.1 What is Semantic Search? . . . 108
9.1.2 Why Semantic Full-text Search? . . . 109
9.1.3 Semantic Search Queries . . . 110
9.1.4 Relevance Scoring and Retrieval . . . 111
9.1.5 Semantic Search Full-text Platforms . . . 111
9.1.6 Ontology-based Faceted Search . . . 114
9.1.7 Form-based Semantic Search Interfaces . . . 116
9.1.8 Semantic Search over Social Media Streams . . . 118
9.2 Semantic-Based User Modeling . . . 122
9.2.1 Constructing Social Semantic User Models from Semantic Annotations . . . 122
9.2.2 Discussion . . . 125
9.3 Filtering and Recommendations for Social Media Streams . . . 125
9.4 Browsing and Visualization of Social Media Streams . . . 126
9.5 Discussion and Future Work . . . 132

10 Conclusions . . . 135

10.1 Summary . . . 135
10.2 Future Directions . . . 135
10.2.1 Cross-media Aggregation and Multilinguality . . . 136
10.2.2 Integration and Background Knowledge . . . 137
10.2.3 Scalability and Robustness . . . 137
10.2.4 Evaluation, Shared Datasets, and Crowdsourcing . . . 138

Bibliography . . . 141

CHAPTER 1

Introduction

Natural Language Processing (NLP) is the automatic processing of text written in natural (human) languages (English, French, Chinese, etc.), as opposed to artificial languages such as programming languages, to try to "understand" it. It is also known as Computational Linguistics (CL) or Natural Language Engineering (NLE). NLP encompasses a wide range of tasks, from low-level tasks, such as segmenting text into sentences and words, to high-level complex applications such as semantic annotation and opinion mining. The Semantic Web is about adding semantics, i.e., meaning, to data on the Web, so that web pages can be processed and manipulated by machines more easily. One central aspect of the idea is that resources are described using unique identifiers, called *uniform resource identifiers (URIs)*. Resources can be entities, such as "Barack Obama," concepts such as "Politician" or relations describing how entities relate to one another, such as "spouse-of." NLP techniques provide a way to enhance web data with semantics, for example by automatically adding information about entities and relations and by understanding which real-world entities are referenced so that a URI can be assigned to each entity.

The goal of this book is to introduce readers working with, or interested in, Semantic Web technologies, to the topic of NLP and its role and importance in the field of the Semantic Web. Although the field of NLP has existed long before the advent of the Semantic Web, it has only been in recent years that its importance here has really come to the fore, in particular as Semantic Web technologies move toward more application-oriented realizations. The purpose of this book is therefore to explain the role of NLP and to give readers some background understanding about some of the NLP tasks that are most important for Semantic Web applications, plus some guidance about choosing methods and tools that fit most appropriately for a particular scenario. Ultimately, the reader should come away armed with the knowledge to understand the main principles and, if necessary, to choose suitable NLP technologies that can be used to enhance their Semantic Web applications.

The overall structure of the book is as follows. We first describe some of the core low-level components, in particular those which are commonly found in open source NLP toolkits and used widely in the community. We then show how these tools can be combined and used as input for the higher-level tasks such as Information Extraction, semantic annotation, social media analysis, and opinion mining, and finally how applications such as semantically enhanced information retrieval and visualization, and the modeling of online communities, can be built on top of these.

One point we should make clear is that when we talk about NLP in this book, we are referring principally to the subtask of Natural Language Understanding (NLU) and not to the related subtask of Natural Language Generation (NLG). While NLG is a useful task which is also relevant to the Semantic Web, for example in relaying the results of some application back to the user in a way that they can easily understand, and particularly in systems that require voice output of results, it goes outside the remit of this book, as it employs some very different techniques and tools. Similarly, there are a number of other tasks which typically fall under the category of NLP but are not discussed here, in particular those concerned with speech rather than written text. However, many applications for both speech processing and natural language generation make use of the low-level NLP tasks we describe. There are also some high-level NLP-based applications that we do not cover in this book, such as Summarization and Question Answering, although again these make use of the same low-level tools.

Most early NLP tools such as parsers (e.g., Schank's conceptual dependency parser [1]) were rule-based, due partly to the predominance of certain linguistic theories (primarily those of Noam Chomsky [2]), but also due to the lack of computational power which made machine learning methods infeasible. In the 1980s, machine learning systems started coming to the fore, but were still mainly used just to automatically create sets of rules similar to existing manually developed rule systems, using techniques such as decision trees. As statistical models became more popular, particularly in fields such as Machine Translation and Part-of-Speech tagging, where hard rule-based systems were often not sufficient to resolve ambiguities, Hidden Markov Models (HMMs) became popular, introducing the idea of weighted features and probablistic decision-making. In the last few years, deep learning and neural networks have also become very popular, following their spectacular success in the field of image recognition and computer vision (for example in the technology behind self-driving cars), although their success for NLP tasks is currently nowhere near as dramatic. Deep learning is essentially a branch of Machine Learning that uses multiple hierarchical levels of features that are learned in an unsupervised fashion. This makes it very suitable for working with big data, because it is fast and efficient, and does not require the manual creation of training data, unlike supervised machine learning systems. However, as will be demonstrated throughout the course of this book, one of the problems of NLP is that tools almost always need adapting to specific domains and tasks, and for real-world applications this is often easier with rule-based systems. In most cases, combinations of different methods are used, depending on the task.

1.1 INFORMATION EXTRACTION

Information extraction is the process of extracting information and turning it into structured data. This may include populating a structured knowledge source with information from an unstructured knowledge source [3]. The information contained in the structured knowledge base can then be used as a resource for other tasks, such as answering natural language queries or improving on standard search engines with deeper or more implicit forms of knowledge than that expressed in

the text. By unstructured knowledge sources, we mean free text, such as that found in newspaper articles, blog posts, social media, and other web pages, rather than tables, databases, and ontologies, which constitute structured text. Unless otherwise specified, we use the word *text* in the rest of this book to mean unstructured text.

When considering information contained in text, there are several types of information that can be of interest. Often regarded as the key components of text are proper names, also called *named entities* (NEs), such as persons, locations, and organizations. Along with proper names, *temporal expressions*, such as dates and times, are also often considered as named entities. Figure 1.1 shows some simple Named Entities in a sentence. Named entities are connected together by means of *relations*. Furthermore, there can be relations between relations, for example the relation denoting that someone is CEO of a company is connected to the relation that someone is an employee of a company, by means of a sub-property relation, since a CEO is a kind of employee. A more complex type of information is the *event*, which can be seen as a group of relations grounded in time. Events usually have participants, a start and an end date, and a location, though some of this information may be only implicit. An example for this is the opening of a restaurant. Figure 1.2 shows how entities are connected to form relations, which form events when grounded in time.

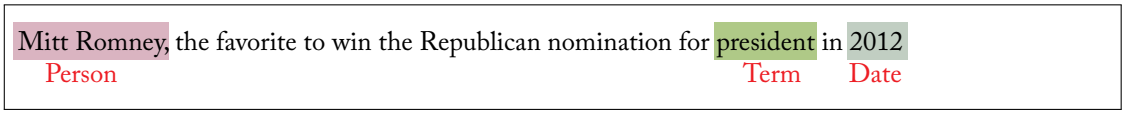


Figure 1.1: Examples of named entities.

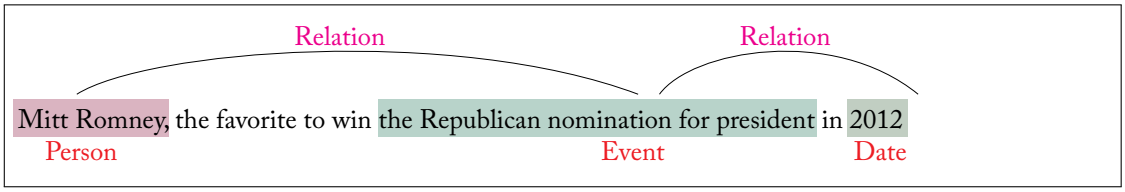


Figure 1.2: Examples of relations and events.

Information extraction is difficult because there are many ways of expressing the same facts:

- BNC Holdings Inc. named Ms. G. Torretta as its new chairman.
- Nicholas Andrews was succeeded by Gina Torretta as chairman of BNC Holdings Inc.
- Ms. Gina Torretta took the helm at BNC Holdings Inc.

Furthermore, information may need to be combined across several sentences, which may additionally not be consecutive.

- After a long boardroom struggle, Mr. Andrews stepped down as chairman of BNC Holdings Inc. He was succeeded by Ms. Torretta.

Information extraction typically consists of a sequence of tasks, comprising:

1. linguistic pre-processing (described in Chapter 2);
2. named entity recognition (described in Chapter 3);
3. relation and/or event extraction (described in Chapter 4).

Named entity recognition (NER) is the task of recognizing that a word or a sequence of words is a proper name. It is often solved jointly with the task of assigning types to named entities, such as Person, Location, or Organization, which is known as named entity classification (*NEC*). If the tasks are performed at the same time, this is referred to as *NERC*. NERC can either be an annotation task, i.e., to annotate a text with NEs, or the task can be to populate a knowledge base with these NEs. When the named entities are not simply a flat structure, but linked to a corresponding entity in an ontology, this is known as *semantic annotation* or *named entity linking* (NEL). Semantic annotation is much more powerful than flat NE recognition, because it enables inferences and generalizations to be made, as the linking of information provides access to knowledge not explicit in the text. When semantic annotation is part of the process, the information extraction task is often referred to as Ontology-Based Information Extraction (OBIE) or Ontology Guided Information Extraction (see Chapter 5). Closely associated with this is the process of ontology learning and population (OLP) as described in Chapter 6. Information extraction tasks are also a pre-requisite for many opinion mining tasks, especially where these require the identification of relations between opinions and their targets, and where they are based on ontologies, as explained in Chapter 7.

1.2 AMBIGUITY

It is impossible for computers to analyze language correctly 100% of the time, because language is highly ambiguous. Ambiguous language means that more than one interpretation is possible, either syntactically or semantically. As humans, we can often use world knowledge to resolve these ambiguities and pick the correct interpretation. Computers cannot easily rely on world knowledge and common sense, so they have to use statistical or other techniques to resolve ambiguity. Some kinds of text, such as newspaper headlines and messages on social media, are often designed to be deliberately ambiguous for entertainment value or to make them more memorable. Some classic examples of this are shown below:

- Foot Heads Arms Body.
- Hospitals Sued by 7 Foot Doctors.
- British Left Waffles on Falkland Islands.
- Stolen Painting Found by Tree.

In the first headline, there is syntactic ambiguity between the proper noun *(Michael) Foot*, a person, and the common noun *foot*, a body part; between the verb and plural noun *heads*, and the same for *arms*. There is also semantic ambgiuity between two meanings of both *arms* (weapons and body parts), and *body* (physical structure and a large collection). In the second headline, there is semantic ambiguity between two meanings of *foot* (the body part and the measurement), and also syntactic ambiguity in the attachment of modifiers (7 [Foot Doctors] or [7 Foot] Doctors). In the third example, there is both syntactic and semantic ambiguity in the word *Left* (past tense of the verb, or a collective noun referring to left-wing politicians). In the fourth example, there is ambiguity in the role of the preposition *by* (as agent or location). In each of these examples, for a human, one meaning is possible, and the other is either impossible or extremely unlikely (doctors who are 7-foot tall, for instance). For a machine, understanding without additional context that leaving pastries in the Falkland Islands, though perfectly possible, is an unlikely news item, is almost impossible.

1.3 PERFORMANCE

Due not only to ambiguity, but a variety of other issues, as will be discussed throughout the book, performance on NLP tasks varies widely, both between different tasks and between different tools. Reasons for the variable performance of different tools will be discussed in the relevant sections, but in general, the reason for this is that some tools are good at some elements of the task but bad at others, and there are many issues regarding performance when tools are trained on one kind of data and tested on another. The reason for performance between tasks varying so widely is largely based on complexity, however.

The influence of domain dependence on the effectiveness of NLP tools is an issue that is all too frequently overlooked. For the technology to be suitable for real-world applications, systems need to be easily customizable to new domains. Some NLP tasks in particular, such as Information Extraction, have largely focused on narrow subdomains, as will be discussed in Chapters 3 and 4. The adaptation of existing systems to new domains is hindered by various bottlenecks such as training data acquisition for machine learning–based systems. For the adaptation of Semantic Web applications, ontology bottlenecks may be one of the causes, as will be discussed in Chapter 6.

An independent, though related, issue concerns the adaptation of existing systems to different text genres. By this we mean not just changes in domain, but different media (e.g., email, spoken text, written text, web pages, social media), text type (e.g., reports, letters, books), and structure (e.g., layout). The genre of a text may be influenced by a number of factors, such as author, intended audience, and degree of formality. For example, less formal texts may not follow standard capitalization, punctuation, or even spelling formats, all of which can be problematic for the intricate mechanisms of IE systems. These issues will be discussed in detail in Chapter 8.

Many natural language processing tasks, especially the more complex ones, only become really accurate and usable when they are tightly focused and restricted to particular applications and domains. Figure 1.3 shows a three-dimensional tradeoff graph between generality vs. specificity of domain, complexity of the task, and performance level. From this we can see that the highest performance levels are achieved in language processing tasks that are focused on a specific domain and that are relatively simple (for example, identifying named entities is much simpler than identifying events).

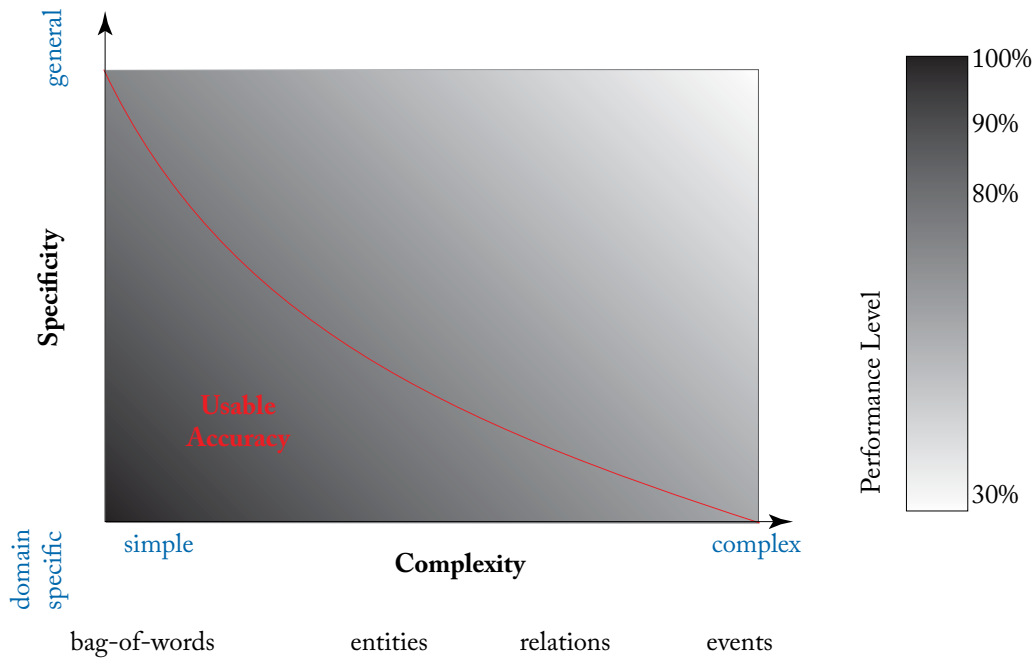


Figure 1.3: Performance tradeoffs for NLP tasks.

In order to make feasible the integration of semantic web applications, there must be some kind of understanding reached between semantic web and NLP practitioners as to what constitutes a reasonable expectation. This is of course true for all applications where NLP should be integrated. For example, some applications involving NLP may not be realistically usable in the

real world as standalone automatic systems without human intervention. This is not necessarily the case, however, for other kinds of semantic web applications which do not rely on NLP. Some applications are designed to assist a human user rather than to perform the task completely autonomously. There is often a tradeoff between the amount of autonomy that will most benefit the end user. For example, information extraction systems enable the end user to avoid having to read in detail hundreds or even thousands of documents in order to find the information they want. For humans to search manually through millions of documents is virtually impossible. On the other hand, the user has to bear in mind that a fully automated system will not be 100% accurate, and it is important for the design of the system to be flexible in terms of the tradeoff between precision and recall. For some applications, it may be more important to retrieve everything, although some of the information retrieved may be incorrect; on the other hand, it may be more important that everything retrieved is accurate, even if some things are missed.

1.4 STRUCTURE OF THE BOOK

Each chapter in the book is designed to introduce a new concept in the NLP pipeline, and to show how each component builds on the previous components described. In each chapter we outline the concept behind the component and give examples of common methods and tools. While each chapter stands alone to some extent, in that it refers to a specific task, the chapters build on each other. The first five chapters are therefore best read sequentially.

Chapter 2 describes the main approaches used for NLP tasks, and explains the concept of an NLP processing pipeline. The linguistic processing components comprising this pipeline—language identification, tokenization, sentence splitting, part-of-speech tagging, morphological analysis, and parsing and chunking—are then described, and examples are given from some of the major NLP toolkits.

Chapter 3 introduces the task of named entity recognition and classification (NERC), which is a key component of information extraction and semantic annotation systems, and discusses its importance and limitations. The main approaches to the task are summarized, and a typical NERC pipeline is described.

Chapter 4 describes the task of extracting relations between entities, explaining how and why this is useful for automatic knowledge base population. The task can involve either extracting binary relations between named entities, or extracting more complex relations, such as events. It describes a variety of methodologies and a typical extraction pipeline, showing the interaction between the tasks of named entity and relation extraction and discussing the major research challenges.

Chapter 5 explains how to perform entity linking by adding semantics into a standard flat information extraction system, of the kind that has been described in the preceding chapters. It discusses why this flat information extraction is not sufficient for many tasks that require greater richness and reasoning and demonstrates how to link the entities found to an ontology and to

Linked Open Data resources such as DBpedia and Freebase. Examples of a typical semantic annotation pipeline and of real-world applications are provided.

Chapter 6 introduces the concept of automated ontology development from unstructured text, which comprises three related components: learning, population, and refinement. Some discussion of these terms and their interaction is given, the relationship between ontology development and semantic annotation is discussed, and some typical approaches are described, again building on the notions introduced in the previous chapters.

Chapter 7 describes methods and tools for the detection and classification of various kinds of opinion, sentiment, and emotion, again showing how the NLP processes described in previous chapters can be applied to this task. In particular, aspect-based sentiment analysis (such as which elements of a product are liked and disliked) can benefit from the integration of product ontologies into the processing. Examples of real applications in various domains are given, showing how sentiment analysis can also be slotted into wider applications for social media analysis. Because sentiment analysis is often performed on social media, this chapter is best read in conjunction with Chapter 8.

Chapter 8 discusses the main problems faced when applying traditional NLP techniques to social media texts, given their unusual and inconsistent usage of spelling, grammar, and punctuation amongst other things. Because traditional tools often do not perform well on such texts, they often need to be adapted to this genre. In particular, the core pre-processing components described in Chapters 2 and 3 can have a serious knock-on effect on other elements in the processing pipeline if errors are introduced in these early stages. This chapter introduces some state-of-the-art approaches for processing social media and gives examples of some real applications.

Chapter 9 brings together all the components described in the previous chapters by defining and describing a number of application areas in which semantic annotations are required, such as semantically enhanced information retrieval and visualization, the construction of social semantic user models, and modeling online communities. Common approaches and open source tools are described for these areas, including evaluation, scalability, and state-of-the-art results.

The concluding chapter summarizes the main concepts described in the book, and gives some discussion of the current state-of-the-art, major problems still to be overcome, and an outlook to the future.

CHAPTER 2

Linguistic Processing

2.1 INTRODUCTION

There are a number of low-level linguistic tasks which form the basis of more complex language processing algorithms. In this chapter, we first explain the main approaches used for NLP tasks, and the concept of an NLP processing pipeline, giving examples of some of the major open source toolkits. We then describe in more detail the various linguistic processing components that are typically used in such a pipeline, and explain the role and significance of this pre-processing for Semantic Web applications. For each component in the pipeline, we describe its function and show how it connects with and builds on the previous components. At each stage, we provide examples of tools and describe typical performance of them, along with some of the challenges and pitfalls associated with each component. Specific adaptations to these tools for non-standard text such as social media, and in particular Twitter, will be discussed in Chapter 8.

2.2 APPROACHES TO LINGUISTIC PROCESSING

There are two main kinds of approach to linguistic processing tasks: a knowledge-based approach and a learning approach, though the two may also be combined. There are advantages and disadvantages to each approach, summarized in Table 2.1.

Knowledge-based or rule-based approaches are largely the more traditional methods, and in many cases have been superseded by machine learning approaches now that processing vast quantities of data quickly and efficiently is less of a problem than in the past. Knowledge-based approaches are based on hand-written rules typically written by NLP specialists, and require knowledge of the grammar of the language and linguistic skills, as well as some human intuition. These approaches are most useful when the task can easily be defined by rules (for example: "a proper noun always starts with a capital letter"). Typically, exceptions to such rules can be easily encoded too. When the task cannot so easily be defined in this way (for example, on Twitter, people often do not use capital letters for proper nouns), then this method becomes more problematic. One big advantage of knowledge-based approaches is that it is quite easy to understand the results. When the system incorrectly identifies something, the developer can check the rules and find out why the error has occurred, and potentially then correct the rules or write additional rules to resolve the problem. Writing rules can, however, be quite time-consuming, and if specifications for the task change, the developer may have to rewrite many rules.

Machine learning approaches have become more popular recently with the advent of powerful machines, and because no domain expertise or linguistic knowledge is required. One can set

up a supervised system very quickly if sufficient training data is available, and get reasonable results with very little effort. However, acquiring or creating sufficient training data is often extremely problematic and time-consuming, especially if it has to be done manually. This dependency on training data also means that adaptation to new types of text, domain, or language is likely to be expensive, as it requires a substantial amount of new training data. Human readable rules therefore typically tend to be easier to adapt to new languages and text types than those built from statistical models. The problem of sufficient training data can be handled by incorporating unsupervised or semi-supervised methods for machine learning: these will be discussed further in Chapters 3 and 4. However, these typically produce less accurate results than supervised learning.

Table 2.1: Summary of knowledge-based vs. machine learning approaches to NLP

Knowledge-Based	Machine Learning Systems
Based on hand-coded rules	Use statistics or other machine learning
Developed by NLP specialists	Developers do not need NLP expertise
Make use of human intuition	Requires large amounts of training data
Easy to understand results	Cause of errors is hard to understand
Development could be very time consuming	Development is quick and easy
Changes may require rewriting rules	Changes may require re-annotation

2.3 NLP PIPELINES

An NLP pre-processing pipeline, as shown in Figure 2.1, typically consists of the following components:

- Tokenization.
- Sentence splitting.
- Part-of-speech tagging.
- Morphological analysis.
- Parsing and chunking.

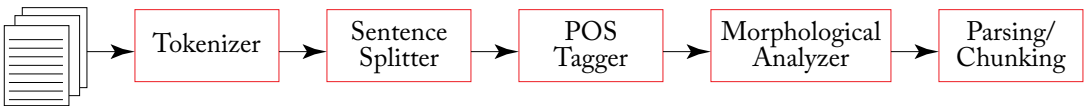


Figure 2.1: A typical linguistic pre-processing pipeline.

The first task is typically tokenization, followed by sentence splitting, to chop the text into tokens (typically words, numbers, punctuation, and spaces) and sentences respectively. Part-of-speech (POS) tagging assigns a syntactic category to each token. When dealing with multilingual text such as tweets, an additional step of language identification may first be added before these take place, as discussed in Chapter 8. Morphological analysis is not compulsory, but is frequently used in a pipeline, and essentially consists of finding the root form of each word (a slightly more sophisticated form of stemming or lemmatization). Finally, parsing and/or chunking tools may be used to analyze the text syntactically, identifying things like noun and verb phrases in the case of chunking, or performing a more detailed analysis of grammatical structure in the case of parsing.

Concerning toolkits, **GATE** [4] provides a number of open-source linguistic pre-processing components under the LGPL license. It contains a ready-made pipeline for Information Extraction, called ANNIE, and also a large number of additional linguistic processing tools such as a selection of different parsers. While GATE does provide functionality for machine learning-based components, ANNIE is mostly knowledge-based, making for easy adaptation. Additional resources can be added via the plugin mechanism, including components from other pipelines such as the Stanford CoreNLP Tools. GATE components are all Java-based, which makes for easy integration and platform independence.

Stanford CoreNLP [5] is another open-source annotation pipeline framework, available under the GPL license, which can perform all the core linguistic processing described in this section, via a simple Java API. One of the main advantages is that it can be used on the command line without having to understand more complex frameworks such as GATE or UIMA, and this simplicity, along with the generally high quality of results, makes it widely used where simple linguistic information such as POS tags are required. Like ANNIE, most of the components other than the POS tagger are rule-based.

OpenNLP[1] is an open-source machine learning-based toolkit for language processing, which uses maximum entropy and Perceptron-based classifiers. It is freely available under the Apache license. Like Stanford CoreNLP, it can be run on the command line via a simple Java API. While, like most other pipelines, the various components further down the pipeline mainly rely on tokens and sentences, the sentence splitter can be run either before or after the tokenizer, which is slightly unusual.

NLTK [6] is an open-source Python-based toolkit, available under the Apache license, which is also very popular due to its simplicity and command-line interface, particularly where Java-based tools are not a requirement. It provides a number of different variants for some components, both rule-based and learning-based.

In the rest of this chapter, we will describe the individual pipeline components in more detail, using the relevant tools from these pipelines as examples.

[1] http://opennlp.apache.org/index.html

2.4 TOKENIZATION

Tokenization is the task of splitting the input text into very simple units, called tokens, which generally correspond to words, numbers, and symbols, and are typically separated by white space in English. Tokenization is a required step in almost any linguistic processing application, since more complex algorithms such as part-of-speech taggers mostly require tokens as their input, rather than using the raw text. Consequently, it is important to use a high-quality tokenizer, as errors are likely to affect the results of all subsequent NLP components in the pipeline. Commonly distinguished types of tokens are numbers, symbols (e.g., $, %), punctuation, and words of different kinds, e.g., uppercase, lowercase, mixed case. A representation of a tokenized sentence is shown in Figure 2.2, where each pink rectangle corresponds to a token.

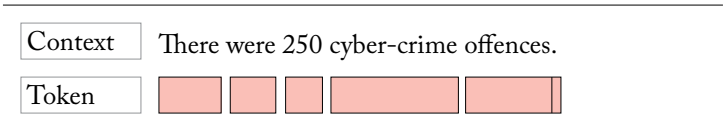


Figure 2.2: Representation of a tokenized sentence.

Tokenizers may add a number of features describing the token. These include details of orthography (e.g., whether they are capitalized or not), and more information about the kind of token (whether it is a word, number, punctuation, etc.). Other components may also add features to the existing token annotations, such as their syntactic category, details of their morphology, and any cleaning or normalization (such as correcting a mis-spelled word). These will be described in subsequent sections and chapters. Figure 2.3 shows a token for the word *offences* in the previous example with some features added: the kind of token is a word, it is 8 characters long, and the orthography is lowercase.

Tokenizing well-written text is generally reliable and reusable, since it tends to be domain-independent. However, such general purpose tokenizers typically need to be adapted to work correctly with things like chemical formulae, twitter messages, and other more specific text types. Other non-standard cases are hyphenated words in English, which some tools treat as a single token and some tools treat as three (the two words, plus the hyphen itself). Some systems also perform a more complex tokenization that takes into account number combinations such as dates and times (for example, treating *07:56* as a single token). Other tools leave this to later processing stages, such as a Named Entity Recognition component. Another issue is the apostrophe: for example in cases where an apostrophe is used to denote a missing letter and effectively joins two words without a space between, such as *it's*, or in French *l'homme*. German compound nouns suffer the opposite problem, since many words can be written together without a space. For German tokenizers, an extra module which splits compounds into their constituent parts can therefore be very useful, in particular for retrieval purposes. This extra segmentation module is critical to define

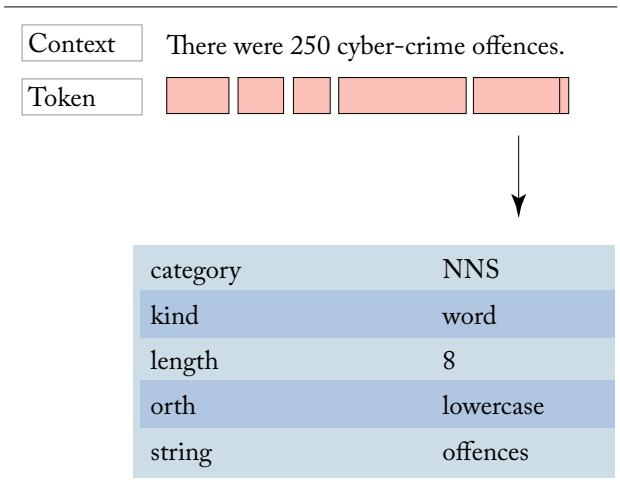


Figure 2.3: Representation of a tokenized sentence.

word boundaries also for many East Asian languages such as Chinese, which have no notion of white space between words.

Because tokenization generally follows a rigid set of constraints about what constitutes a token, pattern-based rule matching approaches are frequently used for these tools, although some tools do use other approaches. The **OpenNLP TokenizerME**,[2] for example, is a trainable maximum entropy tokenizer. It uses a statistical model, based on a training corpus, and can be re-trained on a new corpus.

GATE's **ANNIE Tokenizer**[3] relies on a set of regular expression rules which are then compiled into a finite-state machine. It differs somewhat from most other tokenizers in that it maximizes efficiency by doing only very light processing, and enabling greater flexibility by placing the burden of deeper processing on other components later in the pipeline, which are more adaptable. The generic version of the ANNIE tokenizer is based on Unicode[4] and can be used for any language which has similar notions of token and white space to English (i.e., most Western languages). The tokenizer can be adapted for different languages either by modifying the existing rules, or by adding some extra post-processing rules. For English, a specialized set of rules is available, dealing mainly with use of apostrophes in words such as *don't*.

The **PTBTokenizer**[5] is an efficient, fast, and deterministic tokenizer, which forms part of the suite of Stanford CoreNLP tools. It was initially designed to largely mimic Penn Treebank 3 (PTB) tokenization, hence its name. Like the ANNIE Tokenizer, it works well for English and

[2] http://incubator.apache.org/opennlp/documentation/manual/opennlp.html
[3] http://gate.ac.uk
[4] A good explanation of Unicode can be found at http://www.unicode.org/standard/WhatIsUnicode.html.
[5] http://nlp.stanford.edu/software/tokenizer.shtml

other Western languages, but works best on formal text. While deterministic, it uses some quite good heuristics, so as with ANNIE, it can usually decide when single quotes are parts of words, when full stops imply sentence boundaries, and so on. It is also quite customizable, in that there are a number of options that can be tweaked.

NLTK[6] also has several similar tokenizers to ANNIE, one based on regular expressions, written in Python.

2.5 SENTENCE SPLITTING

Sentence detection (or sentence splitting) is the task of separating text into its constituent sentences. This typically involves determining whether punctuation, such as full stops, commas, exclamation marks, and question marks, denote the end of a sentence or something else (quoted speech, abbreviations, etc.). Most sentence splitters use lists of abbreviations to help determine this: a full stop typically denotes the end of a sentence unless it follows an abbreviation such as *Mr.*, or lies within quotation marks. Other issues involve determining sentence structure when line breaks are used, such as in addresses or in bulleted lists. Sentence splitters vary in how such things are handled.

More complex cases arise when the text contains tables, titles, formulae, or other formatting markup: these are usually the biggest source of error. Some splitters ignore these completely, requiring a punctuation mark as a sentence boundary. Others use two consecutive new lines or carriage returns as an indication of a sentence end, while there are also cases when even a single newline or carriage return character would indicate end of a sentence (e.g., comments in software code or bulleted/numbered lists which have one entry per line). GATE's ANNIE sentence splitter actually provides several variants in order to let the user decide which is the most appropriate solution for their particular text. HTML formatting tags, Twitter hashtags, wiki syntax, and other such special text types are also somewhat problematic for general-purpose sentence splitters which have been trained on well-written corpora, typically newspaper texts. Note that sometimes tokenization and sentence splitting are performed as a single task rather than sequentially.

Sentence splitters generally make use of already tokenized text. GATE's **ANNIE** sentence splitter uses a rule-based approach based on GATE's JAPE pattern-action rule-writing language [7]. The rules are based entirely on information produced by the tokenizer and some lists of common abbreviations, and can easily be modified as necessary. Several variants are provided, as mentioned above.

Unlike ANNIE, the **OpenNLP** sentence splitter is typically run before the tokenization module. It uses a machine learning approach, with the models supplied being trained on untokenized text, although it is also possible to perform tokenization first and let the sentence splitter process the already tokenized text. One flaw in the OpenNLP splitter is that because it cannot identify sentence boundaries based on the contents of the sentence, it may cause errors on articles which have titles since these are mistakenly identified to be part of the first sentence.

[6] http://www.nltk.org/

NLTK uses the Punkt sentence segmenter [8]. This uses a language-independent, unsupervised approach to sentence boundary detection, based on identifying abbreviations, initials, and ordinal numbers. Its abbreviation detection, unlike most sentence splitters, does not rely on pre-compiled lists, but is instead based on methods for collocation detection such as log-likelihood.

Stanford CoreNLP makes use of tokenized text and a set of binary decision trees to decide where sentence boundaries should go. As with the ANNIE sentence splitter, the main problem it tries to resolve is deciding whether a full stop denotes the end of a sentence or not.

In some studies, the Stanford splitter scored the highest accuracy out of common sentence splitters, although performance will of course vary depending on the nature of the text. State-of-the-art sentence splitters such as the ones described score about 95–98% accuracy on well-formed text. As with most linguistic processing tools, each one has strengths and weaknesses which are often linked to specific features of the text; for example, some splitters may perform better on abbreviations but worse on quoted speech than others.

2.6 POS TAGGING

Part-of-Speech (POS) tagging is concerned with tagging words with their part of speech, e.g., noun, verb, adjective. These basic linguistic categories are typically divided into quite fine-grained tags, distinguishing for instance between singular and plural nouns, and different tenses of verbs. For languages other than English, gender may also be included in the tag. The set of possible tags used is critical and varies between different tools, making interoperability between different systems tricky. One very commonly used tagset for English is the Penn Treebank (PTB) [9]; other popular sets include those derived from the Brown corpus [10] and the LOB (Lancaster-Oslo/Bergen) Corpus [11], respectively. Figure 2.4 shows an example of some POS-tagged text, using the PTB tagset.

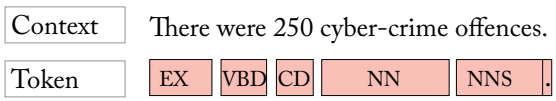


Figure 2.4: Representation of a POS-tagged sentence.

The POS tag is determined by taking into account not just the word itself, but also the context in which it appears. This is because many words are ambiguous, and reference to a lexicon is insufficient to resolve this. For example, the word *love* could be a noun or verb depending on the context (*I love fish* vs. *Love is all you need*).

Approaches to POS tagging typically use machine learning, because it is quite difficult to describe all the rules needed for determining the correct tag given a context (although rule-based methods have been used). Some of the most common and successful approaches use Hidden Markov models (HMMs) or maximum entropy. The **Brill** transformational rule-based tagger

[12], which uses the PTB tagset, is one of the most well-known taggers, used in several major NLP toolkits. It uses a default lexicon and ruleset acquired from a large corpus of training data via machine learning. Similarly, the **OpenNLP** POS tagger also uses a model learned from a training corpus to predict the correct POS tag from the PTB tagset. It can be trained with either a Maximum Entropy or a Perceptron-based model. The **Stanford** POS tagger is also based on a Maximum Entropy approach [13] and makes use of the PTB tagset. The **TNT** (Trigrams'n'Tags) tagger [14] is a fast and efficient statistical tagger using an implementation of the Viterbi algorithm for second-order Markov models.

In terms of major NLP toolkits, some (such as Stanford CoreNLP) have their own POS taggers, as described above, while others use existing implementations or variants on them. For example, **NLTK** has Python implementations of the Brill tagger, the Stanford tagger, and the TNT tagger. **GATE**'s ANNIE English POS tagger [15] is a modified version of the Brill tagger trained on a large corpus taken from the *Wall Street Journal*. It produces a POS tag as an annotation on each word or symbol. One of the big advantages of this tagger is that the lexicon can easily be modified manually by adding new words or changing the value or order of the possible tags associated with a word. It can also be retrained on a new corpus, although this requires a large pre-tagged corpus of text in the relevant domain/genre, which is not easy to find.

The accuracy of these general-purpose, reusable taggers is typically excellent (97–98%) on texts similar to those on which the taggers have been trained (mostly news articles). However, the accuracy can fall sharply when presented with new domains, genres, or noisier data such as social media. This can have a serious knock-on effect on other processes further down the pipeline such as Named Entity recognition, ontology learning via lexico-syntactic patterns, relation and event extraction, and even opinion mining, which all need reliable POS tags in order to produce high-quality results.

2.7 MORPHOLOGICAL ANALYSIS AND STEMMING

Morphological analysis essentially concerns the identification and classification of the linguistic units of a word, typically breaking the word down into its root form and an affix. For example, the verb *walked* comprises a root form *walk* and an affix *-ed*. In English, morphological analysis is typically applied to verbs and nouns, because these may appear in the text as variants created by inflectional morphology. Inflectional morphology refers to the different forms of words reflected by mood, tense, number, and so on, such as the past tense of a verb or the plural of a noun. Inflection in English is typically expressed by adding a suffix to the root form (e.g., *walk, walked*, *box, boxes*) or another internal modification such as a vowel change (e.g., *run, ran*, *goose, geese*). In other languages, prefixes (adding to the beginning of a word), infixes (adding in the middle of a word), and other changes may be used. Some morphological analysis tools represent these internal modifications as an alternative representation of the default affix. What we mean by this is that if the plural of a noun is commonly represented by adding *-s* as a suffix, the output of the tool will show the value of the affix as *-s* even in the case of plural forms such as *geese*. Essentially,

it treats an irregular vowel change form simply as a kind of surface representational variant of the standard affix. The GATE morphological analyzer, for example, depicts the word *geese* as having the root *goose* and affix *-s*.

Typically, NLP tools which perform morphological analysis deal only with inflectional morphology, as described above, but do not handle derivational morphology. Derivation is the process of adding derivational morphemes, which create a new word from existing words, usually involving a change in grammatical category (for example, creating the noun *worker* from the verb *work*, or the noun *loudness* from the adjective *loud*.

Morphological analyzers for English are often rule-based, since the majority of inflectional variants follow grammatical rules and set patterns (for example, plural nouns are typically created by adding *-s* or *-es* to the end of the singular noun). Exceptions can also be handled quite easily by rules, and unknown words are assumed to follow default rules. The English morphological analyzer in **GATE** is rule-based, with the rule language (flex) supporting rules and variables that can be used in regular expressions in the rules. POS tags can be taken into account if desired, depending on a configuration parameter. The analyzer takes as input a tokenized document, and considering one token and its POS tag at a time, it identifies its lemma and affix. These values are than added as features of the token.

The **Stanford** Morphology tool also uses a rule-based approach, is based on a finite-state transducer, and is written in flex. Unlike the GATE tool, however, it requires the use of POS tags as well as tokens, and generates lemmas but not affixes.

NLTK provides an implementation of morphological analysis based on WordNet's built-in *morphy* function. WordNet [16] is a large lexical database of English resembling a thesaurus, where nouns, verbs, adjectives, and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a concept. Synsets are interlinked by means of conceptual-semantic and lexical relations. The morphy function is designed to allow users to query an inflectional form against a base form listed in WordNet. It uses a rule-based method involving lists of inflectional endings, based on syntactic category, and an exception list for each syntactic category, in which a search for an inflected form is done. Like the Stanford tool, it returns only the lemma but not the affix. Furthermore, it can only handle words present in WordNet.

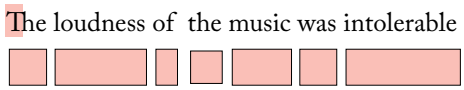
OpenNLP does not currently provide any tools for morphological analysis.

2.7.1 STEMMING

Stemmers produce the stem form of each word, e.g., *driving* and *drivers* have the stem *drive*, whereas morphological analysis tends to produce the root/lemma forms of the words and their affixes, e.g., *drive* and *driver* for the above examples, with affixes *-ing* and *-s* respectively. There is much confusion about the difference between stemming and morphological analysis, due to the fact that stemmers can vary considerably in how they operate and in their output. In general, stemmers do not attempt to perform an analysis of the root or stem and its affix, but simply strip the word down to its stem. The main way in which stemmers themselves vary is due to the presence

or absence of the constraint that the stem must also be a real word in the given language. Basic stemming algorithms simply strip off the affix, e.g., *driving* would be stripped to the stem *driv-* by removing the suffix *-ing*. The distinction between verbs and nouns is often not maintained, so both *driver* and *driving* would be stripped down to the stem *driv-*. Information retrieval (IR) systems often make use of this kind of suffix stripping, since it can be performed by a simple algorithm and does not require other linguistic pre-processing such as POS tagging. Stemming is useful for IR systems because it brings together lexico-syntactic variants of a word which have a common meaning (so one can use either the singular or plural form of a word in the search query, and it will match against either form in a web page). Note that unlike most morphological analysis tools, stemming tools may also consider variants arising from derivational morphology, since they ignore the syntactic category of the word. A further difference is that typically, stemmers do not refer to the context surrounding the word, but only to the word in isolation, while morphological analyzers may also use the context.

Figure 2.5 shows an example of how stemming and morphological analysis may differ. The stemmer in GATE strips off the derivational affix *-ness*, reducing the noun *loudness* to the base adjective *loud*, as shown by the *stem* feature. The morphological analyzer, on the other hand, is not concerned with derivational morphology, and leaves the word in its entirety, as shown by the *root* feature *loudness* and producing a zero affix.



affix	
category	NNS
kind	word
length	8
orth	lowercase
root	loudness
stem	loud
string	loudness

Figure 2.5: Comparison of stemming and morphological analysis in GATE.

Suffix-stripping algorithms may differ in results for a variety of reasons. One such reason is whether the algorithm constrains whether the output word must be a real word in the given language. Some approaches do not require the word to actually exist in the language lexicon (the set of all words in the language).

The most well-known stemming algorithm is the **Porter Stemmer** [17], which has been re-implemented in many forms. Due to the problems of many different variants being created, Porter later invented the Snowball language, which is a small string processing language designed specifically for creating stemming algorithms for use in Information Retrieval. A variety of useful open-source stemmers for many languages have since been created in Snowball. **GATE** provides a wrapper for a number of these, covering 11 European languages, while **NLTK** provides an implementation of them for Python. Because the stemmers are rule-based and easy to modify, following Porter's original approach, this makes them very straightforward to combine with the other low-level linguistic components described previously in this chapter. **OpenNLP** and **Stanford CoreNLP** do not provide any stemmers.

2.8 SYNTACTIC PARSING

Syntactic parsing is concerned with analysing sentences to derive their syntactic structure according to a grammar. Essentially, parsing explains how different elements in a sentence are related to each other, such as how the subject and object of a verb are connected. There are many different syntactic theories in computational linguistics, which posit different kinds of syntactic structures. Parsing tools may therefore vary widely not only in performance but in the kind of representation they generate, based on the syntactic theory they make use of.

Freely available wide-coverage parsers include the Minipar[7] dependency parser, the RASP [18] statistical parser, the Stanford [19] statistical parser, and the general-purpose SUPPLE parser [20]. These are all available within GATE, so that the user can try them all and decide which is the most appropriate for their needs.

Minipar is a dependency parser, i.e., it determines the dependency relationships between the words in a sentence. It processes the text one sentence at a time, and thus only needs a sentence splitter as a prerequisite. It works on the basis of identifying linguistic constructions and parts-of-speech like apposition, relative clauses, subjects and objects of verbs, and determiners, and how they relate to each other. Apposition is the construction where two noun phrases next to each other refer to the same thing, e.g., "my brother John," or "Paris, the capital of France." Relative clauses typically start with a relative pronoun (such as "who," "which," etc.) and modify a preceding noun, e.g., "who was wearing the hat" in the phrase "the man who was wearing the hat."

In contrast to dependency relations, constituency parsers are based on the idea of constituency relations, and may involve a number of different Constituency Grammar theories such as Phrase-Structure Grammars, Categorial Grammars and Lexical Functional Grammars, amongst others. The constituency relation is hierarchical and derives from the subject-predicate division of Latin and Greek grammars, where the basic clause structure is divided into the subject (noun phrase) and predicate (verb phrase). Further subdivisions of each are then made at a more fine-grained level.

[7] http://www.cs.ualberta.ca/~lindek/minipar.htm

A good example of a constituency parser is the **Shift-Reduce Constituency Parser** which is part of the Stanford CoreNLP Tools.[8] Shift-and-reduce operations have long been used for dependency parsing with high speed and accuracy, but only more recently have they been used for constituency parsing. The Shift-Reduce parser aims to improve on older constituency parsers which used chart-based algorithms (dynamic programming) to find the highest scoring parse, which were accurate but very slow. The latest Shift-Reduce Constituency parser is faster than the previous Stanford parsers, while being more accurate than almost all of them.

Figure 2.6 shows a parse tree generated using a dependency grammar, while Figure 2.7 shows one generated using a constituency grammar for the same sentence.

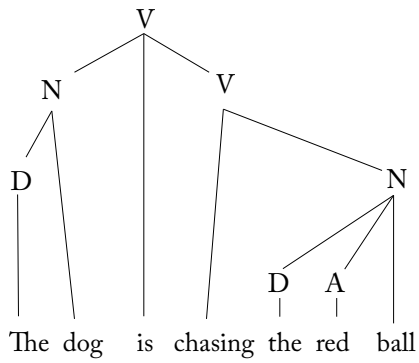


Figure 2.6: Parse tree showing dependency relation.

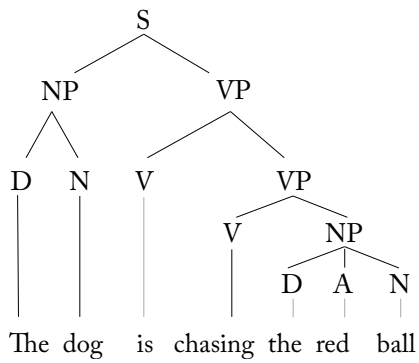


Figure 2.7: Parse tree showing constituency relation.

The **RASP** statistical parser [18] is a domain-independent, robust parser for English. It comes with its own tokenizer, POS tagger, and morphological analyzer included, and as with

[8] http://nlp.stanford.edu/software/srparser.shtml

Minipar, requires the text to be already segmented into sentences. RASP is available under the LGPL license and can therefore be used also in commercial applications.

The **Stanford** statistical parser [19] is a probabilistic parsing system. It provides either a dependency output or a phrase structure output. The latter can be viewed in its own GUI or through the user interface of GATE Developer. The Stanford parser comes with data files for parsing Arabic, Chinese, English, and German and is licensed under GNU GPL.

The **SUPPLE** parser is a bottom-up parser that can produce a semantic representation of sentences, called simplified quasilogical form (SQLF). It has the advantage of being very robust, since it can still produce partial syntactic and semantic results for fragments even when the full sentence parses cannot be determined. This makes it particularly applicable for deriving semantic features for the machine learning–based extraction of semantic relations from large volumes of real text.

2.9 CHUNKING

Parsing algorithms can be computationally expensive and, like many linguistic processing tools, tend to work best on text similar to that on which they have been trained. Because it is a much more difficult task than some of the lower-level processing tasks, such as tokenization and sentence splitting, performance is also typically much lower, and this can have knock-on effects on any subsequent processing modules such as Named Entity recognition and relation finding. Sometimes it is better therefore to sacrifice the increased knowledge provided by a parser for something more lightweight but reliable, such as a chunker which performs a more shallow kind of analysis. Chunkers, also sometimes called shallow parsers, recognize sequences of syntactically correlated words such as Noun Phrases, but unlike parsers, do not provide details of their internal structure or their role in the sentence.

Tools for chunking can be subdivided into Noun Phrase (NP) Chunkers and Verb Phrase (VP) Chunkers. They vary less than parsing algorithms because the analysis is at a more coarse-grained level—they perform identification of the relevant "chunks" of text but do not try to analyze it. However, they may differ in what they consider to be relevant for the chunk in question. For example, a simple Noun Phrase might consist of a consecutive string containing an optional determiner, one or more optional adjectives, and one or more nouns, as shown in Figure 2.8. A more complex Noun Phrase might also include a Prepositional Phrase or Relative Clause modifying it. Some chunkers include such things as part of the Noun Phrase, as shown in Figure 2.9, while others do not (Figure 2.10). This kind of decision is highly dependent on what the chunks will be used for later. For example, if they are used as input for a term recognition tool, it should be considered whether the possibility of a term that contains a Prepositional Phrase is relevant or not. For ontology generation, such a term is probably not required, but for use as a target for sentiment analysis, it might be useful.

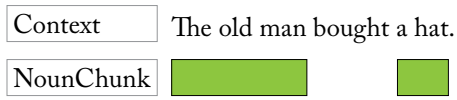


Figure 2.8: Simple NP chunking.

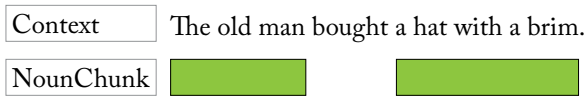


Figure 2.9: Complex NP chunking excluding PPs.

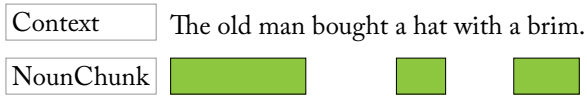


Figure 2.10: Complex NP chunking including PPs.

Verb Phrase chunkers delimit verbs, which may consist of a single word such as *bought* or a more complex group comprising modals, infinitives and so on (for example *might have bought* or *to buy*). They may even include negative elements such as *might not have bought* or *didn't buy*. An example of chunker output combining both noun and verb phrase chunking is shown in Figure 2.11.

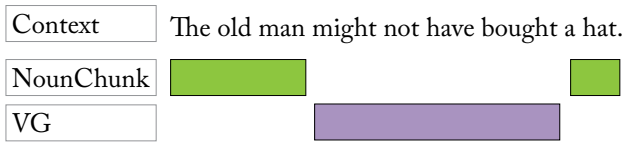


Figure 2.11: Complex VP chunking.

Some tools also provide additional chunks; for example, the **TreeTagger** [21] (trained on the Penn Treebank) can also generate chunks for prepositional phrases, adjectival phrases, adverbial phrases, and so on. These can be useful for building up a representation of the whole sentence without the requirement for full parsing.

As we have already seen, linguistic processing tools are not infallible, even assuming that the components they rely on have generated perfect output. It may seem simple to create an NP chunker based on grammatical rules involving POS tags, but it can easily go wrong. Consider the

two sentences *I gave the man food* and *I bought the baby food*. In the first case, *the man* and food are independent NPs which are respectively the indirect and direct objects of the verb *gave*. We can rephrase this sentence as *I gave food to the man* without any change in meaning, where it is clear these NPs are independent. In the second example, however, *the baby food* could be either a single NP which contains the compound noun *baby food*, or follow the same structure as the previous example (*I bought food for the baby*). An NP chunker which used the seemingly sensible pattern "Determiner + Noun + Noun" would not be able to distinguish between these two cases. In this case, a learning-based model might do better than a rule-based approach.

GATE provides both NP and VP chunker implementations. The NP Chunker is a Java implementation of the Ramshaw and Marcus BaseNP chunker [22], which is based on their POS tags and uses transformation-based learning. The output from this version is identical to the output of the original C++/Perl version.

The GATE VP chunker is written in JAPE, GATE's rule-writing language, and is based on grammar rules for English [23, 24]. It contains rules for the identification of non-recursive verb groups, covering finite (*is investigating*), non-finite (*to investigate*), participles (*investigated*), and special verb constructs (*is going to investigate*). All the forms may include adverbials and negatives. One advantage of this tool is that it explicitly marks negation in verbs (e.g., *don't*, which is extremely useful for other tasks such as sentiment analysis. The rules make use of POS tags as well as some specific strings (e.g., the word *might* is used to identify modals).

OpenNLP's chunker uses a pre-packaged English maximum entropy model. Unlike GATE, whose two chunkers are independent, it analyses the text one sentence at a time and produces both NP and VP chunks in one go, based on their POS tags. The OpenNLP chunker is easily retrainable, making it easy to adapt to new domains and text types if one has a suitable pre-annotated corpus available.

NLTK and **Stanford CoreNLP** do not provide any chunkers, although they could be created using rules and/or machine learning from the other components (such as POS tags) in the relevant toolkit.

2.10 SUMMARY

In this chapter we have introduced the idea of an NLP pipeline and described the main components, with reference to some of the widely used open-source toolkits. It is important to note that while performance in these low-level linguistic processing tasks is generally high, the tools do vary in performance, not just in accuracy, but also in the way in which they perform the tasks and their output, due to adhering to different linguistic theories. It is therefore critical when selecting pre-processing tools to understand what is required by other tools downstream in the application. While mixing and matching of some tools is possible (particularly in frameworks such as GATE, which are designed precisely with interoperability in mind), compatibility between different components may be an issue. This is one of the reasons why there are several different toolkits available offering similar but slightly different sets of tools. On the performance side, it is also important

to be aware of the effect of changing domain and text type, and whether the tools are easily modifiable or not if this is necessary. In particular, moving from tools trained on standard newswire to processing social media text can be problematic; this is discussed in detail in Chapter 8. Similarly, some tools can be adapted easily to new languages (in particular, the first components in the chain such as tokenizers), while more complex tools such as parsers may be more difficult to adapt. In the following chapter, we introduce the task of Named Entity Recognition and show how the linguistic processing tools described in this chapter can be built on to accomplish this.

CHAPTER 3

Named Entity Recognition and Classification

3.1 INTRODUCTION

As discussed in Chapter 1, information extraction is the process of extracting information from unstructured text and turning it into structured data. Central to this is the task of *named entity recognition and classification (NERC)*, which involves the identification of proper names in texts (*NER*), and their classification into a set of predefined categories of interest (*NEC*). Unlike the pre-processing tools discussed in the previous chapter, which deal with syntactic analysis, NERC is about automatically deriving **semantics** from textual content. The traditional core set of named entities, developed for the shared NERC task at MUC-6 [25], comprises Person, Organization, Location, and Date and Time expressions, such as *Barack Obama*, *Microsoft*, *New York*, *4th July 2015*, etc.

NERC is generally an annotation task, i.e., to annotate a text with named entities (NEs), but it can involve simply producing a list of NEs which may then be used for other purposes, including creating or extending gazetteers to assist with the NE annotation process in future. It can be subdivided into two tasks: the recognition task, involving identifying the boundaries of an NE (typically referred to as NER); and named entity classification (NEC), involving detecting the class or type of the NE. Slightly confusingly, NER is often used to mean the combination of the two tasks, especially in older work; here we stick to using NERC for the combined task and NER for only the recognition element. For more fine-grained NEC than the standard Person, Organization, and Location classification, classes are often taken from an ontology schema and are subclasses of these [26]. The main challenge for NEC is that NEs can be highly ambiguous (e.g., "May" can be a person's name or a month of the year; "Mark" can be a person's name or a common noun). Partly for this reason, the two tasks of NER and NEC are typically solved as a single task.

A further task regarding named entities is *named entity linking (NEL)*. The NEL task is to recognize if a named entity mention in a text corresponds to any NEs in a reference knowledge base. A named entity mention is an expression in the text referring to a named entity: this may be under different forms, e.g., "Mr. Smith" and "John Smith" are both mentions (textual representations) of the same real-world entity, expressed by slightly different linguistic realizations. The reference knowledge base used is typically Wikipedia. NEL is even more challenging than NEC because distinctions do not only have to be made on the class-level, but also within classes. For

example, there are many persons with the name "John Smith." The more popular the names are, the more difficult the NEL task becomes. A further problem, which all knowledge base–related tasks have, is that knowledge bases are incomplete; for example, they will only contain the most famous people named "John Smith." This is particularly challenging when working on tasks involving recent events, since there is often a time lag between newly emerging entities appearing in the news or on social media and the updating of knowledge bases with their information. More details on named entity linking, along with relevant reference corpora, are given in Chapter 5.

3.2 TYPES OF NAMED ENTITIES

The reason that Person, Organization, Location, Date, and Time have become so popular as standard types of named entity is due largely to the Message Understanding Conference series (MUC) [25], which introduced the Named Entity Recognition and Classification task in 1995 and which drove the initial development of many systems which are still in existence today. Due to the expansion of NERC evaluation efforts (described in more detail in Section 3.3) and the need for using NERC tools in real-life applications, other kinds of proper nouns and expressions gradually also started to be considered as named entities, according to the task, such as newspapers, monetary amounts, and more fine-grained classifications of the above, such as authors, music bands, football teams, TV programs, and so on. NERC is the starting point for many more complex applications and tasks such as ontology building, relation extraction, question answering, information extraction, information retrieval, machine translation, and semantic annotation. With the advent of open information extraction scenarios focusing on the whole of the web, analysis of social media where new entities emerge constantly, and named entity linking tasks, the range of entities extracted has widened dramatically, which has brought many new challenges (see for example Section 4.4, where the role of knowledge bases for Named Entity Linking is discussed). Furthermore, the standard kind of 5- or 7-class entity recognition problem is now often less useful, which in turn means that new paradigms are required. In some cases, such as the recognition of Twitter user names, the distinction between traditional classes, such as Organization and Location, has become blurred even for a human, and is no longer always useful (see Chapter 8).

Defining what exactly should constitute each entity type is never easy, and guidelines differ according to the task. Traditionally, people have used the standard guidelines from the evaluations, such as MUC and CONLL, since these allow methods and tools to be compared with each other easily. However, as tools have been used for practical purposes in real scenarios, and as the types of named entities have consequently changed and evolved, so the ways in which entities are defined have also had to be adapted for the task. Of course, this now makes comparison and performance evaluation more difficult. The ACE evaluation [27], in particular, attempted to solve some of the problems caused by metonymy, where an entity which theoretically depicts one type (e.g., Organization) is used figuratively. Sports teams are an example of this, where we might use the location *England* or *Liverpool* to mean their football team (e.g., *England won the World Cup in*

1966). Similarly, locations such as *The White House* or *10 Downing Street* can be used to refer to the organization housed there (*The White House announced climate pledges from 81 countries.*). Other decisions involve determining, for example, if the category Person should include characters such as God or Santa Claus, and furthermore, if so, whether they should be included in all situations, such as when using God and Jesus as part of profanities.

3.3 NAMED ENTITY EVALUATIONS AND CORPORA

As mentioned above, the first major evaluation series for NERC was MUC, which first addressed the named entity challenge in 1996. The aim of this was to recognize named entities in newswire text, and led not only to system development but the first real production of gold standard NE-annotated corpora for training and testing. This was followed in 2003 by ConLL [28], another major evaluation compaign, providing gold standard data for newswire not only in English but also Spanish, Dutch, and German. The corpus produced for this evaluation effort is now one of the most popular gold standards for NERC, with NERC software releases typically quoting performance on it.

Other evaluation campaigns later started to address NERC for genres other than newswire, specifically ACE [27] and OntoNotes [29], and introduced new kinds of named entities. Both of those corpora contain subcorpora with the genres newswire, broadcast news, broadcast conversation, weblogs, and conversational telephone speech. ACE additionally contains a subcorpus with usenet newsgroups, and addressed not only English but also Arabic and Chinese in later editions. Both ACE and OntoNotes also involved tasks such as coreference resolution, relation and event extraction, and word sense disambiguation, allowing researchers to study the interaction between these tasks. These tasks are addressed in Section 3.5 and in Chapters 4 and 5.

While NERC corpora mostly use the traditional entity types, such as Person, Organization and Location, which are not motivated by a concrete Semantic Web knowledge base (such as DBpedia, Freebase, or YAGO), these types are very general. This means that when developing NERC approaches on those corpora for Semantic Web purposes, it is relatively easy to build on top of them and to include links to a knowledge base later. For example, NERD [30] uses an OWL ontology[1] containing the set of mappings of all entity categories (e.g., *criminal* is a sub-class of *Person* in the NERD ontology).

3.4 CHALLENGES IN NERC

One of the main challenges of NERC is to distinguish between named entities and entities. The difference between these two things is that named entities are instances of types (such as Person, Politician) and refer to real-life entities which have a single unique referent, whereas entities are often groups of NEs which do not refer to unique referents in the real world. For example, "Prime Minister" is an entity, but it is not a named entity because it refers to any one of a group of named

[1] http://nerd.eurecom.fr/ontology

entities (anyone who has been or currently is a prime minister). It is worth noting though that the distinction can be very difficult to make, even for humans, and annotation guidelines for tasks differ on this.

Another challenge is to recognize NE boundaries correctly. In Example 3.1, it is important to recognize that *Sir* is part of the name *Sir Robert Walpole*. Note that tasks also differ in where they place the boundaries. MUC guidelines define that a Person entity should include titles; however, other evaluations may define their tasks differently. A good discussion of the issues in designing NERC tasks, and the differences between them, can be found in [31]. The entity definitions and boundaries are thus often not consistent between different corpora. Sometimes, boundary recognition is considered as a separate task from detecting the type (Person, Location, etc.) of the named entity. There are several annotation schemes commonly used to recognize where NEs begin and end. One of the most popular ones is the *BIO* schema, where B signifies the *Beginning* of an NE, I signifies that the word is *Inside* an NE, and O signifies that the word is just a regular word *Outside* of an NE. Another very popular scheme is BILOU [32], which has the additional labels L (*Last* word of an NE) and U (*Unit*, signifying that the word is an entire unit, i.e., NE).

Example 3.1 Sir Robert Walpole was a British statesman who is generally regarded as the first Prime Minister of Great Britain. Although the exact dates of his dominance are a matter of scholarly debate, 1721-1742 are often used.[2]

Politician: Government positions held (Officeholder, Office/position/title, From, To)
Person: Gender
Sir Robert Walpole: Politician, Person

Government positions held (Sir Robert Walpole, Prime Minister of Great Britain, 1721, 1742)
Gender (Sir Robert Walpole, male)

Ambiguities are one of the biggest challenges for NERC systems. These can affect both the recognition and the classification component, and sometimes even both simultaneously. For example, the word *May* can be a proper noun (named entity) or a common noun (not an entity, as in the verbal use *you may go*), but even when a proper noun, it can fall into various categories (month of the year, part of a person's name (and furthermore a first name or surname), or part of an organization name). Very frequent categorization problems occur with the distinction between Person and Organization, since many companies are named after people (e.g., the clothing company *Austin Reed*). Similarly, many things which may not be named entities, such as names of diseases and laws, are named after people too. While technically one could annotate the person's name here, it is not usually desirable (we typically do not care about annotating *Parkinson* as a Person in the term *Parkinson's disease* or *Pythagoras* in *Pythagoras' Theorem*).

[2]Example from `http://en.wikipedia.org/wiki/Robert_Walpole`

3.5 RELATED TASKS

Temporal normalization takes the recognition of temporal expressions (NEs classified as Date or Time) a step further, by mapping them onto a standard date and time format. Temporal normalization, and in particular that of relative dates and times, is critical for event recognition tasks. The task is quite easy if a text already refers to time in an absolute way, e.g., "8am." It becomes more challenging, however, if a text refers to time in a relative way, e.g., "last week." In this case we first have to find the date the text was created, so that it can be used as a point of reference for the relative temporal expression. One of the most popular annotation schema for temporal expressions is TimeML [33]. Most NERC tools do not include temporal normalization as a standard part of the NERC process, but some tools have additional plugins that can be used. GATE, for example, has a Date Normalizer plugin that can be added to ANNIE in order to perform this task. It also has a temporal annotation plugin, GATE-Time, based on the HeidelTime tagger [34], and which conforms to TimeML, an ISO standard for temporal semantic annotation of documents [35]. SUTime [36] is another library for recognizing and normalizing time expressions, available as part of the Stanford CoreNLP pipeline. It makes use of a deterministic rule-based system, and thus is easily extendable. It produces a set of annotations with one of four temporal types (DATE, TIME, DURATION, and SET), which correspond to the TIMEX3 standard for type and value. The slightly unusual "SET" type refers to a set of times, such as a recurring event.

Co-reference resolution aims at connecting together different mentions of the same entity. This task is important because it helps with finding relations between entities later, and it also helps with named entity linking. The different mentions may be identical references, in which case the task is easy, or the task may be more complicated because the same entity can be mentioned in different ways. For example, *John Smith*, *Mr. Smith*, *John*, *J. S. Smith*, and *Smith* could all refer to the same person. Similarly, we may have acronyms (*U.K.* and *United Kingdom*) or even aliases which bear no surface resemblance to their alternative name (*IBM* and *The Big Blue*). With the exception of the latter form, where lists of explicit name pairs are often the best solution, rule-based systems tend to be quite effective for this task. For example, even though acronyms are often highly ambiguous, in the context of the same document it is rare that an acronym and a longer name that matches the relevant letters would not be a match. Of course, explicit lists of pairs can also be used; similarly, lists of exceptions can also be added. ANNIE's Orthomatcher is a good example of a co-reference tool which relies entirely on hand-coded rules, performing on news texts with around 95% accuracy [37]. The Stanford Coref tool is integrated in the Stanford CoreNLP pipeline, and implements the multi-pass sieve co-reference and anaphor resolution system described in [38]. SANAPHOR [39] extends this further by adding a semantic layer on top of this and improving the results. It takes as input co-reference clusters generated by Stanford Coref, and then splits those containing unrelated mentions, and merges those which should belong together. It uses the output from an NEL process involving DBpedia/YAGO to disambiguate those mentions which are linked to different entities, and merges those which are linked to the same one. It can also be used with other NERC and NEL tools as input.

3.6 APPROACHES TO NERC

Approaches to NERC can be roughly divided into rule- or pattern-based, and machine learning or statistical extraction methods [40], although quite often the two techniques are mixed (see [41][42][43]). Most learning-based techniques rely on some form of human supervision, with the exception of purely structural IE techniques performing unsupervised machine learning on unannotated documents [44]. As we have already seen, language engineering platforms, such as GATE, Stanford CoreNLP, OpenNLP, and NLTK, enable the modular implementation of techniques and algorithms for information extraction, by inserting different pre-processing and NERC modules into the pipeline, thereby allowing repeatable experimentation and evaluation of their results. An example of a typical processing pipeling for NERC is shown in Figure 3.1.

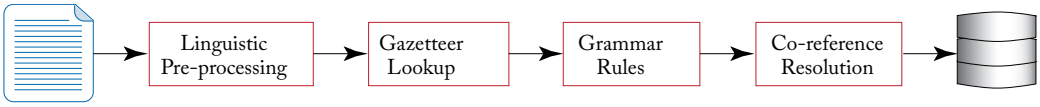


Figure 3.1: Typical NERC pipeline.

3.6.1 RULE-BASED APPROACHES TO NERC

Linguistic rule-based methods for NERC, such as those used in ANNIE, GATE's information extraction system, typically comprise a combination of gazetteer lists and hand-coded pattern-matching rules. These rules use contextual information to help determine whether candidate entities from the gazetteers are valid, or to extend the set of candidates. The gazetteer lists act as a starting point from which to establish, reject, or refine the final entity to be extracted. A typical NERC processing pipeline consists of linguistic pre-processing (tokenization, sentence splitting, POS tagging) as described in the pervious chapter, followed by entity finding using gazetteers and grammars, and then co-reference resolution.

Gazetteer lists are designed for annotating simple, regular features such as known names of companies, locations, days of the week, famous people, etc. A typical set of gazetteers for NERC might contain hundreds or even thousands of entries. However, using gazetteers alone is insufficient for recognizing and classifying entities, because on the one hand many names are too ambiguous (e.g., "London" could be part of an Organization name, a Person name, or just the Location), and on the other hand they cannot specify every named entity (e.g., in English one cannot pre-specify every single possible surname). When gazetteers are combined with other linguistic pre-processing annotations (part-of-speech tags, capitalization, other contextual evidence), however, they can be very powerful.

Using pattern matching for NERC requires the development of patterns over multi-faceted structures that consider many different properties of words, such as orthography (capitalization), morphology, part-of-speech information and so on. Traditional pattern-matching languages,

such as PERL, quickly become unmanageable due to complexity, when used for such tasks. Therefore, attribute-value notations are normally used, that allow for conditions to refer to token attributes arising from multiple analysis levels. An example of this is JAPE, the Java-based pattern matching language used in GATE, based on CPSL [45]. JAPE employs a declarative notation that allows for context-sensitive rules to be written and for non-deterministic pattern matching to be performed. The rules are divided into phases (subsets) which run sequentially; each phase typically consists of rules for the same entity type (e.g., Person) or rules that have the same specific requirements for their being run. A variety of priority mechanisms enable dealing with competing rules, which make it possible to handle ambiguity: for example, one can prefer patterns occurring in a particular context, or one can prefer a certain entity type over another in a given situation. Other rule-based mechanisms work in a similar way.

A typical simple pattern-matching rule might try to match all university names, e.g., *University of Sheffield*, *University of Bristol*, where the pattern consists of the specific words *University of* followed by the name of a city. From the gazetteer, we can check for the mention of a city name such as Sheffield or Bristol. A more complex rule might try to identify the name of any organization by looking for a keyword from a gazetteer list, such as *Company*, *Organization*, *Business*, *School*, etc. occurring together with one or more proper nouns (as found by the POS Tagger), and potentially also containing some function words. While these kinds of rules are quite good at matching typical patterns (and work very well for some entity types such as Persons, Locations, and Dates), they can be highly ambiguous. Compare for example the company name *General Motors*, the person name *General Carpenter*, and the phrase *Major Disaster* (which does not denote any entity), and it can easily be seen that such patterns are insufficient. Learning approaches, on the other hand, may be good at recognizing that *disaster* is not typically part of a person or organization's name, because it never occurs as such in the training corpus.

As mentioned above, rule-based systems are developed based on linguistic features, such as POS tags or context information. Instead of manually developing such rules, it is possible to label training examples, then automatically learn rules, using rule learning (also known as rule induction) systems. These automatically induce sets of rules from labeled training examples using supervised learning. They were popular among the early NERC learning systems, and include SRV [46], RAPIER [47], WHISK [48], BWI [49], and LP^2 [50].

3.6.2 SUPERVISED LEARNING METHODS FOR NERC

Rule learning methods were historically followed by supervised learning approaches, which learn weights for features, based on their probability of appearing with negative vs. positive training examples for specific NE types. The general supervised learning approach consists of five stages:

- linguistic pre-processing;
- feature extraction;
- training models on training data;
- applying models to test data;

- post-processing the results to tag the documents.

Linguistic pre-processing at the minimal level includes tokenization and sentence splitting. Depending on the features used, it can also include morphological analysis, part-of-speech tagging, co-reference resolution, and parsing, as described in Chapter 2. Popular features include:

- Morphological features: capitalization, occurrence of special characters (e.g., $, %);
- Part-of-speech features: tags of the occurrence;
- Context features: words and POS of words in a window around the occurrence, usually of 1–3 words;
- Gazetteer features: appearance in NE gazetteers;
- Syntactic features: features based on parse of sentence;
- Word representation features: features based on unsupervised training on unlabeled text using e.g., Brown clustering or word embeddings.

Statistical NERC approaches use a variety of models, such as Hidden Markov Models (HMMs) [51], Maximum Entropy models [52], Support Vector Machines (SVMs) [53] [54] [55], Perceptrons [56][57], Conditional Random Fields (CRFs) [58, 59], or neural networks [60]. The most successful NERC approaches include those based on CRFs and, more recently, multi-layer neural networks. We refer readers interested in learning more about those machine learning algorithms to [61, 62].

CRFs model NERC as a sequence labeling approach, i.e., the label for a token is modeled as dependent on the label of preceding and following tokens in a certain window. Examples of frameworks which are available for CRF-based NERC are Stanford NER[3] and CRFSuite.[4] Both are distributed with feature extractors and models trained on the ConLL 2003 data [28].

Multi-layer neural network approaches have two advantages. First, they learn latent features, meaning they do not require linguistic processing beyond sentence splitting and tokenization. This makes them more robust across domains than architectures based on explicit features, since they do not have to compensate for mistakes made during pre-processing. Second, they can easily incorporate unlabeled text, on which representation feature extraction methods can be trained. The state-of-the-art system for NERC, SENNA [60], uses such a multi-layer neural network architecture with unsupervised pre-training. It is available as a stand-alone distribution[5] or as part of the DeepNL framework.[6] Like the frameworks above, it is distributed with feature extractors and offers functionality for training models on new data.

There are advantages and disadvantages to a supervised learning approach for NERC compared with a knowledge engineering, rule-based approach. Both require manual effort—rule-based approaches require specialist language engineers to develop hand-coded rules, whereas supervised learning approaches require annotated training data, for which language engineers are

[3] http://nlp.stanford.edu/software/CRF-NER.shtml
[4] http://www.chokkan.org/software/crfsuite/
[5] http://ronan.collobert.com/senna/
[6] https://github.com/attardi/deepnl

not needed. Which stream of approach is better suited for an application scenario is dependent on the application and the domain. For popular domains, such as newswire, hand-labeled training data is already available, whereas for others, it might need to be created from scratch. If the linguistic variation in the text is very small and quick results are desired, hand-coding rules might be a better starting point.

3.7 TOOLS FOR NERC

GATE's general purpose named entity recognition and classification system, ANNIE, is a typical example of a rule-based system. It was designed for traditional NERC on news texts but, being easily adaptable, can form also the starting point for new NERC applications in other languages and for other domains. GATE contains tools for ML, so can be used to train models for NERC also, based on the pre-processing components described in Chapter 2. Other well known systems are UIMA,[7] developed by IBM, which focuses more on architectural support and processing speed, and offers a number of similar resources to GATE; OpenCalais,[8] which provides a web service for semantic annotation of text for traditional named entity types, and LingPipe[9] which provides a (limited) set of machine learning models for various tasks and domains. While these are very accurate, they are not easily adaptable to new applications. Components from all these tools are actually included in GATE, so that a user can mix and match various resources as needed, or compare different algorithms on the same corpus. However, the components provided are mainly in the form of pre-trained models, and do not typically offer the full functionality of the original tools.

The Stanford NER package, included in the Stanford CoreNLP pipeline, is a Java implementation of a Named Entity Recognizer. It comes with well-engineered feature extractors for NERC, and has a number of options for defining these. In addition to the standard 3-class model (Person, Organization, Location), it also comes with other models for different languages and models trained on different sets. The methodology used is a general implementation of linear chain Conditional Random Field (CRF) sequence models, and thus the user can easily retrain it on any labeled data they have. The Stanford NER package is also used in NLTK, which does not have its own NERC tool.

OpenNLP contains a NameFinder module for English NERC which has separate models for the standard 7-type MUC classification (Person, Organization, Location, Date, Time, Money, Percent), trained on standard freely available datasets. It also has models for Spanish and Dutch trained on CONLL data. As with the Stanford NER tool, the user can easily retrain the NameFinder on any labeled dataset. Similarly to the other learning-based tools mentioned above, because they rely on supervised learning, these tools work well only when large amounts

[7] http://uima.apache.org
[8] http://www.opencalais.com/
[9] http://alias-i.com/lingpipe/index.html

of annotated training data are available, so applying them to new domains and text types can be quite problematic if such data does not exist.

An example of a system that performs fine-grained NERC is FIGER [63],[10] which is trained on Wikipedia. The tag set for FIGER is made up of 112 types, which are derived from Freebase by selecting the most frequent types and merging fine-grained types. The goal is to perform multi-class multi-label classification, i.e., each sequence of words is assigned one or several of multiple types, or no type. Training data for FIGER is created by exploiting the anchor text of entity mentions annotated in Wikipedia, i.e., for each sequence of words in a sentence, the sequence is automatically mapped to a set of Freebase types and used as positive training data for those types. The system is trained using a two-step process: training a CRF model for named entity boundary recognition, then an adapted perceptron algorithm for named entity classification. Typically, a CRF model would be used for doing both at once (e.g. [64]), but this is avoided here due to the large set of NE types. As for the other NERC tools, it can easily be retrained on new data.

3.8 NERC ON SOCIAL MEDIA

Research on NERC in tweets is currently a hot research area, since there are many tasks which rely on the analysis of social media, as we will discuss in Chapter 8. Social media is a particular challenge for NERC due to its noisy nature (incorrect spelling, punctuation, capitalization, novel use of words, etc.), which affects both the pre-processing components required (and thus has a knock-on effect on the NERC component performance) and the named entities themselves, which become harder to recognize. Due to the lack of annotated corpora, performing NERC on social media data using a learning approach is generally viewed as a domain adaptation problem from newswire text, often integrating the two kinds of data for training [65] and including a tweet normalization step [66]. One particular challenge is recency: the kinds of NEs that we want to recognize in social media are often newly emerging (recent news stories about people who were not previously famous, for example) and so are not typically found in gazetteers or even in Linked Data sets such as DBpedia. Another challenge is that a diverse context [67] as well as a smaller context window [68] make NERC more difficult: unlike in longer news articles, there is a low amount of discourse information per tweet, and threaded structure is fragmented across multiple documents, flowing in multiple directions. NERC from social media will be discussed explicitly in Chapter 8.

3.9 PERFORMANCE

In general, NERC performance is lower than performance of NLP pre-processing tasks, such as POS tagging, but can still reach F1 scores above 90%. NERC performance depends on a variety of factors, including the type of text (e.g., newswire, social media), the NE type (e.g., PER, LOC,

[10] https://github.com/xiaoling/figer

ORG), the size of the available training corpus and, most notably, how different the corpus the NER was developed on is from the text the NERC is applied to [69]. In the context of NERC evaluation campaigns, the task is typically to train and test systems on different splits of the same corpus (also called in-domain performance), meaning the test corpus is very similar to the training corpus.

To give an indication of such in-domain NERC performance, the current state-of-the-art result on ConLL 2003, the most popular newswire corpus with NERC annotations, is an F1 of 90.10%. The best-performing system is currently [70].[11] On the other hand, the winning tool for NERC for the social media domain in the 2015 shared task WNUT [71, 72] only achieved 56.41% F1, and 70.63 for NER. It is clear that NERC is much more difficult than NER, and that NERC for existing social media corpora is more challenging than for newswire corpora. Notably, the corpora also differ in size, which is fairly typical. Large NERC-annotated corpora exist for the newswire genre, but these are still largely lacking for the social media genre. This is a big part of the reason that performance on social media corpora is so much worse [69].

In real-world or application scenarios, such an in-domain setting as described above typically does not apply. Even if a hand-annotated NERC corpus is created for the specific application at some point, the test data might change. Typically, the greater the time difference between the creation time of a training corpus and test data, the less useful it is for extracting NEs from that test corpus [69]. This is particularly true for the social media genre, where entities change very quickly. In practice this means that after a couple of years, training data can be rendered almost useless.

3.10 SUMMARY

In this chapter, we have described the task of Named Entity Recognition and Classification and its two subtasks of boundary identification and classification into entity types. We have shown why the linguistic techniques described in the previous chapter are required for the task, and how they are used in both rule-based and machine-learning approaches. Like most of the following NLP tasks we describe in the rest of the book, this is the point at which tasks begin to get more complicated. The linguistic pre-processing tasks all essentially have a very similar goal and definition which does not vary according to what they will be used for. NE recognition and other tasks, such as relation extraction, sentiment analysis, etc., vary enormously in their definition, depending on why they are required. For example, the classes of NEs may differ widely from the standard MUC types of Person, Organization, and Location to a much more fine-grained classification involving many more classes and thus making the task very different. From there one can also go a stage further and perform a more semantic form of annotation, linking entities to external data sources such as DBpedia and Freebase, as will be described in Chapter 5. Despite this, methods for NERC are typically reusable (at least to some extent) even when the task itself varies substantially, although for example some kinds of learning methods may work better for

[11] http://www.aclweb.org/aclwiki/index.php?title=CONLL-2003_(State_of_the_art)

different levels of classification. In the following chapter, we look at how named entities can be connected via relations, such as authors and their books, or employees and their organizations.

CHAPTER 4

Relation Extraction

4.1 INTRODUCTION

Relation extraction (RE) is the task of extracting links between entities, which builds on the task of NERC discussed in the previous chapter. The focus is often to extract binary relations between named entities, but can also be to extract more complex relations, such as events. Typical relation types include *birthdate(PER, DATE)* and *founder-of(PER, ORG)*, with examples for relations being *birthdate(John Smith, 1985-01-01)* or *founder-of(Bill Gates, Microsoft)*.

RE can be an annotation task, i.e., to annotate a text with relations, but is typically considered a *slot filling* task, also called *knowledge base population*, i.e., to populate a knowledge base with relations for a given set of relation types (known as a *relation schema*). It can be subdivided into three tasks: relation argument identification (finding the boundaries of the arguments), relation argument classification (assigning types to the arguments), and relation classification (assigning a type to the relation) [73]. The first two tasks are generally approached using NERC. For semantic annotation (see Section 5), a further step is to link relation arguments to entries in a knowledge base using *named entity linking (NEL)* methods.

One of the problems RE approaches face is the vast difference between relation schemas; unlike for NER, there is no standard small set of types that is shared across systems. The schema used largely depends on the application. In some cases an existing ontology schema is used, e.g., the YAGO schema; in other cases a schema is created specially for the task. For this reason, there are fewer off-the-shelf RE systems available than NER systems.

Another problem is that relation types can overlap or even entail one another, e.g., the *ceo-of(PER, ORG)* relation is completely subsumed by the *employee-of(PER, ORG)* relation, whereas there is only a strong overlap, but no entailment relation between *country-of-birth(PER, LOC)* and *country-of-residence(PER, LOC)*. Such entailment relations are sometimes defined by the underlying relation schema and can then be used to improve relation extraction performance [74].

Lastly, it is worth noting that the more general and frequent a relation, the easier it is to achieve a high relation extraction performance for that relation.

4.2 RELATION EXTRACTION PIPELINE

This section aims at describing a typical relation extraction approach. A graphical overview of an RE pipeline is given in Figure 4.1. Note that there are several variations of this approach, as described in the following sections.

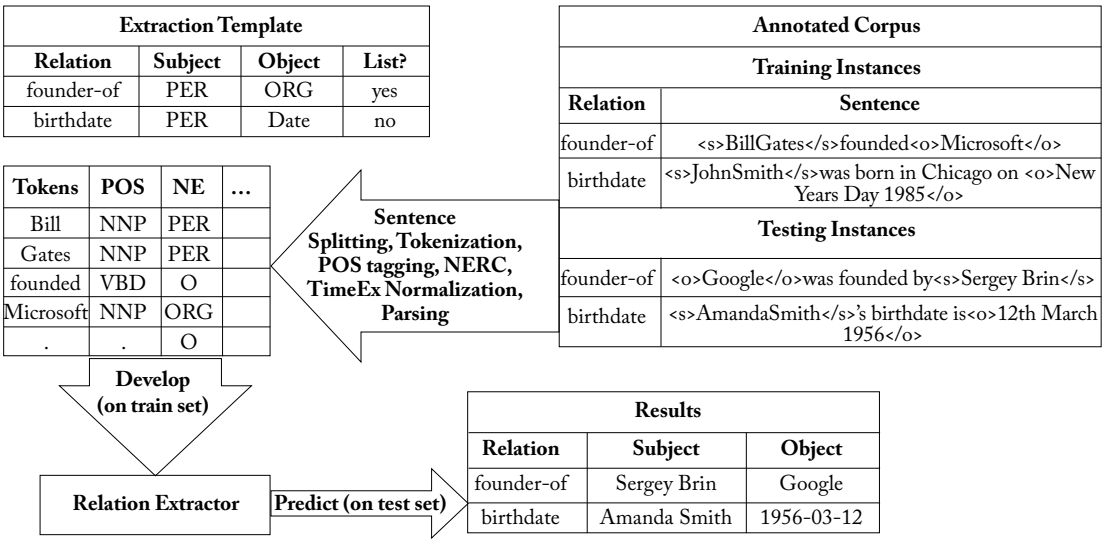


Figure 4.1: Typical relation extraction pipeline.

The input to the relation extraction task is usually a set of training documents, a set of testing documents, and an extraction template. The extraction template defines which relations are to be extracted and how they are defined, i.e., how many arguments they have and what concepts those arguments belong to. For instance, *founder-of* is defined as a relation between a person (PER) and an organization *(ORG): founder-of(PER, ORG)*, and is a list relation, i.e., it may have more than one object (*founder*) per subject and relation. Detailed NE types are not always given, e.g., the TAC KBP 2014 Slot Filling challenge does not provide the NE type of the object of the relation [75]. Next, the documents are pre-processed with several NLP steps to determine morphology, syntax, and semantics of the sentence. These pre-processing steps aim to help "understand" text in order to facilitate the extraction of relations.

One of the most important pre-processing steps is NERC. This is because, as mentioned in the previous section, relations are either extracted between named entities only, or between a mixture of named entities and general concepts (e.g., a person). For example, *Bill Gates* would be assigned the type PER and *Microsoft* the type ORG. Historically, the first series of relation evaluation efforts at the MUC conferences distinguished between the named entity types person (PER), location (LOC), organization (ORG), and miscellaneous (MISC) [25], though depending on the extraction template, more fine-grained types (e.g., Politician, Film) may be used.

Following pre-processing, the training set is used to develop relation extractors, after which they are applied to the test set to extract relations. If more than one relation per template is extracted, those extractions are validated. The definition of relations can help with this. For instance,

a company may have more than one founder, but every person only has two biological parents, which determines how many extractions per subject of each relation should be returned.

The output of the relation extraction task is a set of annotated test documents (often called *sentence-level extraction*) or a list of extraction triples (*instance-level extraction*). If the output is a list of extractions, they can be used to populate knowledge bases. More details on knowledge bases and their role in the relation extraction task are given in the next section.

4.3 RELATIONSHIP BETWEEN RELATION EXTRACTION AND OTHER IE TASKS

Relation extraction is generally defined as extracting relation mentions with their arguments from text. For traditional relation extraction, relation types and their arguments are defined in a schema, whereas for open information extraction [76], relation types are not pre-defined. An example of a template for a binary relation would be *Person, born on, Date*, though relations can have more than two arguments, such as *Government positions held*. As we have already seen, relation extraction builds on the task of NERC, since in order to extract relations between entities, entities have to be detected first.

Relation extraction has many challenges: in addition to those of the NERC task, the main challenge is that relations can be expressed in different ways. For instance *born on* can be expressed as *was born on*, *birthdate is on*, or *saw the light of day for the first time on*. Relation expressions are also not always unique to one relation, e.g., *works at* can imply *employee of* or *CEO of*. Some relation expressions are also very vague, for instance *Alfred Hitchcock's "The Birds" was very popular*. In that case, context is very helpful, i.e., that Alfred Hitchcock was a film maker and therefore *The Birds* is very likely to be a film. Relations can also span several sentences or only contain an indirect reference to one of the entities involved in the relation (e.g., *They*), as in the following example.

Example 4.1 In November 1963 *Capitol Records* finally signed a contract with *the Beatles* and announced plans to release the Beatles' single "I Want To Hold Your Hand" in December 1963 as well as their second album "With The Beatles" in January.

Pre-processing steps such as coreference resolution are therefore useful. As for NEs, relations can also be annotated in the text, or extracted and used to populate a knowledge base. For knowledge base population, a further step is to combine extractions, which is also part of the TAC KBP challenges.[1] In order to combine extractions, it is important to judge if extractions are synonymous, if they entail one another, or if they contradict one another. Both *recognizing textual entailment (RTE)*, i.e., recognizing if a statement can be inferred from another statement, and *contradiction detection (CD)*, i.e., if two statements cannot be true at the same time, are therefore important related tasks.

Event extraction is the task of identifying events, which are groups of relations that usually have participants, a start and end date, and a location. An example could be the opening of a

[1] http://www.nist.gov/tac/2014/KBP/SFValidation/index.html

restaurant. A restaurant is opened at a certain point in time at a certain location, but it might be closed and reopen in a different location, maybe even under a new owner. Events are very complex to extract, partly because the extraction process also tends to involve temporal analysis, and partly because the definition of event is quite fuzzy.

Although relation extraction is often solved in the form of a pipeline architecture, as shown in Figure 4.1, this can lead to errors being propagated. If an error is made at an early stage of the pipeline, it cannot be corrected again later. For example, if a NERC fails to recognize a named entity, a relation extractor cannot recover from the mistake. For this reason, alternative solutions to this may be proposed, which learn different tasks jointly. This allows for information from later processing stages (such as RE) to be used for earlier stages (such as NERC) to recover from errors. Methods to address this have been proposed for joint NERC and RE [73, 77] and for joint NERC, RE, and co-reference resolution [78, 79].

4.4 THE ROLE OF KNOWLEDGE BASES IN RELATION EXTRACTION

Knowledge bases are an integral part of the relation extraction process. They consist of a schema, sometimes also called extraction template, and data associated with the schema. The schema defines the structure of information, e.g., it might define that persons can be politicians or musicians, and that they have names and birthdates, that politicians are in addition associated with a party and musicians play instruments in bands with other musicians. The schema thus defines *classes* (e.g., Person), their *subclasses* (e.g., Politician), and *properties* (e.g., in-party). What is relevant for the task of relation extraction is that properties define which relations can hold between instances of classes, whereas their classes restrict the types of the relations' arguments. The data associated with the schema would then be examples of such politicians and musicians with their respective names, birthdates, parties, instruments, and bands. The relation extraction process typically starts with such a schema and the goal is then to annotate text with relations, or to populate the knowledge base with information, i.e., extract and add data. The latter is called *knowledge base population (KBP)* and has become popular, among other reasons, due to the TAC KBP series of challenges.[2] This series of evaluation efforts comprises several parts of the relation extraction pipeline, including extracting relations (slot filling) [75] and validating extractions (slot filler validation). For slot filling, the subjects or relations are already given and the task is then to find the objects of relations in a corpus.

Shared task evaluation efforts often use locally defined templates. However, with the rise of the World Wide Web and then the Semantic Web, web-based publicly available knowledge bases also became popular for the KBP task [80, 81].

[2] http://www.nist.gov/tac/2014/

4.5 RELATION SCHEMAS

There are two kinds of information that need to be described for relation extraction. First, we need information about classes (e.g., artist, track) and their relationships (e.g., released-track). This kind of information is published as a schema. Second, we need information about instances of these classes (e.g., David Bowie, Changes), which can be published in a dataset. Note though that this is optional: some websites contain semantic markup, often using `http://schema.org/`, but do not publish this in a separate dataset.

While for the purpose of relation extraction, schemas serve a similar purpose to locally defined templates (Section 4.4), they have a clear advantage in the way data is described, using unique identifiers for entities, called *uniform resource identifiers (URIs)*. Imagine a slot-filling task, for which the subjects of relations are given and the goal is to extract values for the objects of those relations. Some of the subjects may be ambiguous and refer to many different real-world entities. This ambiguity may be across classes (a jaguar can be an animal or a car brand), or within classes (there are many people named John Smith). Especially for the latter, it is very useful to have URIs as input for each subject. For instance, if the task is to extract birthdates, the RE approach would be expected to return only one result per subject entity, but would likely find more than one for *John Smith*. If there are several URIs with the name *John Smith* in the knowledge base, the RE approach can make use of this information and return several results, or, if other information about people named *John Smith* is already contained in the knowledge base, try to return the likely birthdate for that specific John Smith, based on that additional information.

There are several cross-domain datasets, with DBpedia having the most links to other datasets and effectively functioning as a hub for Linked Data. Other prominent examples of cross-domain datasets include Freebase [82], Yago [83], and Wikidata [84]. Domain-specific datasets exist for several different domains: governments release their data using Semantic Web standards, sciences make use of the technology to describe complex processes with ontologies, libraries and museums structure and release their data about books and artifacts and media, and social media providers enrich their websites with semantic information. One relation extraction method, distant supervision (see Section 4.10), relies to a large degree on both the schema and the data contained in Linked datasets.

It is important to know for relation extraction that information in different datasets is often interlinked. Information about the same entities may be found in more than one dataset, and to indicate this, datasets have links between them. This means relation extraction approaches which make use of information already contained in datasets can combine information from several datasets, as will become clear later. Furthermore, there are also links on the schema level (e.g., the property *birthdate* in one schema may be linked to the property *born-on* in another schema, or the class *album* may be linked to the class *musicalbum*), which allows for even easier combination of information in datasets, but also for combination of extraction schemas. For instance, one schema may define that music artists have birthdates, and another that they release albums. These could then be combined for extracting both relations.

4.6 RELATION EXTRACTION METHODS

Having seen how a typical relation extraction approach works, this section now details different relation extraction streams which are variations of the typical relation extraction approach described in the previous section. Relation extraction approaches can be broadly divided into rule-based methods, supervised methods, semi-supervised bootstrapping methods, unsupervised/Open IE methods, distantly supervised approaches, and universal schemas.

4.6.1 BOOTSTRAPPING APPROACHES

Bootstrapping approaches, a type of semi-supervised approach, were among the first relation extraction approaches, prominent pioneers being DIPRE [85] and Snowball [86]. A description of DIPRE is given below, since subsequent approaches used a similar architecture.

Algorithm 4.1 DIPRE [85]: extract(R, D)

```
while R < n do
    O ← findOccurrences(R, D)
    P ← generatePatterns(O)
    R ← M_D(P)
end while
return R
```

DIPRE consists of four simple steps (see Algorithm 4.1). The input to DIPRE is R, a set of $5 < s, o >$ tuples for the relation *PERSON author-of BOOK*, and D, a document collection, in this case the Web. The first step is to find occurrences of tuples on the Web. Next, patterns are generated. Third, pattern matches are generated; $M_D(p)$ is the set of tuples for which any of the patterns $p \in P$ is matched on a web page. This process is repeated until n relation occurrences are found.

This basic algorithm is used by almost all bootstrapping approaches, with slight variations. For instance, the input to the algorithm might be examples as well as extraction patterns or extraction rules. Matching of patterns can be handled in different ways, using exact or inexact matching. The most interesting part of the algorithm is how patterns are generated. In DIPRE, this is very basic: a pattern is created by grouping sentences for which the sequence of words between *person* and *book* match, and for which *person* and *book* are in the same order. Next, specificity is measured: if a pattern matches too many sentences and as a result specificity is above a manually tuned threshold t, the pattern is rejected. If specificity is too low and only the same book is found with that pattern, the pattern is rejected too. This already hints at one downside of bootstrapping approaches called semantic drift, which is their tendency to move too far away from R and create patterns which express different, related relations, which often co-occur with the same entity tuples, e.g., for *author-of*, this could be *editor-of*.

Subsequently, bootstrapping models have been researched to improve on the DIPRE model. Prominent large-scale bootstrapping models include KnowItAll [87] and NELL [88].

KnowItAll [87] is a Web-scale information extraction system, which relies on the scale and redundancy of the Web to provide enough information and validate it. By redundancy, we mean here that much information on the Web is available in multiple places, which means that these multiple information sources can be used to verify facts or fill in missing information. In contrast to DIPRE, it does not start with a single relation, but with several, and also contains methods for extending the extraction schema. KnowItAll consists of four modules: an extractor, a search engine interface, an assessor, and a bootstrapping component.

The extractor component applies Hearst patterns [89] to extract instances for classes (these would be instances of books in DIPRE). Hearst patterns, described further in Chapter 6, are lexico-syntactic extraction rules such as *NP1 is a NP2*, where NP2 is the name of a class such as *books*, and NP1 is the name of an instance of that class. Using the search engine interface, these patterns (with NP1 left blank) are then formulated as search queries to retrieve web pages containing NP1. The component further contains relation extraction rules, e.g., *NP1 plays for NP2*, representing the relation *playsFor(Athlete,SportsTeam)*. Once all extraction rules have been applied, extracted patterns are validated by the assessor.

The assessor measures co-occurence statistics of candidate extractions with discriminator phrases, which are highly frequent extraction patterns. This means for each search query (e.g., *Tom Cruise starred in X*), the number of search results is recorded and the PMI (Pointwise Mutual Information) of the entity *Tom Cruise* and the pattern is computed.

KnowItAll then uses bootstrapping in combination with the assessor to validate extractions. For each class, the 20 instances with the highest average PMI are retrieved. These are then used to train conditional probabilities for each extraction pattern. Seeds for negative instances are taken from positive instances for other classes. The best five extraction patterns for each class are saved; the rest are discarded. A Naive Bayes classifier is then trained, combining evidence from those five extraction patterns to classify if an entity (e.g., *Tom Cruise*) is an instance of a class (e.g., *actor*). Instead of just selecting the best extraction patterns once, a bootstrapping process can be used: once the best five extraction patterns have been determined, they can be used to find a new set of instances with high PMI. To ensure high-quality extraction patterns, incorrect instances are also removed manually.

NELL [88] is a bootstrapping system that extracts information from the Web to populate a knowledge base, and learns to extract information more accurately over time. Like KnowItall, NELL is based on the hypothesis that the large amount of redundant information on the Web is a huge advantage for learning mechanisms. The main differences are that the bootstrapping component is more sophisticated and that NELL combines extractions from different sources on the Web: text, lists, and tables. Similar to KnowItAll, it learns to extract which instances belong to which classes, and which relations hold between instances of those classes.

Information is extracted from unstructured information on the Web (text), as well as semi-structured data (lists and tables). Extractors are trained in concert using coupled learning, based on CPL for free text and CSEAL for lists and tables [90]. CPL, similarly to KnowItAll, relies on co-occurrence statistics between noun phrases and context patterns to learn extraction patterns. CSEAL uses mutual exclusion relationships to provide negative examples, which are then used to filter overly general lists and tables.

In addition, NELL learns morphological regularities of instances, and uses probabilistic Horn clause rules to infer new relations from relations it has already learned. For learning morphological regularities, NELL uses a coupled morphological classifier (CMC). For each class, a logistic regression model is trained to classify noun phrases based on morphological and syntactic features (e.g., words, capitalization, affixes, POS tags). The Rule Learner learns probablistic Horn clauses to infer new relations from relations that are already present in the knowledge base.

The learning system starts with a knowledge base (123 classes, 55 relations, and a few instances for classes and relation triples), and gradually populates and extends it. After the extraction component has extracted a belief, the precision of the belief is improved by consulting external data resources or humans. The most strongly supported beliefs are promoted to facts and integrated into the knowledge base. For the remaining extraction steps, the extractor always uses the updated knowledge base.

NELL typically enables instances and relations to be extracted with a relatively high precision initially [88], and the different extractors tend to be complementary. However, it demonstrates a problem that is very typical of bootstrapping approaches: extraction precision declines over time. This could, however, be solved by allowing a human to interact with the system during learning, using active learning [91].

4.7 RULE-BASED APPROACHES

Another method of developing a relation extraction system is to use a rule-based or pattern-based approach. Rule-based relation extraction approaches make use of domain knowledge, which are encoded as relation extraction rules [92–94]. There are two different types of rule-based approaches: those which are stand-alone, and those which learn rules for inference to complement other relation extraction approaches. The former typically relies on a rule grammar to encode complex rules and dependencies between them. An example of a rule from [94] is *BandMember followed within 30 characters by Instrument*. For both *BandMember* and *Instrument*, pre-compiled gazetteers plus regular expressions are used to recognize them. A downside of such rule-based approaches is that they are not able to generalize to unseen textual patterns, and generally have poor recall.

Rule-based approaches used for inference purposes include **KnowledgeVault** [95], which uses a path ranking algorithm. This starts with a pair of entities that are known to be in a relation according to a (seed) knowledge base, then performs random walk over the knowledge graph to find other paths that connect these entities. Thus it can learn that if two people share a child, they

are very likely to be married, or that people often study at the same university as their siblings. A downside of using rules learned for inference purposes is that rules learned on a small knowledge base might not generalize to new relations, e.g., for some of the runs submitted to TAC KBP 2014, using such learned rules caused a drop in performance [75]. To mitigate this, only rules learned based on a large enough amount of evidence should ideally be used.

4.8 SUPERVISED APPROACHES

Supervised approaches are currently the best performing stream of relation extraction approaches, provided that a sufficient amount of labeled training data is available. They closely follow the general relation extraction pipeline (Figure 4.1): given a corpus annotated with relations, they pre-process the sentences with typical NLP pre-processing steps (POS tagging, parsing, identifying NEs, etc.), then extract features, train a model, and predict relations on a test set.

Features are extracted from both positive and negative training examples, and serve as cues for learning whether two NEs are related or not. During training, the model observes how often a feature co-occurs with positive as opposed to negative training examples, and based on that learns a weight for each feature, which can again be positive or negative. For instance, if for the relation *author-of* the phrase between two entities is *is the author of*, it would likely get a high positive weight, whereas the phrase *is the director of* would likely get a negative weight.

Typical relation extraction features (used for example in [73, 81]) include:

- n-grams of words to left and right of entities;
- n-grams of POS of words to left and right of entities;
- flag indicating which entity came first in sentence;
- sequence of POS tags and bag of words (BOW) between the two entities;
- dependency path between subject and object;
- POS tags of words on the dependency path between the two entities; and
- lemmas on the dependency path.

Other options for features include kernel methods [96, 97] or, more recently, relation embeddings [98], which learn higher-dimensional representations of the labeled data. Such representations can be seen as latent features and thus eliminate the need for feature engineering, which can be cumbersome. In terms of models, a large variety is used, such as SVM, Maximum Entropy models, Markov logic networks, or (deep) neural networks.

A typical supervised RE tool is the **Stanford relation extractor**,[3] which is built on top of the Stanford CoreNLP framework. It detects the relations *Live_In*, *Located_In*, *OrgBased_In* and *Work_For*. It is trained on TREC data, but is easy to retrain on other corpora and customize.

[3] http://nlp.stanford.edu/software/relationExtractor.html

4.9 UNSUPERVISED APPROACHES

Unsupervised relation extraction approaches became popular soon after supervised systems, with *open information extraction* systems such as TextRunner [99], ReVerb [100], and OLLIE [101]. Open information extraction is a paradigm to use simple and scalable methods to extract information which is not restricted beforehand. This is in contrast to the semi-supervised approaches described in the previous section, which use pre-defined extraction schemas. Open IE can thus be seen as a subgroup of unsupervised approaches. For Open IE methods, this means they have to infer the classes of entities, and relations between them. We describe below the first Open IE approach, in order to introduce the research stream, and point out shortcomings and improvements of subsequent research.

TextRunner [99] was the first fully implemented and evaluated Open IE system. It learns a Conditional Random Field (CRF) model for relations, classes, and entities from a corpus using a relation-independent extraction model. First, it runs over the whole corpus once and annotates sentences with POS tags and NP chunks. To determine whether the relation should be extracted or not, the system uses a supervised classifier. This supervised classifier is trained by parsing a small subset of the corpus and then heuristically labeling sentences as positive (trustworthy) and negative (not trustworthy) examples, using a small set of hand-written rules. The classifier then makes the decision for unseen sentences based on POS tags instead of the parse tree, because it would be too expensive to parse the whole corpus. To resolve synonyms, TextRunner performes unsupervised clustering of relations and entities based on string and a distributional similarity [99].

ReVerb [100] addresses two shortcomings previous Open IE systems have: incoherent extractions and uninformative extractions. Incoherent extractions occur when the extracted relation phrase has no meaningful interpretation. This is due to the fact that decisions in TextRunner are made sequentially. An example would be the relation *contains omits* which is extracted from the sentence *The guide contains dead links and omits sites.* To solve this, syntactic constraints on which relations to extract are introduced. The first is that a relation phrase either has to be a verb (e.g., *invented*), a verb followed by a preposition (e.g., *located in*), or a verb followed by nouns, adjectives, or adverbs and a preposition (e.g., *has atomic weight of*). Also, if there are multiple possible matches, the longest possible match is chosen. If adjacent sequences are found (e.g., *wants*, *to extend*), these are merged (e.g., *wants to extend*). Lastly, the relation has to appear between the two arguments in a sentence.

Uninformative extractions omit important information, e.g., for the sentence *Faust made a deal with the devil*, TextRunner would extract *Faust, made, a deal* instead of *Faust, made a deal with, the devil*. These can partly be captured by syntactic constraints. However, this may cause the extraction of overly specific relations such as *is offering only modest greenhouse gas reduction targets at*. To tackle this, a lexical constraint is introduced: a relation has to appear with at least 20 distinct arguments in a sentence in order to be meaningful.

Although open information extraction is a promising research paradigm and it is possible to map clusters of relations to extraction schemas afterwards, it also provides unneccessary restriction

for the task of knowledge base population. A relation extraction approach that is developed for a schema can be expected to have higher performance than one which is not restricted to a schema. This is due to the problems mentioned above of incoherent and uninformative relations. These issues are less pronounced for bootstrapping methods.

On the other hand, Open IE methods which do not use a pre-defined schema make it more broadly applicable for different application scenarios. It can be seen as a benefit that, depending on the scenario, the output can be mapped to different schemas in a post-processing step. Tools and demos for Open IE are available as standalone distributions for the systems by the University of Washington's KnowItAll project (TextRunner, ReVerb, Ollie, Srlie, Relnoun),[4] and distributed by Stanford NLP researchers, integrated with Stanford CoreNLP [102].[5]

4.10 DISTANT SUPERVISION APPROACHES

Distant supervision is a method for automatically annotating training data using existing relations in knowledge bases. The first approach was proposed by Craven and Kulien [103] in 1999 as a method for knowledge base population for the biomedical domain, though they called their approach "weakly labeled." Although results were promising, this approach did not gain popularity until 10 years later, when the term "distant supervision" was coined. The re-surfacing of these approaches may be partly due to the increasing availability of large knowledge bases on the Web. Mintz et al. [81] define the distant supervision assumption as:

> If two entities participate in a relation, any sentence that contains those two entities might express that relation.

How such an approach works in practice is visualized in Figure 4.2. The input to the approach is a knowledge base, containing a set of classes and relations, instances of those classes and examples of those relations, and training and test corpora. The training corpus is pre-processed to recognize named entities, then searched for sentences containing both the subject and the object of known relations (e.g., *Virginia* and *Richmond* for the relation *contains(LOC, LOC)*). Sentences containing both the subject and the object of known relations are considered positive training data for the relation; others are negative training examples (NIL). A supervised classifier (e.g., Naive Bayes, SVM, MaxEnt) is then trained and applied to a test corpus. Overall, the learning process is the same as that for supervised learning, merely the training data labeling process is different (automatic instead of manual). As such, the approach has all the advantages of supervised learning (high precision extraction, output with respect to extraction schema), and additional advantages, since no manual effort is required to label training data. Extraction performance is slightly lower than for supervised approaches, due to incorrectly labeled training examples. One of the main causes of incorrectly labeled training data is the ambiguity of surface forms (e.g., *Virginia* can be

[4] http://ai.cs.washington.edu/projects/open-information-extraction
[5] http://nlp.stanford.edu/software/openie.html

a person name or a location) [104, 105]. Improving the automatic labeling process has been the main focus of distant supervision research since, as a survey by [106] outlines.

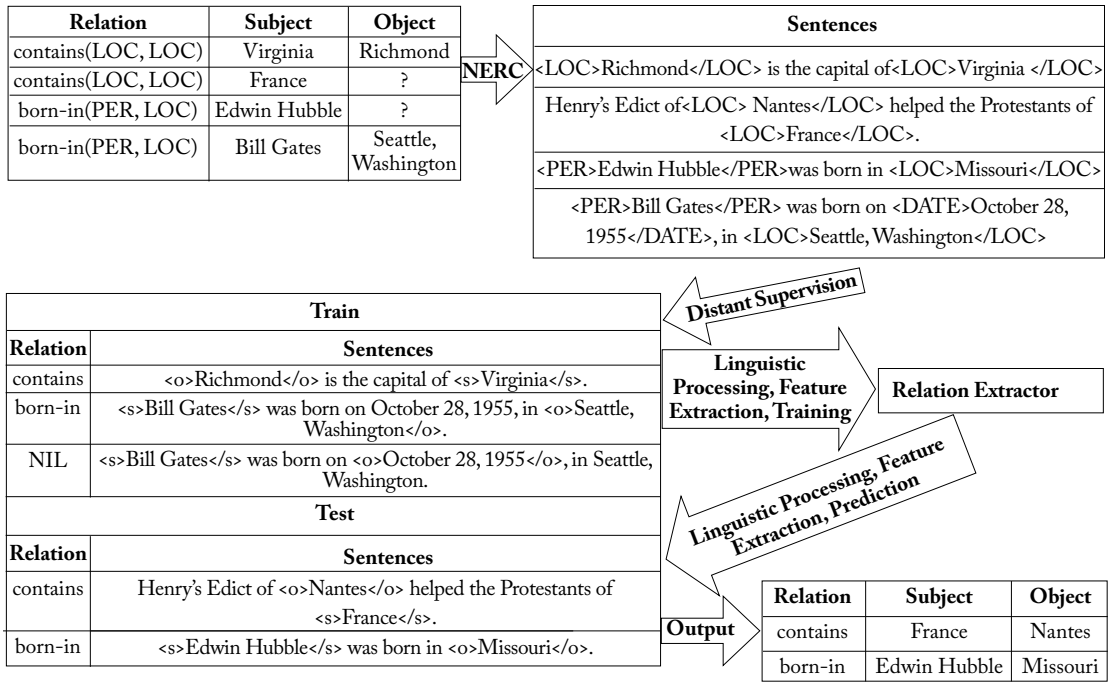


Figure 4.2: [81] Distant supervision method overview.

4.10.1 UNIVERSAL SCHEMAS

The idea of univeral schemas [107] combines the benefits of open information extraction and distant supervision. Methods for modeling missing data to reduce false negatives assume that not all relation mentions (e.g., *Microsoft founded-by Bill Gates*) are contained in the KB, which leads to them being labeled as negative training data. Universal schemas, on the other hand, address the idea that not all relations (e.g., *founded-by*) are contained in the KB. They then aim at combining relations defined by the KB schema with relations discovered in text with Open IE methods. Recall that Open IE methods do not rely on an extraction schema, and instead cluster surface patterns (e.g., *founded*, *was founded by*) to relations. To do this, a matrix is constructed with rows representing entity pairs and columns representing both relations, defined by a KB and Open IE style patterns. In order to predict values for unseen relations, matrix factorization is used.

4.10.2 HYBRID APPROACHES

Finally, it is worth noting that, in addition to universal schemas, there are a large number of other hybrid approaches aimed at combining the advantages of several types of approaches. There are methods that combine hybrid pattern-based and supervised approaches; approaches that combine distant supervision and rule-based [108]; approaches that combine distant supervision and (direct) supervision [109]; and finally methods that combine universal schema and rule-based [110] approaches.

A popular state-of-the-art hybrid relation extraction tool is **SampleJS** [109].[6] It uses distant supervision to automatically obtain noisy training examples, and active learning to iteratively improve the quality of the training data. The approach addresses some of the problems outlined in the introduction, e.g., that relations can overlap. The distribution comes with a pre-trained model, using a mixture of the Freebase relation schema and the TAC KBP 2013 schema, resulting in 41 relations, and Wikipedia as a training corpus. The approach can be re-trained for other schemas and/or corpora.

4.11 PERFORMANCE

There a several training corpora for supervised relation extraction, though not nearly as many as for named entity recognition. Corpora include ACE, Ontonotes, and TREC and TAC KBP corpora. ACE and Ontonotes also include annotations for related NLP tasks, such as NER and co-reference resolution, making them ideal for studying the interdependence between those tasks.

The performance of relation extraction approaches heavily depends on the relation type. For learning-based approaches it depends on the number of training examples per relation; for approaches which make use of background knowledge such as distant supervision and rule-based approaches, it depends on the quality of the background data and moreover on the genre of the corpus (e.g., newswire, Wikipedia, biomedical data). Evaluation initiatives such as TAC KBP enable the objective comparison of different approaches by controlling for some of these parameters. In the TAC KBP 2014 challenge, submissions used all the different kinds of relation extraction approaches discussed in this chapter, i.e., direct supervision, distant supervision, pattern-based, rule-based, bootstrapping, Open IE approaches, and universal schema approaches. Trends that emerged are that 14 out of 18 systems submitted to the task used distant supervision, most systems combined distant supervision with rules, and that the top three systems all used distant supervision. A successful way of incorporating direct supervision and distant supervision is active learning, one of them being the foundation for **SampleJS** [109]. The one universal schema approach also performed well, though not as well as distant supervision combined with direct supervision or combined with rules. Groups that used hand-crafted patterns performed average or worse than average. The best out of those approaches was one that combined Open IE and hand-crafted patterns. This suggests that for knowledge base population with relations, machine

[6] http://nlp.stanford.edu/software/mimlre.shtml

learning-based approaches vastly outperform pattern-based approaches. Overall, human performance on the TAC KBP 2014 task is an F1 of 70.36% and the best-performing approach reaches 36.77%.

A downside of the TAC KBP evaluation setup is that the number of training examples per relation differs widely, making it difficult to compare performance across relations. To give some indication of relation extraction difficulty, Table 4.1 lists the P, R, and F1 of the most frequent relations in the **SampleJS** [109] evaluation setup, which are partly TAC KBP 2014 relations and partly Freebase relations.

Table 4.1: Comparison of performance for different relations

Method	P	R	F1
Employee of	32	46	38
Top members	26	60	36
(Org:) alt names	48	39	43
Title	26	35	30
Spouse	54	85	66
Origin	43	70	53
Cause of Death	93	39	55
Children	62	18	27
Date of Death	64	39	48
Age	97	90	93

As the table shows, performance varies widely depending on the relation, e.g., for *age* the F1 is 93%, whereas for *children* it is merely 27%. It is worth noting that those evaluation challenges do not necessarily give a realistic insight into real-world relation extraction performance. Extraction performance dramatically increases with more training data, and also with discarding relations extracted with low confidence. A hybrid Web-scale relation extraction system built at Google [95] managed to achieve an AUC score (area under the precision-recall curve) of 0.927 by discarding all relations extracted at a confidence below 0.9.

To summarize, the most successful relation extraction approaches are hybrid learning-based approaches, extracting information using a number of different methods. They use large quantities of training data and extract relations from a number of different sources.

4.12 SUMMARY

Table 4.2 summarizes the key points of the different types of approaches. All relation extraction streams have different advantages and disadvantages. They differ with respect to how much initial input is required, whether human intervention is required during the learning process, and

Table 4.2: Comparison of different minimally supervised relation extraction methods

Method	Input	Output	Description	Advantages	Disadvantages
Boot-strapping	Unlabeled text, relation schema, rules and/or examples	Extraction rules, relations	Using a small set of extraction rules, extract examples, keep prominent ones, iteratively learn more extraction rules and examples	Easy to add new rules, can also be supplied by user	Often low recall and/or manual refinement needed for high precision
Rule-based	Unlabeled text, relation schema, rules and NE gazetteers	Relations	Using extraction rules and gazetteers, extract relations	Easy to add new rules, can also be supplied by user	Often low recall, much manual effort to develop
Supervised	Labeled text, relation schema	Relations	Using a schema and labeled training data, train model	Currently highest precision and recall for schema-specific relation extraction	Up-front effort of labeling data, risk of overfitting training set
Open IE	Unlabeled text	Groups of relations	Discover groups of relations from text using clustering, keep prominent ones	No knowledge about text needed	Difficult to make sense of groups and map to relation schemas
Distantly Supervised	Unlabeled text, relation schema, examples	Extraction model, relations	Using a schema and examples of relations, automatically annotate training data, train a model to extract more relations	Extracting relations with high recall and precision	Initial examples required
Universal Schema	Several partly populated knowledge bases	Unified knowledge	Take several KBs defined by different schemas, partly populated with relations, predict union of KBs	Integrate relations defined by different schemas after extraction	For small KBs it can be faster to do this manually

how suitable they are for knowledge base population. Bootstrapping methods may only need a handful of initial examples, but as discussed in Section 4.6.1, the problem of semantic drift may require additional human intervention during the learning process. They are suitable for knowledge base population, as extraction is performed with respect to an extraction schema. Rule-based approaches need a large number of hand-crafted rules plus NE gazetteers and often have a low recall. In real-world scenarios rule-based approaches are still used often, even though they do not represent the state of the art in performance. This is because they are easy to develop and expand and they do not require much up-front effort, such as labeling a training corpus. A twist on rule-based extraction which does not require effort are rule learning systems, which learn high-precision inference rules from seed knowledge bases, and which can be used in conjunction with other relation extraction methods.

Supervised relation extraction methods require labeled training examples with respect to a relation schema. They are the best-performing relation extraction stream for knowledge base population, however, they can also require a large amount of up-front effort if no appropriate training data is available. Open IE approaches do not require any input to start with. This means, however, that the output of such approaches are merely clusters of relations and there is no straightforward way of mapping them to an existing relation schema. Therefore, they are interesting for scenarios for which such an extraction schema is not available or for which the goal is to extend an extraction schema, but they are less suitable for knowledge base population.

Distant supervision approaches require a small amount of input, around 30 examples per relation at least, and use this information to label training data, then perform supervised learning. The abundance of such information on the Web in existing knowledge bases makes it possible to gather such information automatically, and therefore they do not require human input. Because they then also use the schema associated with relation examples for training, they are very suitable for knowledge base population. Even if labeled training data is available, as for the TAC KBP evaluation campaigns, adding additional distantly labeled data increases performance. Universal schemas are an approach for unifying relations defined by different schemas. These relations can be extracted using different RE methods, such as distant supervision and Open IE, which is one of their main strengths.

Which relation extraction method is best really therefore depends on the task at hand. If the task is exploratory, Open IE is eminently suitable, and many tools are available to get an idea of their performance. For knowledge base population, the current state of the art consists of hybrid approaches of supervised relation extraction methods combined with distant supervision or with inference rules learned from a seed knowledge base.

CHAPTER 5

Entity Linking

Having determined which expressions in text are mentions of entities, a follow-up task is entity linking (or entity disambiguation) [111]. It typically requires annotating a potentially ambiguous entity mention in a document (e.g., Paris) with a link to a canonical identifier describing a unique entity in a database or an ontology (e.g., `http://dbpedia.org/resource/Paris`). Approaches have used different entity databases as a disambiguation target (e.g., Wikipedia pages [112–114]) and Linked Open Data resources (e.g., DBpedia [115, 116], YAGO [117], Freebase [118]). Many disambiguation targets have commonalities and links with each other, and it is often possible to map between them [119]. Linking entity mentions to such resources is core to automatic semantic annotation of web documents, knowledge base population, semantic search, cross-lingual information access, and other related tasks.

Entity linking is a challenging task, as methods need to handle firstly *name variations*, where the same entity can be referred to in many different ways (e.g., New York and the Big Apple). The second challenge is the high *entity ambiguity*, i.e., the same string can refer to more than one entity (e.g., Paris, France vs. Paris, Texas vs. Paris Hilton). Since DBpedia contains millions of instances, entity ambiguity is a very significant challenge, as it is not uncommon for a textual mention to have more than a hundred candidates in the knowledge base. The last very significant challenge is that of *missing entities*, i.e., detecting when there is no appropriate target entity in the knowledge base.

Named Entity Linking (NEL) approaches typically include a candidate selection phase, which identifies all candidate knowledge base entries for a given entity mention in text. This is followed by a reference disambiguation (or entity resolution) phase, which determines which is the most probable target entity amongst all candidates. This disambiguation step tends to use contextual information from the text, as well as knowledge from the ontology to choose the correct URI. Text mentions can be disambiguated either independently of each other, or jointly across the entire document [116, 120].

Much of the work on entity linking makes the closed world assumption, i.e., that there is always a target entity in the knowledge base. For many document types (particularly social media) and applications, however, this is very limiting, since the entities mentioned are often not noteworthy or established enough to have already been included in Wikipedia or an LOD resource (see the previous discussion in Chapter 3 about newly emerging entities). Therefore, the harder, open NEL task is to either return a matching entry from the target knowledge base (e.g., DBpedia URI, Wikipedia URL) or NIL to indicate that there is no matching entity.

5.1 NAMED ENTITY LINKING AND SEMANTIC LINKING

Semantic linking is concerned with the wider problem of determining which topics (e.g., technology) and entities (e.g., iPad) best capture the meaning of a document. Semantic linking is also referred to as the "aboutness" task [121], or as the "C2W" (Concepts to Wikipedia) and "Sc2W" (Scored concepts to Wikipedia) tasks [122].

Correct semantic linking is often reliant on subtle contextual clues, and needs to be combined with world knowledge. For example, a tweet mentioning iPad makes Apple a relevant entity, because of the implicit relation between the two entities. Consequently, semantically linked entities and topics do not need to be mentioned explicitly in the document text. From an implementational perspective, the aboutness task involves identifying relevant entities at the whole-document level, skipping the NER step of determining explicit entity mentions first.

In contrast, the NEL task, which is the focus of this chapter, is concerned with disambiguating only explicitly mentioned entities. In that case, not only do the entity mentions need to be identified via NERC, but also a target unique entity identifier (or NIL) needs to be assigned to that entity mention. Since entity mentions that are not recognized will not be disambiguated, NEL peformance is heavily dependent on NERC performance.

5.2 NEL DATASETS

The first NEL corpora were created as part of the TAC-KBP entity linking challenges [123, 124]. These consist of documents and one given entity for each document, which needs to be disambiguated to a knowledge base entry or NIL. Since the entity mention is already provided and there is only one per document, these corpora are somewhat limiting.

Another older dataset is AQUAINT,[1] which unfortunately is annotated against a now rather outdated version of Wikipedia. It also links not just named entities but also terms to their Wikipedia pages. This makes it more suitable for evaluating semantic linking, rather than LOD-based NEL approaches.

The AIDA/CoNLL corpus [116] consists of news articles annotated with YAGO URIs and split into training, development, and testing portions. The testing part alone contains 231 documents with 4,485 target annotations.

In follow-up work, the authors released the smaller AIDA-EE dataset [125], which contains 300 documents with 9,976 entity names, linked to a 2010 version of Wikipedia. This dataset is biased in that all entity mentions were first identified automatically with the Stanford NER tool, and only those mentions were then linked manually to the correct Wikipedia page. In practice, this means that entity mentions missed by the Stanford system will be considered incorrect during evaluation, even though the given NEL system may have been correct.

[1] http://www.nzdl.org/wikification/docs.html

Another recent dataset is N3[2], which contains three corpora of English and German news articles with manually annotated entities, linked to DBpedia URIs.

Microblog corpora created specifically for LOD-based NEL are very limited. Some, e.g., Ritter's corpus [126], contain only entity types, whereas those from the MSM challenges [127, 128] have anonymized the URLs and user name mentions. Corpora created for semantic linking, such as Meij [121], are not well suited for evaluating named entity linking, due to the presence of implicit entities and generic topics (e.g., "website," "usability," and "audience").

The YODIE Twitter corpus contains just under 800 tweets, annotated with DBpedia URIs by multiple experts [129]. The tweets contain hashtags, URLs, and user mentions, including many with corresponding DBpedia URIs (e.g., @eonenergyuk). The publicly available dataset[3] is split into equally sized training and evaluation parts.

5.3 LOD-BASED APPROACHES

Ontology-based entity linking and disambiguation methods typically collect a dictionary of labels for each entity URI, using the Wikipedia entity pages, redirects (used for synonyms and abbreviations), disambiguation pages (for multiple entities with the same name), and anchor text used when linking to a Wikipedia page. This dictionary is used for identifying all candidate entity URIs for a given text mention. Next is the disambiguation stage, where all candidate URIs are ranked and a confidence score is assigned. If there is no matching entity in the target knowledge base, a NIL value is returned.

Typically methods use Wikipedia corpus statistics coupled with techniques (e.g., TF/IDF) which match the context of the ambiguous mention in the text against the Wikipedia pages for each candidate entity [115]. Michelson et al. [130] demonstrate how such an approach can be used to derive a user's topic profile from their tweets, based on Wikipedia categories.

5.3.1 DBPEDIA SPOTLIGHT

One widely used DBpedia-based semantic annotation system is *DBpedia Spotlight* [115]. It is a freely available and customizable web-based system, which annotates text documents with DBpedia URIs. It targets the DBpedia ontology, which has more than 30 top-level classes and 272 classes overall. It is possible to restrict which classes (and their sub-classes) are used for named entity recognition, either by listing them explicitly or through a SPARQL query. The algorithm first selects entity candidates through lookup against a Wikipedia-derived dictionary of URI lexicalizations, followed by a URI ranking stage using a vector space model. Each DBpedia resource is associated with a document, constructed from all paragraphs mentioning that concept in Wikipedia. The method has been shown to out-perform OpenCalais and Zemanta (see Section 5.4) on a small gold-standard of newspaper articles [115].

[2] http://aksw.org/Projects/N3NERNEDNIF.html
[3] https://gate.ac.uk/applications/yodie.html

RT @XXXX Eyeopener vs. Ryerson Quidditch team this Sunday at 4 p.m. Anyone know where to get cheap brooms? #Ryerson @XXXX #Rams

@XXXX http://www.youtube.com/watch?v=eLMui7zBiXo we beat kilkenny after they beat us for the last 4 years in the hurling. Woo!!!

Kk its 22:48 friday nyt :D really tired so imma go to sleep :) good nyt x god bles xxxx

Amazon U.K. Offering HTC Desire Z Unlocked http://dbpedia.org/resource/Irish_Museum_of_Modern_Art earlier in Lo... http://bit.ly/bsyz9H URL

RT @XXXX: Eventful morning for Oklahoma State's Darrell Williams. Won Big 12 Rookie of the Week Award- and got charged with f...

Figure 5.1: DBpedia Spotlight results on tweets.

Figure 5.1 shows several tweets annotated with DBpedia Spotlight. The results clearly demonstrate the need for tweet spelling normalization, as well as the difficulties Spotlight has with recognizing URLs. As exemplified here, by default the algorithm is designed to maximize recall (i.e., annotate as many entities as possible, using the millions of instances from DBpedia). Given the short, noisy nature of tweets, this may lead to low accuracy results. Further formal evaluation on a shared, large dataset of short social media messages is required, in order to establish the best values for the various DBpedia Spotlight parameters (e.g., confidence, support).

5.3.2 YODIE: A LOD-BASED ENTITY DISAMBIGUATION FRAMEWORK

YODIE[4] is a NED framework built on top of GATE. It combines GATE's ANNIE NER system with a number of widely used URI candidate selection strategies, similarity metrics, and a machine learning model for entity disambiguation, which determines the best candidate URI. For each NE mention and for every candidate, YODIE calculates a number of normalized scores, which reflect the semantic similarity between the entity referred to by the candidate and the context of its mention:

- *Relatedness Score*: introduced in [131], uses the proportion of incoming links that overlap in the Wikipedia graph to favor congruent candidate choices.
- *LOD-based Similarity Score*: similar to above but based on the number of relations between each pair of URIs in the DBpedia graph (introduced next).
- *Text-based Similarity Scores*: these measure the similarity between the textual context of the mentioned named entity and text associated with each candidate URI for that mention (see below).

[4] https://gate.ac.uk/applications/yodie.html

The process of deciding how to combine these scores to select the best candidate URI is non-trivial. YODIE uses LibSVM[5] to select the best candidate.

Training data for the model consists of one training instance for each candidate generated by the system on the training corpus. Each instance receives a target of `true` if the candidate is the correct disambiguation target and `false` otherwise. The values of the various similarity metrics are used as features. This means that at application time, the model assigns to each candidate a class of true or false, along with a probability. This classification is independent of the other candidates on that entity, but ranking of the candidate list can be performed on the basis of the probability. The most probable URI is thus assigned as the target disambiguation for this entity, unless its probability is below a given confidence threshold, in which case "nil" is assigned. It was trained on TAC KBP data from 2009 to 2013, excluding the 2010 set,[6] along with the AIDA training set [116], and the tweet training set introduced in Section 5.2.

5.3.3 OTHER KEY LOD-BASED APPROACHES

Two other openly available, state-of-the-art LOD-based NED systems are AIDA [116, 125] and AGDISTIS [120]. They are both graph-based disambiguation approaches, which aim to jointly disambiguate all entities mentioned within a text. While such approaches tend to work very well on longer documents, their performance on tweets and other short social media posts is typically worse.

AGDISTIS [120] is a graph-based NEL approach which was designed to be knowledge base-agnostic. It combines the Hypertext-Induced Topic Search (HITS) algorithm with label expansion strategies and string similarity measures. It has been tested both with DBpedia and YAGO2 and, similar to most other NEL systems covered here, disambiguates with respect to the three standard classes of Person, Organization, and Place. First, for each named entity, a number of candidates are identified and then, in a second step, the HITS algorithm is used to compute the optimal assignment by constructing a disambiguation graph. All algorithms were chosen to have polynomial time complexity, so AGDISTIS is applicable to larger web documents.

Another notable example is TagMe, which was designed specifically for annotating short texts with respect to Wikipedia [132]. A comparative evaluation of all openly available state-of-the-art approaches, except the most recent AGDISTIS, is reported in [122], using several available news datasets.

Lastly, a NEL system linking to YAGO is the LINDEN [117] framework. It makes use of the richer semantic information in YAGO (semantic similarity), in addition to Wikipedia-based information (using link structure for semantic associativity). The method is heavily dependent on the Wikipedia-Miner[7] toolkit [114], which is used to analyze the context of the ambiguous entity mention and detect the Wikipedia concepts that appear there. Evaluation on the TAC-

[5] http://www.csie.ntu.edu.tw/~cjlin/libsvm/
[6] http://www.nist.gov/tac/2013/KBP/
[7] http://wikipedia-miner.cms.waikato.ac.nz/

KBP2009 dataset showed LINDEN outperforming the highest ranked Wikipedia-only systems, which participated in the original TAC evaluation. Unfortunately, LINDEN has not been compared directly to DBpedia Spotlight on a shared evaluation dataset.

5.4 COMMERCIAL ENTITY LINKING SERVICES

There are a number of commercial online entity linking services which assign Linked Data URIs. The NERD online tool [119] allows their easy comparison on user-uploaded datasets. It also unifies their results and maps them to the Linking Open Data cloud. Here we focus only on the services used by research methods surveyed here [133–135].

Zemanta[8] is an online semantic annotation tool, originally developed for blog and email content to help users insert tags and links through recommendations. Figure 5.2 shows an example text and the recommended tags, potential in-text link targets (e.g., the W3C Wikipedia article and the W3C home page), and other relevant articles. It is then for the user to decide which of the tags should apply and which in-text link targets they wish to add. In this example, in-text links have been added for the terms highlighted in orange, all pointing to the Wikipedia articles on the respective topics.

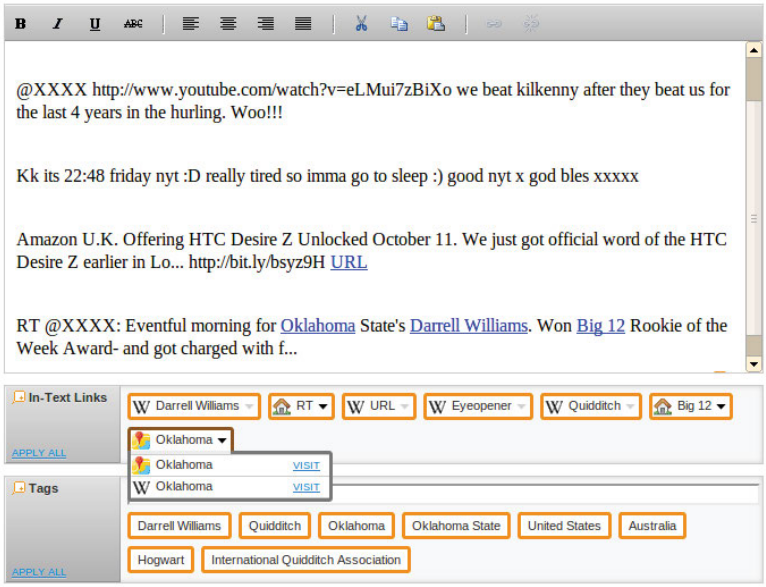


Figure 5.2: Zemanta's online tagging interface.

Open Calais is another commercial web service for semantic annotation, which has been used by some researchers on social media. For instance, Abel et al. [134] harness OpenCalais to

[8] http://www.zemanta.com

recognize named entities in news-related tweets.[9] The target entities are mostly locations, companies, people, addresses, contact numbers, products, movies, etc. The events and facts extracted are those involving the above entities, e.g., acquisition, alliance, company competitor. Figure 8.1 shows an example text annotated with some entities.

The entity annotations include URIs, which allow access via HTTP to obtain further information on that entity via Linked Data. Currently OpenCalais links to eight Linked Data sets, including its own knowledge base, DBpedia, Wikipedia, IMDB, and Shopping.com. These broadly correspond to the entity types covered by the ontology.

The main limitation of Calais comes from its proprietary nature, i.e., users send documents to be annotated by the web service and receive results back, but they do not have the means to give Calais a different ontology to annotate with or to customize the way in which the entity extraction works.

5.5 NEL FOR SOCIAL MEDIA CONTENT

The state-of-the-art LOD-based NEL approaches discussed above have been developed and evaluated predominantly on news articles and other carefully written, longer texts [111, 122]. As discussed in Section 5.2, very few microblog corpora annotated with LOD URIs exist and they are also small and incomplete.

Moreover, where researchers have evaluated microblog NEL, e.g., [67], state-of-the-art approaches have shown poor performance, due to the limited context, linguistic noise, and use of emoticons, abbreviations, and hashtags. Each microblog post is treated in isolation, without taking into account the wider available context. In particular, only tweet text is typically processed, despite the fact that the complete tweet JSON object also includes author profile data (full name, optional location, profile text, and web page). Around 26% of all tweets also contain URLs [136], 16.6% hashtags, and 54.8% at least one user name mention.

Microblog named entity linking is a relatively new, under-explored task. Recent tweet-focused evaluations uncovered problems in using state-of-the-art NEL approaches in this genre [67, 134], largely due to the brevity of tweets (140 characters). There has been limited research on analysing Twitter hashtags and annotating them with DBpedia entries, to assist semantic search over microblog content, e.g. [137]. Approaches based on knowledge graphs have been proposed, in order to overcome the challenge of having a very limited context, with some success [138].

Shen et al. [139] use additional tweets from a user's timeline to find user-specific topics and use those to improve the disambiguation. Huang et al. [140] present an extension of graph-based disambiguation which introduces "Meta Paths" that represent context from other tweets through shared hash tags, authors, or mentions. Gattani et al. [141] make use of URL expansion and use context derived from tweets by the same author and containing the same hashtag, but do not

[9] Unfortunately they do not evaluate the named-entity recognition accuracy of OpenCalais on their dataset.

evaluate the contribution of this context to end performance, and do not make use of hashtag definitions or user biographies.

More recently, [129] investigated the impact on NEL performance of using context expansion, user biography information, and hashtag definitions. In particular, in the case of hashtags, tweet content is enriched with hashtag definitions, which are retrieved automatically from the web. Similarly, tweets containing @mentions are enriched with the textual information from that Twitter profile. In the case of URLs, the corresponding web content is appended to the tweet. Disambiguation performance is measured both when such context expansion is performed *individually* (i.e., only hashtags, only URLs, etc.), as well as when all three types of contextual information are used *jointly*.

5.6 DISCUSSION

Our overview of Wikipedia- and LOD-based entity linking has demonstrated that the majority of research has concentrated on a small number of common, well understood entities, namely persons, locations, organizations, and sometimes products. Real challenges exist in widening this set toward new entity types, as this also tends to increase ambiguity and consequently lower the performance of NEL methods. Another important, but hitherto little studied problem, is optimizing NEL algorithms for social media posts, where context and textual content are very different, and harder, to address accurately.

The other key challenge is scaling up to languages other than English, where researchers also need new training and evaluation datasets, especially of social media content. While there are some approaches that address multiple languages (e.g., DBpedia Spotlight, YODIE), the bulk of NEL research is still carried out on English language datasets.

CHAPTER 6

Automated Ontology Development

6.1 INTRODUCTION

In this chapter, we will introduce the concept of automated ontology development, which comprises three related components: learning, population, and refinement. Ontology learning (or generation) denotes the task of creating an ontology from scratch, and mainly concerns the task of defining the concepts and generating relevant relations between them. Ontology population consists of adding instances to an existing ontology structure (as created by the ontology learning task, for instance). Ontology refinement involves adding, deleting, or changing new concepts, relations, and/or instances in an existing ontology. Ontology learning may also be used to denote all three tasks, in particular where the tasks of learning and population are performed via a single methodology. For all three components of ontology development, the starting point is typically a large corpus of unstructured text (which may be the entire web itself, or a set of domain-specific documents). We do not concern ourselves here with ontology development from structured text, since this does not typically involve the use of Natural Language Processing.

In the rest of this chapter, we will describe the task in more detail, explaining how it resembles and differs from that of semantic annotation, and giving examples of how it is useful. We will then describe some typical approaches, again building on the tools described in previous chapters. It should be noted that there are already several excellent books on ontology learning and population, written from various perspectives—see, e.g., [142–144]. In this chapter we therefore summarize only some key concepts from an NLP perspective.

6.2 BASIC PRINCIPLES

It is clear that ontologies are of paramount importance in Semantic Web applications. While there are already thousands of existing ontologies, ranging from small domain- and application-specific ontologies to vast all-encompassing ones such as DBpedia, nevertheless they are often insufficient or unsuitable for a particular task. Furthermore, new kinds of tools and applications may demand new kinds of ontologies: for example, the recent surge of interest in opinion mining from product reviews demands specific ontologies for recognizing aspects of products. If one wants to analyze reviews about cameras, one needs to know about all the various components of a camera and how they relate to each other—lens, battery type, dimensions, manufacturer, etc. Similarly, hotels have

aspects such as number of rooms, restaurant, bar, swimming pool, service, and so on. These are not strictly components of the hotel and so would not be necessarily represented in a typical "hotel ontology." We will look more at aspect-related opinion mining in Chapter 4.

The manual creation of ontologies is generally unfeasible except for very small toy domains or for very specific cases, and is both labor- and cost-intensive, as well as being subjective. On the other hand, automatic creation of ontologies is error-prone: at best, it is only as good as the data from which the ontology is generated, which is rarely complete, and is problematic in that extracting the correct relations between ontology elements is not an easy task, since this information is rarely explicit in the data. A compromise must be sought between maximum automation with minimum loss to performance, and subjectivity.

While domain-specific ontologies do exist, and in some fields are even quite comprehensive (the medical domain, for example, has enormous knowledge bases such as UMLS[1] and the Gene Ontology[2]), it is nevertheless unlikely that any pre-existing knowledge base will be entirely adequate for a Semantic Web application. Apart from containing potential errors, omissions, and redundancies, it may also be highly ambiguous. Furthermore, different kinds of applications in the same domain may require different kinds of ontologies; a generalized medical ontology might not be specific enough for working in the subdomain of eye pathology, for example.

Another problem is non-standardization of terminology. Even when terms are standardized, different variants may still be used in textual sources, such as *heart attack* and *myocardial infarction*. Also, many terms are ambiguous, not just across domains (e.g., mouse in the computing domain is different from mouse in the zoology domain), but even within them (usually due to underspecification, e.g., *leg* in medicine could refer to a human leg or a prosthetic leg). Furthermore, a text in a particular domain might still refer to an out-of-domain concept ambiguous with an in-domain concept (e.g., in a medical report *concussion caused by hitting her head on a table leg*). Methods of adapting ontologies to the task and domain need to be considered in order to realize their full potential in applications. Customization of lexical resources is thus a critical task, for which clustering and term recognition both play a significant role by structuring the knowledge required.

The essential elements and approaches comprising ontology development can be characterized in the form of an ontology learning layer cake (Figure 6.1), based on the idea of the famous Semantic Web layer cake [145]. Starting from the bottom of the cake and working upwards, the most fundamental tasks are term and synonym recognition, e.g., cities and countries might be terms. The next levels involve concepts, classes, and relationships (properties), e.g., cities belong to countries, some cities are capital cities, countries have capital cities. Finally at the top we have axioms such as disjointness (something cannot be both a river and a mountain). This is, of course, a rather simplified view of things, and has some limitations, as it is based on a lexical approach to

[1] http://www.nlm.nih.gov/research/umls/
[2] http://geneontology.org/

ontology acquisition [146]. However, this is the exact approach we take in this chapter, since we are concerned with NLP methods for ontology development, so it fits well.

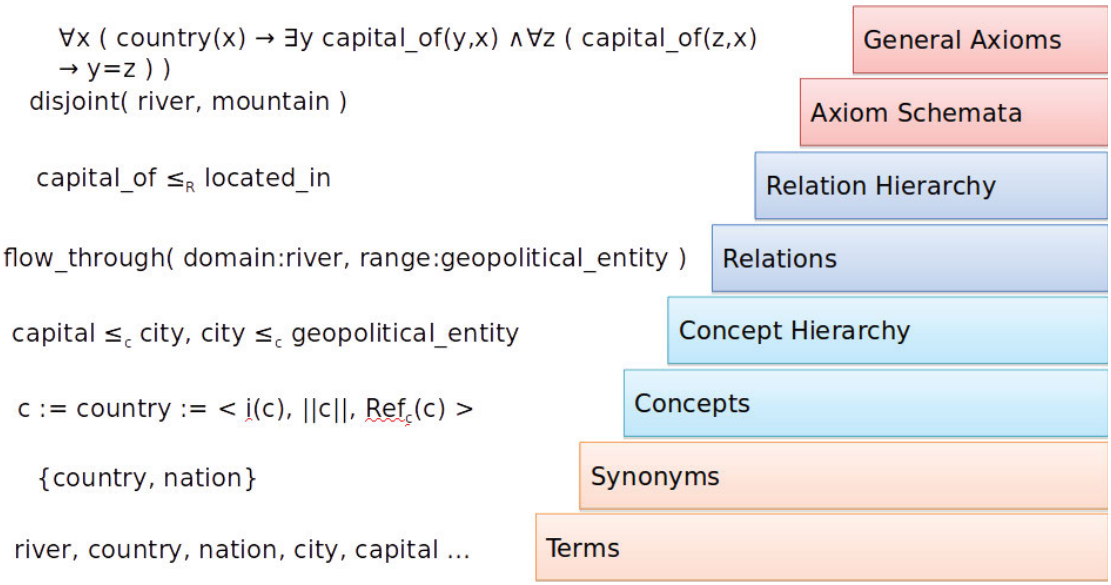


Figure 6.1: Ontology learning layer cake (reproduced from Cimiano, P.: *Ontology Learning and Population from Text: Algorithms, Evaluation and Application*, Springer-Verlag, New York, 2006).

6.3 TERM EXTRACTION

The identification of terms that are relevant to the domain is a vital first step in both the ontology population and generation tasks. This task is known as term extraction or term recognition, also abbreviated to ATR (Automatic Term Recognition). Automatic ontology population is generally performed by means of a kind of ontology-based information extraction (OBIE), as descrbed in Chapter 5. Whereas typically OBIE concerns identifying named entities and relating them to an ontology, for ontology population, it consists of identifying the key terms in the text and then relating them to concepts in the ontology (relation extraction). For ontology generation, terms are first found and then relations between them are extracted, which form the basis for the ontology itself.

The definition of "term" is fraught with controversy. In general, a term can be said to refer to a specific concept which is characteristic of a domain or sublanguage. Unlike named entities such as Person or Location, which are typically generic across all domains, a technical term such as *myocardial infarction* is only considered a relevant term when it occurs in a medical domain; if we were interested in sporting terms then it would probably not be considered a relevant term, even if

it occurred in a sports article. Terms, like named entities, are generally formed from noun phrases. In some contexts, and especially in existing ontologies, verbs may also be considered terms, but most corpus-based term recognition techniques do not consider these. Even the definition of a noun phrase can vary; as discussed in Chapter 2, some noun phrase chunkers may extract noun phrases that include prepositional phrases, while others may not.

Term recognition can be performed in a variety of ways. The main distinction we can make is between algorithms that only take the distributional properties of terms into account, such as frequency and tf/idf [147], and extraction techniques that use the contextual information associated with terms. However, many approaches combine the two types of knowledge. Linguistic techniques are typically used in the first instance to find candidate terms; these are then ranked in order of term likelihood. A cut-off point (threshold) can then be used to make an absolute decision between what is and is not considered a term, which is critical for most applications. Since the evaluation of term ranking and term recognition is quite a difficult and subjective task, where the ideal solution may also vary depending on the nature of the task, a number of term extraction frameworks have been developed, where different solutions or variations can all be tried and compared. TermRaider (described below) and JATE[3] are good examples of this.

6.3.1 APPROACHES USING DISTRIBUTIONAL KNOWLEDGE

These approaches typically use frequency-based methods, based around the tf/idf model. Tf/idf (term frequency/inverse document frequency) reflects how important a word is to a document in a collection. Since some words will appear very frequently in all domains, the tf/idf value adjusts for this; it increases proportionally to the number of times a word appears in the document, but is offset by the frequency of the word in the corpus. The principle behind its use in term extraction is that we would expect terms to occur more frequently in a corpus relevant to the domain than they would in a non-relevant corpus, whereas non-terms would appear evenly distributed in the two corpora, or even less frequently in the domain-specific corpus. For example, we would expect our medical term *myocardial infarction* to occur more frequently in a medical corpus than in a corpus of sports texts. Typically, therefore, we use the tf/idf model to compare a domain-specific corpus with a general corpus, rather than to compare a single document with a corpus.

There are many variations and improvements on the basic tf/idf model. **TermRaider**[4] is a GATE plugin for term extraction that produces term candidates from a corpus, together with a statistically derived termhood score. Like most term extraction methods, it first identifies candidate terms based on linguistic principles, and then filters and ranks them. The initial candidate term identification in TermRaider relies on linguistic pre-processing (tokenization, sentence splitting, POS-tagging, lemmatization, and NP chunking), normally performed in GATE by ANNIE or TwitIE (though other tools could be used instead). Candidate terms are then extracted from the text by means of grammar rules that further restrict the noun phrases, such as excluding cer-

[3] http://code.google.com/p/jatetoolkit/
[4] https://gate.ac.uk/projects/arcomem/TermRaider.html

tain frequent stopwords. Finally, tf/idf is applied to the corpus, producing a score that indicates the salience of each term candidate for each document. All term candidates with a tf/idf score higher than a manually determined threshold (set as a runtime parameter) are then selected as terms.

Two main variations to tf/idf are additionally implemented in TermRaider. Augmented tf/idf incorporates information about term hyponyms. The principle behind this is that terms which have hyponyms present are more likely to be valid terms. The score represents a term candidate's maximum value of local augmented tf/idf, which for each occurrence of a term candidate is its tf/idf score plus the tf/idf scores of all hyponyms of the term candidate found surrounding that occurrence. A second variation is the Kyoto domain relevance score [148], which incorporates also the number of distinct hyponyms of a term candidate found in the entire corpus. Again, this stems from the principle that terms which have more hyponyms are more likely to be valid.

The **NC-value method** [149] uses similar methodology, and is used as the foundation for tools such as TerMine.[5] The method is based on tf/idf over candidate terms extracted in a similar fashion to TermRaider, but is extended by adding information about frequency of co-occurrence with context words. The TRUCKS approach [150] extended this further by identifying salient parts of the context surrounding a term, and measuring their strength of association with relevant candidate terms.

6.3.2 APPROACHES USING CONTEXTUAL KNOWLEDGE

Approaches using contextual knowledge take into account the words in the context of the candidate terms, in order to help rank them. Different kinds of knowledge can be used, either individually or combined together. Sometimes this information is used to exclude certain terms being candidates. But in most cases, it is used in the form of weights to help with the term ranking.

Terminological knowledge concerns the terminological status of context words. A context word which is also a term is likely to be a better indicator of a term than one which is not also a term itself. This is based on the premise that terms tend to occur together. For example, in TRUCKS [150] a weight is produced for each candidate term based on its total frequency of occurrence with other terms in its context.

Syntactic knowledge is based on *boundary words*, i.e., the words immediately before and after a candidate term. The *barrier word* approach [151, 152] requires a term to be considered only in the presence of certain syntactic categories following or preceding the candidate term. Other systems allocate a weight to each syntactic category of immediate context words based on co-occurrence frequency analysis. For example, a verb occurring immediately before a candidate term is statistically a much better indicator of a true term than an adjective is. Each candidate term is then assigned a syntactic weight, calculated by summing the category weights for all the context boundary words occurring with it.

[5] http://www.nactem.ac.uk/software/termine/

Semantic knowledge is based on the idea of incorporating semantic information about the context. This relies on the principle that words in the context which have a high degree of similarity to the candidate term are more likely to be relevant. Similarity can be calculated in a variety of ways; see Section 6.4 for some examples.

6.4 RELATION EXTRACTION

Once the relevant terms are extracted, relations between them must then be generated. Recently, many relation extraction approaches have been proposed that focus on the task of ontology development (learning, extension, population). These approaches aim to learn taxonomic relations between concepts, instead of lexical items. The kind of relation extraction required for ontology development differs slightly from the relation extraction task covered in Chapter 4, where the focus was on non-taxonomic relations, such as authors of books, while here we are concerned with taxonomic relations such as hyponymy (e.g., apple is a kind of fruit).

6.4.1 CLUSTERING METHODS

Clustering methods aim to organize terms in a hierarchy that can be transformed directly into an ontology, using some kind of distance measure to create or merge clusters of terms. This measures how similar one term is to another or to a set; for example, it may be used to compute the most typical instances of a concept as the ones closest to the centroid (the hypothetical "average" instance of a set). For this methodology, one has to first select an appropriate semantic distance measure and clustering algorithm. A good overview of methodologies can be found in [153]. Examples of clustering methods include vector space [154], associative networks [155], or set-theoretic approaches [156].

6.4.2 SEMANTIC RELATIONS

Ontology-based semantic relations are based on the idea that words which are semantically related will occur closer together in an ontology than those which are less strongly related. This can be helpful for positioning terms correctly in an ontology and for term disambiguation. There are a number of different metrics for measuring the relatedness, which can be categorized into three main types: frequency-based methods, thesaurus-based methods, and example-based methods. A longer description of these can be found in [157]; here, we summarize some of the main ones.

Frequency-based methods are frequently used for information retrieval, and are based on statistical properties of words in corpora. They include the weighted Jaccard measure [158], simple co-occurrence techniques (e.g., co-occurrence frequency, mutual information, association ratio) and vector-based techniques, which measure the similarity between words using, e.g., the dot product, cosine function, or Euclidean distance between two vectors which represent the contexts of words presented in their definitions. The vector for a context is calculated by adding the co-occurrence information vectors of the words in the definition, found using simple co-occurrence.

Thesaurus-based methods rely on a hierarchically structured thesaurus or ontology, where the nodes are generally assigned weights based on frequency or probability. Common techniques for computing similarity involve conceptual distance, semantic distance, and variations. Conceptual distance [159] is the length of the shortest path connecting the two instances in the hierarchy. Semantic distance [160] is measured by the information content of the Most Specific Common Abstraction (MSCA)—the most specific class in the hierarchy which subsumes both classes. The information content is calculated by estimating the probability of occurrence of the class in a corpus. The depth of the node in the hierarchy may also be taken into account, since nodes deeper in the hierarchy tend to be more similar.

Example-based methods are frequently used in machine translation, and aim to select the most similar experience for a given problem. These methods typically combine hierarchical structures with a set of examples taken from a corpus. They include weighted feature graphs [161], word closeness [162], best-match algorithm [163], and example-based weighted semantic distance [164].

Corpus-based semantic methods are most frequently used for the task of relation extraction for ontology creation. These are based on the idea that words which are semantically related will occur together in some text. Furthermore, such words will co-occur more frequently than non-related (or less strongly related) words. For example, apples are more closely related to oranges than they are to shoes, since both are types of fruit and shoes are not. We would therefore expect the word *apples* to occur in the same text more frequently with the word *oranges* than with the word *shoes*. By comparing the relative frequencies of the two co-occurrences, we can determine that apples and oranges are more strongly related than apples and shoes. Corpus-based approaches have the advantage that they are self-contained and do not require any external sources, which means that they are very suitable for specialized domains, and tend to ensure that the information is appropriate to that domain. However, using information from such a corpus may result in statistical skewing, and there may be gaps in the coverage of the corpus. Table 6.1 shows some advantages and disadvantages of a corpus-based approach [157].

Table 6.1: Advantages and disadvantages of a corpus-based approach to semantic relation extraction

Advantages	Disadvantages
Real examples of language in use	Techniques may be unreliable
Information tailored to domain	Coverage may be inadequate
Statistical information available	Large corpus needed
	Gaps in coverage
	Information may be ambiguous

6.4.3 LEXICO-SYNTACTIC PATTERNS

The **Hearst patterns** are a set of lexico-syntactic patterns that indicate hyponymic relations [165], and have been widely used for finding relations between terms and ontology creation. They are used in both Text2Onto and SPRAT (see below). Typically they achieve a very high level of precision, but quite low recall: in other words, they are very accurate but only cover a small subset of the possible patterns for finding hyponyms and hypernyms. For this reason, they are usually combined with other kinds of patterns.

The Hearst patterns can be described by the following rules, where NP stands for a Noun Phrase and the regular expression symbols have their usual meanings[6]:

1. `such NP as (NP,)* (or|and) NP`
   Example: …works by such *authors* as *Herrick*, *Goldsmith*, and *Shakespeare*.

2. `NP (,NP)* (,)? (or|and) (other|another) NP`
   Example: *Bruises*, *wounds*, or other *injuries*…

3. `NP (,)? (including|especially) (NP,)* (or|and) NP`
   Example: All *common-law countries*, including *Canada* and *England*…

There are also cases where these do not work. For example one can extract *Italians* as a hyponym of *Europeans* from the phrase *Europeans, especially Italians*, but one should not extract *Democrats* as a hyponym of *US Presidents* from the phrase *US presidents, especially Democrats*.

As a follow-up to this, Berland and Charniak [166] also developed some patterns to deal with meronymy, e.g., to extract that *speedometer* is a part of a *car*. Two example patterns are given below:

1. `NN's NN`
   … building's basement …

2. `NN of DET (JJ|NN)* NN`
   … basement of a building…

The SPRAT system developed as a GATE plugin and described in Section 6.6 also includes further patterns.

6.4.4 STATISTICAL TECHNIQUES

Whereas lexico-syntactic patterns typically produce paradigmatic relations (such as hyponymy) between terms, syntagmatic relations (such as collocations) can be identified using statistical techniques. Pointwise Mutual Information [167] is a well-known technique that measures the mutual dependence of the two variables. It is commonly used in corpus linguistics as a significance function for computing collocations [168]. For relation finding, we can use it to measure how strongly related two terms are within the same document or corpus [169].

[6] `()` for grouping; `|` for disjunction; `*`, `+`, and `?` for iteration.

6.5 ENRICHING ONTOLOGIES

Ontologies are typically not static but constantly evolving. First, new concepts (classes) may be added, deleted, or moved. When such changes are made, they need to be reflected also in the instances and relations (properties). Second, new instances may need to be added, deleted, or moved in order to make the ontology more complete or rectify existing problems. For structural changes to the ontology, principled mechanisms need to be put in place for dealing with this, so that valid information is not lost (for example, moving instances up the hierarchy if the concept to which they belong is deleted). However, such changes typically do not require NLP technology. We shall restrict ourselves here, therefore, to discussing methods for enriching ontologies by adding new instances and relations.

One of the main reasons why ontologies are often not complete is due to the problem of sparse data. When constructing an ontology from a corpus, the information contained in the corpus will never be complete—one cannot expect any collection of texts to contain all terms in a domain or to provide obvious lexico-syntactic patterns for collecting relationships between terms. This kind of **lexical acquisition bottleneck** is ubiquitous in language processing tasks, and is frequently resolved using clustering techniqeus. For ontology enrichment, **semantic frames** can be used. This idea dates back to the late 1960s with Harris' **distributional hypothesis** [167] (namely, words that appear in the same context tend to have similar meanings), and work in the 1970s [170, 171] which focused on determining sets of sublanguage-specific word classes using syntactic patterns from domain-specific text. In particular, research in this area has been used in specific domains such as medicine, where a relatively small number of syntactic structures is often found, for example in patient reports. Here the structures are also quite simple, with short and relatively unambiguous sentences typically found: this makes syntactic pattern matching much easier. The idea is that by examining sets of lexical items found in specific syntactic environments, semantic word classes (clusters) can be established. For example, in the field of clinical reporting, by collecting instances of the lexical items found as objects of the verb *develop* together with the subject *patient*, Hirschman et al. [172] developed a class *sign or symptom*, consisting of lexical items such as *mild cold, fever, slight cough, etc.* An example of what they called an **information format** is shown in Table 6.2.

Table 6.2: Information format for the class *sign or symptom*

Subject	Verb	Object
patient	develop	mild cold
patient	develop	fever
patient	develop	slight cough
patient	develop	headache

Since then, much work on semantic knoweldge acquisition has followed a similar approach. For example, Rocha [173] pioneered the use of case frames for what he calls Event Definition models (very similar to the frames used in Information Extraction for defining events, and used in the MUC evaluations). An example of such a case frame is shown in Table 6.3.

Table 6.3: Example of Rocha's case frame

Slot	Filler
Procedure:	Chest x-ray
Link:	Shows

6.6 ONTOLOGY DEVELOPMENT TOOLS

In this section we describe some typical tools for automatic ontology creation and enrichment that use NLP techniques.

6.6.1 TEXT2ONTO

Text2Onto [174] was one of the first and most well-known tools for automatic ontology development. It performs synonym extraction on the basis of patterns, combining machine learning approaches with basic linguistic processing such as tokenization, lemmatization, and shallow parsing. Since it is based on the GATE framework, it offers flexibility in the choice of algorithms to be applied.

6.6.2 SPRAT

SPRAT (Semantic Pattern Recognition and Annotation Tool) [175] is an example of an ontology development system for the fisheries domain, though the methodology could be applied equally to other domains. It is capable of either creating an ontology from scratch, or modifying an existing ontology, and was based on the principle of lexico-syntactic patterns. Compared with Text2Onto, it has more lexico-syntactic patterns, but does not use statistical clustering and parsing for relation extraction. This means that it generates less data, but is potentially more accurate.

6.6.3 FRED

FRED [176] is an online tool for converting text into linked-data-ready ontologies, using Deep Parsing. It combines Discourse Representation Theory (DRT), linguistic frame semantics, and Ontology Design Patterns (ODP). It is based on Boxer [177], a linguistic tool that generates formal semantic representations of text, based on event semantics. While other tools typically focus mainly on helping users to identify the key terms to be added to the ontology, FRED differs in that it aims to present ontologies and linked data ready to use.

6.6.4 SEMI-AUTOMATIC ONTOLOGY CREATION

In the Ontology Engineering domain, *Ontology Design Patterns* [178] have emerged as a way of assisting ontology developers to model OWL ontologies in a top-down fashion. Ontology Design Patterns (ODPs) are essentally sets of conceptual patterns designed to help a user construct or refine a domain ontology. Tools for supporting the semi-automatic reuse of these have also been developed [179]. These tools take as input text relevant to the domain, and obtain as output a set of ODPs for solving the initial ontological needs. The correspondence between ODPs and NL formulations is performed through Lexico-Syntactic Patterns.

So far in this chapter we have focused on describing methods for bottom-up ontology creation from corpora. An alternative to ODPs for the non-expert user is the use of syntaxes or *controlled languages* specifically designed to make ontology languages more readable and understandable by others. Examples include Attempto Controlled English (ACE) [180], Rabbit [181], Sydney OWL Syntax [182], and CLOnE (Controlled Language for Ontology Editing) [183]. Some example sentences from these are shown in Table 6.4. The main idea underlying these controlled languages is to allow non-expert users to express their modeling needs following certain syntactic rules. Here one has to know about the terms and relations in advance that one wishes to model: the difficulty lies in converting these into the correct ontological form. For example, with CLoNE a domain expert can use a Natural Language Interface to transform their text into a simple ontology—as text is typed in the interface, it is automatically converted (using NLP processing) into classes and relations in the ontology. The catch is that the user must type their text in a very prescribed way, according to the controlled language used.

Table 6.4: Examples of controlled languages for ontology generation

Language	Example Sentence
ACE	Every river-stretch has-part at-most 2 confluences.
Rabbit	Every Bourne is a kind of stream.
Sydney Syntax	The classes petrol station and gas states are equivalent.
CLoNE	Projects have string names

6.7 SUMMARY

In this chapter, we have discussed the task of automated ontology development and its main components: ontology learning, population, and refinement. While there are many approaches to automated ontology development, we have focused here on the methods which are based on NLP, and which build on the NLP components we have described in previous chapters: preprocessing, named entity recognition, and relation extraction. We have focused here particularly on term extraction, since this is the key component for ontology development, and on methods for assigning hierarchical structure to these terms. Relation extraction is another key component:

since this has already been described in detail in Chapter 6.4, we have simply summarized here the main types of relations useful for ontology generation, focusing particularly on lexico-syntactic patterns. We have concluded by pointing to varous related elements of ontology development, such as semi-automatic ontology creation, and given some examples of tools typically used in this field.

CHAPTER 7

Sentiment Analysis

7.1 INTRODUCTION

An important aspect of understanding text is the detection and classification of opinion, sentiment, and emotion. This can range from the classification of user reviews about products (did this reviewer like it or not? Which aspects of the product did they like/dislike?) to understanding sentiments and emotions in tweets, tracking opinions over time, detecting opinion influencers and leaders, and creating summaries based on opinions. This chapter describes the key components of a typical sentiment analysis tool, introducing a variety of different possible methods, and gives examples of real applications in various domains, showing how sentiment analysis can also be slotted into wider applications for social media analysis.

Sentiment analysis (from text) is about analysing text in order to understand people's opinions. We do not deal here with sentiment analysis from other forms of media such as images and videos, as they do not fall within the scope of NLP. At the simplest level, this means understanding whether a person is talking in a positive or negative way about something, but of course opinions can be much more subtle: they may express all kinds of different emotions and strengths of emotion (do they like something a little or a lot, are they fearful, shocked, angry, relieved, pleasantly surprised, etc?). They may also express sentiment about particular aspects of a product or event, leading overall to some contradiction (liking some elements but disliking others).

Sentiment analysis tools can be enormously useful in almost every aspect of industry. Product reviews are a typical example: someone who wants to buy a camera might look for comments and reviews online, while someone who has bought a camera may comment on it and write about their experience; camera manufacturers can get feedback from their customers which may help them improve their products or service and/or adjust their marketing strategies. Trying to analyze these reviews and opinions manually is often not feasible, especially for large companies who may get millions of reviews about each product. While official review sites often have star rating systems, the most useful information is often to be found in the free text, so simply aggregating the numerical scores is not sufficient to get the full picture. Furthermore, comments on social media such as Twitter often need to be dealt with urgently: while fully automated systems for responding to these should certainly not be relied on, opinion mining tools can nevertheless help flag critical issues or show trends. Question answering systems can also benefit hugely from opinion mining components, in order to deal with questions such as "Which is the best Japanese restaurant in London?" and so on. One could even attempt to answer queries that require more complex understanding, such as "Which is the camera with the best battery life?"

While customer reviews and comments are an obvious target for opinion mining tools, and much research has focused on these (partly because of the obvious need, but also because it is easy to create training and test sets of large volumes of data using the rating systems as a gold standard), there are many other uses for opinion mining tools. Understanding political and social sentiment about governments, events, elections, and so on is another important task. Traditionally, this analysis is performed by polls (such as YouGov in the UK), but these are time-consuming and expensive to carry out. Predictive analysis, in particular, is a huge market, from understanding which films will win Oscars and other awards (and thereby leading to increased revenue), to investigating how public mood can influence the stock market and making predictions based on social media chatter. Social analytics can also be used to draw important inferences; not only via explicit connections (people who like travel might want to buy travel products) but also through implicit associations (people who buy Nike products also tend to buy Apple products, for example).

Opinion mining tools take a piece of text as input, and output information such as whether the piece of text is opinionated, what kind of opinion is expressed (positive, negative, etc.), the degree of strength of the opinion, and possibly other information such as what the opinion is about, who is holding the opinion, and some kind of opinion summary over multiple sentences or statements. These subtasks will be discussed in more detail in Section 7.3.

The task of opinion mining might at first seem staightforward: a naive system would simply look for positive and negative words (like, hate, good, bad, etc.) and produce a resulting opinion accordingly. In practice, the task is much more complex than this, even for basic polarity detection (knowing whether a statement is positive or negative). This is because, as we have already seen earlier in the book, natural language is incredibly complex and ambiguous. This is particularly the case for social media, where much opinion mining work is focused; people use unusual terms to describe their feelings, they qualify statements with negatives, they do not use correct grammar or spelling, they use conditional statements and phrase sentiment as questions, they can be sarcastic or assume that the reader has additional world knowledge to decipher the meaning without explicit reference (for example, references to Voldemort or Hitler are generally negative). This means that complex linguistic analysis is often required to decipher meaning correctly, as will be seen in Sections 7.2 and 7.3.

We should clarify finally in this section a point about terminology. Theoretically, opinions and sentiment are two different things, and thus by extension opinion mining and sentiment analysis. Sentiments typically express a particular polarity (positive, negative, or neutral). For example "I think your dress is pretty" is a positive sentiment expressed by me. Opinions could express something rather more generic, e.g., "I think that it will rain tomorrow" is an opinion expressed by me about the weather, but it does not express any particular positive or negative sentiment. However, "opinion" can also be used to mean a positive or negative sentiment; in the first example, I am expressing a positive opinion about your dress.

In the early days of opinion-mining research, the term "opinion mining" was thus used for something quite encompassing, while sentiment analysis was used specifically for the task of

polarity detection. However, in recent years the two terms have come to be used interchangeably, in particular where sub-tasks and side-tasks have been formed (e.g., detecting whether something is opinionated or not; detecting emotions; detecting the reliability of opinions, and so on—see the following sections). In this chapter, we use the term "opinion mining" to cover the tasks of detecting whether something expresses sentiment, what the polarity of the sentiment is, how strong that sentiment is, who is holding the opinion, what the opinion is about, and what emotions are being expressed. We do not attempt to distinguish opinions as a non-factual statement with neutral sentiment (as in the weather example) from a factual statement (e.g., "it is raining").

7.2 ISSUES IN OPINION MINING

A naive approach to sentiment analysis would simply use a lexicon of opinionated words (good, bad, happy, sad, etc.) and aggregate these over the text to be analysed (e.g., sentence, tweet, or document) in order to decide an overall polarity. Indeed, many baseline approaches use precisely this method, and get reasonable scores. However, even if we take into account issues such as negation ("good" vs. "not good"), there are many subtle nuances which impede this kind of simplistic analysis. For example, conditional sentences can change the meaning substantially ("If Scotland lost the match, it would be a tragedy."). An opinion may also be very different depending on who is holding it and what the opinion is actually about. "It's great that Scotland lost the match." implies positive sentiment by the author about the match outcome, but some kind of negative sentiment about Scotland. On the other hand, we would not expect the Scotland team or Scottish supporters to be happy about the outcome. Even swear words and negative terminology may, in the right context, be used positively: British people, in particular, often refer to their friends with quite negative terminology without being negative in any way about them (for example, calling someone a mucker is a term of endearment, but literally means someone who cleans away waste products).

One must also be careful about distinguishing between an opinion about a person or thing, and an event involving that person or thing. For example, expressing sadness or shock about someone's death does not indicate dislike of that person, even though the tone of the message overall is negative, but many sentiment analysis tools will get this wrong because they do not distinguish between the two things.

Sarcasm can also be difficult to deal with, but is a common feature on social media. First, the system must recognize when sarcasm is present, which is not always an easy task, even for a human with more contextual knowledge. Second, the system must understand how the sarcasm impacts on the polarity of the opinion: it may reverse the expected polarity of the whole phrase or sentence, just one small part of it, or even multiple sentences [184]. While the ability to perform sarcasm detection may appear a trivial aim, its implications are critical: in 2014 the U.S. Secret Service announced plans to purchase software to watch users of social networks in real time, which would include specifically the ability to detect sarcasm.[1]

[1] http://www.bbc.co.uk/news/technology-27711109

7.3 OPINION-MINING SUBTASKS

It is clear from the above discussion that there are a number of issues in opinion mining which need to be addressed by a tool to perform this task automatically. These can be broken down into a set of optional subtasks which tools may deploy. We give a brief description of these and typical methods which may be used.

7.3.1 POLARITY RECOGNITION

Polarity recognition is the task of deciding whether a statement is positive, negative, or neutral. Sometimes this forms part of the opinion detection task (is this statement opinionated?), where neutral indicates that the statement is not opinionated, and the other two categories indicate that it is. Other systems first classify statements into opinionated or non-opinionated, and then further subclassify the opinionated statements in a separate subtask. These may also be evaluated as a single task or as two separate ones. Some systems make a separate distinction between neutral and no sentiment, mainly when the system is used on longer documents. In this case, neutral is typically the case where there is an equal number of positive and negative elements: for example on a review site a score of 3/5 stars could be seen as equally positive and negative, where there are some good and bad points about the product. Alternatively, neutral sentiment is sometimes used to describe the case where the author clearly is expressing some sentiment but it is unclear what exactly that sentiment is. In these cases, no sentiment is different from neutral sentiment. However, both manual annotators and automated tools have great difficulty in distinguishing between the two cases, especially in shorter documents, and so they are often lumped together with no distinction.

7.3.2 OPINION TARGET DETECTION

It is often not enough to simply know whether an opinion is positive or negative, unless we also know what exactly it is positive or negative about. As discussed earlier, liking a person is very different from liking the fact that they are dead. Similarly, liking a particular aspect of a person or thing (a person's hair, the color of a car, etc.) can be very different from liking the person or thing as a whole. Target detection concerns the recognition of what the opinion is about, and has two main approaches. The first, top-down, approach is where the target is pre-defined and is typically an aspect or property of an object which is defined in an ontology or other classification system (for example, hotels have aspects such as rooms, catering, location; cameras have price, size, battery life, and so on.) Aspect-based opinion mining with ontologies is described further in Section 7.6. The second approach is a bottom-up one where the possible targets are not known in advance but are derived automatically from the text. Usually these will consist of terms, entities, or events that are identified in a previous stage of the NLP pipeline. Correctly connecting the opinion to the right entity or event is still, however, challenging: simply using distance-based approaches is largely insufficient and, ideally, a linguistically motivated approach should be taken for best results

(i.e., using parsing or at least chunking to ensure the correct relationship is maintained between opinionated words and target). However, this is still not easy, partly due to parsing inaccuracies (especially on social media text) and partly due to the complexity of constructions. Examples of entity-based approaches are found in [185] and [186]. Examples of approaches with pre-defined targets, also known as stance detection, are found in [187] and [188].

7.3.3 OPINION HOLDER DETECTION

In the same vein as opinion target detection, opinion holder detection is about recognizing who is holding the opinion mentioned. In many cases, this may be straightforward, for example in customer reviews it is usually the opinion of the person writing the review, athough in some cases, it may not be as simple ("My girlfriend liked the book, but I found it quite boring"). In cases where the author of the text is not the opinion holder, it is often a case of reported speech (used in a loose sense to refer to verbs of thinking, feeling, etc.). These kinds of structures can be recognized using a good quality linguistic analysis which will recognize names or types of possible opinion holders (typically people or organizations), semantic categories of verbs (think, feel, say, etc.) and syntactic patterns of a form such as $holder - opinion_verb - opinion$. The other case is as in the example above ("my girlfriend liked the book") where the subject of the opinionated verb needs to be recognized as the opinion holder. In tweets, opinion holders may also be the author of an original tweet which has been retweeted; here, care must be taken to establish whether one wants to recognize the original author or the retweeting author, or both, as the holder of the opinion. Note that the latter is somewhat controversial, since someone may retweet an opinion without necessarily agreeing with the original opinion, especially where one wants to highlight a controversial statement. As with opinion target detection, entity recognition is a useful first step for author detection, although also identifying nominal phrases pertaining to people and organizations, such as "my girlfriend," might be necessary.

7.3.4 SENTIMENT AGGREGATION

Sentiment can be identified at various levels: typically either at the sentence/phrase level or at the document/post level. Tweets usually comprise a single sentence and are thus treated as the former category, but sometimes comprise more than one. Opinion is therefore usually identified at the tweet level but using sentence-level methodologies, however, since only a single opinion is usually expressed per tweet. Sentiment analysis performed on a longer article or post (such as a movie review) typically starts with the extraction of opinions at the sentence level, working on a per-sentence or per-phrase basis and breaking down the review or article into a number of potentially different opinions about different facets of the opinion target (for example "The food was delicious, but the service was very slow."). This idea of aspect-based opinion mining is discussed further in Section 7.6, and is typical for the analysis of product rating sites.

There are two main ways of aggregating sentiment. The first approach, which is the most common, involves simply combining all the positive and negative scores from each sentence or

phrase and coming up with a single score as the total, which with any luck will correspond to the star rating, if one is given. Indeed, these star ratings are frequently used as training data for such systems, although this can be problematic since they do not always correspond (one might give a 4-star rating and then use the free text only to explain the negative points). For documents such as articles and blogs, or collections of comments, there is not always a straightforward relation between the aggregated positive and negative points: some theories hold that a neutral sentiment actually holds slightly more positive value than no sentiment expressed at all, and so these may be weighted accordingly. Similarly, negative sentiment often tends to outweigh positive sentiment (people are more likely to post when they are unhappy about something). A second, less common, way of getting a single score for opinion over a longer document is the collect-as-you-go method, where the document is traversed a word at a time and the score is updated. This is known as collective (rather than aggregated) analysis [189].

7.3.5 FURTHER LINGUISTIC SUBCOMPONENTS

In order to deal with some of the remaining issues mentioned earlier, an opinion-mining tool may make use of a number of additional linguistic subcomponents. Parsing, or at the least chunking, is useful in order to break sentences into smaller parts, so that correct correlations can be made between constituents such as opinions, targets, and holders. The simplest method of doing this is to break chunks according to punctuation and co-ordinating words, though this is not foolproof by any means. Parsing will give better results as it enables proper dependencies to be extracted (see Chapter 2), but is often problematic in terms of performance on social media texts and particularly tweets, due to the lack of grammaticality of the text.

It is useful to be able to recognize structures, such as questions and conditional phrases, since these can impact the opinionated text substantially. While questions can convey (usually implicit) sentiment, this is fairly unusual: asking "Do you think this dress is pretty?" normally does not convey a positive or negative sentiment on the part of the questioner. Similarly, "this dress would be pretty if it were in blue" and "If I had wanted a cheap dress, I would have bought a different one" both express complex sentiment, so special care must be taken. In fact, one can go further and identify specific rules concerning sentiment based on the type of conditional statement: these are implemented, for example, in the GATE tools for sentiment analysis [190], where adding such subcomponents is quite straightforward.

Swear words are a particular case where care must be taken with what would superficially seem to be simple negative expressions. Swear words are typically included in negative sentiment lexicons, but people do not always use them in a negative way. In fact, they are generally used as a kind of sentiment enhancer, especially when they occur as modifiers of positive or negative adjectives or nouns (for example, "bloody awful" vs. "bloody good").

As mentioned previously, sarcasm detection is another area where care must be taken. Traditionally, opinion systems have ignored sarcasm and irony, since they are hard to detect automatically, but recently they have been the object of increasing research [184, 191]. A typical approach

is to train a classifier on tweets containing hashtags such as #sarcasm and #sarcastic, and those without such hashtags [192]. Reasonable success has been achieved with such methods in terms of detecting whether a tweet is sarcastic or not, but very little research has investigated the problem of how sarcasm impacts polarity itself, as this is not straightforward (see [184] for a discussion of this problem).

7.4 EMOTION DETECTION

Opinion mining tools for real-world tasks are increasingly moving from the standard kind of positive/negative sentiment detection scheme to an emotion-based approach, which classifies opinionated texts according to the emotions expressed, e.g., [193]. The main reason for this is that it is more useful for practical purposes: for instance, a company would generally prefer to know specifically if people are fearful about a product or angry about it, rather than just negative about it. Another strand of research has investigated the correlation between emotion (especially fear) and changes in stock market prices [194]. Emotions can have fine-grained polarity values expressed as concepts from well-defined ontologies.

Defining a complete and clear set of emotions is difficult, however. There have been a number of attempts to define standards (see, e.g., [195] and `http://www.emotion-research.net`), but there is still no consensus on a basic set of emotions. One of the most commonly cited representations is Plutchik's wheel of emotions, illustrated in Figure 7.1. This attempts to show how different emotions are related, but is perhaps too complex for emotion detection representation. The representation depicts eight primary bipolar emotions shown in the second innermost circle: joy vs. sadness; anger vs. fear; trust vs. disgust; and surprise vs. anticipation. The idea then is that, as with colors, primary emotions can be expressed at different intensities and can be amalgamated to form other emotions. For example, combining anticipation and joy gives you optimism, the opposite of which is disapproval. It is the opposites in particular which are the most troubling; for example one would expect the opposite of optimism to be pessimism. Similary, the wheel classifies the opposite of the basic concept of fear as anger, and trust as disgust. Even just taking the categories as a starting point without considering their interaction, there are a number of expected missing categories, but the basic eight emotions are nevertheless frequently used for automatic classification purposes.

The Parrott tree-structured list of emotions [196], first described by [197], uses Plutchik's basic categories but extends them differently. It uses three levels, the first two of which are shown in Table 7.1.

A third representation, EARL (Emotion Annotation Representation Language), was developed specifically for emotion annotation by the Human-Machine Interaction Network on Emotion (HUMAINE[2]), and classifies 48 emotions, shown in Tables 7.2 and 7.3.

An important point to consider is that, unlike generic opinion polarities (positive/negative), emotion opposites are not necessarily the same as emotion negatives. For example, even though

[2] `http://www.emotion-research.net`

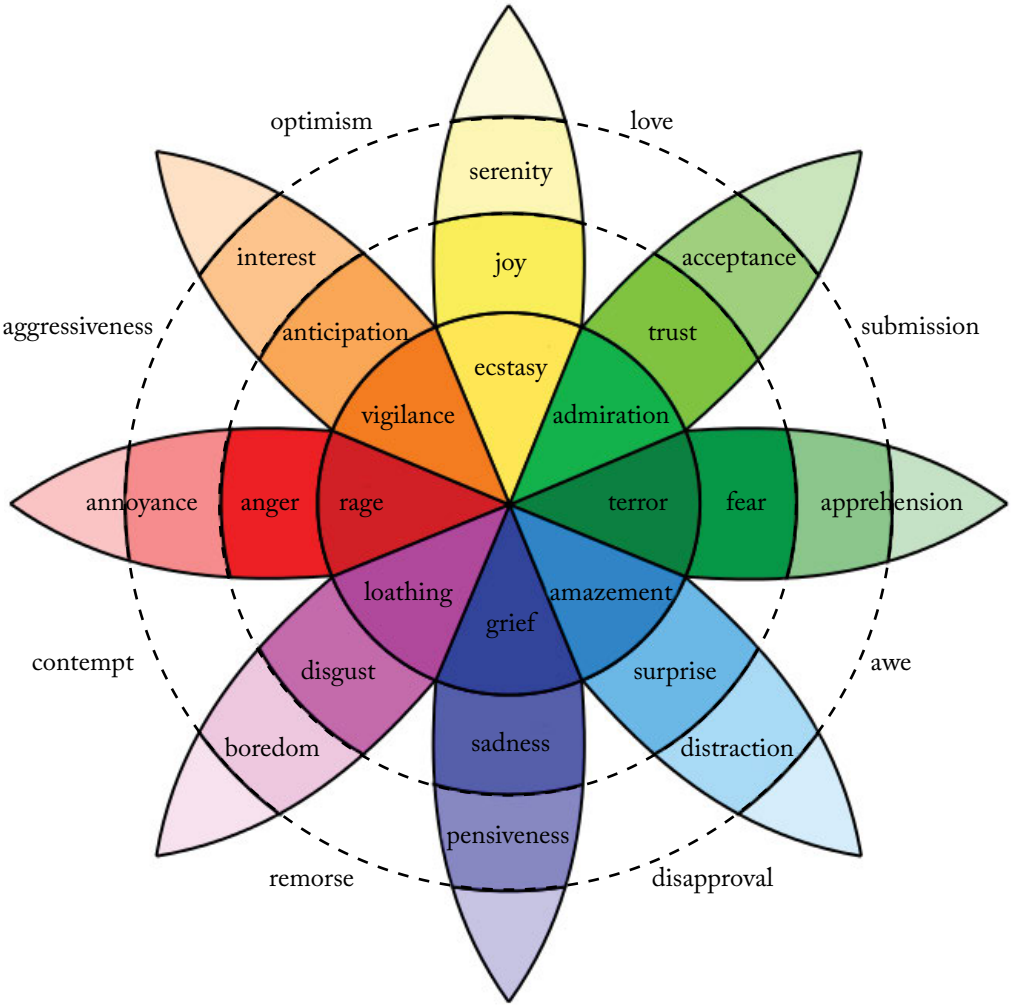


Figure 7.1: Plutchik's wheel of emotions (by Machine Elf 1735. Licensed under Public Domain via Commons).

happy and sad are typically considered to be opposite emotions, and while "I am not happy" could generally be rephrased as "I am sad," on the other hand "I am not sad" does not have the same meaning as "I am happy." Actually this can be generalized further: negated positive emotions are typically negative, but negated negative emotions can often be neutral rather than positive. This means that the typical technique of polarity flipping when negatives are encountered is not necessarily a good solution where emotion detection is concerned. This does not really seem to have been addressed in the literature.

Table 7.1: Parrott's emotion classification

Primary Emotion	Secondary Emotion
Love	Affection Lust/Sexual Desire Longing
Joy	Cheerfulness Zest Contentment Pride Optimism Enthrallment Relief
Surprise	Surprise
Anger	Irritability Exasperation Rage Dislike Disgust Envy Torment
Sadness	Suffering Depression Disappointment Shame Neglect Sympathy
Fear	Horror Nervousness

7.5 METHODS FOR OPINION MINING

While opinion mining is a relatively new field of research, there has nevertheless been much research over the last decade (and beyond) into techniques for identifying and classifying opinions. A wide-ranging and detailed review of traditional automatic sentiment detection techniques is presented in [198], including many sub-components. In general, techniques can be roughly divided into lexicon-based methods and machine-learning methods. Lexicon-based methods rely

Table 7.2: EARL representation of negative emotions

Forceful	Anger Annoyance Contempt Disgust Irritation	**Passive**	Boredom Despair Disappointment Hurt Sadness	**Negative Thoughts**	Doubt Envy Frustration Guilt Shame
Not in Control	Anxiety Embarrassment Fear Helplessness Powerlessness Worry	**Agitation**	Stress Shock Tension		

Table 7.3: EARL representation of positive emotions

Lively	Amusement Delight Elation Excitement Happiness Joy Pleasure	**Caring**	Affection Empathy Friendliness Love	**Positive Thoughts**	Courage Hope Pride Satisfaction Trust
		Quiet Positive	Calmness Contentment Relaxation Relief Serenity	**Reactive**	Interest Politeness Surprise

on a sentiment lexicon, which is a collection of known and pre-compiled sentiment terms. Machine learning approaches make use of syntactic and/or linguistic features, and hybrid approaches are very common, with sentiment lexicons playing a key role in the majority of methods. Even simple approaches can be quite effective; for example, establishing the polarity of product reviews by identifying the polarity of the adjectives that appear in them (reportedly, this approach achieved a 10% higher accuracy than pure machine learning techniques [199]). However, such relatively successful techniques often fail when moved to new domains or text types, because they are inflexible regarding the ambiguity of sentiment terms. The context in which a term is used can change its meaning, particularly for adjectives in sentiment lexicons [200]. For example, a quiet car is generally considered a positive asset, but a quiet alarm clock is generally not. Several

evaluations have shown the usefulness of contextual information [201, 202], and have identified context words with a high impact on the polarity of ambiguous terms [203]. A further bottleneck is the time-consuming creation of these sentiment dictionaries, though solutions have been proposed in the form of crowdsourcing techniques.

Recently, techniques for opinion mining have begun to focus on social media, combined with a trend toward its application as a proactive rather than a reactive mechanism. Understanding public opinion in this way can have important consequences for the prediction of future events, for governments and the media wanting to know about reaction to events and policies, to people looking to predict the stock market, and many other things. Adapting tools to deal with social media, however, is often far from trivial, as explained in Chapter 8. In particular, pre-processing components do not work so well, the short messages on Twitter lack useful contextual information, and there are many misspellings which mean that sentiment words can be missed; furthermore, slang is rife and messages are often (sometimes deliberately) ambiguous.

The vast majority of opinion mining techniques make use of machine learning, partly because it is quick and easy to set up, and because reasonable results can be obtained with minimal effort. Supervised approaches are particularly beneficial when large amounts of training data are available, such as user reviews where an explicit rating system accompanies the free-form text. However, such approaches do not adapt well to tweets and other forms of social media [204], especially those in a specific domain. For specific cases, training data can be created using hashtags or emoticons, but these often constitute only a small proportion of the relevant data since most people do not use these indicators in their tweets. A body of work has thus focused on adapting machine learning methods to new domains [205], but these typically focus on the use of different keywords in similar kinds of text, e.g., product reviews about books vs. reviews about electronics. For targeted opinion mining tasks, especially industrial applications rather than speculative research, a knowledge-based approach is often preferred as it enables the developers to more easily make the opinion mining specific to the task, for example to focus specifically on opinion targets and types, rather than just finding generic positive and negative tweets or emotion categories.

A typical knowledge-based opinion mining approach uses linguistic pre-processing, as described in Chapter 2, gazetteers of sentiment lexicons, and some rules for combining the sentiment scores with other linguistic features (connecting to entities for target recognition, modifying the scores when negatives, adverbs, etc., are found, contextual dependencies, and so on). These methods are thus very easy for a user to tweak when errors are found, for example if new sentiment words or phrases are discovered not in the lexicon, when sentiment words are used in a particular way, when particular linguistic expressions are used, and so on. Typical examples of knowledge-based opinion mining tools are found in GATE, VADER [206], and SO-CAL [207].

7.6 OPINION MINING AND ONTOLOGIES

Concept-level sentiment analysis is a term typically used to refer to approaches that go beyond the word-level analysis and focus instead on a semantic analysis based on ontologies, linked data,

or other semantic resources. By semantic analysis, we mean here that they move away from the more traditional and explicit use of lexicons and co-occurrence information to an approach which relies on the implicit features associated with natural language concepts [208]. For example, SentiWordNet is a resource based on WordNet that adds sentiment information (scores for positivity, negativity, and objectivity) to each WordNet synset. Linking sentiment words found in the text with SentiWordNet thus enables synonyms and variants to be easily found. The CLSA (Concept-Level Semantic Analysis) challenges in 2014 and 2015 were designed exactly to promote the development of semantic opinion mining technology, and showcase some excellent examples [208, 209]; the series is planned to continue at least into 2016.

An example of such a system is given in [210], which uses an ontology to model the space of online reviews, populated with instances from DBpedia. Instances from the DBpedia Lexicalization Dataset [211] are expanded using contextual frames (i.e., the set of words surrounding a term is used to find new relevant terms, as described in Chapter 6). Sentiment lexicons and associated concept triples (e.g., *beer*, *cold*, *positive*) are also included. Other systems such as [212] encode terms corresponding to a concept (aspects) in an ontology, and then expand the set of terms with synonyms and hyponyms found in the text. For example, {zoom, battery life, shutter lag, etc.} is a set of aspects shared by all products in the category *digital camera* [213]. This is often known as aspect-based opinion mining. Figure 7.2 gives another example of an aspect ontology for the camera domain. Note that most of these approaches are designed to deal with closed domains such as product reviews, where products and their features can be easily modeled. It is much harder to use these kinds of approaches for open-domain opinion mining where the set of possible opinion targets is unknown.

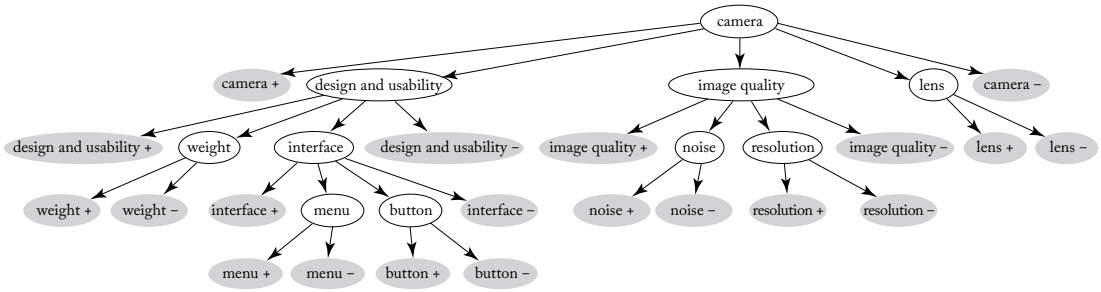


Figure 7.2: Section of an opinion-aspect ontology, reproduced from the presentation "Opinion Retrieval: Looking for Opinions in the Wild," Dr. Giorgos Paltoglou.

One of the challenges hampering the development of such tools, however, is the need to integrate existing linguistic resources for sentiment analysis with such semantic resources. The Linguistic Linked Open Data Cloud (LLOD)[3] is one such initiative to make available linguistic resources in a similar vein to the Linked Open Data Cloud, using vocabularies such as OWL,

[3] http://linguistic-lod.org/

lemon,[4] and NIF[5] to express them; but converting and integrating legacy resources to this is nevertheless by no means a trivial task.

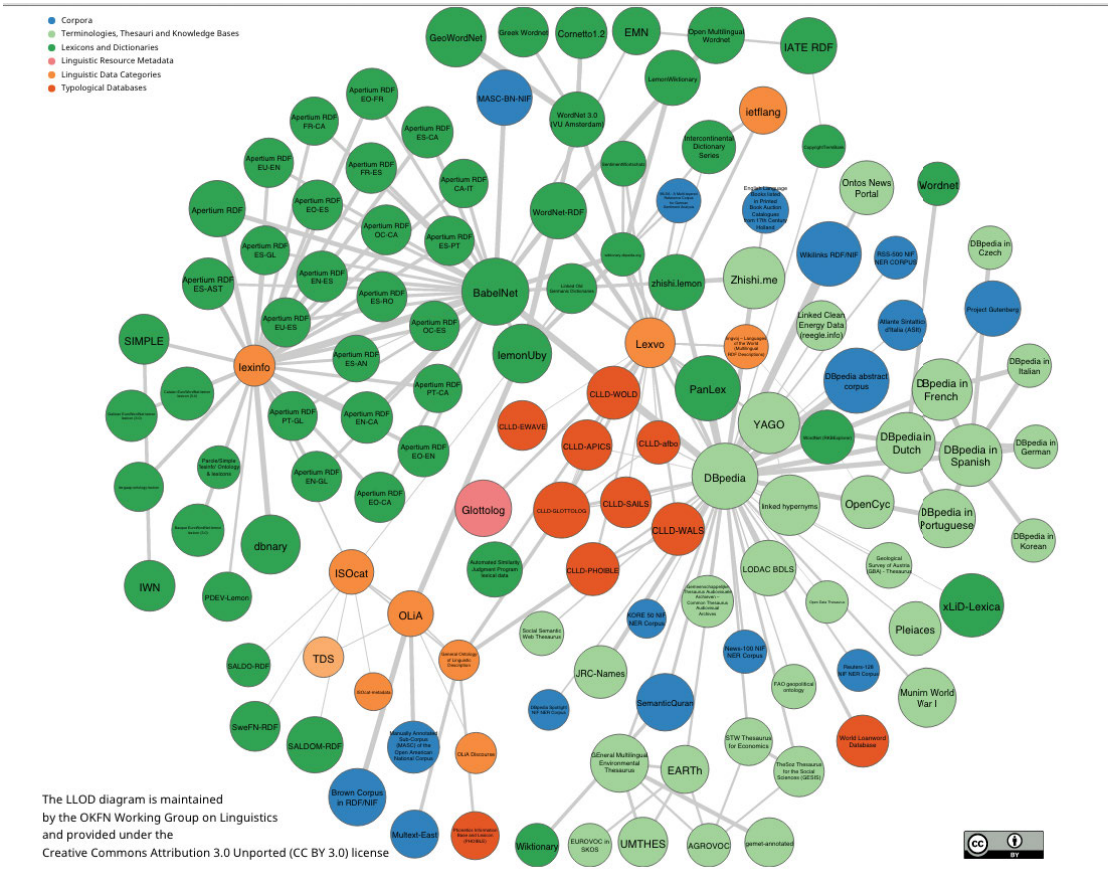


Figure 7.3: Linguistic linked open data cloud, as of January 2016 (automatically generated from the data contained in Linghub and maintained by the OKFN Working Group on Linguistics).

7.7 OPINION-MINING TOOLS

One of the most popular off-the-shelf tools used, both by researchers and in some industrial applications, is SentiStrength [214], mainly because it is freely available, works reasonably well, and is easy to set up and use as standalone or integrated in other applications. It is designed to estimate the strength of positive and negative sentiment in short texts, and deals well with informal

[4] http://lemon-model.net/
[5] http://persistence.uni-leipzig.org/nlp2rdf/

language such as tweets. Unlike most other tools, SentiStrength reports two sentiment strengths separately: negativity on a scale of −1 to −5 (where −5 is extremely negative), and positivity on a scale of 1 to 5 (where 5 is extremely positive). Both Windows and standalone Java versions are available,[6] and it has also recently been integrated with GATE as a plugin; all are customizable via various parameters. However, it suffers from the typical problems of current opinion mining tools: it deals well with explicit sentiment but less well with more complex expressions or those requiring some world knowledge, performance being based largely on the quality of its lexicons.

Most of the major NLP toolkits also have opinion mining components or can at least be applied to the task. This includes NLTK, UIMA, Lingpipe, the Stanford Toolkit, GATE, and also R's Text Mining package, Weka and Rapid Miner, which have classification packages. These mostly use machine learning methods (other than GATE, which has both) and are thus dependent mainly on the quality of the training data and chosen features.

7.8 SUMMARY

In this chapter, we have explained the concept of opinion mining and outlined the various tasks that typically form part of it. We have shown how the tools and methods described in the previous chapters (in particular linguistic pre-processing, named entity recognition, and term recognition) may all be used for the opinion mining task, and how a tool can thus be built up from such components. There are many challenges that still face the development of opinion mining tools, and performance is below that of many other typical NLP tasks, but it is very much an ongoing area of research and development, and tools are already used in real business scenarios nevertheless. The incorporation of semantic technology, such as the Linguistic Linked Open Data Cloud, is currently contributing much to the improvement of both performance and coverage of such tools, and most recently, investigation into Deep Learning methods for opinion mining may also prove fruitful.

[6] http://sentistrength.wlv.ac.uk/

CHAPTER 8

NLP for Social Media

The widespread adoption of social media is based on tapping into the social nature of human interactions, by making it possible for people to voice their opinion, become part of a virtual community and collaborate remotely. If we take micro-blogging as an example, Twitter has over 300 million active users, posting millions of tweets daily.[1]

Engaging actively with such high-value, high-volume, brief life-span media streams has now become a daily challenge for both organizations and ordinary people. Automating this process through intelligent, semantic-based information access methods is therefore increasingly needed. This is an emerging research area, combining methods from many fields, in addition to semantic technologies, namely natural language processing, social science, machine learning, personalization, and information retrieval.

Traditional search methods are no longer able to address the more complex information seeking behavior in social media, which has evolved toward sense making, learning and investigation, and social search [215]. Semantic technologies have the potential to help people cope better with social media–induced information overload. Automatic semantic-based methods that adapt to an individual's information-seeking goals and summarize briefly the relevant social media, could ultimately support information interpretation and decision making over large-scale, dynamic media streams.

Unlike carefully authored news and other textual web content, social media streams pose a number of new challenges for semantic technologies, due to their large-scale, noisy, irregular, and social nature. This chapter discusses the following NLP tasks and research challenges:

- how social media analysis differs from that of longer, less noisy texts;
- ontologies developed for modeling social media content and analysis results; and
- semantic annotation of social media, with focus on keyphrase/term extraction; named entity recognition and linking; event detection; sentiment and opinion mining; and cross-media analysis.

Searching and visualizing the results of large-scale, social media semantic analysis are also very challenging tasks, which are addressed in Chapter 9.

[1] http://www.statista.com/statistics/282087/number-of-monthly-active-twitter-users/ (Visited 29 January, 2016).

8.1 SOCIAL MEDIA STREAMS: CHARACTERISTICS, CHALLENGES, AND OPPORTUNITIES

Social media sites allow users to connect with each other for the purpose of sharing content (e.g., web links, photos, videos), experiences, professional information, and online socializing with friends. Users create posts or status updates and social media sites circulate these to the user's social network. The key difference from traditional web pages is that users are not just passive information consumers, but many are also prolific content creators.

Social media can be categorized on a spectrum, based on the type of connection between users, how the information is shared, and how users interact with the media streams:

- Interest-graph media [216], such as Twitter, encourage users to form connections with others based on shared interests, regardless of whether they know the other person in real life. Connections do not always need to be reciprocated. Shared information comes in the form of a stream of messages in reverse chronological order.

- Social networking sites (SNS) encourage users to connect with people they have real-life relationships with. Facebook, for example, provides a way for people to share information, as well as comment on each other's posts. Typically, short contributions are shared, outlining current events in users' lives or linking to something on the internet that users think their friends might enjoy. These status updates are combined into a time-ordered stream for each user to read.

- Professional Networking Services (PNS), such as LinkedIn, aim to provide an introductions service in the context of work, where connecting to a person implies that you vouch for that person to a certain extent, and would recommend them as a work contact for others. Typically, professional information is shared and PNS tend to attract older professionals [217].

- Content sharing and discussion services, such as blogs, video sharing (e.g., YouTube, Vimeo), slide sharing (e.g., SlideShare), and user discussion/review forums (e.g., CNET). Blogs usually contain longer contributions. Readers might comment on these contributions, and some blog sites create a time stream of blog articles for followers to read. Many blog sites also advertise new blog posts automatically through their users' Facebook and Twitter accounts.

These different kinds of social media, coupled with their complex characteristics, make semantic interpretation extremely challenging. State-of-the-art automatic semantic annotation, browsing, and search algorithms have been developed primarily on news articles and other carefully written, long web content [218]. In contrast, most social media streams (e.g., tweets, Facebook messages) are strongly inter-connected, temporal, noisy, short, and full of slang, leading to severely degraded results.[2]

[2]For instance, named entity recognition methods typically have 85–90% accuracy on news but only 30–50% on tweets [219, 220].

These challenging social media characteristics are also opportunities for the development of new semantic technology approaches, which are better suited to media streams:

Short messages (microtexts): Twitter and most Facebook messages are very short (140 characters for tweets). Many semantic-based methods reviewed below supplement these with extra information and context coming from embedded URLs and hashtags.[3] For instance, Abel et al. [134] augment tweets by linking them to contemporaneous news articles, whereas Mendes et al. exploit online hashtag glossaries to augment tweets [221].

Noisy content: social media content often has unusual spelling (e.g., 2moro), irregular capitalization (e.g., all capital or all lowercase letters), emoticons (e.g., :-P), and idiosyncratic abbreviations (e.g., ROFL, ZOMG). Spelling and capitalization normalization methods have been developed [222], coupled with studies of location-based linguistic variations in shortening styles in microtexts [223]. Emoticons are used as strong sentiment indicators in opinion mining algorithms (see Section 8.3.4).

Temporal: in addition to linguistic analysis, social media content lends itself to analysis along temporal lines, which is a relatively under-researched problem. Addressing the temporal dimension of social media is a pre-requisite for much-needed models of conflicting and consensual information, as well as for modeling change in user interests. Moreover, temporal modeling can be combined with opinion mining, to examine the volatility of attitudes toward topics over time.

Social context is crucial for the correct interpretation of social media content. Semantic-based methods need to make use of social context (e.g., who is the user connected to, how frequently they interact), in order to derive automatically semantic models of social networks, measure user authority, cluster similar users into groups, as well as model trust and strength of connection.

User-generated: since users produce as well as consume social media content, there is a rich source of both explicit and implicit information about the user, e.g., demographics (gender, location, age, etc.), interests, opinions. The challenge here is that in some cases, user-generated content is relatively small, so corpus-based statistical methods cannot be applied successfully.

Multilingual: Social media content is strongly multilingual. For instance, less than 50% of tweets are in English, with Japanese, Spanish, Portuguese, and German also featuring prominently [136]. Unfortunately, semantic technology methods have so far mostly focused on English, while low-overhead adaptation to new languages still remains an open issue. Automatic language identification [136, 224] is an important first step, allowing applications to

[3] A recent study of 1.1 million tweets has found that 26% of English tweets contain a URL, 16.6% – a hashtag, and 54.8% contain a user name mention [136].

Table 8.1: Ontologies and what they model

Ontology	People	Online Posts	Social Networks	Micro Blogs	User Interests	Tags	Geo-location	User Behavior
FOAF	Yes		Knows		Partial			
SIOC(T)	Yes	Yes		Partial	Yes			
MOAT						Yes		
Bottari	Yes	Yes	Yes	Yes		Yes	Yes	
DLPO	Yes	Yes	Yes	Yes	Yes	Yes		
SWUM	Yes				Yes		Yes	Yes
UBO	Yes		Yes		Yes			Yes

first separate social media in language clusters, which can then be processed using different algorithms.

The rest of this chapter discusses how these challenges have been addressed in research to date and where open issues remain.

8.2 ONTOLOGIES FOR REPRESENTING SOCIAL MEDIA SEMANTICS

Ontologies are used heavily in semantic annotation and other NLP tools. Consequently, in this section we focus specifically on ontologies, which can help NLP methods with processing different kinds of social media and accompanying content, including user profiles, sharing, tagging, and liking. Table 8.1 provides an overview of these ontologies, alongside different dimensions, which are discussed in more detail next:

Describing People and Social Networks: Friend-of-a-Friend[4] (FOAF) is a vocabulary for describing people, including names, contact information, and a generic `knows` relation. FOAF also supports limited modeling of interests by modeling them as pages on the topics of interest. As acknowledged in the FOAF documentation itself, such an ontological model of interests is somewhat limited.

Modeling Social Media Sites: The Semantically Interlinked Online Communities[5] (SIOC) ontology models social community sites (e.g., blogs, wikis, online forums). Key concepts are forums, sites, posts, user accounts, user groups, and tags. SIOC supports modeling of user interests through the `sioc:topic` property, which has a URI as a value (posts and user groups also have topics).

[4] http://xmlns.com/foaf/0.1/
[5] http://sioc-project.org/

Modeling microblogs: SIOC has recent extensions (SIOCT), modeling microblogs through the new concept of `MicroblogPost`, a `sioc:follows` property (representing follower/followee relationships on Twitter), and a `sioc:addressed_to` property for posts that mention a specific user name. *Bottari* [225] is an ontology, which has been developed specifically to model relationships in Twitter, especially linking tweets, locations, and user sentiment (positive, negative, neutral), as extensions to the SOIC (Socially Interlinked Online Communities) ontology. A new TwitterUser class is introduced, coupled with separate `follower` and `following` properties, similar to those in SIOCT. The Tweet class is a type of sioc:Post and, unlike SIOCT, Bottari also distinguishes retweets and replies. Locations (points-of-interest) are represented using the W3C Geo vocabulary,[6] which enables location-based reasoning.

Interlinking Social Media, Social Networks, and Online Sharing Practices: DLPO (The LivePost Ontology) provides a comprehensive model of social media posts, going beyond Twitter [226]. It is strongly grounded in fundamental ontologies, such as FOAF, SOIC, and the Simple Knowledge Organization System (SKOS).[7] DLPO models personal and social knowledge discovered from social media, as well as linking posts across personal social networks. The ontology captures six main types of knowledge: online posts, different kinds of posts (e.g., retweets), microposts, online presence, physical presence, and online sharing practices (e.g., liking, favoriting). However, while topics, entities, events, and time are well covered, user behavior roles and individual traits are not addressed as comprehensively as in the SWUM ontology [227] discussed below.

Modeling Tag Semantics: The MOAT (Meaning-Of-A-Tag) ontology [228] allows users to define the semantic meaning of a tag through Linking Open Data and, ultimately, to create manually semantic annotations of social media. The ontology defines two kinds of tags: global (across all content) and local (particular tag on a given resource). MOAT can be combined with SIOCT to tag microblog posts [229]. The DLPO ontology, introduced above, also models topics and tags associated with online posts (including microblogs).

User modeling ontologies are key to the representation, aggregation, and sharing of information about users and their social media interactions. The General User Modeling Ontology (GUMO) [230], for instance, aims to cover a wide range of user-related information, such as demographics, contact information, personality, etc. However, it falls short of representing user interests, which makes it unsuitable for social media.

Based on an analysis of 17 social web applications, Plumbaum et al. [227] have derived a number of user model dimensions required for a social web user modeling ontology. Their taxonomy of dimensions includes demographics, interests and preferences, needs and goals, mental and physical state, knowledge and background, user behavior, context, and individual traits (e.g., cognitive style, personality). Based on these, they have created the SWUM

[6] `http://www.w3.org/2003/01/geo/`

[7] `http://www.w3.org/2004/02/skos/`. Developed to model thesauri, term lists, and controlled vocabularies.

(Social Web User Model) ontology. A key shortcoming of SWUM, however, is its lack of grounding in other ontologies. For instance, user location attributes, such as Country and City, are coded as strings, which severely limits their usefulness for reasoning (e.g., it is hard to find all users based in South West England, based on their cities). A more general approach would have been to define these through URIs, grounded in commonly used Linked Data resources, such as DBpedia and Freebase.

Lastly, the User Behavior Ontology [231] models user interactions in online communities. It has been used to model user behavior in online forums [231] and also Twitter discussions [232]. It has classes that model the impact of posts (replies, comments, etc), user behavior, user roles (e.g., popular initiator, supporter, ignored), temporal context (time frame), and other interaction information. Addressing the temporal dimension of social media is particularly important, especially when modeling changes over time (e.g., in user interests or opinions).

To summarize, there are a number of specialized ontologies, aimed at representing and reasoning with automatically derived semantic information from social media. However, given that they address different phenomena, many NLP applications adopt or extend more than one, in order to meet their requirements. In some cases, NLP methods are used to populate these ontologies with instances automatically, based on social media content (e.g., populating user and community models for a specific set of users/community).

8.3 SEMANTIC ANNOTATION OF SOCIAL MEDIA

Researchers have investigated a wide range of semantic annotation tasks on social media content. This section will discuss some of these in more detail, starting from keyphrase extraction.

8.3.1 KEYPHRASE EXTRACTION

Automatically selected keyphrases are useful in representing the topic of a document or collection of documents, although not very effective in delivering arguments or full statements contained therein. Keyphrase extraction can therefore be considered as a form of shallow knowledge extraction, giving a topical overview. Keywords can also be used in the context of semantic annotation and retrieval, as a means of dimensionality reduction and allowing systems to deal with smaller sets of important terms rather than whole documents.

Keyphrase extraction is closely related to term extraction, but differs mainly in its representative nature. Keyphrase extraction aims to represent the topic by extracting the most significant words and phrases, thereby giving a kind of overview of the document, and thus has a clear end goal. Term extraction does not attempt to represent the document directly, but only seeks to find all the domain-specific terminology used (no matter how significant it is to the document itself). Also, where extracted terms are linked to an ontology or other vocabulary, this is not the case with keyphrase extraction.

Some keyword extraction approaches exploit term co-occurrence; forming a graph of terms with edges derived from the distance between occurrences of a pair of terms and assigning weights to vertices [233]. This class of keyword extraction was found to perform favorably on Twitter data compared to methods which relied on text models [234].

These graph-based approaches to extracting keywords from Twitter perhaps perform well because the domain contains a great deal of redundancy [235]. For example, in the context of trending topics on Twitter (frequently denoted by hashtags), [236] extracted keyphrases by exploiting textual redundancy and selecting common sequences of words. While redundancy in Twitter and other social media is somewhat beneficial when producing keyword summaries, a less helpful trait is the sheer variety of topics discussed. In cases where documents discuss more than one topic, it can be more difficult to extract a coherent and faithful set of keywords from it.

Personal Twitter timelines, when treated as single documents, present this problem. Users are generally capable of posting on multiple topics. While [234] use TextRank on the whole of a user's stream, they do not attempt to model or address topic variation, unlike [237], who incorporated topic modeling into their approach. Theirs is not the only application of Topic Modeling to Twitter data, as it is similar to [238]. However in the latter work topics are discovered but never summarized.

In the context of social tagging and bookmarking services such as Flickr, Delicious, and Bibsonomy, researchers have studied the automatic tagging of new documents with folksonomy tags. One of the early approaches is the AutoTag system [239], which assigns tags to blog posts. First, it finds similar pre-indexed blog posts using standard information retrieval methods, using the new blog post as the query. Then it composes a ranked list of tags, derived from the top-most relevant posts, boosted with information about tags used previously by the given blogger.

More recent approaches use keyphrase extraction from blog content, in order to suggest new tags. For instance, [240] generate candidate keyphrases from n-grams, based on their POS tags, then filter these using a supervised, logistic regression classifier. The keyphrase-based method can be combined with information from the folksonomy [241], in order to generate tag signatures (i.e., associate each tag in the folksonomy with weighted, semantically related terms). These are then compared and ranked against the new blog post, in order to suggest the most relevant set of tags.

8.3.2 ONTOLOGY-BASED ENTITY RECOGNITION IN SOCIAL MEDIA

Named entity recognition methods, which are typically trained on longer, more regular texts (e.g., news articles), have been shown to perform poorly on shorter and noisier social media content [220]. However, while each post in isolation provides insufficient linguistic context, additional information can be derived from the user profiles, social networks, and interlinked posts (e.g., replies to a tweet message). This section discusses what we call *social media–oriented* semantic annotation approaches, which integrate both linguistic and social media-specific features.

Table 8.2: Ontology-based semantic annotation: selected research tools

	Ontology/ LOD Resource Used	Annotations Produced	Disamb. Performed	Target Domain	Corpora Used	Evaluated On
DBpedia Spotlight [115]	DBpedia, Freebase	Over 30 classes	Yes	Open domain	Wikipedia	News
LINDEN [117]	YAGO	YAGO classes	Yes	Open domain	Wikipedia	TAC-KBP 2009
Ritter [220]	Freebase	10 classes	No	Open domain	Tweets	Tweets
Ireson [242]	GeoPlanet	Locations	Yes	Photos	Flickr	Flickr
Laniado&Mika [243]	Freebase	Freebase	Yes	Open domain	Tweets	Tweets
Meij [121]	Wikipedia	Wikipedia	Yes	Open domain	Wikipedia	Tweets
Gruhl [244]	MusicBrainz	Songs and albums	Yes	Music domain	MySpace	MySpace posts
Rowe [134]	DBpedia	Conference-related	Yes	Conferences	Tweets	200 tweets
Choudhury [245]	Wikipedia	Cricket players, games	No	Sports events	Wikipedia	Cricket tweets

Ritter et al. [220] address the problem of named entity classification (but not disambiguation) by using Freebase as the source of a large number of known entities. The straightforward entity lookup and type assignment baseline, without considering context, achieves only 38% f-score (35% of entities are ambiguous and have more than one type, whereas 30% of entities in the tweets do not appear in Freebase). NE classification performance improves to 66% through the use of labeled topic models, which take into account the context of occurrence and the distribution over Freebase types for each entity string (e.g., Amazon can be either a company or a location).

Ireson et al. [242] study the problem of location disambiguation (toponym resolution) of name tags in Flickr. The approach is based on the Yahoo! GeoPlanet semantic database, which provides a URI for each location instance, as well as a taxonomy of related locations (e.g., neighboring locations). The tag disambiguation approach makes use of all other tags assigned to the photo, the user context (all tags assigned by this user to all their photos), and the extended user context, which takes into account the tags of the user contacts. The use of this wider, social network–based context was shown to improve significantly the overall disambiguation accuracy.

Another source of additional, implicit semantics are hashtags in Twitter messages, which have evolved as means for users to follow conversations on a given topic. Laniado and Mika [243] investigate hashtag semantics in 369 million messages, using four metrics: frequency of use, specificity (use of the hashtag vs. use of the word itself), consistency of usage, and stability over time. These measures are then used to determine which hashtags can be used as identifiers and linked to Freebase URIs (most of them are named entities). Hashtags have also been used as an additional source of semantic information about tweets, by adding textual hashtag definitions from crowd-sourced online glossaries [221]. Mendes et al. [221] also carry out semantic annotation through a simple entity lookup against DBpedia entities and categories without further disambiguation. User-related attributes and social connections are coded in FOAF, whereas semantic annotations are coded through the MOAT ontology (see Section 8.2).

Wikipedia-based entity linking approaches (see Section 5.3) benefit significantly from the larger linguistic context of news articles and web pages. Evaluation of DBpedia Spotlight [115] and the Milne and Witten method [114] on a tweet dataset has shown significantly poorer performance [121]. Meij et al. [121] propose a Twitter-specific approach for linking such short, noisy messages to Wikipedia articles. The first step uses n-grams to generate a list of candidate Wikipedia concepts, then supervised learning is used to classify each concept as relevant or not (given the tweet and the user who wrote it). The method uses features derived from the n-grams (e.g., number of Wikipedia articles containing this n-gram), Wikipedia article features (e.g., number of articles linking to the given page), and tweet-specific features (e.g., using hashtag definitions and linked web pages).

Gruhl et al. [244] focus in particular on the disambiguation element of semantic annotation and examine the problem of dealing with highly ambiguous cases, as is the case with song and music album titles. Their approach first restricts the part of the MusicBrainz ontology used for

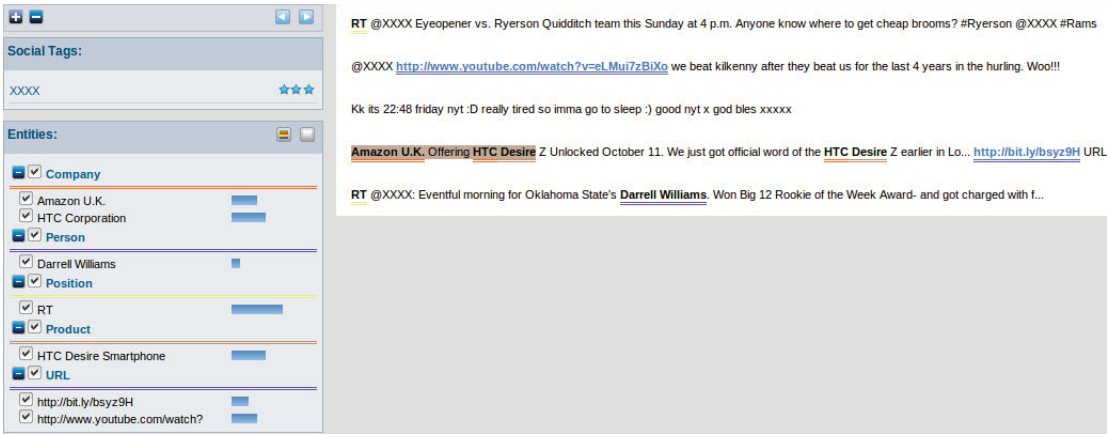


Figure 8.1: Calais results on tweets.

producing the candidates (in this case by filtering out all information about music artists not mentioned in the given text). Secondly, they apply shallow language processing, such as POS tagging and NP chunking, and then use this information as input to a support vector machine classifier, which disambiguates on the basis of this information. The approach was tested on a corpus of MySpace posts for three artists. While the ontology is very large (thus generating a lot of ambiguity), the texts are quite focused, which allows the system to achieve good performance. As discussed by the authors themselves, the processing of less focused texts, e.g., Twitter messages or news articles, is likely to prove much more challenging.

With respect to entity linking, recent tweet-focused evaluations uncovered problems in using state-of-the-art approaches in this genre [67, 134], largely due to the brevity of tweets (140 characters) and also due to treating each post in isolation, without considering the wider available context. In particular, only tweet text is typically processed, despite the fact that the complete tweet JSON object also includes author profile data (full name, optional location, profile text, and web page). Around 26% of all tweets also contain URLs [136], 16.6% – hashtags, and 54.8% – at least one user name mention.

NER systems targeted at microblog text do not commonly utilize social media cues, for example treating hashtags as common words, e.g., [219, 246] or not considering them, as in TwiNER [247]. Shen et al. [139] use additional tweets from a user's timeline to find user-specific topics and use those to improve the disambiguation. Huang et al. [140] present an extension of graph-based disambiguation which introduces "Meta Paths" that represent context from other tweets through shared hash tags, authors, or mentions. Gattani et al. [141] make use of URL expansion and use context derived from tweets by the same author and containing the same hashtag, but do not evaluate the contribution of this context to end performance, and don't make use of hashtag definitions or user profile text.

In the context of the YODIE system (see Section 5.3.2), a systematic investigation was carried out [129], studying the impact of wider social context on performance of LOD-based entity disambiguation in tweets. In particular, in the case of hashtags, tweet content was enriched with hashtag definitions, retrieved automatically from the web. Similarly, tweets containing @mentions were enriched with the textual information from that Twitter profile. In the case of URLs, the corresponding web content was appended to the tweet. Disambiguation performance was measured both when such context expansion was performed *individually* (i.e., only hashtags, only URLs, etc.), as well as when all three types of contextual information was used *jointly*. The experiments demonstrated that tweet expansions lead to significantly improved entity linking performance on microblog content. In particular, overall accuracy improved by 7.3 percentage points. Performance gain was slightly lower for F1–6.2 percentage points.

The main gains came from the ability to disambiguate @mentions, where the tweet-text only baseline fails to identify their DBpedia referent. The dominant contribution in this case, therefore, is in terms of recall. It should also be noted that even without mention expansions, URL and hashtag expansions also lead to significant improvements.

Processing Social Media with GATE

Due to the challenging nature of social media (see Section 8), the standard pre-processing and entity reconition tools from GATE (see Chapters 2 and 3) have, therefore, been adapted to this specific genre.

Therefore, GATE provides the TwitIE plugin [248]—a customization of ANNIE, specific to social media content, which has been tested extensively on microblog messages. The latter content is both readily available as a large public stream and also is the most challenging to process with generic IE tools, due to the shortness, noisy nature, and prevalence of slang and Twitter-specific conventions.

Figure 8.2 shows the TwitIE pipeline and its components. TwitIE is distributed as a plugin in GATE, which needs to be loaded for these processing resources to appear in GATE Developer. Components reused from ANNIE without any modification are shown in blue, whereas the red ones are new and specific to social media.

The first step is language identification, which is based on a social media–adapted version of TextCat [136]. Due to the shortness of tweets, it makes the assumption that each tweet is written in only one language. The choice of languages used for categorization is specified through a cofiguration file, supplied as an initialization parameter. Given a collection of tweets in a new language, it is possible to train TwitIE TextCat to support that new language as well. This is done by using the Fingerprint Generation PR, included in the `Language_Identification` GATE Plugin [249]. It builds a new fingerprint from a corpus of documents.

The TwitIE tokenizer is an adaptation of the ANNIE English tokenizer. It follows Ritter's tokenization scheme [220]. More specifically, it treats abbreviations (e.g., RT, ROFL) and URLs as one token each. Hashtags and user mentions are two tokens (i.e., # and nike in the above

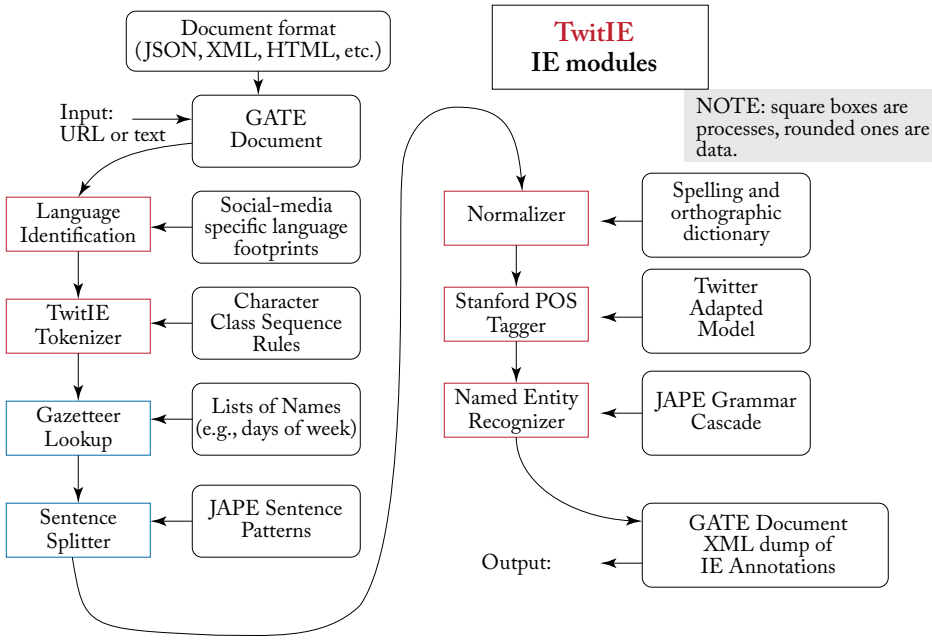


Figure 8.2: The TwitIE information extraction pipeline.

example) plus a separate annotation HashTag covering both. Capitalization is preserved, but an orthography feature is added: all caps, lowercase, mixCase. Lowercasing and emoticons are optionally done in separate modules, since they are not always needed. Consequently, tokenization is faster and more generic, as well as more tailored to the needs of named entity recognition.

The **gazetteer** consists of lists such as cities, organizations, days of the week, etc. It not only consists of entities, but also of names of useful *indicators*, such as typical company designators (e.g., "Ltd."), titles, etc. The gazetteer lists are compiled into finite state machines, which can match text tokens. TwitIE reuses the ANNIE gazetteer lists, at present, without any modification.

The **sentence splitter** is a cascade of finite-state transducers which segments text into sentences. This module is required for the POS tagger. Again, at present, the ANNIE sentence splitter is reused without modification, although when processing tweets, it is also possible to just use the text of the tweet as one sentence, without further analysis.

The TwitIE Normalizer is currently a combination of a generic spelling-correction dictionary and a spelling-correction dictionary specific to social media. The latter contains entries such as "2moro" and "brb," similar to Han et al. [250].

The TwitIE POS tagger contains an adapted model for the Stanford POS tagger, trained on PTB-tagged tweets. Extra tag labels have been added for retweets, URLs, hashtags, and user mentions. The Stanford POS tagger was re-trained [251] using some hand-annotated tweets [220],

the NPS IRC corpus [252], and news texts (the Wall Street Journal part of the Penn Treebank [253]). The resulting model achieves 83.14% POS tagging accuracy, which is still below the 97% achieved on news content. In order to ensure the best possible performance, the TwitIE POS tagger needs to be run after the TwitIE tokenizer and normalizer. Since it is currently trained only on English content, it should only be run on tweets identified as English by the TwitIE language identifier.

Lastly, the TwitIE NER component is a manual adaptation of the ANNIE rule-based entity recognizer. Thanks to the social media adaptation of ANNIE, TwitIE achieves a +30% absolute precision and +20% absolute F1 performance increase, when compared to ANNIE.

8.3.3 EVENT DETECTION

Much as trending topics can be used to monitor global opinions and reactions, social media streams can be used as a discussion backchannel to real-world events [254], and even to discover and report upon such events, almost as soon as they occur. While it may at first appear that trending topics alone are sufficient for this task, there are a few reasons why they are unsatisfactory:

- *Generality*: trending topics may discuss events, but may also refer to celebrities, products, or online memes.
- *Scale*: only the topics with which a huge margin of Twitter users engage can appear as trending topics.
- *Censorship*: it is believed by many that the trending topics displayed by the official Twitter service are censored for political and language content.
- *Algorithm*: the method used to select trending topics is not published anywhere and is generally not understood.

Automatic event detection therefore presents an interesting task for social media streams. While it is possible to have access to an enormous quantity of tweets, enough to reveal global trends and events, the problem of developing and evaluating scalable event detection algorithms which can handle such magnitudes of streaming text remains.

The majority of approaches to event detection do not utilize ontologies or other sources of semantic information. One class of methods uses clustering on tweets [255–257] or blog posts [258]. Another class takes inspiration from signal processing, analysing tweets as sensor data. For example, [259] used such an approach to detect earthquakes in Japan on the basis of tweets with geolocation information attached to them. Similarly, individual words have been treated as wavelet signals in order to discover temporally significant clusters of terms [260].

Once an event is detected in social media streams, the next problem is how to generate useful thematic/topical descriptors for this event. Point-wise mutual information has been coupled with user geolocation and temporal information, in order to derive n-gram event descriptors

from tweets [261]. By making the algorithm sensitive to the originating location, it is possible to see what people from a given location are saying about an event (e.g., those in the U.S)., as well as how this differs from tweets elsewhere (e.g., those from India).

Collections of events in a larger sequence could be referred to as sagas; they may be perfectly legitimate events in their own right, or their individual constituents might similarly be coherent on their own. Citing the example of an academic conference, [135] point out that tweets may refer to the conference as a whole, or to specific sub-events such as presentations at a specific time and place. Using semantic information about the conference event and its sub-events from the Web of Data, tweets are aligned to these sub-events automatically, using machine learning. The method includes a concept enrichment phase, which uses Zemanta to annotate each tweet with DBpedia concepts. Tweets are described semantically using the SIOC and Online Presence semantic ontologies (see Section 8.2).

Another semantic, entity-based approach to sub-event detection has been proposed by [245], who use manually created background knowledge about the event (e.g., team and player names for cricket games), coupled with domain-specific knowledge from Wikipedia (e.g., cricket-related sub-events like getting out). In addition to annotating the tweets with this semantic information, the method utilizes tweet volume (similarly to [262]) and re-tweet frequency as sub-event indicators. The limitation of this approach, however, comes from the need for manual intervention, which is not always feasible outside of limited application domains.

8.3.4 SENTIMENT DETECTION AND OPINION MINING

The existence and popularity of websites dedicated to reviews and feedback on products and services is something of a homage to the human urge to post what they feel and think online. When the most common type of message on Twitter is about "me now" [263], it is to be expected that users talk often about their own moods and opinions. Bollen et al. [194] argue that users express both their own mood in tweets about themselves and more generally in messages about other subjects. Another study [264] estimates that 19% of microblog messages mention a brand and from those that do, around 20% contain brand sentiment.

The potential value of these thoughts and opinions is enormous. For instance, mass analysis could provide a clear picture of overall mood, exploring reactions to ongoing public events [194] or feedback to a particular individual, government, product, or service [265]. The resulting information could be used to improve services, shape public policy, or make a profit on the stock market.

The user activities on social networking sites are often triggered by specific events and related entities (e.g., sports events, celebrations, crises, news articles, persons, locations) and topics (e.g., global warming, financial crisis, swine flu). In order to include this information, semantically- and social network–aware approaches are needed.

There are many challenges inherent in applying typical opinion mining and sentiment analysis techniques to social media [266]. Microposts are, arguably, the most challenging text type

for opinion mining, since they do not contain much contextual information and assume much implicit knowledge. Ambiguity is a particular problem since we cannot easily make use of coreference information: unlike in blog posts and comments, tweets do not typically follow a conversation thread, and appear much more in isolation from other tweets. They also exhibit much more language variation, tend to be less grammatical than longer posts, contain unorthodox capitalization, and make frequent use of emoticons, abbreviations, and hashtags, which can form an important part of the meaning. Typically, they also contain extensive use of irony and sarcasm, which are particularly difficult for a machine to detect. On the other hand, their terseness can also be beneficial in focusing the topics more explicitly: it is very rare for a single tweet to be related to more than one topic, which can thus aid disambiguation by emphasizing situational relatedness.

Unlike some of the more recent concept-level sentiment analysis tools designed for text, such as product and travel reviews (as discussed in Chapter 7.6), which focus on aspect-based approaches, the majority of sentiment and opinion mining methods tested on social media utilise no or very little semantics. For instance, [267, 268] classify tweets as having positive, negative, or neutral sentiment, based on n-grams and part-of-speech information, whereas [269] use a sentiment lexicon to initially annotate positive and negative sentiment in tweets related to political events.

The use of such shallow linguistic information leads to a data sparsity problem. Saif et al. [133] demonstrate that by using semantic concepts, instead of words such as iPhone, polarity classification accuracy is improved. The approach uses AlchemyAPI for semantic annotation of 30 entity classes, the most frequent ones being Person, Company, City, Country, and Organization. The method is evaluated on the Stanford Twitter Sentiment Dataset[8] and shown to outperform semantics-free, state-of-the-art methods, including [268].

Semantic annotation has also been used for the more challenging opinion mining task. In particular, [270] identify people, political parties, and opinionated statements in tweets using a rule-based entity recognizer, coupled with an affect lexicon derived from WordNet. Subsequent semantic analysis uses patterns to generate triples representing opinion holders and voter intentions. Negation is dealt with by capturing simple patterns such as "isn't helpful" or "not exciting" and using them to negate the extracted sentiment judgments. This work was later extended with semantic annotation of political terms (organized into a topic hierarchy) and MPs, in a tool for analysing discussions on Twitter about the 2015 UK elections [271].

8.3.5 CROSS-MEDIA LINKING

The short nature of Twitter and Facebook messages, coupled with their frequent grounding in real-world events, means that often short posts cannot be understood without reference to external context. While some posts already contain URLs, the majority do not. Therefore automatic methods for cross-media linking and enrichment are required.

[8] http://twittersentiment.appspot.com/

Abel et al. [134] link tweets to current news stories in order to improve the accuracy of semantic annotation of tweets. Several linkage strategies are explored: utilising URLs contained in the tweet, TF-IDF similarity between tweet and news article, hashtags, and entity-based similarity (semantic entities and topics are recognized by OpenCalais), with the entity-based one being the best one for tweets without URLs. The approach bears similarities with the keyphrase-based linking strategy for aligning news video segments with online news pages [272]. [273] go one step further by aggregating social media content on climate change from Twitter, YouTube, and Facebook with online news, although details of the cross-media linking algorithm are not supplied in the paper.

An in-depth study comparing Twitter and *New York Times* news [274] has identified three types of topics: event-oriented, entity-oriented, and long-standing topics. Topics are also classified into categories, based on their subject area. Nine of the categories are those used by NYT (e.g., arts, world, business) plus two Twitter-specific ones (Family&Life and Twitter). Family&Life is the predominant category on Twitter (called "me now" by [263]), both in terms of number of tweets and number of users. Automatic topic-based comparison showed that tweets abound with entity-oriented topics, which are much less covered by traditional news media.

Going beyond news and tweets, future research on cross-media linking is required. For instance, some users push their tweets into their Facebook profiles, where they attract comments, separate from any tweet replies and retweets. Similarly, comments within a blog page could be aggregated with tweets discussing it, in order to get a more complete overall view.

8.3.6 RUMOR ANALYSIS

One specific kind of semantic analysis of social media is rumor analysis. Research firstly demonstrated the damage that the diffusion of false rumors can cause in society, as well as the slow spread of debunking tweets [275, 276]. Being able to determine the accuracy of circulating information in social media is therefore crucial. However, the veracity of rumors is usually hard to establish [390], since as many views and testimonies as possible need to be assembled and examined in order to reach a final judgement. Examples of rumors that were later disproven, after being widely circulated, include a 2010 earthquake in Chile, where rumors of a volcano eruption and a tsunami warning in Valparaiso spawned on Twitter [277]. Another example is the England riots in 2011, where false rumors claimed that rioters were going to attack Birmingham's Children's Hospital and that animals had escaped from London Zoo [278].

The first step of rumor analysis is to detect tweets pertaining to rumors [279, 280].

One particularly influential work is Mendoza et al. [277], who analysed manually 7 confirmed truths and 7 false rumors regarding the earthquake in Chile in 2010, with each rumor consisting of around 1,000 tweets. Each tweet was then classified manually according to the stance it expresses toward the rumor claim: affirmation, denial, questioning, unknown, or unrelated. The study showed that a much higher percentage of tweets about false rumors are shown to deny the respective rumors (approximately 50%). This is in contrast to rumors later proven to

be true, where only 0.3% of tweets were denials. Based on this, authors claimed that rumors can be detected using aggregate analysis of the stance expressed in tweets.

This inspired a large body of subsequent work on rumor stance classification. One of the first approaches of Qazvinian et al. [281] classified each tweet automatically as supporting, denying or questioning a given rumor. However, they chose to conflate the denying and questioning tweets for each rumor into a single class, converting it into a 2-way classification problem of supporting vs. denying-or-questioning. Hamidian and Diab [282] use Tweet Latent Vectors to assess the ability of performing 2-way classification of the stance of tweets as either supporting or denying a rumor. They study the extent to which a model trained on historical tweets can be used for classifying new tweets on the same rumor.

More recent work has cast this back to the more realistic 3-way classification [283]. Other notable approaches are Liu et al. [284] who introduce rule-based methods for stance classification, which outperforms [281]. Similarly, [279] use regular expressions for rumor stance classification.

In all those cases, the most challenging aspect is generalization to new, unseen rumors, which often differ from what the classifier has observed in the training data. Earlier work ignored this distinction and pooled together tweets from all rumors using cross-validation. More recent work [285] on rumor stance classification defines the problem as transfer learning and evaluates only on unseen rumors. Zeng et al. [286] explored the use of three different classifiers (Random Forest, Naive Bayes, and Logistic Regression) for automated rumor stance classification on unseen rumors, but only focusing on a 2-way support/deny definition of the problem.

The key challenge for researchers of rumors in social media is the lack of large, widely available datasets. The 2017 RumourEval challenge is aiming to address this,[9] as well as to provide method comparison on rumor veracity and rumor stance classification. Another recent dataset is [287].

8.3.7 DISCUSSION

Even though some inroads have been made already, current methods for semantic annotation of social media streams have many limitations. Firstly, most methods address the more shallow problems of keyword and topic extraction, while ontology-based entity and event recognition do not reach the significantly higher precision and recall results obtained on longer text documents. One way to improve the currently poor automatic performance is through crowdsourcing. The ZenCrowd system [288], for instance, combines algorithms for large-scale entity linking with human input through micro-tasks on Amazon Mechanical Turk. In this way, textual mentions that can be linked automatically and with high confidence to instances in the LOD cloud are not shown to the human annotators. The latter are only consulted on hard-to-solve cases, which not only significantly improves the quality of the results, but also limits the amount of manual intervention required. We return to crowdsourcing in more detail in Section 10.2.

[9] http://alt.qcri.org/semeval2017/task8/

Another way to improve semantic annotation of social media is to make better use of the vast knowledge available on the Web of Data. Currently this is limited mostly to Wikipedia and resources derived from it (e.g., DBpedia and YAGO). One of the challenges here is ambiguity. For instance, song and album titles in MusicBrainz are highly ambiguous and include common words (e.g., Yesterday), as well as stop words (The, If) [244]. Consequently, an automatic domain categorization step might be required, in order to ensure that domain-specific LOD resources, such as MusicBrainz, are used to annotate only social media content from the corresponding domain. The other major challenges are robustness and scalability. Firstly, the semantic annotation algorithms need to be robust in the face of noisy knowledge in the LOD resources, as well as being robust with respect to dealing with the noisy, syntactically irregular language of social media. Secondly, given the size of the Web of Data, designing ontology-based algorithms which can load and query efficiently these large knowledge bases, while maintaining high computational throughput is far from trivial.

The last obstacle to making better use of Web of Data resources lies in the fairly limited lexical information available. With the exception of resources grounded in Wikipedia, lexical information in the rest is mostly limited to RDF labels. This in turn limits their usefulness as a knowledge source for ontology-based information extraction and semantic annotation. One recent strand of work has focused on utilizing the Wiktionary [289] collaboratively built, multilingual lexical resources. It is particularly relevant to analysing user-generated content, since it contains many neologisms and is updated continuously by its contributor community. For English and German, in particular, there is also related ongoing work on creating UBY [290]—a unified, large-scale, lexico-semantic resource, grounded in Wikipedia and Wordnet, and thus, indirectly, to other LOD resources as well. Another relevant strand is work on linguistically grounded ontologies [291], which has proposed a more expressive model for associating linguistic information to ontology elements. While these are steps in the right direction, further work is still required, especially with respect to building multilingual semantic annotation systems.

In addition, it is axiomatic that semantic annotation methods are only as good as their training and evaluation data. Algorithm training on social media gold standard datasets is currently very limited. For example, there are currently fewer than 10,000 tweets annotated with named entity types and events. Bigger, shared evaluation corpora from different social media genres are therefore badly needed. Creating these through traditional manual text annotation methodologies is unaffordable, if a significant mass is to be reached. Research on crowdsourcing evaluation gold standards has been limited, primarily with focus on using Amazon Mechanical Turk to acquire small datasets (e.g., tweets with named entity types) [292]. We will revisit this challenge again in Section 10.2.

In the area of sentiment analysis, researchers have investigated the problems of sentiment polarity detection, subjectivity classification, prediction through social media, and user mood profiling, however, most methods use no or very little semantics. Moreover, evaluation of opinion mining is particularly difficult for a number of methodological reasons (in addition to the lack

of shared evaluation resources discussed above). First, opinions are often subjective, and it is not always clear what was intended by the author. For example, a person cannot necessarily tell if a comment such as "I love Baroness Warsi," in the absence of further context, expresses a genuine positive sentiment or is being used sarcastically. Inter-annotator agreement performed on manually annotated data therefore tends to be low, which affects the reliability of any gold standard data produced.

Lastly, social media streams impose a number of further outstanding challenges on opinion and sentiment mining methods:

- *Relevance*: In social media, discussions and comment threads can rapidly diverge into unrelated topics, as opposed to product reviews which rarely stray from the topic at hand.

- *Target identification*: There is often a mismatch between the topic of the social media post, which is not necessarily the object of the sentiment held therein. For example, the day after Whitney Houston's death, TwitterSentiment and similar sites all showed an overwhelming majority of tweets about Whitney Houston to be negative; however, almost all these tweets were negative only in that people were sad about her death, and not because they disliked her.

- *Volatility over time*: More specifically, opinions can change radically over time, from positive to negative and vice versa. To address this problem, the different types of possible opinions can be associated as ontological properties with the classes describing entities, facts, and events, discovered through semantic annotation techniques, similar to those in [293] which aimed at managing the evolution of entities over time. The extracted opinions and sentiments can be time-stamped and stored in a knowledge base, which is enriched continuously, as new content and opinions come in. A particularly challenging question is how to detect emerging new opinions, rather than adding the new information to an existing opinion for the given entity. Contradictions and changes also need to be captured and used to track trends over time, in particular through opinion aggregation.

- *Opinion aggregation*: Another challenge is the type of aggregation that can be applied to opinions. In entity-based semantic annotation, this can be applied to the extracted information in a straightforward way: data can be merged if there are no inconsistencies, e.g., on the properties of an entity. Opinions behave differently here, however: multiple opinions can be attached to an entity and need to be modeled separately, for which we advocate populating a knowledge base. An important question is whether one should just store the mean of opinions detected within a specific interval of time (as current opinion visualization methods do), or if more detailed approaches are preferable, such as modeling the sources and strength of conflicting opinions and how they change over time. A second important question in this context involves finding clusterings of the opinions expressed in social media, according to influential groups, demographics, and geographical and social cliques. Conse-

quently, the social, graph-based nature of the interactions requires new methods for opinion aggregation.

CHAPTER 9

Applications

Semantic annotations have diverse applications, such as *semantic search*—finding documents that mention one or more concepts/instances from an ontology/linked open data; constructing social semantic user models, including demographics, user interests, and online behavior; modeling online communities; and semantic-based information visualization. All of these applications make use of the output of the earlier text processing stages, such as named entity recognition and linking, relation and term extraction, sentiment analysis, etc.

This chapter will introduce each of these applications in turn, explaining not just the basic principles of each, but also pointing to some key examples from the literature. We will conclude with a discussion on open questions and future directions.

9.1 SEMANTIC SEARCH

An in-depth introduction and review of the state-of-the-art in semantic search is beyond the scope of this book, but see [294, 295] for details. This section will only provide a high-level overview.

Semantic search over documents is about finding information that is based not just on the presence of words, but also on their meaning [296, 297]. This task is a modification of classical Information Retrieval (IR), but documents are retrieved on the basis of relevance to ontology concepts, as well as words. Nevertheless the basic assumption is quite similar—a document is characterized by the bag of tokens constituting its content, disregarding its structure. While the basic IR approach considers word stems as tokens, there has been considerable effort toward using word-senses or lexical concepts (see [298, 299]) for indexing and retrieval. In the case of semantic search, what is being indexed is typically a combination of words, ontological concepts conveying the meaning of some of these words (e.g., Cambridge is a location), and optionally relations between such concepts (e.g., Cambridge is in the UK) [296]. The latter enable somebody searching for documents about the UK to find also documents mentioning Cambridge.

However, Cambridge (as well as many other names and words) has multiple meanings, i.e., is ambiguous. The token "Cambridge" may refer to the city of Cambridge in the UK, to Cambridge in Massachusetts, the University of Cambridge, etc. Similarly, different tokens may have the same meaning, e.g., New York and the Big Apple. Therefore, semantic search tries to offer users more precise and relevant results, by using semantic annotations and external knowledge, typically encoded in ontologies and/or Linked Open Data resources.

In practice, the output of semantic annotation techniques (such as those discussed in Chapter 5) is used to allow users to find documents that mention one or more instances, classes, and/or relations. Some semantic search platforms support queries that mix free-text keywords with semantic annotations and even SPARQL queries. Most retrieval tools also provide document browsing functionality as well as search refinement capabilities. Due to the fact that documents can have hundreds of annotations (especially if every concept mention in the document is annotated), annotation retrieval on a large document collection is a very challenging task.

Annotation-based search and retrieval is different from traditional information retrieval, because of the underlying graph representation of annotations, which encode structured information about text ranges within the document. The encoded information is different from the words and inter-document link models used by Google and other search engines. Many semantic annotations also refer to ontologies via URIs. While augmented full-text indexes can help with efficient access, the data storage requirements can be very substantial, as the cardinality of the annotation sets grows. Therefore different, more optimized solutions have been investigated.

The main difference from Semantic Web search engines, such as Swoogle [300], is the focus on annotations and using those to find documents, rather than forming queries against ontologies or navigating ontological structures. Similarly, semantic-based facet search and browse interfaces, such as /facet [301], tend to be ontology based, whereas annotation-based facet interfaces (see KIM below) tend to hide the ontology and instead resemble more closely "traditional" string-based faceted search.

9.1.1 WHAT IS SEMANTIC SEARCH?

In order to understand the different kinds of semantic search tasks and approaches, it is useful to consider two aspects: (i) what is being searched; and (ii) what are the results. We discuss these in turn.

With respect to what is being searched, there are three main kinds of content to consider:

- *Documents*: This is traditional full-text search, where queries are answered on the basis of word co-occurrence in text content. For example, a query for "Cambridge university" returns all documents that contain the words Cambridge and/or university somewhere. This does not mean the results are only documents about that university. This kind of search has problems answering entity-type queries, e.g., which cities in the UK have a population of less than 100,000.

- *Ontologies and other semantic knowledge, e.g., LOD*: This is search over structured formal data, typically expressed as RDF [302] or OWL [303], and stored in a database or a semantic repository. Consequently, such formal queries are expressed in structured query languages, such as SPARQL [304] or SQL. This kind of search is often referred to as semantic search, because it uses semantics and inference to find the matching formal knowledge. In this

chapter, we will refer to this kind of search as *ontology-based search*. This kind of search is particularly suited to answering entity-type queries, such as our example above.

- *Both documents and formal knowledge*: This is what this chapter refers to as semantic search over documents, or multi-paradigm [297], or semantic full-text search [305]. This kind of search draws both on document content and on semantic knowledge, in order to answer queries such as: "flooding in cities in the UK" or "flooding in places within 50 miles of Sheffield." In this case, information about which cities are in the UK or within 50 miles of Sheffield is the result of ontology-based search (e.g., against DBpedia or GeoNames). Documents are then searched for the co-occurrence of the word "flooding" and the matching entities from the ontology-based search. In other words, what is being searched here is the document content for keywords, the index of semantically annotated entities that occur within these documents, and the formal knowledge.

With respect to the results returned by searches, there are four main kinds:

- *Documents*: The search returns a ranked list of documents, typically displayed with their title and, optionally, some additional metadata (e.g., author). This kind of result is typically produced by full-text searches, although some also include snippets.

- *Documents + highlighted snippets*: In addition to document titles, one or more snippets are returned, where the query hits are highlighted, in an attempt to make it apparent to users why this document is relevant to their query. Semantic search systems typically return matching documents in this way, e.g., the KIM system [296], Mímir [297], and Broccoli [306].

- *Information summary*: This is a human-readable rendering of formal knowledge, returned by ontology-based searches for entities. For instance, a search in Google for "Tony Blair" would display on the right a summary showing several photos and basic facts, such as date of birth, generated automatically from their formal knowledge graph representation [307].

- *Structured results*: Ontology-based searches, which result in a list of entities, are often shown in a structured form, e.g., a list of UK city names. See for example the KIM entity searches[1] [296] or Broccoli [306].

9.1.2 WHY SEMANTIC FULL-TEXT SEARCH?

As argued by [305], full-text search works well for precision-oriented searches, when the relevant documents contain the keywords that describe the user need. However, there are many cases when recall is paramount and also implicit knowledge is needed in order to answer parts of the query. A frequent class of such queries is the entity-based one, e.g., "plants with edible leaves" [305]. In this case, most likely there is no single document containing the answer and, furthermore, documents

[1] http://ln.ontotext.com/KIM

typically refer to the specific plants by name (e.g., broccoli), instead of using the generic term "plants."

Environmental science is another example where there is a strong need to go beyond keyword-based search [308, 309]. The British Library carried out a survey of environmental science researchers, and analysed the kinds of information needs they struggled to satisfy through keyword search [310]. The top requirement was for geographically specific queries, including proximity search (e.g., "documents about flooding within 50 miles of Sheffield") and implied locations (e.g., the query "documents about flooding in South West England" needs to return a document about flooding in Exeter, even though South West England is not mentioned explicitly).

A further example is patent search [295, 311], where recall is crucial, since failure to find pre-existing, relevant patents may result in legal proceedings and financial losses. Examples of hard-to-find information using keywords alone are searches for references to papers cited in a specific section of the patent, and also searches for measurements and quantities (e.g., in chemical patents). Measurements in particular are numeric and can show great variation—the same value can be expressed using different measurement systems, e.g., inches or centimeters, or different multipliers even when using the same measurement system, e.g., mm, cm, or meters.

9.1.3 SEMANTIC SEARCH QUERIES

Since semantic search queries need to contain both text-based keywords and formal SPARQL-like queries over the ontology, they are often referred to as hybrid queries. The Semplore system [312], for example, uses conjunctive hybrid query graphs, similar to SPARQL, but enhanced with a "virtual" concept called keyword concept W. A similar approach has been taken in the Broccoli system [306], which has a special *occurs – with* relation, the value of which is the free text keyword.

Mímir[2] [295] has an even richer query language, which also supports the inclusion of linguistic annotations in queries. For example, a Mímir query "PER says" will return documents where an entity of type Person is followed by the keyword "says." Morphological variations for keywords are also supported (e.g., "PER root:say"), as are distance restrictions (e.g., "Person [0..5] root:say" which matches text with up to 5 words separating the two components, such as "Sebastian James of Dixons Group said"). Additional semantic restrictions based on knowledge from the ontology are expressed by adding a SPARQL query. For example, this query is for documents mentioning people born in Sheffield:

```
{Person sparql = "SELECT ?inst
    WHERE { ?inst :birthPlace <http://dbpedia.org/resource/Sheffield>}"}
```

[2]A set of example queries and several test Mímir indexes are available for experimentation at: http://demos.gate.ac.uk/mimir/

9.1.4 RELEVANCE SCORING AND RETRIEVAL

In the context of semantic full-text search, [313] propose a modification of tf.idf, based on the frequency of occurrence of instances from the semantic annotations in the document collection. They also combine semantic similarity with a standard keyword-based similarity for ranking, in order to cater for cases when there are no sufficiently relevant semantic annotations.

The Mímir semantic full-text search framework [295] supports different ranking functions, and new ones can easily be integrated. In addition to tf.idf, it also implements ranking based on hit length and the BM25 algorithm.

The CE^2 system goes one step further and uses a graph-based approach to compute the ranking of the hybrid search results [314]. The graph structure comes from the formal semantic knowledge.

With respect to ranking individuals returned via knowledge base search, [315] propose ObjectRank—a PageRank-based approach.

9.1.5 SEMANTIC SEARCH FULL-TEXT PLATFORMS

We describe below some of the major semantic search frameworks/prototypes; many others are also available.

GoNTogle [316] is a search system that provides keyword, semantic, and hybrid search over semantically annotated documents. The semantic search replaces keywords with ontological classes. Results are obtained based on occurrences of the ontological classes from the query within the annotations associated with a document. Finally, the *hybrid search* comprises a standard boolean AND or OR operation between the result sets produced by a keyword search and a semantic search. The only type of annotation supported is associating an ontology class with a document segment. Another similar system is Semplore [312] which uses conjunctive hybrid query graphs, similar to SPARQL, but enhanced with a "virtual" concept called keyword concept W. However, both GoNTogle and Semplore do not have support for searches over document structure, nor for searches over other types of linguistic annotations.

The **Broccoli** system [306] also provides a user interface for building queries, combining text-based and semantic constraints (encoded as entity mentions in the input text, with URIs). The association between text and semantics is encoded by means of the *occurs-with* relation which is implied whenever mentions of words and ontological entities occur within the same *context*. The *contexts* are automatically extracted at indexing time, and rely mainly on shallow syntactic analysis of the document and extraction of syntactic dependency relations. The *occurs-with* relation provides access to the underlying phrase structure of the input document. However, the system is designed to only use this particular relation, so indexing other kinds of document structure (e.g., abstract, sections) is likely to prove problematic. Consequently, there is no support for richer linguistic annotations, such as part-of-speech or morphology, document metadata, or structural search other than based on co-occurrences within *contexts*.

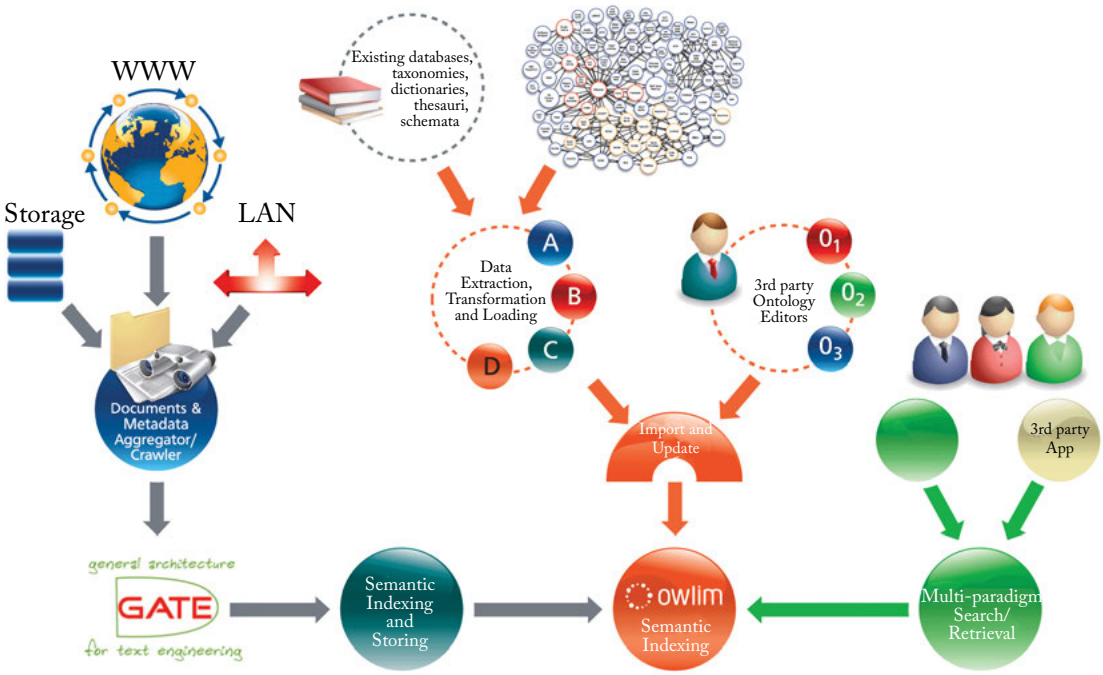


Figure 9.1: KIM architecture.

The **KIM** (Knowledge and Information Management) platform [296, 317] was among the first systems to implement semantic search, both over RDF knowledge bases via SPARQL, and over semantically annotated document content, including hybrid queries mixing keywords and semantic restrictions. KIM has a number of user interfaces for semantic search and browsing and can be customized easily for specific applications. It is freely available for research use from `http://www.ontotext.com/kim/getting-started/download`.

KIM is an extendible platform for knowledge management, which offers tools for semantic annotation, indexing, and semantic-based search (referred to as multi-paradigm search in KIM). Figure 9.1 shows KIM's architecture, which also includes a web crawler for content harvesting; a knowledge ETL component which interfaces to thesauri, dictionaries, and LOD resources; and a set of web-based user interfaces for entity-based and semantic-based full text search (see Section 9.1.6 for details on the KIM faceted search).

Semantic annotation in KIM is based on GATE's NLP tools. The essence of KIM's semantic annotation is the recognition of named entities with respect to the KIM ontology. The entity instances all bear unique identifiers that allow annotations to be linked both to the entity type and to the exact individual in the instance base. For new (previously unknown) entities, new identifiers are allocated and assigned; then minimal descriptions are added to the semantic repos-

itory. The annotations are kept separately from the content, and an API for their management is provided.

KIM can also use Linked Data ontologies for semantic annotation and search. At present it has been tested with DBPedia, Geonames, Wordnet, Musicbrainz, Freebase, UMBEL, Lingvoj, and the CIA World Factbook. Those datasets are preprocessed and loaded to form an integrated dataset of about 1.2 billion explicit statements. Forward-chaining is performed to materialize another 0.8 billion implicit statements.

GATE **Mímir**[3] [295] is an integrated semantic search framework, which offers indexing and search over full text, document structure, document metadata, linguistic annotations, and any linked, external semantic knowledge bases. It supports hybrid queries that arbitrarily mix full-text, structural, linguistic, and semantic constraints. A key distinguishing feature from previous work are the containment operators, that allow flexible creation and nesting of full-text, structural, and semantic constraints.

Figure 9.2 shows the Mímir semantic query UI. The goal is to find documents, mentioning locations in the UK, where the population density is more than 500 people per square km. The knowledge about population density is coming from DBpedia. The documents being searched are metadata descriptions of government reports on climate change and flooding, created by the British Library as part of the EnviLOD project.[4]

The high-level concept behind Mímir is that a document collection is processed with NLP algorithms, typically including semantic annotation using Linked Open Data accessed via a triple store, such as OWLIM [318] or Sesame. The annotated documents are then indexed in Mímir, together with their full-text content, document metadata, and document structure markup (the latter can also be discovered automatically via the NLP tools). At search time, the triple store is used as a source of implicit knowledge, to help answer the hybrid searches that combine full-text, structural, and semantic constraints. The latter are formulated using a SPARQL query, executed against the triple store.

Mímir uses inverted indexes for indexing the document content (including additional linguistic information, such as part-of-speech or morphological roots), and for associating instance of annotations with the position in the input text where they occur. The inverted index implementation used by Mímir is based on MG4J [319]. Beside document text, the other main kind of data are the structural and NLP-generated annotations. In Mímir both kinds are represented in the same data structure, comprising a start and end position, an annotation type (e.g., Location), and an optional set of attributes (called features in the GATE framework).

Mímir is highly scalable: in one application 150 million web pages were indexed successfully, using two hundred Amazon EC2 Large Instances running for a week to produce a federated index [293]. Since Mímir runs on GateCloud.net [320], building Mímir semantic indexes on the Amazon cloud is straightforward.

[3] http://gate.ac.uk/mimir/
[4] http://gate.ac.uk/projects/envilod

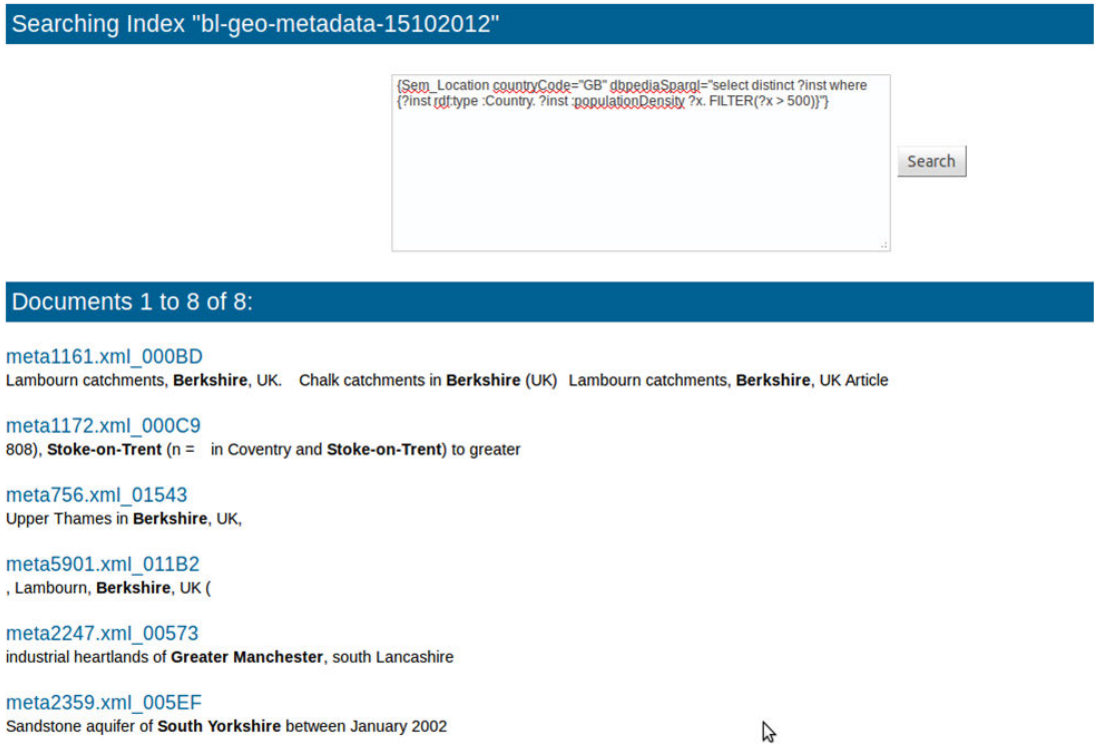


Figure 9.2: Mímir's semantic search UI showing a formal query, the retrieved documents, and short text snippets showing in bold the matched locations.

9.1.6 ONTOLOGY-BASED FACETED SEARCH

As discussed earlier, KIM has a comprehensive set of web browser–based UIs for semantic search. This includes ontology-driven faceted search, where the user can select one or more instances (visualized with their RDF labels, but found via their URIs) and obtain the documents where these co-occur. Timeline and entity-centric views are also supported.

Figure 9.3 shows a case where the user is searching for patents mentioning amoxicillin and gentamicin. This example is taken from the ExoPatent online KIM demo,[5] which uses the FDA Orange Book (23,000 patented drugs) and Unified Medical Language System (UMLS—a database of 370,000 medical terms) to annotate documents with semantic information. The demo runs on a small set of 40,000 patents. ExoPatent supports semantic search for diseases, drug names, body parts, references to literature and other patents, numeric values, and ranges.

[5]Available online at `http://exopatent.ontotext.com`.

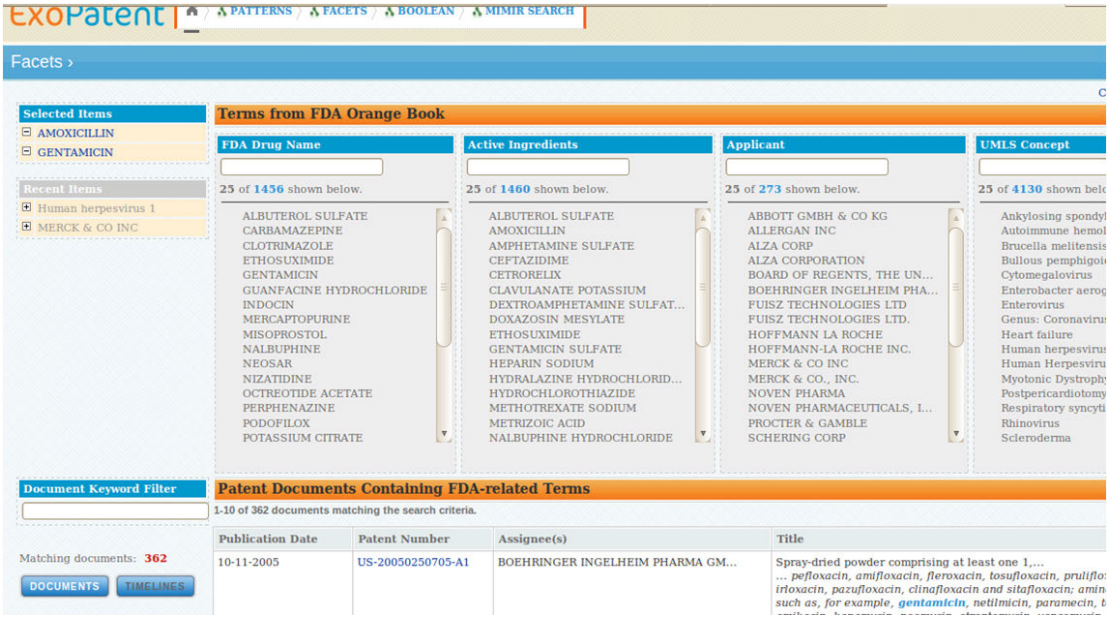


Figure 9.3: KIM's entity-based faceted search UI.

In the faceted search UI, as new entities are selected as constraints (see left column), the number of matching documents is updated dynamically. Optional keyword constraints can also be specified in the keyword filter field on the left. At the bottom of the figure, one can see the titles of the retrieved documents and some relevant content from them. The titles are clickable, in order to view the full document content and the semantic annotations within it. The entities/terms listed in the entity columns (drug name, ingredients, applicant, and UMLS concept) are also updated to show only entities co-occurring with the already selected entity constraints.

The Broccoli system mentioned earlier has a similar interactive query building UI, which updates dynamically as the user is typing concepts or keywords to search for. The documents being searched are Wikipedia articles, indexed with classes and instances from the YAGO ontology. Figure 9.4 shows an example query for documents mentioning UK cities, which also contain the keyword "flood." The semantic query is displayed as a graph on top, making explicit the relations between the concepts searched for. Keywords have a special relation "occurs-with," whereas all other semantic relations come from the YAGO ontology. As the user starts typing a query term (e.g., city), the lists of matching classes, instances, and relations on the left are updated dynamically. Once a query term is selected, only relations applicable to this class are shown in the list of relation candidates. Due to the entity-centric queries, the result list is structured as a list of entities, where relevant information from the YAGO ontology is provided for each entity returned, as well as documents from Wikipedia about this entity which also contain the given keyword(s).

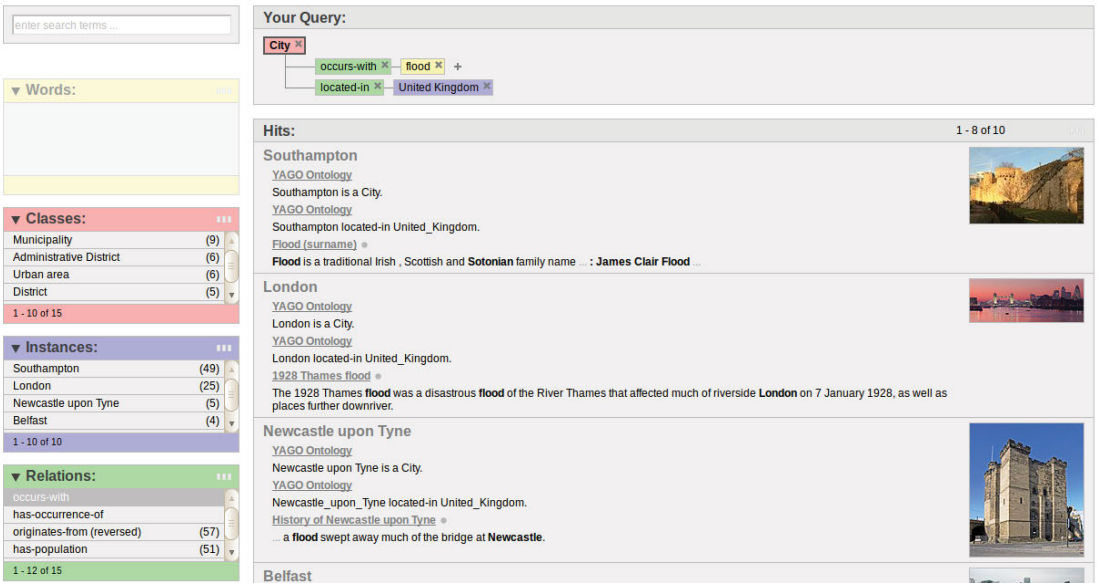


Figure 9.4: The Broccoli interactive query building UI.

9.1.7 FORM-BASED SEMANTIC SEARCH INTERFACES

One of the challenges faced by semantic search interfaces, especially in subject-specific cases, is to indicate to users what they can search for. A form-based interface makes this explicit, in a manner similar to the facet-based UIs discussed above.

An example form-based interface is shown in Figure 9.5 from the EnviLOD UI [308], which was developed as a user-friendly semantic search front end to a Mímir index of environmental science documents, terms, and LOD entities.

There is a keyword search field, complemented with optional semantic search constraints, through a set of inter-dependent drop-down lists. In the first list, users can search for specific entity types (Locations, Organizations, Persons, Rivers, Dates), and can also specify constraints on document-level attributes. More than one semantic constraint can be added, through the plus button, which inserts a new row underneath the current row of constraints.

For example, if "Location" is chosen as a semantic constraint, then further constraints can be specified by choosing an appropriate property constraint, as shown in the figure. "Population" allows users to pose restrictions on the population number of the locations that are being searched for. Similar numeric constraints can be imposed on the latitude, longitude, and population density attribute values.

Restrictions can also be imposed in terms of location name or the country it belongs to. For string-value properties, if "is" is chosen from the third list instead of "none," then the value

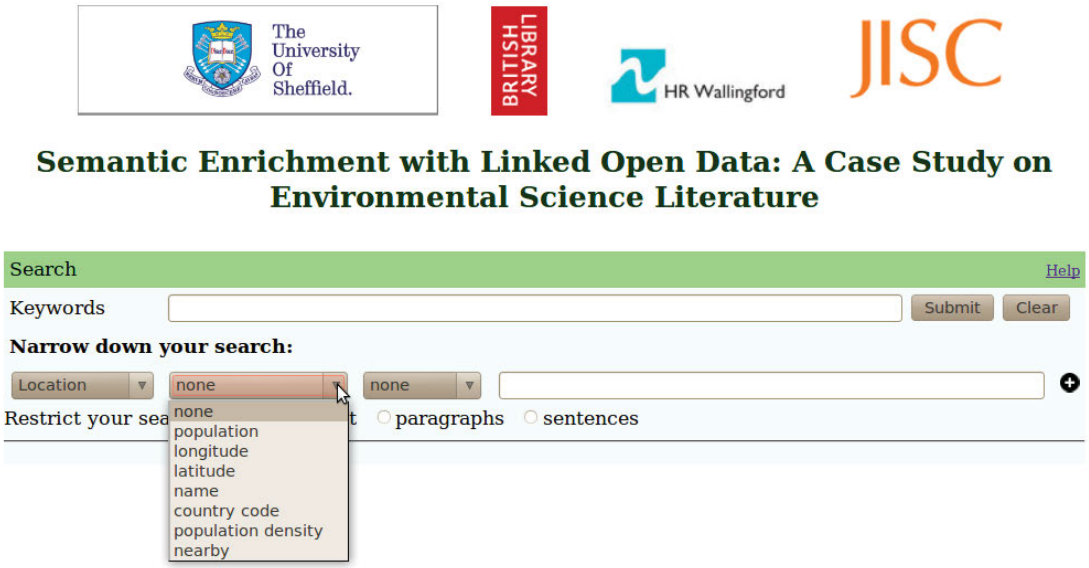


Figure 9.5: The EnviLOD semantic search UI.

must be exactly as specified (e.g., Oxford), whereas "contains" triggers sub-string matching, (e.g., Oxfordshire is matched as a location name containing Oxford). In this way, a user searching for documents mentioning locations with a name containing "Oxford" will be shown not only documents mentioning Oxford explicitly, but also documents mentioning Oxfordshire and other locations in Oxfordshire (e.g., Wytham Woods, Banbury). In the latter case, the knowledge from DBpedia and GeoNames will be used to identify which other locations are in Oxfordshire, in addition to Oxford.

One problem with an EnviLOD-style UI is that it hides from users information about what instances of these classes occur in the indexed document collection (e.g., which UK counties are mentioned). In order to provide such high-level entity-based overviews of the documents, one approach is to list all instances, for each class, as done in the KIM and Broccoli interfaces.

An alternative is to use tag clouds and other visualizations of entity co-occurrences. Mímir has recently been extended with such a user interface, called GATE Prospector (see Figure 9.6). The top half of the UI shows ontology classes and instances (UMLS in this example) and the user selects the desired ones. Additional search restrictions could be imposed via document metadata filters. The bottom half of the picture shows the matching instances (terms in the case of UMLS) as well as the number of times they occur in the document collection. A frequency-based term cloud is also shown. The set of terms/instances can be saved for later use, e.g., to generate entity/term co-occurrence visualizations.

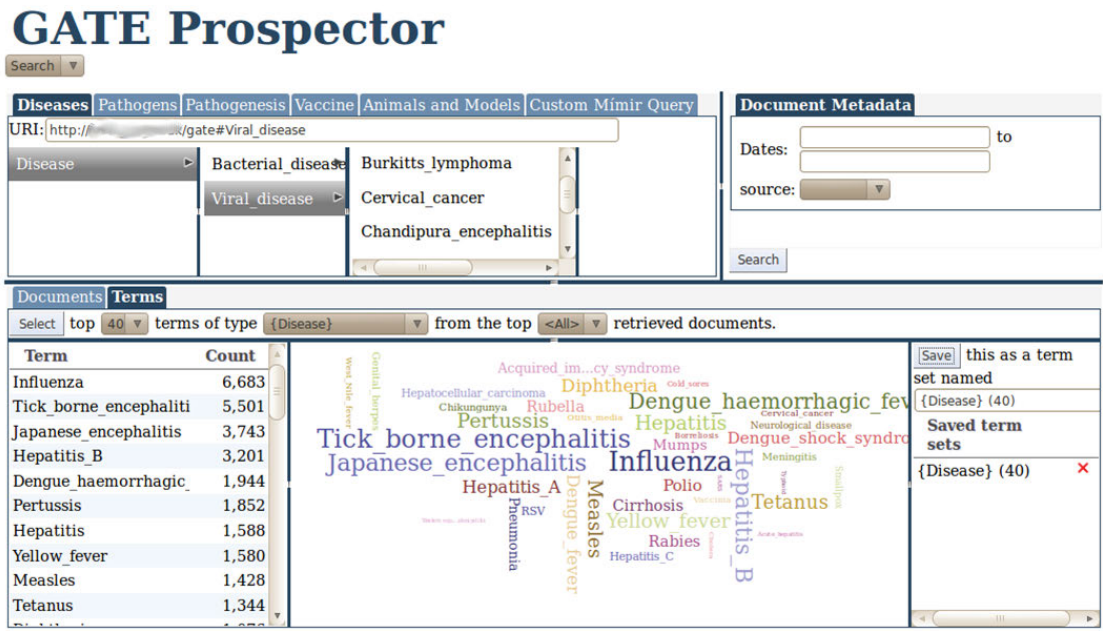


Figure 9.6: The GATE prospector semantic search UI.

Figure 9.7 shows an example of co-occurrence visualization, where the most frequently mentioned instances of diseases are plotted against the most frequently mentioned instances of pathogens. Examples from other domains include plotting which sentiment terms co-occur most frequently with which political parties or politicians, given a large collection of tweets about an election.

9.1.8 SEMANTIC SEARCH OVER SOCIAL MEDIA STREAMS

Searching social media streams differs significantly from web searches [321] in a number of important ways. Firstly, users search message streams, such as Twitter, for temporally relevant information, and are mostly interested in people. Secondly, searches are used to monitor Twitter content over time, and can be saved as part of user profiles. Thirdly, Twitter search queries are significantly shorter and results include more social chatter, whereas web searches look for facts. Coupled with the short message length, noisy nature, and additional information hidden in URLs and hashtags, these differences make traditional keyword-based search methods sub-optimal on media streams.

A comparison of social media monitoring tools conducted in October 2014 by Ideya Ltd[6] shows that there are at least 245 tools for social media monitoring available, of which 197 are

[6] http://ideya.eu.com/reports.html

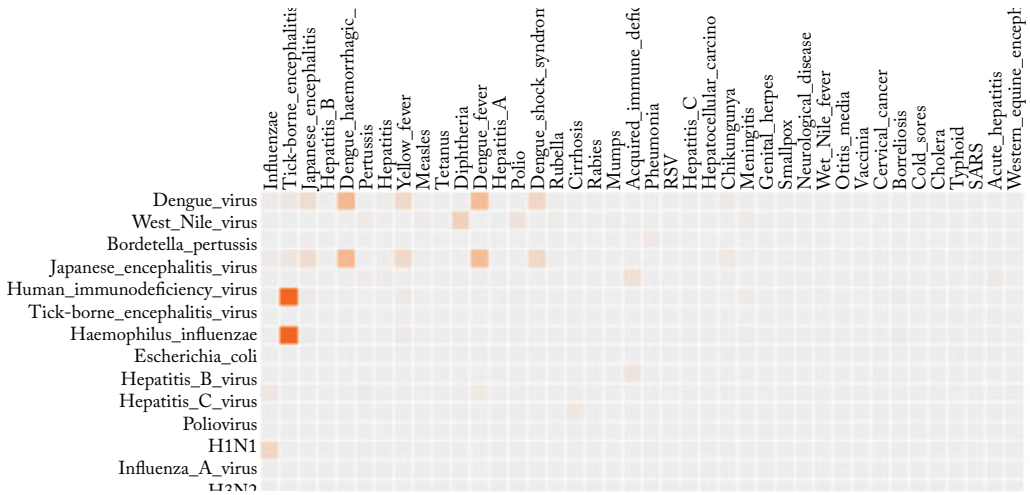


Figure 9.7: GATE prospector: instance/term co-occurrence view.

paid, with the remainder free or using a freemium model. Most of the free tools, at least, do not allow the in-depth and customizable analysis ideally required. Published research has principally concentrated on number-crunching exercises based on topic and entity identification by hashtag, simple keyword, or easily available Twitter metadata, such as author name, language, number of retweets, and so on [322–326]. While some of these methods do involve more complex language processing techniques, these typically comprise simple off-the-shelf sentiment analysis tools such as SentiStrength [214] and SentiWordNet [327] and/or generic basic entity and topic recognition tools such as DBpedia Spotlight [115], or core open source NLP tools such as ANNIE [328], and are not adapted to the domain and task. This section will focus therefore on recent work on semantic search, which aims to address these challenges.

The TREC 2011 Microblog track[7] has given impetus to research by providing a set of query topics, a time point, and a corpus of 16 million tweets, a subset of which was hand-annotated for relevance as a gold standard. In addition to the widely used keyword-based and tweet syntax features (e.g., whether it contains a hashtag), Tao et al. [329] experimented with entity-based semantic features produced by DBpedia Spotlight, which provided significantly better results.

The Twarql system [330] generates RDF triples from tweets, based on metadata from the tweets themselves, as well as entity mentions, hashtags, and URLs [221]. These are encoded using standard Open Data vocabularies (FOAF, SIOC) (see Section 8.2) and can be searched through SPARQL queries. It is also possible to subscribe to a stream of tweets matching a complex semantic query, e.g., what competitors are mentioned with my product (Apple iPad in their use

[7] http://sites.google.com/site/trecmicroblogtrack/

case). At the time of writing, Twarql has not been evaluated formally, so its effectiveness and accuracy are yet to be established.

Abel et al. propose an adaptive faceted search framework for social media streams [331]. It uses semantic entity annotations by OpenCalais, coupled with a user model (see Section 9.2.1), in order to create and rank facets semantically. Keyword search and hashtag-based facets are used as the two baselines. The best results are achieved when facets are personalized, i.e., ranked according to which entities are interesting for the given user (as coded in their entity-based user model). Facet ranking also needs to be made sensitive to the temporal context (essentially the difference between query time and post timestamp).

There is also a GATE-based framework for analysing and searching over large volumes of social media content. The real-time analytics framework comprises the GATE semantic annotation components discussed in the earlier chapters, the Mímir semantic search framework, and a dynamic result aggregation component. Exploratory search and sense-making are supported through information visualization interfaces, such as co-occurrence matrices, term clouds, treemaps, and choropleths. There is also a Prospector-based interactive semantic search interface, where users can save, refine, and analyze the results of semantic search queries over time. Practical use of the framework in real-time and at scale has been demonstrated on analysing tweets from UK politicians and the public's response to them in the run up to the 2015 UK general election, and analysing over 64 million tweets related to the 2016 UK referendum on EU membership (Brexit).

The GATE-based framework can perform all the steps in the analytics process: data collection, semantic annotation, indexing, search, and visualization. In the data collection process, user accounts and hashtags can be followed through the Twitter "statuses/filter" streaming API. This produces a JSON file which is saved for later processing. The tweet stream can also (optionally) be analysed as it comes in, in near real-time, and the results indexed for aggregation, search, and visualization. Twitter's own "hosebird" client library is used to handle the connection to the API, with auto reconnection and backoff-and-retry.

In the case of **non-live processing**, the collected JSON is processed using the GATE Cloud Parallelizer (GCP) to load the JSON files into GATE documents (one document per tweet), annotate them, and then index them for search and visualization in the GATE Mímir framework [295]. GCP is a tool designed to support the execution of GATE pipelines over large collections of millions of documents, using a multi-threaded architecture.[8] GCP tasks or batches are defined using an extensible XML syntax, describing the location and format of the input files, the GATE application to be run, and the kinds of outputs required. A number of standard input and output data format handlers are provided (e.g., XML, JSON), but all the various components are pluggable, so custom implementations can be used if the task requires it. GCP keeps track of the progress of each batch in a human- and machine-readable XML format, and is designed so

[8]For more information about GCP, see `https://gate.ac.uk/gcp/`.

that if a running batch is interrupted for any reason, it can be re-run with the same settings and GCP will automatically continue from where it left off.

In cases where **real-time live stream analysis** is required, the Twitter streaming client is used to feed the incoming tweets into a message queue. A separate semantic annotation process (or processes) then reads messages from the queue, analyzes them, and pushes the resulting annotations and text into Mímir. If the rate of incoming tweets exceeds the capacity of the processing side, more instances of the message consumer are launched across different machines to scale the capacity.

The live processing system is made up of several distinct components:

- The *collector* component receives tweets from Twitter via their streaming API and forwards them to a reliable messaging queue. It also saves the raw JSON of the tweets in backup files for later re-processing if necessary.

- The *processor* component consumes tweets from the message queue, processes them with the GATE analysis pipeline and sends the annotated documents to Mímir for indexing.

- Mímir receives the annotated tweets and indexes their text and annotation data, making it available for searching after a short (configurable) delay.

Once tweets are annotated semantically and stored in Mímir for searching, we can use Prospector to query and visualize the semantic search results. In this example, two sets of semantic annotations (political topics vs. UK political parties in this case) are mapped to the two dimensions of a matrix, while the color intensity of each cell conveys co-occurrence strength. The matrix can be re-ordered by clicking on any row/column, which sorts the axis according to the association strength with the clicked item. This example demonstrates the 10 topics most frequently talked about in the run-up to the UK elections in 2015 by the 10 most frequent groups of politicians tweeting, where a group represents a political party and a category (MP or Candidate).[9]

The underlying Mímir query which identifies which topics were mentioned by which party in the election tweets is:

```
{DocumentAuthor author_party =
"Green Party"}| OVER
{Topic theme = "uk_economy"}
```

The information about which party the tweet author belongs to and what terms are contained within each tweet, is added automatically from DBpedia during the semantic annotation phase.

[9]"SNP Other" denotes the odd case where the leader of the SNP party was not an MP or candidate, but was still interesting enough for us to follow. "Other MP" denotes MPs from the minor political parties.

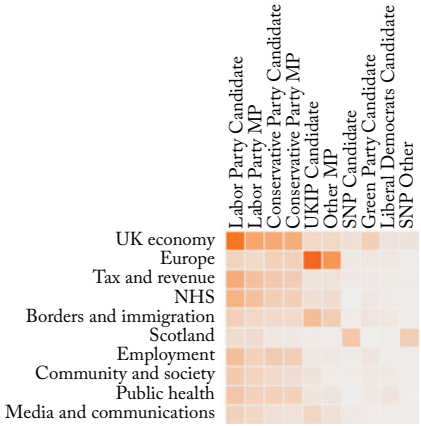


Figure 9.8: Prospector's topic to party candidate co-occurrence matrix.

9.2 SEMANTIC-BASED USER MODELING

Another application area of Semantic Web research that has made heavy use of NLP techniques is *semantic user/community modeling*, e.g., [134, 332]. A detailed overview of user modeling for the Semantic Web is beyond the scope of this chapter, but see [333].

In more detail, a user model (UM) is a knowledge resource containing explicit semantic information about various aspects of the user, which is available *a priori* (e.g., from metadata in a Facebook profile) or is inferred automatically from user behavior, user-generated content, social networks, or other sources. Typically, NLP methods, such as entity recognition and linking, are used for the latter task.

The rationale for automatically deriving ontology-based user models from social data is that they form the basis of semantic-based Personal Information Management (PIM) and other similar applications. In particular, PIM work originated in research on the social semantic desktop [334], where information from the user's computer (e.g., email, documents) is analysed with NLP methods and used to derive models of the user.

9.2.1 CONSTRUCTING SOCIAL SEMANTIC USER MODELS FROM SEMANTIC ANNOTATIONS

Among the various kinds of social media, folksonomies have probably received most attention from researchers studying how semantic models of user interactions and interests can be derived from user-generated content. Many approaches focused on exploring the social and interaction graphs, using techniques from social network analysis (e.g., [335]). In this section, however, we are concerned with methods that discover and exploit the semantics of textual tags instead (including hashtags), as well as semantic-based user modeling research on social media.

Based on the kinds of semantic information used, methods can be classified as follows.

- Bag of words ([336]).
- Semantically disambiguated entities: mentioned by the user (e.g., [134, 337]) or from a linked longer web document (e.g., [134]).
- Topics: Wikipedia categories (e.g., [134, 338]), latent topics (e.g., [339]), or tag hierarchies (e.g., [340]). One solution to modeling tag semantics more explicitly is to ground tags into WordNet and then using WordNet-based semantic similarity measures to derive the semantic relatedness of folksonomy tags [341].

This is typically supplemented with more quantitative social network information (e.g., how many connections/followers a user has [231]) and interaction information (e.g., post frequency [232], average number of posts per thread [231]).

Discovering User Demographics

Discovering user demographics is a critical task in constructing user models from semantically annotated social media. Every Twitter user has a profile which reveals some details of their identity. The profile is semi-structured, including a textual bio field, a full name, the user's location, a profile picture, a time zone and a homepage URL (most of these are optional and often empty). The user's attributes can be related to the content of their posts, for example their physical location can determine to a degree the language they use [342] or the events on which they comment [343].

One of the applications of NLP techniques has been in deriving user demographics information, when it is not readily available in social media profiles. One commonly addressed task is classifying users as male or female based on the text of their tweets, their description fields, and their names, e.g., [344]. The authors report better-than-human accuracy, compared to a set of annotators on Mechanical Turk. A general framework for user classification has also been developed, which can learn to automatically discover political alignment, ethnicity, and fans of a particular business [345].

Another important dimension is automatically locating Twitter users, by analysing the content of their posts and user profile.[10] Methods typically use NLP techniques to analyze the textual content produced by the user and infer their location based on features, such as mentions of local place names [346] and use of local dialect. In the work of [342, 347], region-specific terms and language that might be relevant to the geolocation of users were discovered automatically. A classification approach is devised in [348] that also incorporates specific mentions of places near to the user. One disadvantage to this method is the fact that someone might be writing about a popular global event which is of no relevance to their actual location. Another is that users might take deliberate steps to hide their true location by alternating the style of their posts or not referencing local landmarks.

[10]Only around 36% of users actually filled in their location field in their profile with a valid location as specific as their nearest city [342].

Using Semantic Annotations to Derive User Interests

Another intensely researched area of semantic-based user modeling is that of deriving implicit user interests, by using term and entity recognition techniques, as well as topic models. For instance, Abel et al. [134] have used semantic annotation tools to derive automatically entity- and topic-based user profiles. The entity-based profile for a given user is modeled as a set of weighted entities, where the weight each entity e is computed based either on the number of user tweets that mention e, or based on frequency of entity occurrences in the tweets, combined with the related news articles (which are identified in an earlier, linking step). Topic-based profiles are defined in a similar fashion, but represent higher-level Wikipedia categories (e.g., sports, politics). Both entities and topics are identified using OpenCalais (see Section 5.4).

Kapanipathi et al. [337] similarly use semantic annotations to derive user interests (entities or concepts from DBpedia), weighted by strength (calculated on the basis of frequency of occurrence). They also demonstrate how interests can be merged based on information from different social media (LinkedIn, Facebook, and Twitter). Facebook likes and explicitly stated interests in LinkedIn and Facebook are combined with the implicit interest information from the tweets. The Open Provenance Model[11] is used to keep track of interest provenance.

A similar entity- and topic-based approach to modeling user interests is proposed by Michelson and Macskassy [130] (called Twopics). All capitalized, non-stop words in a tweet are considered as entity candidates and looked up against Wikipedia (page titles and article content). A disambiguation step then identifies the Wikipedia entity which matches best the candidate entity from the tweet, given the tweet content as context. For each disambiguated entity, the sub-tree of Wikipedia categories is obtained. In a subsequent, topic-assignment step, all category sub-trees are analysed to discover the most frequently occurring categories, which are then assigned as user interests in the topic-based profile. The authors also argue that such more generic topics, generated by leveraging the Wikipedia category taxonomy, are more appropriate for clustering and searching for users than the term-based topic models derived using bag-of-words or LDA methods.

Capturing User Behavior

As demonstrated above, user behavior is key to understanding interactions in social media. In this section we focus primarily on approaches which utilise automatically derived semantics, in order to classify user behavior.

In the case of online forums, the following user behavior roles have been identified [349]: *elitist*, *grunt*, *joining conversationalist*, *popular initiator*, *popular participant*, *supporter*, *taciturn*, and *ignored*. For social tagging systems, researchers [350] have classified users according to their tagging motivation, into *categorizers* and *describers*. In Twitter, the most common role distinction is drawn on the basis of tweet content, and users are classified into *meformers* (80% of users) and *informers* (20% of users) [263].

[11] http://openprovenance.org

In order to assign behavior roles in online forums automatically, Angeletou et al. [231] create skeleton rules in SPARQL that map semantic features of user interaction to a level of behavior (high, medium, and low). These levels are constructed dynamically from user exchanges and can be altered over time, as the communities evolve. User roles, contexts, and interactions are modeled semantically through the User Behavior Ontology (see Section 8.2) and are used ultimately to predict the health of a given online forum.

The problem of characterizing Twitter user behavior, based on the content of their posts has yet to be fully explored. [237] generated keyphrases for users with the aid of topic modeling and a PageRank method. Similarly, [234] use a combination of POS filtering and TextRank to discover tags for users. It should also be noted that while [263] went some way toward categorizing user behavior and tweet intention, their method is not automatic and it remains unclear whether or not similar categories could be assigned by a classifier.

9.2.2 DISCUSSION

With respect to tweets, automatically derived user interests could be separated into "global" ones (based on the user's tweets on trending topics) vs. "user-specific" (topics which are of more personal interest, e.g., work, hobby, friends). Further work is required on distinguishing globally interesting topics (e.g., trending news) from interests specific to the given user (e.g., work-related, hobby, gossip from a friend, etc.). In other words, we need to go beyond using semantic annotation to profile users automatically, and toward also capturing rationale and provenance.

What is interesting to a user also ties in with user behavior roles (see Section 9.2.1). In turn, this requires more sophisticated methods for automatic assignment of user roles, based on the semantics of posts, in addition to the current methods based primarily on quantitative interaction patterns.

Lastly, another challenging question is how to go beyond interest-based models and interaction-based social networks. For instance, Gentile et al. [351] have demonstrated how people's expertise could be captured from their email exchanges and used to build dynamic user profiles. These are then compared with each other, in order to derive automatically an expertise-based user network, rather than one based on social interactions. Such an approach could be extended and adapted to blogs (e.g., for discovery and recommendation of blogs), as well as to information sharing posts in Twitter and LinkedIn streams.

9.3 FILTERING AND RECOMMENDATIONS FOR SOCIAL MEDIA STREAMS

The unprecedented rise in the volume and perceived importance of social media content has resulted in individuals starting to experience information overload. In the context of Internet use, research on information overload has shown already that high levels of information can lead to ineffectiveness, as "a person cannot process all communication and informational inputs" [352].

Consequently, researchers have studied semantic-based methods for information filtering and content recommendation of social media streams. Since Facebook timelines are predominantly private, the bulk of work has so far focused on Twitter.

As discussed in [336], social media streams are particularly challenging for recommender methods, and different from other types of documents/web content. Firstly, relevance is tightly correlated with recency, i.e., content stops being interesting after just a few days. Secondly, users are active consumers and generators of social content, as well as being highly connected with each other. Thirdly, recommenders need to strike a balance between filtering out noise and supporting serendipity/knowledge discovery. Lastly, interests and preferences vary significantly from user to user, depending on the volume of their personal stream; what they use social media for and how they use it (see Section 9.2.1 on user roles); and user context (e.g., mobile vs. tablet, work vs. home).

Chen et al. [336] and Abel et al. [353] focused on recommending URLs to Twitter users, since it is a common information sharing task. The approach of Chen et al. is based on a bag-of-words model of user interests, based on the user tweets, what is trending globally, and the user's social network. URL topics are modeled similarly as a word vector, and tweet recommendations are computed using cosine similarity.

Abel et al. [353] improve on this approach by using semantic annotation tools to derive semantic-based user interest models (see Section 9.2.1 for details). They also capture more in-depth semantics through analysing hashtag semantics, replies, and, crucially, by modeling temporal dynamics of user interests.

Recently, Chen et al. [354] extended their work toward recommending interesting conversations, i.e., threads of multiple messages. The rationale comes from the widespread use of Facebook and Twitter for social conversations [263], coupled with the difficulties that users experience with following these conversations over time, in Twitter in particular. Conversations are rated based on thread length, topic (using bag-of-words as above) and tie-strength (higher priority for content from tightly connected users). The shallow nature of the approach leaves ample scope for improvement through using semantic annotations and other NLP techniques discussed in this book.

9.4 BROWSING AND VISUALIZATION OF SOCIAL MEDIA STREAMS

The main challenge in browsing and visualization of high-volume stream media is in providing a suitably aggregated, high-level overview. Timestamp-based list interfaces that show the entire, continuously updating stream (e.g., the Twitter timeline-based web interface) are often impractical, especially for analysing high-volume, bursty events. For instance, during the royal wedding in 2011, tweets during the event exceeded 1 million. Similarly, monitoring long running events, such as presidential election campaigns, across different media and geographical locations, is equally complex.

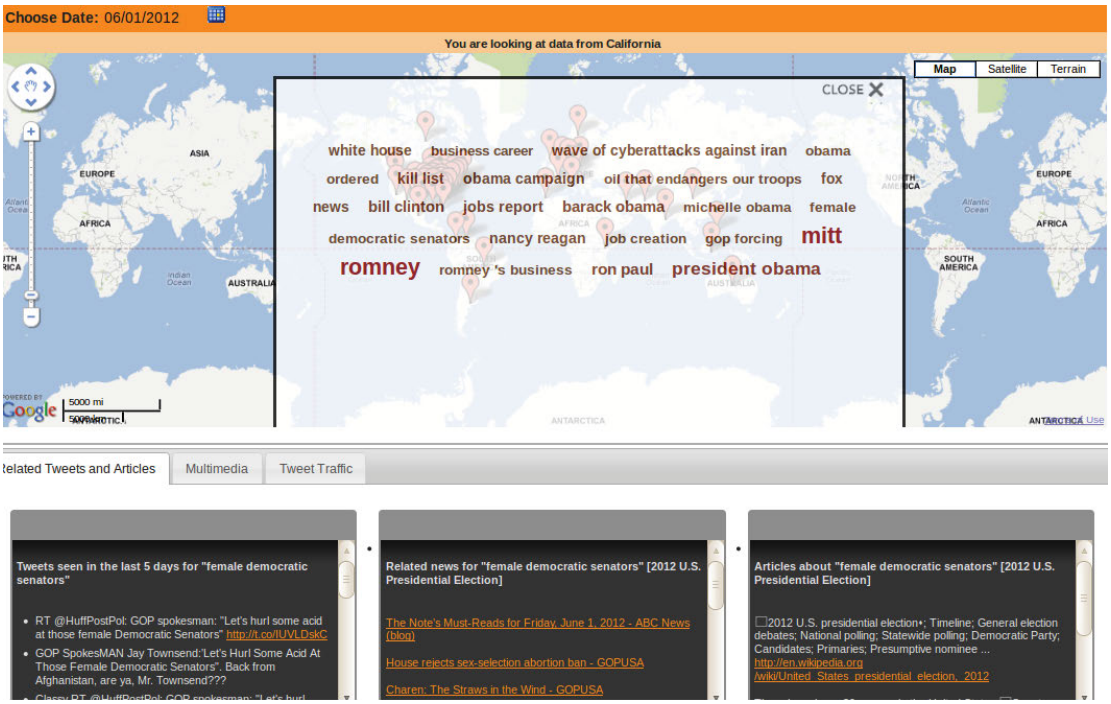


Figure 9.9: The Twitris social media event monitoring portal (`http://twitris.knoesis.org`).

One of the simplest and most widely used visualizations is word clouds. These generally use single word terms, which can be somewhat difficult to interpret without extra context. Word clouds have been used to assist users in browsing social media streams, including blog content [355] and tweets [261, 356]. For instance, Phelan et al. [357] use word clouds to present the results of a Twitter-based recommendation system. The Eddi system [358] uses topic clouds, showing higher-level themes in the user's tweet stream. These are combined with topic lists, which show who tweeted on which topic, as well as a set of interesting tweets for the highest ranked topics. The Twitris system (see Figure 9.9) derives even more detailed, contextualized phrases, by using 3-grams, instead of uni-grams [261]. More recently, the concept has been extended toward image clouds [254].

The main drawback of cloud-based visualizations is their static nature. Therefore, they are often combined with timelines showing keyword/topic frequencies over time [260, 273, 358, 359], as well as methods for discovery of unusual popularity bursts [355]. [269] use a timeline which is synchronized with a transcript of a political broadcast, allowing navigation to key points in a video of the event, and displaying tweets from that time period. Overall sentiment is shown on a timeline at each point in the video, using simple color segments. Similarly, TwitInfo (see Figure 9.11 [262]) uses a timeline to display tweet activity during a real-world event (e.g., a football

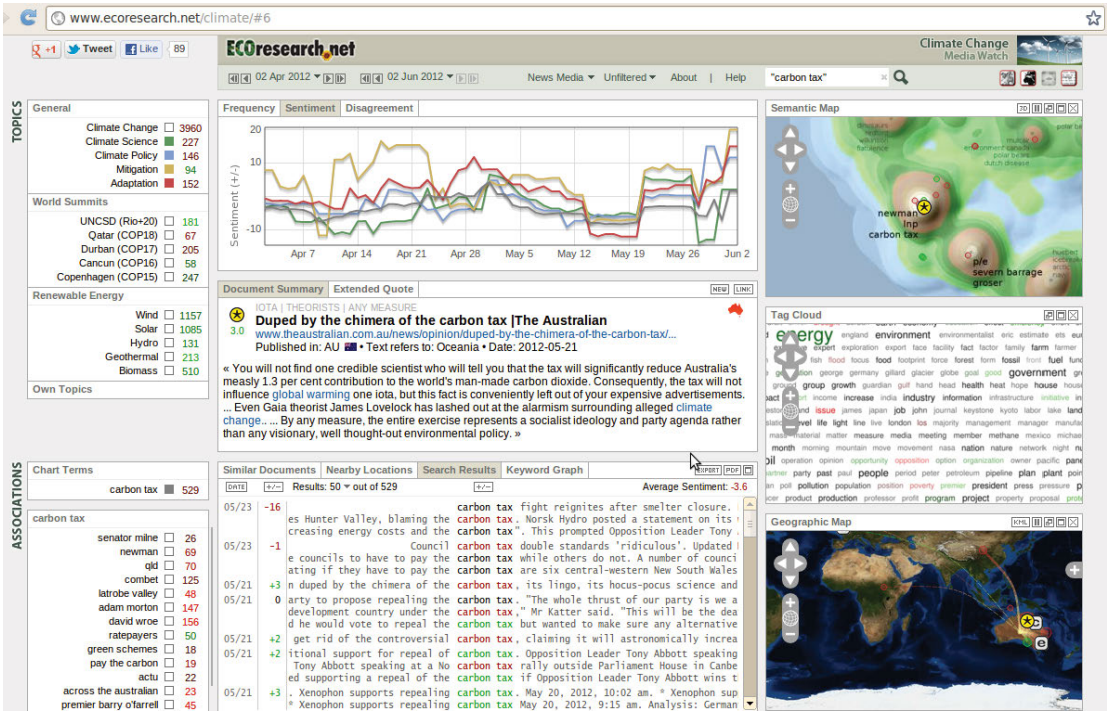


Figure 9.10: Media watch on climate change portal (`http://www.ecoresearch.net/climate`).

game), coupled with some example tweets, color-coded for sentiment. Some of these visualizations are dynamic, i.e., update as new content comes in (e.g., topic streams [254], falling keyword bars [273], dynamic information landscapes [273]), or topic bars comparing tweets alongside different criteria (in this case, tweet authors were split by their support for the UK leaving/remaining in the EU; Figure 9.13).

In addition, some visualizations try to capture the semantic relatedness between topics in the media streams. For instance, BlogScope [355] calculates keyword correlations, by approximating mutual information for a pair of keywords using a random sample of documents. Another example is the information landscape visualization, which conveys topic similarity through spatial proximity [273] (see Figure 9.10). Topic-document relationships can be shown also through force-directed, graph-based visualizations [360]. Lastly, Archambault et al. [361] propose multi-level tag clouds, in order to capture hierarchical relations.

Another important dimension of user-generated content is its place of origin. For instance, some tweets are geo-tagged with latitude/longitude information, while many user profiles on Facebook, Twitter, and blogs specify a user location. Consequently, map-based visualizations of topics have also been explored [261, 262, 273, 362] (see also Figures 9.10 and 9.11). For instance,

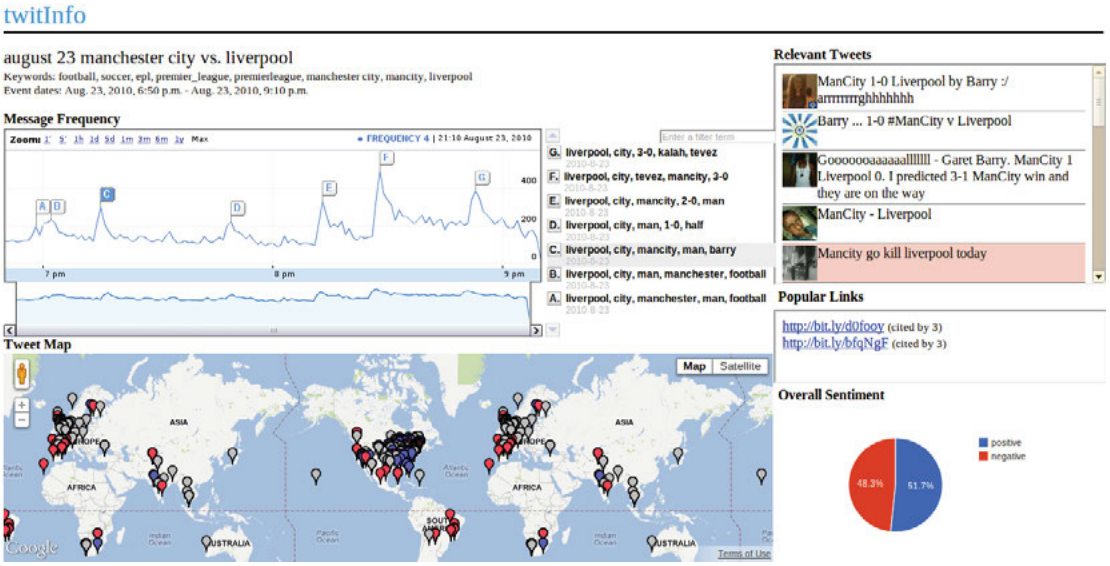


Figure 9.11: TwitInfo tracks a football game (`http://twitinfo.csail.mit.edu/`).

Figure 9.12: Different topics extracted by Twitris for Great Britain.

Twitris [261] allows users to select a particular state from the Google map and shows the topics discussed in social media from this state only. Figure 9.9 shows the Twitris US 2012 presidential elections monitor, where we have chosen to see the related topics discussed in social media originating from California. Clicking on the topic "female democratic senators" displays the relevant tweets, news, and Wikipedia articles. For comparison, Figure 9.12 shows the most discussed topics related to the election, extracted from social media originating from Great Britain. While there is significant topic overlap between the two locations, the differences become also clearly visible.

It is also possible to aggregate and visualize tweets based on the location of their tweet author, e.g., investigate regional variation of topic mentions. The example below shows a Mímir-

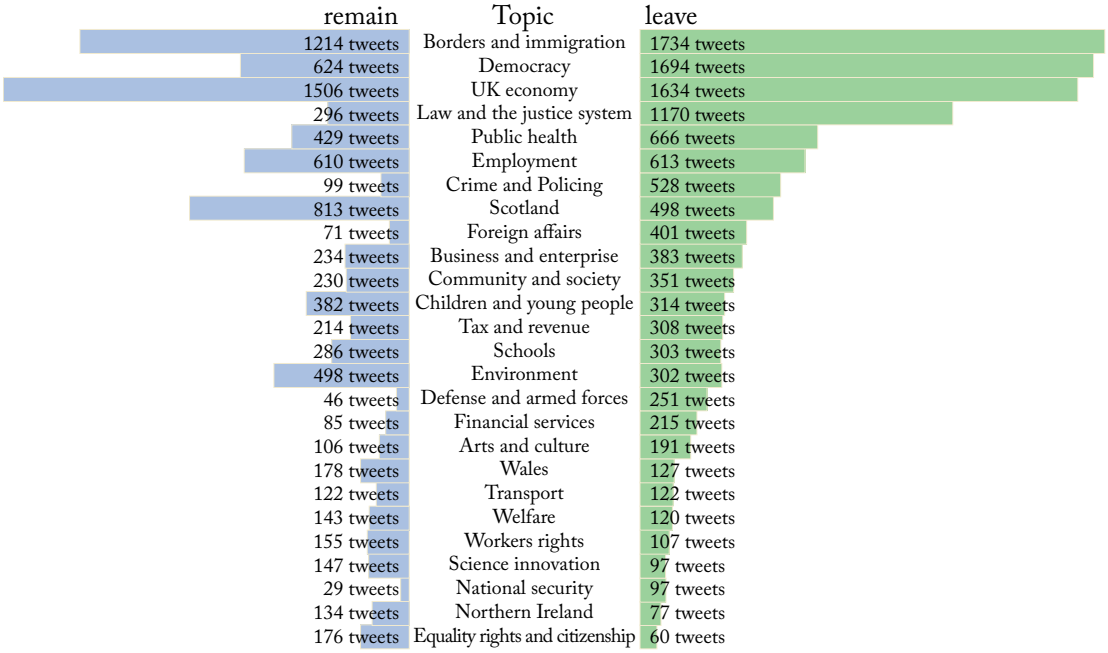


Figure 9.13: Topic bars comparing tweets on that topic posted by leave/remain EU referendum supporters.

based visualization showing which topics are talked about more in different parts of the country, based on NUTS aggregation of the tweets by UK election candidates. This involves issuing a series of Mímir queries over the tweets for each topic, to find how many tweets mentioning each topic in turn were written by an MP representing each region. The information about which region an MP represents is not expressed in the tweet itself, but uses our knowledge base in two stages: first to find which constituency an MP represents, and then to match the constituency with the appropriate NUTS region. Figure 9.14 shows a choropleth depicting the distribution of MPs' tweets which discuss the UK economy (the most frequent theme) in tweets on the UK 2015 General Election, collected in the week beginning the 2nd of March 2015. This is a dynamic visualization, based on the Leaflet library[12] and the aggregated query results returned by Mímir for each theme and NUTS1 region. The choropleth has a pull-down menu from which the user can select the topic of interest, and this re-draws the map accordingly. Demos of the interactive choropleth and treemap on this dataset, as well as examples of the topic cloud and a sentiment visualization, are publicly available at `http://www.nesta.org.uk/blog/4-visualizations-uk-general-election`.

[12] `http://leafletjs.com/`

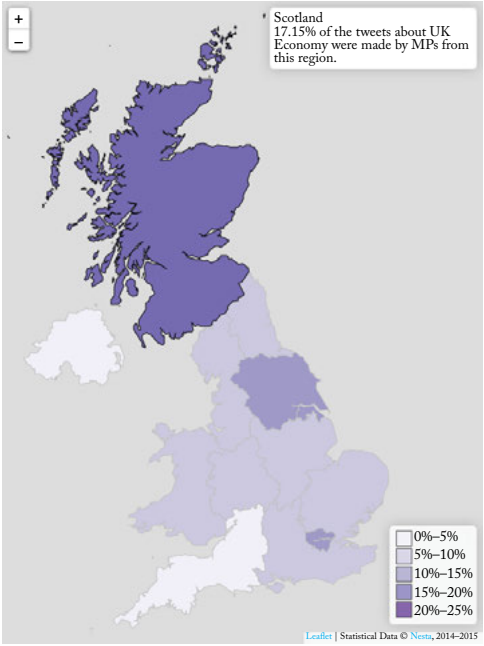


Figure 9.14: Choropleth depicting distribution of tweets about the economy.

Opinions and sentiment also feature frequently in visual analytics interfaces. For instance, Media Watch (Figure 9.10 [273]) combines word clouds with aggregated sentiment polarity, where each word is colored in a shade of red (predominantly negative sentiment), green (predominantly positive), or black (neutral/no sentiment). Search results snippets and faceted browsing terms are also sentiment colored. Others have combined sentiment-based color coding with event timelines [359], lists of tweets (Figure 9.11 [262]), and mood maps [359]. Aggregated sentiment is typically presented using pie charts [260] and, in the case of TwitInfo, the overall statistics are normalized for recall (Figure 9.11 [262]).

Researchers have also investigated specifically the problem of browsing and visualizing social media conversations about real-world events, e.g., broadcast events [356], football games (Figure 9.11 [262]), conferences [254], and news events [359, 362]. A key element here is the ability to identify sub-events and combine these with timelines, maps, and topic-based visualizations.

Other visualizations have been designed to exploit the user-generated and social nature of the media streams. For instance, the PeopleSpiral visualization [254] plots Twitter users who have contributed to a topic (e.g., posted using a given hashtag) on a spiral, starting with the most active and "original" users first. User originality is measured as the ratio between the number of tweets authored by the user vs. re-tweets made. OpinionSpace [363] instead clusters and visualizes users

in a two-dimensional space, based on the opinions they have expressed on a given set of topics. Each point in the visualization shows a user and their comment, so the closer two points, the more similar the users and opinions are. However, the purely point-based visualization was found hard to interpret by some users, since they could not see the textual content until they clicked on a point. ThemeCrowds [361] instead derives hierarchical clusters of Twitter users through agglomerative clustering and provides a summary of the tweets generated by this user cluster, through multilevel tag clouds (inspired by treemap visualization). Tweet volumes over time are shown in a timeline-like view, which also allows the selection of a time period.

9.5 DISCUSSION AND FUTURE WORK

Most current search, recommendation, and visualization methods tend to use shallow textual and frequency-based information. For instance, a comparison between TF-IDF weighted topic models and LDA topic modeling has shown the former to be superior [238, 354]. However, these can be improved further through integration of semantic information, as suggested by [354]. In the case of personalized recommendations, these could be improved by incorporating user behavior roles, making better use of the latent semantics and implicit user information, as well as better integration of the temporal dimension in the recommender algorithms.

Browsing and visualization interfaces can also be improved by taking into account the extra semantic knowledge about the entities mentioned in the media streams. For instance, when entities and topics are annotated with URIs to LOD resources, such as DBpedia, the underlying ontology can underpin hierarchically-based visualizations, including semantic relations. In addition, the exploration of media streams through topic-, entity-, and time-based visualizations can be enriched with ontology-based faceted search and semantic query interfaces. One such example is the KIM semantic platform, which is, however, aimed at largely static document collections [317].

Algorithm scalability and efficiency are particularly important, due to the large-scale, dynamic nature of social media streams. For instance, the interactive Topic Stream visualization takes 45 seconds to compute on 1 million tweets and 325,000 contributing users, which is too long for most usage scenarios [254]. Similarly, calculating keyword correlations through pointwise mutual information is computationally too expensive on high-volume blog posts [355]. A frequently used solution is to introduce a sliding window over the data (e.g., between one week and one year) and thus limit the content used for IDF and other such calculations.

Most of the systems and approaches reviewed here are not easily extendable or adaptable to a new problem domain, new visualization, or with extended semantic annotation capabilities. The advantage of the GATE-based open source tools for semantic search and visualization (Mímir and Prospector) and the GATE real-time analytics pipeline is exactly in their open, extensible, and scalable nature. In the most recent application on analysing UK Brexit referendum tweets (i.e., the Brexit Analyser; see Figure 9.15), there were on average 500,000 tweets a day, with a peak on well over 2 million tweets on voting day. This required very high-performance semantic

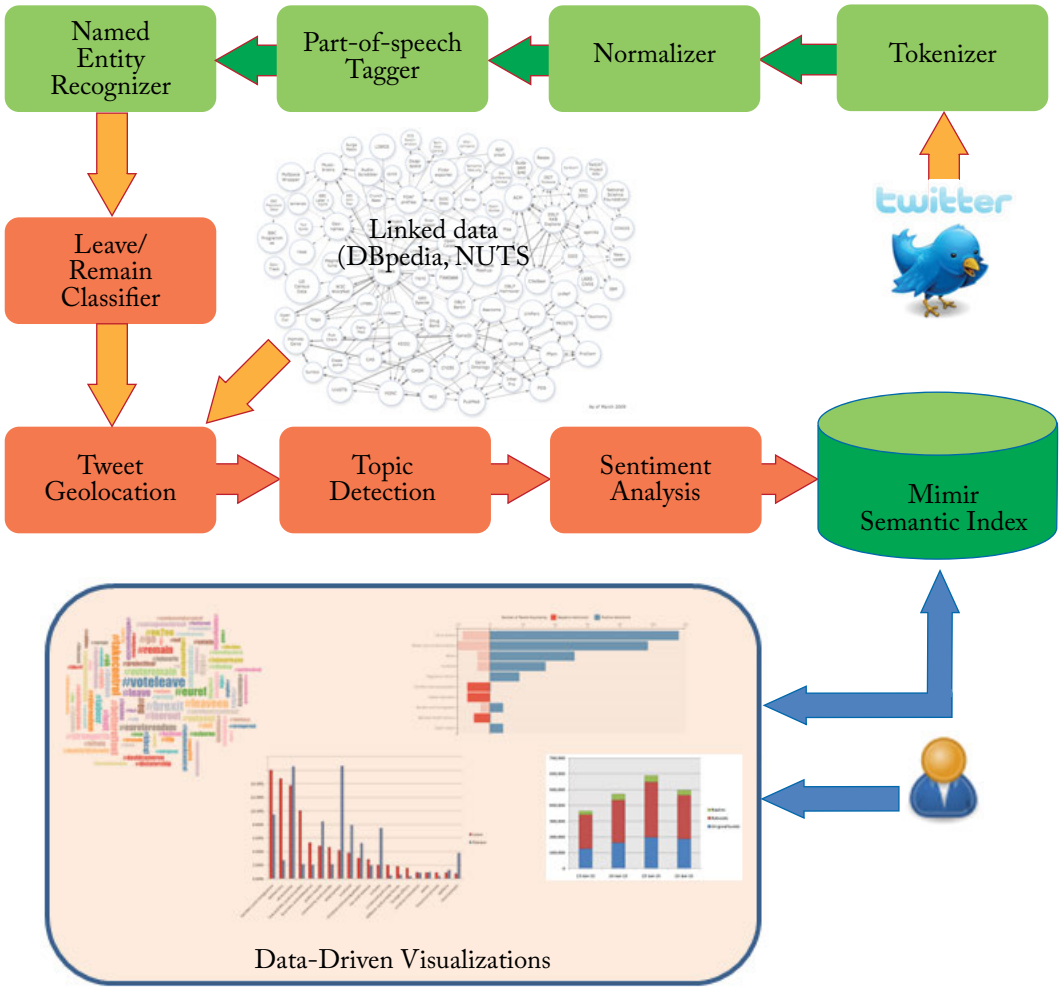


Figure 9.15: Architecture of the Brexit semantic analysis, search, and visualization system.

analytics, indexing, search, and visualization components, which were designed to analyze up to 100 tweets per second.

For analysis, we are using GATE's TwitIE system [248], which consists of a tokenizer, normalizer, part-of-speech tagger, and a named entity recognizer. After that, we added a Leave/Remain classifier, to identify a reliable sample of tweets with unambiguous stance. Next is a tweet geolocation component, which uses latitude/longitude, region, and user location metadata to geolocate tweets within the UK NUTS2 regions. Key themes and topics discussed in the tweets were detected (more than one topic/theme can be contained in each tweet), followed by topic-centric sentiment analysis. The key benefit from reusing so many already available semantic annotation components was that application development time was very short.

The Mímir-based searches and visualizations supported efficient exploration of the large dataset of over 64 million tweets. Typical Mímir queries would contain restrictions by timestamp (normalized to GMT), tweet kind (original, reply, or retweet), voting intention (Leave/Remain), mentioning a specific user/hashtag/topic, written by a specific user, containing a given hashtag or a given topic (e.g., all tweets discussing taxes). Example annotation-driven visualizations were shown above. They are all interactive, where users can click on a specific item (e.g., a topic bar, NUTS region) and see immediately all tweets backing this aggregated visualization element. While still undergoing development, this open framework approach to large-scale, interactive semantic search and visualization has demonstrated benefits in reduced development time, robustness, and diverse visualizations.

In conclusion, designing effective semantic search, browsing, and visualizations interfaces for high-volume, high-velocity media streams has proven particularly challenging. Some outstanding issues include:

- designing meaningful and intuitive visualizations, conveying intuitively the complex, multidimensional semantics of user-generated content (e.g., topics, entities, events, user demographics (including geolocation), sentiment, social networks);
- visualizing changes over time;
- supporting different levels of granularity, at the level of semantic content, user clusters, and temporal windows;
- allowing interactive, real-time exploration;
- integration with search, to allow users to select a subset of relevant content;
- exposing the discussion/threaded nature of the social conversations; and
- addressing scalability and efficiency.

CHAPTER 10

Conclusions

We conclude the book with a summary of the key points, some general observations about the use of NLP in Semantic Web applications, and some thoughts about future directions.

10.1 SUMMARY

The aim of this book has been to introduce some of the key concepts, techniques, and tools in Natural Language Processing and text analytics to Semantic Web researchers, explaining why they are necessary and how creating and using such tools and technologies can be achieved. It is important to understand clearly not just why NLP technologies can be useful, but also their limitations. Throughout the book, we have illustrated the technologies with examples of common open-source tools that can be used, discussing issues with integration and dependencies, and giving some idea about expected performance.

The first part of the book has been dedicated to explaining the key underlying concepts of NLP, in order to build a foundation for the more complex tasks in the later stages of the book. We have largely followed the pipeline approach involved in building up an NLP-based application, starting with the low-level tasks such as basic linguistic pre-processing, and moving on to more complex tasks such as relation finding, ontology development, and opinion mining. We have also considered different kinds of tasks and applications, such as social media analysis and the specific kinds of adaptation needed for these, as well as how all these tools can be used for more complex applications such as semantically-enhanced information retrieval and visualization.

Ultimately, the reader should come away armed with the knowledge to understand the main principles of NLP and its role in the Semantic Web, and with the ability to choose suitable NLP technologies that can be used to enhance their Semantic Web applications. There are of course many topics and tools that have not been discussed here, but where appropriate, pointers have been given to more detailed descriptions elsewhere. This book attempts to gather into one place some of the most relevant material to meet these aims.

10.2 FUTURE DIRECTIONS

While established core linguistic pre-processing techniques form the foundations for many NLP tasks, as we have seen throughout the course of this book, there are still many challenges to be faced in adapting NLP methods and tools to the new forms of data and to the new kinds of applications which are constantly emerging. In this section, we discuss some of the imminent

critical directions in which NLP research needs to move in order to keep up with technological advances.

10.2.1 CROSS-MEDIA AGGREGATION AND MULTILINGUALITY

The majority of methods surveyed here have been developed and evaluated only on one kind of media (e.g., news texts, twitter, or blog posts). However, many current applications demand integration of different kinds of text, for example by connecting tweets to news articles and blogs. Furthermore, cross-media linking, which is a crucial open issue, can go beyond this, due to the fact that users are increasingly adopting more than one social media platform, often for different purposes (e.g., personal vs. professional use). In addition, as people's lives are becoming more and more digital, this work will provide a partial answer to the challenge of inter-linking our personal collections (e.g., emails, photos) with our social media online identities.

The challenge is to build computational models of cross-media content merging, analysis, and visualization, and to embed these into algorithms capable of dealing with the large-scale, contradictory and multi-purpose nature of multi-platform social media streams. For example, further work is needed on algorithms for cross-media content clustering, cross-media identity tracking, modeling contradictions between different sources, and inferring change in interests and attitudes over time.

Another related major challenge is multilinguality. Most of the methods surveyed here were developed and tested on English content only, because this tends to be the first port of call for new technologies and applications. We should not overlook, however, the importance of adapting such tools to other languages and/or enabling them to deal with multiple languages simultaneously. As discussed in Section 8.3.7, some initial steps are being made through multilingual lexicons, such as Wiktionary [289] and UBY [290], and linguistically grounded ontologies [291]. Other work has focused on widening the range of available linguistic resources to less studied languages, through crowdsourcing. Amazon Mechanical Turk, in particular, has emerged as useful, since crowdsourcing projects are easily set up there, coupled with the fact that it allows "access to foreign markets with native speakers of many rare languages" [364]. This feature is particularly useful for researchers working on less-resourced languages, such as Arabic [365], Urdu [364] and others [366–368]. Irvine and Klementiev [368], for example, have shown that it is possible to create lexicons between English and 37 out of the 42 low-resource languages that they experimented with. Similarly, Weichselbraun et al. [369] crowd-source domain-specific sentiment lexicons in multiple languages, through games with a purpose. A related aspect is designing crowdsourcing projects, so that they can be re-used easily across languages, e.g., [368, 370] for Mechanical Turk and [371, 372] for games-with-a-purpose. There is also the related issue of annotated corpora and evaluation, to which we return in Section 10.2.4 below.

Lastly, as users are increasingly consuming social media streams on different hardware platforms (desktops, tablets, smart phones), cross-platform and/or platform-independent informa-

tion access methods need to be developed. This is particularly challenging in the case of information visualization on small screen devices.

10.2.2 INTEGRATION AND BACKGROUND KNOWLEDGE

Traditionally, research efforts focus on furthering one particular research stream, e.g., rule-based methods or supervised learning methods. Research streams have different benefits: some are good at learning feature representations and models based on labeled training data, and making predictions on unseen data [60], while others make use of background knowledge, e.g., by learning inference rules based on seed knowledge bases [95, 110] or automatically creating training data for supervised learning based on seed knowledge bases [73, 81, 373].

Something that has proven successful in real-world settings is getting different views on the same problem with different methods [95] or even different extraction schemas [107] and integrating them. While there has been some work on integrating different methods, e.g., using ensemble learning [374] or universal schemas [107, 110], the majority of work does not focus on this. In addition, work on integrating schemas assumes there are two overlapping schemas available. In reality, more than two schemas for defining information are used. Also, schemas do not always overlap, which is one of the reasons to use different schemas in the first place.

There are further open challenges with respect to learning inference rules from knowledge bases. NLP research often considers artificial settings in which schemas do not define relations between concepts or properties. For example, in RDFS relationships are defined by properties which have superproperties, domains, and ranges, while OWL allows the definition of mutually inverse relationships. Work on learning inference rules largely ignores this and assumes that all such relations have to be learned from scratch, thus not focusing on the challenge of going beyond what is already defined.

10.2.3 SCALABILITY AND ROBUSTNESS

In information extraction research, large-scale algorithms (also referred to as data-intensive or web-scale natural language processing) are demonstrating increasingly superior results compared with approaches trained on smaller datasets [375]. This is mostly thanks to addressing the data sparseness issue through the collection of significantly larger numbers of naturally occurring linguistic examples [375]. The need for and the success of data-driven NLP methods mirrors to a large extent recent trends in other research fields, leading to what is referred to as "the fourth paradigm of science" [376].

At the same time, semantic annotation and information access algorithms need to be scalable and robust, in order to cope with the large volumes of data encountered in social media streams. Many use cases require online, near real-time processing, which introduces additional requirements in terms of algorithm complexity. Cloud computing [377] is increasingly regarded as a key enabler of scalable, on-demand processing, giving researchers everywhere affordable ac-

cess to computing infrastructures, which allow the deployment of significant compute power on an on-demand basis, and with no upfront costs.

However, developing scalable and parallelizable algorithms for platforms such as Hadoop is far from trivial. Straightforward deployment and sharing of semantic annotation pipelines and algorithm parallelization are only a few of the requirements which need to be met. Research in this area is still in its infancy, especially around general purpose platforms for scalable semantic processing.

GATE Cloud[1] can be viewed as the first step in this direction [320]. It is a novel cloud-based platform for large-scale text mining research, which also supports ontology-based semantic annotation pipelines. It aims to provide researchers with a platform-as-a-service, which enables them to carry out large-scale NLP experiments by harnessing the vast, on-demand compute power of the Amazon cloud. It also minimizes the need to implement specialized parallelizable text-processing algorithms. Important infrastructural issues are dealt with by the platform, completely transparently for the researcher: load balancing, efficient data upload and storage, deployment on the virtual machines, security, and fault tolerance.

One example application of GATE Cloud was in a project with the UK National Archive [293], which used it to annotate semantically 42TB of web pages and other textual content. The semantic annotation process was underpinned by a large-scale knowledge base, acquired from the LOD cloud, data.gov.uk, and a large geographical database. The results were indexed in GATE Mímir [297], coupled with a user interface for browsing, search, and navigation from the document space into the semantic knowledge base via full-text search, semantic annotations, and SPARQL queries.

10.2.4 EVALUATION, SHARED DATASETS, AND CROWDSOURCING

The fourth major open issue is evaluation. As discussed previously, lack of shared gold-standard datasets can significantly hamper repeatability and comparative evaluation of algorithms. At the same time, comprehensive user- and task-based evaluation experiments are also required, in order to identify problems with existing search and visualization methods. Particularly in the area of intelligent information access, there is a significant body of research which does not report evaluation experiments, or which has only performed small-scale, formative studies. Longitudinal evaluation with larger user groups is particularly lacking.

Similarly, algorithm training and adaptation on social media gold standard datasets is currently very limited. For example, no gold standard datasets of Twitter and blog summaries exist and there are fewer than 10,000 tweets annotated with named entities. Creating sufficiently large, vitally needed datasets through traditional expert-based text annotation methodologies is very expensive, both in terms of time and funding required. The latter can vary between USD 0.36 and 1.0 per word [371], which is unaffordable for corpora consisting of millions of words. Some cost reductions could be achieved through web-based collaborative annotation tools, such as GATE

[1] http://cloud.gate.ac.uk

Teamware [378] and WebAnno [379], which support distributed teams and are tailored to non-expert annotators.

An alternative involves the use of commercial crowdsourcing marketplaces, which are reportedly 33% less expensive than in-house employees on tasks such as tagging and classification [380]. Consequently, in the field of language processing, researchers have started creating annotated corpora with Amazon Mechanical Turk, Crowdflower, and game-based approaches as less expensive alternatives.

With respect to corpus annotation in particular, Poesio et al. [371] estimate that, compared to the cost of expert-based annotation (estimated at $1 million), the cost of 1 million annotated tokens could be reduced to less than 50% by using MTurk ($380.000–$430.000) and to around 20% ($217,927) when using a game-based approach such as their own PhraseDetectives game. With respect to crowdsourcing social media annotations, there have been experiments on categorizing tweets [381] and annotating named entities in tweets [292], amongst other things. In the Semantic Web field, researchers have explored mostly crowdsourcing through games with a purpose, primarily for knowledge acquisition [382, 383] and LOD improvement [384].

At the same time, researchers have turned to crowdsourcing as a means for scaling up human-based evaluation experiments. The main challenge here is in how to define the evaluation task, so that it can be crowdsourced from non-specialists, with high-quality results [385]. This is far from trivial, and researchers have argued that crowdsourcing evaluation tasks need to be designed differently from expert-based evaluations [386]. In particular, Gillick and Liu [386] found that non-expert evaluation of summarization systems produces noisier results, thus requiring more redundancy to achieve statistical significance, and furthermore that Mechanical Turk workers cannot produce score rankings that agree with expert ranking.

One successful design for crowdsourcing-based evaluation has used a four-phase workflow of separate tasks, which has been tried on reading comprehension of machine translation [367]. A simpler task design has been used by [387] for evaluation of tweet summaries: here Mechanical Turk workers were asked to indicate on a five-point scale how much of the information from the human-produced summary was contained in the automatically produced summary. A third evaluation example, which has achieved successful results on Mechanical Turk, is pair-wise ranking [388]. The task in this case is to identify the most informative sentence from a product review. In this case, the crowdworkers were asked to indicate whether a sentence chosen by the baseline system was more informative than a sentence chosen by the author's method. Sentence order was randomized and it was also possible to indicate that none of these sentences were a good summary.

Despite all this work, reusable corpus conversion tools and user interfaces for crowdsourced NLP tasks is still problemantic. The open-source GATE Crowdsourcing plugin [389] addresses this challenge by offering infrastructural support for mapping documents to crowdsourcing units and back automatically, as well as automatically generating reusable crowdsourcing interfaces for NLP classification and selection tasks. The entire workflow has been tested on diverse NLP tasks,

including annotating named entities; disambiguating words and named entities with respect to DBpedia URIs; annotation of opinion holders and targets; and sentiment.

To conclude, crowdsourcing has recently emerged as a promising method for creating shared evaluation datasets, as well as for carrying out user-based evaluation experiments. Adapting these efforts to the specifics of semantic annotation and information visualization, as well as using these to create large-scale resources and repeatable, longitudinal evaluations, are key areas for future work.

Bibliography

[1] Roger C. Schank and Larry Tesler. A conceptual dependency parser for natural language. In *Proc. of the Conference on Computational Linguistics*, pages 1–3. Association for Computational Linguistics, 1969. DOI: 10.3115/990403.990405. 2

[2] Robert B. Lees and N. Chomsky. Syntactic structures. *Language*, 33(3 Part 1), pages 375–408, 1957. DOI: 10.2307/411160. 2

[3] R. Gaizauskas and Y. Wilks. Information extraction: Beyond document retrieval. *Journal of Documentation*, 54(1), pages 70–105, 1998. DOI: 10.1108/eum0000000007162. 2

[4] Hamish Cunningham, Diana Maynard, Kalina Bontcheva, and Valentin Tablan. Gate: An architecture for development of robust hlt applications. In *Proc. of the 40th Annual Meeting on Association for Computational Linguistics*, ACL'02, pages 168–175, Stroudsburg, PA, 2002. DOI: 10.3115/1073083.1073112. 11

[5] Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. The Stanford CoreNLP natural language processing toolkit. In *Proc. of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, 2014. DOI: 10.3115/v1/p14-5010. 11

[6] Steven Bird, Ewan Klein, and Edward Loper. *Natural Language Processing with Python*. O'Reilly Media, Inc., 2009. 11

[7] H. Cunningham, D. Maynard, and V. Tablan. JAPE: A Java Annotation Patterns Engine 2nd ed. Research Memorandum CS–00–10, Department of Computer Science, University of Sheffield, Sheffield, UK, 2000. 14

[8] Tibor Kiss and Jan Strunk. Unsupervised multilingual sentence boundary detection. *Computational Linguistics*, 32(4), pages 485–525, 2006. DOI: 10.1162/coli.2006.32.4.485. 15

[9] Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of english: The penn treebank. *Computational Linguistics*, 19(2), pages 313–330, 1994. 15

[10] W. Nelson Francis and Henry Kucera. Brown corpus manual. *Brown University*, 1979. 15

[11] Stig Johansson. The tagged {LOB} corpus: User\'s manual, 1986. 15

[12] E. Brill. A simple rule-based part-of-speech tagger. In *Proc. of the 3rd Conference on Applied Natural Language Processing*, Trento, Italy, 1992. DOI: 10.3115/974499.974526. 16

[13] Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proc. of the Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, NAACL'03, pages 173–180, 2003. DOI: 10.3115/1073445.1073478. 16

[14] Thorsten Brants. Tnt: A statistical part-of-speech tagger. In *Proc. of the 6th conference on Applied Natural Language Processing*, ANLP'00, pages 224–231, 2000. DOI: 10.3115/974147.974178. 16

[15] M. Hepple. Independence and commitment: Assumptions for rapid training and execution of rule-based part-of-speech taggers. In *Proc. of the 38th Annual Meeting of the Association for Computational Linguistics*, Hong Kong, 2000. DOI: 10.3115/1075218.1075254. 16

[16] G. A. Miller. WordNet: An on-line lexical database. *International Journal of Lexicography*, 3(4), pages 235–312, 1990. 17

[17] M. F. Porter. An algorithm for suffix stripping. *Program*, 14(3),pages 130–137, 1980. DOI: 10.1108/eb046814. 19

[18] Ted Briscoe, John Carroll, and Rebecca Watson. The second release of the rasp system. In *Proc. of the COLING/ACL on Interactive Presentation Sessions*, pages 77–80, 2006. DOI: 10.3115/1225403.1225423. 19, 20

[19] D. Klein and C. Manning. Accurate unlexicalized parsing. In *Proc. of the 41st Meeting of the Association for Computational Linguistics*, 2003. DOI: 10.3115/1075096.1075150. 19, 21

[20] Robert Gaizauskas, Mark Hepple, Horacio Saggion, Mark A. Greenwood, and Kevin Humphreys. SUPPLE: A practical parser for natural language engineering applications. In *Proc. of the 9th International Workshop on Parsing Technology*, pages 200–201. Association for Computational Linguistics, 2005. DOI: 10.3115/1654494.1654521. 19

[21] Helmut Schmid. Probabilistic part-of-speech tagging using decision trees. In *Proc. of the International Conference on New Methods in Language Processing*, volume 12, pages 44–49. Citeseer, 1994. 22

[22] L. Ramshaw and M. Marcus. Text chunking using transformation-based learning. In *Proc. of the 3rd ACL Workshop on Very Large Corpora*, 1995. DOI: 10.1007/978-94-017-2390-9_10. 23

[23] Collins Cobuild, Ed. *English Grammar*. Harper Collins, 1999. 23

[24] S. Azar. *Understanding and Using English Grammar*. Prentice Hall Regents, 1989. 23

[25] R. Grishman and B. Sundheim. Message understanding conference-6: A brief history. In *Proc. of COLING*. Association for Computational Linguistics, 1995. DOI: 10.3115/992628.992709. 25, 26, 38

[26] Asif Ekbal, Eva Sourjikova, Anette Frank, and Simone Paolo Ponzetto. Assessing the challenge of fine-grained named entity recognition and classification. In A. Kumaran and Haizhou Li, Eds., *Proc. of the Named Entities Workshop*, pages 93–101, Uppsala, Sweden, 2010. Association for Computational Linguistics. 25

[27] Christopher Walker, Stephanie Strassel, Julie Medero, and Kazuaki Maeda. ACE 2005 multilingual training corpus. *Linguistic Data Consortium*, Philadelphia, 2006. 26, 27

[28] Erik F. Tjong, Kim Sang, and Fien De Meulder. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In Walter Daelemans and Miles Osborne, Eds., *Proc. of NAACL-HLT*, pages 142–147, 2003. 27, 32

[29] Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. Ontonotes: The 90% solution. In *Proc. of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 57–60, New York City, 2006. Association for Computational Linguistics. 27

[30] Giuseppe Rizzo and Raphaël Troncy. NERD: A framework for evaluating named entity recognition tools in the Web of data. In *ISWC 10th International Semantic Web Conference*, Bonn, Germany, 2011. 27

[31] Lev Ratinov and Dan Roth. Design challenges and misconceptions in named entity recognition. In *Proc. of the 13th Conference on Computational Natural Language Learning*, pages 147–155. Association for Computational Linguistics, 2009. DOI: 10.3115/1596374.1596399. 28

[32] Lev Ratinov and Dan Roth. Design challenges and misconceptions in named entity recognition. In *Proc. of the 13th Conference on Computational Natural Language Learning*, pages 147–155. Association for Computational Linguistics, 2009. DOI: 10.3115/1596374.1596399. 28

[33] James Pustejovsky, José M. Castano, Robert Ingria, Roser Sauri, Robert J. Gaizauskas, Andrea Setzer, Graham Katz, and Dragomir R. Radev. Timeml: Robust specification of event and temporal expressions in text. *New Directions in Question Answering*, 3, pages 28–34, 2003. 29

[34] J. Strötgen and M. Gertz. HeidelTime: High quality rule-based extraction and normalization of temporal expressions. In *Proc. of the 5th International Workshop on Semantic Evaluation*, pages 321–324. Association for Computational Linguistics, 2010. 29

[35] James Pustejovsky, Kiyong Lee, Harry Bunt, and Laurent Romary. ISO-TimeML: An international standard for semantic annotation. In *LREC*, 2010. 29

[36] Angel X. Chang and Christopher D. Manning. Sutime: A library for recognizing and normalizing time expressions. In *LREC*, pages 3735–3740, 2012. 29

[37] K. Bontcheva, M. Dimitrov, D. Maynard, V. Tablan, and H. Cunningham. Shallow Methods for Named Entity Coreference Resolution. In *Chaînes de Références et Résolveurs D'anaphores, Workshop TALN 2002*, Nancy, France, 2002. `http://gate.ac.uk/sale/taln02/taln-ws-coref.pdf` 29

[38] Karthik Raghunathan, Heeyoung Lee, Sudarshan Rangarajan, Nathanael Chambers, Mihai Surdeanu, Dan Jurafsky, and Christopher Manning. A multi-pass sieve for coreference resolution. In *Proc. of the Conference on Empirical Methods in Natural Language Processing*, pages 492–501. Association for Computational Linguistics, 2010. 29

[39] Roman Prokofyev, Alberto Tonon, Michael Luggen, Loic Vouilloz, Djellel Eddine Difallah, and Philippe Cudré-Mauroux. Sanaphor: Ontology-based coreference resolution. In *The Semantic Web-ISWC 2015*, pages 458–473. Springer, 2015. DOI: 10.1007/978-3-319-25007-6_27. 29

[40] P. Cimiano, S. Handschuh, and S. Staab. Towards the self-annotating web. In *Proc. of the 13th International Conference on World Wide Web (WWW'04)*, 2004. DOI: 10.1145/988672.988735. 30

[41] P. Pantel and M. Pennacchiotti. Automatically harvesting and ontologizing semantic relations. In *Proc. of the Conference on Ontology Learning and Population: Bridging the Gap Between Text and Knowledge*, pages 171–195. IOS Press, 2008. 30

[42] P. Cimiano, M. Hartung, and E. Ratsch. Learning the appropriate generalization level for relations extracted from the Genia corpus. In *Proc. of the 5th Language Resources and Evaluation Conference (LREC)*, 2006. 30

[43] A. Schutz and P. Buitelaar. Relext: A tool for relation extraction from text in ontology extension. *The Semantic Web-ISWC*, pages 593–606, 2005. DOI: 10.1007/11574620_43. 30

[44] V. Crescenzi, G. Mecca, P. Merialdo, et al. Roadrunner: Towards automatic data extraction from large web sites. In *Proc. of the International Conference on Very Large Data Bases*, pages 109–118. Citeseer, 2001. 30

[45] D. E. Appelt. The common pattern specification language. Technical report, SRI International, Artificial Intelligence Center, 1996. DOI: 10.3115/1119089.1119095. 31

[46] D. Freitag. Information extraction from html: Application of a general learning approach. *Proc. of the 15th Conference on Artificial Intelligence AAAI-98*, pages 517–523, 1998. 31

[47] M. Califf and R. Mooney. Relational learning of pattern-match rules for information extraction. *Working Papers of the ACL-97 Workshop in Natural Language Learning*, pages 9–15, 1997. 31

[48] S. Soderland. Learning information extraction rules for semi-structured and free text. *Machine Learning*, 34(1), pages 233–272, 1999. DOI: 10.1023/A:1007562322031. 31

[49] Dayne Freitag and Nicholas Kushmerick. Boosted wrapper induction. In *17th National Conference on Artificial Intelligence (AAAI-2000): 12th Innovative Applications of Artificial Intelligence Conference (IAAI-2000)*, pages 577–583, 2000. 31

[50] F. Ciravegna. $(LP)^2$, an adaptive algorithm for information extraction from web-related texts. In *Proc. of the IJCAI Workshop on Adaptive Text Extraction and Mining*, Seattle, 2001. 31

[51] A. McCallum, D. Freitag, and F. Pereira. Maximum entropy Markov models for information extraction and segmentation. In *Proc. of the 17th International Conference on Machine Learning*, pages 591–598. Citeseer, 2000. 32

[52] H. L. Chieu and H. T. Ng. Named entity recognition with a maximum entropy approach. In Walter Daelemans and Miles Osborne, Eds., *Proc. of CoNLL-2003*, pages 160–163. Edmonton, Canada, 2003. 32

[53] H. Isozaki and H. Kazawa. Efficient support vector classifiers for named entity recognition. In *Proc. of the 19th International Conference on Computational Linguistics (COLING'02)*, pages 390–396, Taipei, Taiwan, 2002. DOI: 10.3115/1072228.1072282. 32

[54] J. Mayfield, P. McNamee, and C. Piatko. Named entity recognition using hundreds of thousands of features. In *Proc. of CoNLL-2003*, pages 184–187. Edmonton, Canada, 2003. DOI: 10.3115/1119176.1119205. 32

[55] Y. Li, K. Bontcheva, and H. Cunningham. Using uneven margins SVM and perceptron for information extraction. In *Proc. of 9th Conference on Computational Natural Language Learning (CoNLL-2005)*, 2005. DOI: 10.3115/1706543.1706556. 32

[56] X. Carreras, L. Màrquez, and L. Padró. Learning a perceptron-based named entity chunker via online recognition feedback. In *Proc. of CoNLL-2003*, pages 156–159. Edmonton, Canada, 2003. DOI: 10.3115/1119176.1119198. 32

[57] Yaoyong Li, Kalina Bontcheva, and Hamish Cunningham. Adapting SVM for data sparseness and imbalance: A case study on information extraction. *Natural Language Engineering*, 15(2), pages 241–271, 2009. DOI: 10.1017/s1351324908004968. 32

[58] Andrew McCallum and Wei Li. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In Walter Daelemans and Miles Osborne, Eds., *Proc. of the 7th Conference on Natural Language Learning at HLT-NAACL 2003*, pages 188–191, 2003. DOI: 10.3115/1119176. 32

[59] Jenny Rose Finkel, Trond Grenager, and Christopher Manning. Incorporating non-local information into information extraction systems by Gibbs sampling. In Kevin Knight, Hwee Tou Ng, and Kemal Oflazer, Eds., *Proc. of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 363–370, Ann Arbor, Michigan, 2005. Association for Computational Linguistics. DOI: 10.3115/1219840. 32

[60] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *Journal of Machine Learning Research (JMLR)*, 999888, pages 2493–2537, 2011. 32, 137

[61] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, New York, 2006. 32

[62] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, MA, 2012. 32

[63] Xiao Ling and Daniel S. Weld. Fine-grained entity recognition. In *Proc. of AAAI*, pages 94–100. AAAI Press, 2012. 34

[64] Jenny Rose Finkel, Trond Grenager, and Christopher Manning. Incorporating non-local information into information extraction systems by Gibbs sampling. In Kevin Knight, Hwee Tou Ng, and Kemal Oflazer, Eds., *Proc. of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 363–370, Ann Arbor, Michigan, 2005. Association for Computational Linguistics. DOI: 10.3115/1219840. 34

[65] Colin Cherry and Hongyu Guo. The unreasonable effectiveness of word representations for twitter named entity recognition. In Rada Mihalcea, Joyce Chai, and Anoop Sarkar, Eds., *Proc. of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 735–745, Denver, Colorado, 2015. Association for Computational Linguistics. DOI: 10.3115/v1/n15-1. 34

[66] Bo Han and Timothy Baldwin. Lexical normalisation of short text messages: Makn sens a #twitter. In Yuji Matsumoto and Rada Mihalcea, Eds., *Proc. of the ACL-HLT*, pages 368–378, Portland, Oregon, 2011. Association for Computational Linguistics. 34

[67] Leon Derczynski, Diana Maynard, Giuseppe Rizzo, Marieke van Erp, Genevieve Gorrell, Raphaël Troncy, and Kalina Bontcheva. Analysis of named entity recognition and linking for tweets. *Information Processing and Management*, 51, pages 32–49, 2015. DOI: 10.1016/j.ipm.2014.10.006. 34, 59, 96

[68] Leon Derczynski, Isabelle Augenstein, and Kalina Bontcheva. USFD: Twitter NER with drift compensation and linked data. In *Proc. of the 1st Workshop on Noisy User-generated Text*. Association for Computational Linguistics, 2015. to appear. DOI: 10.18653/v1/w15-4306. 34

[69] Isabelle Augenstein, Leon Derczynski, and Kalina Bontcheva. Generalisation in named entity recognition: A quantitative analysis. *Computer Speech and Language*, 2016. under review. 35

[70] Zhiheng Huang, Wei Xu, and Kai Yu. Bidirectional LSTM-CRF models for sequence tagging. *CoRR*, abs/1508.01991, 2015. 35

[71] Timothy Baldwin, Marie-Catherine de Marneffe, Bo Han, Young-Bum Kim, Alan Ritter, and Wei Xu. Shared tasks of the workshop on noisy user-generated text: Twitter lexical normalization and named entity recognition. In Wei Xu, Bo Han, and Alan Ritter, Eds., *Proc. of the Workshop on Noisy User-generated Text*, pages 126–135, Beijing, China, 2015. Association for Computational Linguistics. DOI: 10.18653/v1/w15-43. 35

[72] Ikuya Yamada, Hideaki Takeda, and Yoshiyasu Takefuji. Enhancing named entity recognition in twitter messages using entity linking. In Rada Mihalcea, Joyce Chai, and Anoop Sarkar, Eds., *Proc. of the Workshop on Noisy User-generated Text*, pages 136–140, Beijing, China, 2015. Association for Computational Linguistics. 35

[73] Isabelle Augenstein, Andreas Vlachos, and Diana Maynard. Extracting relations between non-standard entities using distant supervision and imitation learning. In *Proc. of the Conference on Empirical Methods in Natural Language Processing*, pages 747–757, Lisbon, Portugal, 2015. Association for Computational Linguistics. DOI: 10.18653/v1/d15-1086. 37, 40, 45, 137

[74] Tim Rocktäschel, Sameer Singh, and Sebastian Riedel. Injecting logical background knowledge into embeddings for relation extraction. In *Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2015. DOI: 10.3115/v1/n15-1118. 37

[75] Mihai Surdeanu and Heng Ji. Overview of the english slot filling track at the TAC2014 knowledge base population evaluation. In *Proc. of the TAC-KBP 2014 Workshop*, 2014. 38, 40, 45

[76] Oren Etzioni, Anthony Fader, and Janara Christensen. Open information extraction: The second generation. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2011. 39

[77] Rohit J. Kate and Raymond Mooney. Joint entity and relation extraction using card-pyramid parsing. In Mirella Lapata and Anoop Sarkar, Eds., *Proc. of the 14th Conference*

on Computational Natural Language Learning, pages 203–212, Uppsala, Sweden, 2010. Association for Computational Linguistics. 40

[78] Sameer Singh, Sebastian Riedel, Brian Martin, Jiaping Zheng, and Andrew McCallum. Joint inference of entities, relations, and coreference. In *Proc. of AKBC*, pages 1–6. ACM, 2013. DOI: 10.1145/2509558.2509559. 40

[79] Qi Li and Heng Ji. Incremental joint extraction of entity mentions and relations. In *Proc. of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 402–412, Baltimore, Maryland, 2014. Association for Computational Linguistics. DOI: 10.3115/v1/p14-1038. 40

[80] Heng Ji and Ralph Grishman. Knowledge base population: Successful approaches and challenges. In Yuji Matsumoto and Rada Mihalcea, Eds., *Proc. of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1148–1158, Portland, Oregon, 2011. Association for Computational Linguistics. 40

[81] Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. Distant supervision for relation extraction without labeled data. In Keh-Yih Su, Jian Su, Janyce Wiebe, and Haizhou Li, Eds., *Proc. of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1003–1011, Suntec, Singapore, 2009. Association for Computational Linguistics. DOI: 10.3115/1687878. 40, 45, 47, 48, 137

[82] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proc. of the ACM SIGMOD International Conference on Management of Data*, pages 1247–1250. ACM, 2008. DOI: 10.1145/1376616.1376746. 41

[83] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. YAGO: A large ontology from wikipedia and WordNet. *Web Semantics: Science, Services and Agents on the World Wide Web*, 6(3), pages 203–217, 2008. DOI: 10.1016/j.websem.2008.06.001. 41

[84] Denny Vrandečić and Markus Krötzsch. Wikidata: A free collaborative knowledgebase. *Communications of the ACM*, 57(10), pages 78–85, 2014. DOI: 10.1145/2629489. 41

[85] Sergey Brin. Extracting patterns and relations from the world wide web. In Paolo Atzeni, Alberto Mendelzon, and Giansalvatore Mecca, Eds., *The World Wide Web and Databases*, pages 172–183. Springer, 1999. DOI: 10.1007/10704656. 42

[86] Eugene Agichtein and Luis Gravano. Snowball: Extracting relations from large plain-text collections. In Peter Nürnberg, David Hicks, and Richard Furuta, Eds., *Proc. of the 5th ACM Conference on Digital Libraries*, pages 85–94, 2000. DOI: 10.1145/336597. 42

[87] O. Etzioni, M. Cafarella, D. Downey, S. Kok, A. Popescu, T. Shaked, S. Soderland, D. S. Weld, and A. Yates. Web-scale information extraction in KnowItAll. In Stuart Feldman, Mike Uretsky, Marc Najork, and Craig Wills, Eds., *Proc. of the 13th International Conference on World Wide Web*, Rio de Janeiro, Brazil, 2004. ACM. 43

[88] Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R. Hruschka Jr., and Tom M. Mitchell. Toward an architecture for never-ending language learning. In Maria Fox and David Poole, Eds., *Proc. of the 24th AAAI Conference on Artificial Intelligence*, Palo Alto, California, 2010. AAAI Press. 43, 44

[89] Marti A. Hearst. Automatic acquisition of hyponyms from large text corpora. In *Proc. of the 14th International Conference on Computational Linguistics*, pages 539–545, 1992. DOI: 10.3115/992133.992154. 43

[90] Andrew Carlson, Justin Betteridge, Richard C. Wang, Estevam R. Hruschka, Jr., and Tom M. Mitchell. Coupled semi-supervised learning for information extraction. In *Proc. of the 3rd ACM International Conference on Web Search and Data Mining*, WSDM'10, pages 101–110, New York, NY, 2010. ACM. DOI: 10.1145/1718487.1718501. 44

[91] Saulo D. S. Pedro and Estevam R. Hruschka Jr. Conversing learning: Active learning and active social interaction for human supervision in never-ending learning systems. In Rubén Fuentes-Fernández Juan Pavón, Néstor D. Duque-Méndez, Ed., *Advances in Artificial Intelligence—IBERAMIA 2012*, pages 231–240. Springer, 2012. DOI: 10.1007/978-3-642-34654-5. 44

[92] Stephen Soderland. *Learning Text Analysis Rules for Domain Specific Natural Language Processing*. Ph.D. thesis, University of Massachusetts, Amherst, MA, 1997. 44

[93] Warren Shen, AnHai Doan, Jeffrey F. Naughton, and Raghu Ramakrishnan. Declarative information extraction using datalog with embedded extraction predicates. In Christoph Koch, Johannes Gehrke, Minos N. Garofalakis, Divesh Srivastava, Karl Aberer, Anand Deshpande, Daniela Florescu, Chee Yong Chan, Venkatesh Ganti, Carl-Christian Kanne, Wolfgang Klas, and Erich J. Neuhold, Eds., *VLDB*, pages 1033–1044. ACM, 2007.

[94] Frederick Reiss, Sriram Raghavan, Rajasekar Krishnamurthy, Huaiyu Zhu, and Shivakumar Vaithyanathan. An algebraic approach to rule-based information extraction. In *Proc. of the 24th IEEE International Conference on Data Engineering*, pages 933–942. IEEE, 2008. DOI: 10.1109/icde.2008.4497502. 44

[95] Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *Proc. of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 601–610. ACM, 2014. DOI: 10.1145/2623330.2623623. 44, 50, 137

[96] Aron Culotta and Jeffrey Sorensen. Dependency tree kernels for relation extraction. In *Proc. of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), Main Volume*, pages 423–429, Barcelona, Spain, 2004. DOI: 10.3115/1218955.1219009. 45

[97] Razvan Bunescu and Raymond Mooney. A shortest path dependency kernel for relation extraction. In Raymond Mooney, Chris Brew, Program Co-chair Lee-Feng Chien, Academia Sinica, and Program Co-chair Katrin Kirchhoff, University of Washington, Eds., *Proc. of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 724–731, Vancouver, British Columbia, Canada, 2005. Association for Computational Linguistics. 45

[98] Jason Weston, Antoine Bordes, Oksana Yakhnenko, and Nicolas Usunier. Connecting language and knowledge bases with embedding models for relation extraction. In David Yarowsky, Timothy Baldwin, Anna Korhonen, Karen Livescu, and Steven Bethard, Eds., *Proc. of the Conference on Empirical Methods in Natural Language Processing*, pages 1366–1371, Seattle, Washington, 2013. Association for Computational Linguistics. 45

[99] Alexander Yates, Michele Banko, Matthew Broadhead, Michael Cafarella, Oren Etzioni, and Stephen Soderland. TextRunner: Open information extraction on the Web. In Bob Carpenter, Amanda Stent, and Jason D. Williams, Eds., *Proc. of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 25–26, Rochester, New York, 2007. Association for Computational Linguistics. DOI: 10.3115/1614164. 46

[100] Anthony Fader, Stephen Soderland, and Oren Etzioni. Identifying relations for open information extraction. In David Yarowsky, Timothy Baldwin, Anna Korhonen, Karen Livescu, and Steven Bethard, Eds., *Proc. of the Conference on Empirical Methods in Natural Language Processing*, pages 1535–1545, Seattle, Washington, 2013. Association for Computational Linguistics. 46

[101] Mausam, Michael Schmitz, Stephen Soderland, Robert Bart, and Oren Etzioni. Open language learning for information extraction. In Jun'ichi Tsujii, James Henderson, and Marius Pasça, Eds., *Proc. of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 523–534, Jeju Island, Korea, 2012. Association for Computational Linguistics. 46

[102] Gabor Angeli, Melvin Jose Johnson Premkumar, and Christopher D. Manning. Leveraging linguistic structure for open domain information extraction. In Chengqing Zong and Michael Strube, Eds., *Proc. of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 344–354, Beijing, China, 2015. Association for Computational Linguistics. DOI: 10.3115/v1/p15-1. 47

[103] Mark Craven, Johan Kumlien, et al. Constructing biological knowledge bases by extracting information from text sources. In Thomas Lengauer, Reinhard Schneider, Peer Bork, Douglas Brutlag, Janice Glasgow, Hans-Werner Mewes, and Ralf Zimmer, Eds., *Proc. of the International Conference on Intelligent Systems for Molecular Biology*, volume 1999, pages 77–86, Palo Alto, California, 1999. AAAI Press. 47

[104] Isabelle Augenstein, Diana Maynard, and Fabio Ciravegna. Relation extraction from the Web using distant supervision. In Krzysztof Janowicz, Stefan Schlobach, Patrick Lambrix, and Eero Hyvönen, Eds., *EKAW*, volume 8876 of *Lecture Notes in Computer Science*, pages 26–41, Heidelberg, Germany, 2014. Springer. 48

[105] Isabelle Augenstein, Diana Maynard, and Fabio Ciravegna. Distantly supervised web relation extraction for knowledge base population. *Semantic Web Journal*, 7, 2016. DOI: 10.3233/sw-150180. 48

[106] Benjamin Roth, Tassilo Barth, Michael Wiegand, and Dietrich Klakow. A survey of noise reduction methods for distant supervision. In Fabian Suchanek, Sebastian Riedel, Sameer Singh, and Partha Pratim Talukdar, Eds., *Proc. of the Workshop on Automated Knowledge Base Construction*, pages 73–78, New York, NY, 2013. ACM. DOI: 10.1145/2505515.2505806. 48

[107] Sebastian Riedel, Limin Yao, Andrew McCallum, and Benjamin M. Marlin. Relation extraction with matrix factorization and universal schemas. In Lucy Vanderwende, Hal Daumé III, and Katrin Kirchhoff, Eds., *Proc. of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 74–84, Atlanta, Georgia, 2013. Association for Computational Linguistics. 48, 137

[108] Sebastian Krause, Hong Li, Hans Uszkoreit, and Feiyu Xu. Large-scale learning of relation-extraction rules with distant supervision from the Web. In Philippe Cudré-Mauroux, Jeff Heflin, Evren Sirin, Tania Tudorache, Jérôme Euzenat, Manfred Hauswirth, Josiane Xavier Parreira, Jim Hendler, Guus Schreiber, Abraham Bernstein, and Eva Blomqvist, Eds., *International Semantic Web Conference (1)*, volume 7649 of *Lecture Notes in Computer Science*, pages 263–278. Springer, 2012. DOI: 10.1007/978-3-642-35173-0. 49

[109] Gabor Angeli, Julie Tibshirani, Jean Wu, and Christopher D. Manning. Combining distant and partial supervision for relation extraction. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, Eds., *Proc. of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1556–1567, Doha, Qatar, 2014. Association for Computational Linguistics. DOI: 10.3115/v1/d14-1. 49, 50

[110] Tim Rocktäschel, Sameer Singh, and Sebastian Riedel. Injecting logical background knowledge into embeddings for relation extraction. In Rada Mihalcea, Joyce Chai, and

Anoop Sarkar, Eds., *Proc. of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1119–1129, Denver, Colorado, 2015. Association for Computational Linguistics. 49, 137

[111] D. Rao, P. McNamee, and M. Dredze. Entity linking: Finding extracted entities in a knowledge base. In *Multi-source, Multi-lingual Inf. Extraction and Summarization*. Springer, 2013. DOI: 10.1007/978-3-642-28569-1_5. 53, 59

[112] Silviu Cucerzan. Large-scale named entity disambiguation based on wikipedia data. In *Proc. of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 708–716, 2007. 53

[113] A. Burman, A. Jayapal, S. Kannan, M. Kavilikatta, A. Alhelbawy, L. Derczynski, and R. Gaizauskas. USFD at KBP 2011: Entity linking, slot filling and temporal bounding. In *Proc. of the Text Analysis Conference (TAC'11)*, 2011.

[114] D. Milne and I. H. Witten. Learning to link with wikipedia. In *Proc. of the 17th Conference on Information and Knowledge Management (CIKM)*, pages 509–518, 2008. DOI: 10.1145/1458082.1458150. 53, 57, 95

[115] P. N. Mendes, M. Jakob, A. García-Silva, and C. Bizer. DBpedia spotlight: Shedding light on the web of documents. In *Proc. of I-SEMANTICS*, pages 1–8, 2011. DOI: 10.1145/2063518.2063519. 53, 55, 95, 119

[116] J. Hoffart, M. A. Yosef, I. Bordino, H. Furstenau, M. Pinkal, M. Spaniol, B. Taneva, S. Thater, and G. Weikum. Robust disambiguation of named entities in text. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 782–792, 2011. 53, 54, 57

[117] W. Shen, J. Wang, P. Luo, and M. Wang. LINDEN: Linking named entities with knowledge base via semantic knowledge. In *Proc. of the 21st Conference on World Wide Web*, pages 449–458, 2012. DOI: 10.1145/2187836.2187898. 53, 57

[118] Z.C. Zheng, X.C. Si, F.T. Li, E.Y. Chang, and X.Y. Zhu. Entity disambiguation with freebase. In *Proc. of the Conference on Web Intelligence (WI-IAT'13)*, 2013. DOI: 10.1109/wi-iat.2012.26. 53

[119] G. Rizzo, R. Troncy, S. Hellmann, and M. Bruemmer. NERD meets NIF: Lifting NLP extraction results to the linked data cloud. In *5th Workshop on Linked Data on the Web (LDoW)*, 2012. 53, 58

[120] Ricardo Usbeck, Axel-Cyrille Ngonga Ngomo, Sören Auer, Daniel Gerber, and Andreas Both. Agdistis—graph-based disambiguation of named entities using linked data. In *International Semantic Web Conference*. 2014. DOI: 10.1007/978-3-319-11964-9_29. 53, 57

[121] E. Meij, W. Weerkamp, and M. de Rijke. Adding semantics to microblog posts. In *Proc. of the 5th International Conference on Web Search and Data Mining (WSDM)*, 2012. DOI: 10.1145/2124295.2124364. 54, 55, 95

[122] Marco Cornolti, Paolo Ferragina, and Massimiliano Ciaramita. A framework for benchmarking entity-annotation systems. In *Proc. of the 22nd International Conference on World Wide Web*, WWW'13, pages 249–260, 2013. DOI: 10.1145/2488388.2488411. 54, 57, 59

[123] H. Ji, R. Grishman, H. T. Dang, K. Griffitt, and J. Ellis. Overview of the tac knowledge base population track. In *Proc. of the 3rd Text Analysis Conference*, 2010. 54

[124] H. Ji and R. Grishman. Knowledge base population: Successful approaches and challenges. In *Proc. of ACL'2011*, pages 1148–1158, 2011. 54

[125] Johannes Hoffart, Yasemin Altun, and Gerhard Weikum. Discovering emerging entities with ambiguous names. In *Proc. of the 23rd International Conference on World Wide Web*, WWW'14, pages 385–396, 2014. DOI: 10.1145/2566486.2568003. 54, 57

[126] A. Ritter, S. Clark, Mausam, and O. Etzioni. Named entity recognition in tweets: An experimental study. In *Proc. EMNLP*, 2011. 55

[127] M. Rowe, M. Stankovic, A. S. Dadzie, B. P. Nunes, and A. E. Cano. Making sense of microposts (#msm2013): Big things come in small packages. In *Proc. of the WWW Conference—Workshops*, 2013. 55

[128] Amparo Elizabeth Cano Basave, Giuseppe Rizzo, Andrea Varga, Matthew Rowe, Milan Stankovic, and Aba-Sah Dadzie. Making sense of microposts (#microposts2014) named entity extraction and linking challenge. In *4th Workshop on Making Sense of Microposts (#Microposts2014)*, 2014. 55

[129] Genevieve Gorrell, Johann Petrak, and Kalina Bontcheva. Using @Twitter conventions to improve #lod-based named entity disambiguation. In *The Semantic Web. Latest Advances and New Domains*, pages 171–186. Springer, 2015. DOI: 10.1007/978-3-319-18818-8_11. 55, 60, 97

[130] M. Michelson and S. A. Macskassy. Discovering users' topics of interest on Twitter: A first look. In *Proc. of the 4th Workshop on Analytics for Noisy Unstructured Text Data*, AND'10, pages 73–80, 2010. DOI: 10.1145/1871840.1871852. 55, 124

[131] David Milne and Ian H. Witten. An effective, low-cost measure of semantic relatedness obtained from wikipedia links. In *Proc. of AAAI*, 2008. 56

[132] Paolo Ferragina and Ugo Scaiella. Tagme: On-the-fly annotation of short text fragments (by wikipedia entities). In *Proc. of the 19th ACM International Conference on Information*

and Knowledge Management, CIKM'10, pages 1625–1628, New York, NY, 2010. DOI: 10.1145/1871437.1871689. 57

[133] H. Saif, Y. He, and H. Alani. Alleviating data sparsity for Twitter sentiment analysis. In *Proc. of the #MSM2012 Workshop, CEUR*, volume 838, 2012. 58, 101

[134] F. Abel, Q. Gao, G. J. Houben, and K. Tao. Semantic enrichment of Twitter posts for user profile construction on the social web. In *ESWC (2)*, pages 375–389, 2011. DOI: 10.1007/978-3-642-21064-8_26. 58, 59, 89, 96, 102, 122, 123, 124

[135] M. Rowe and M. Stankovic. Aligning tweets with events: Automation via semantics. *Semantic Web*, 1, 2009. DOI: 10.3233/SW-2011-0042. 58, 100

[136] S. Carter, W. Weerkamp, and E. Tsagkias. Microblog language identification: Overcoming the limitations of short, unedited and idiomatic text. *Language Resources and Evaluation Journal*, 2013. DOI: 10.1007/s10579-012-9195-y. 59, 89, 96, 97

[137] U. Lösch and D. Müller. Mapping microblog posts to encyclopedia articles. *Lecture Notes in Informatics*, 192(150), 2011. 59

[138] Sherzod Hakimov, Salih Atilay Oto, and Erdogan Dogdu. Named entity recognition and disambiguation using linked data and graph-based centrality scoring. In *Proc. of the 4th International Workshop on Semantic Web Information Management*, 2012. DOI: 10.1145/2237867.2237871. 59

[139] Wei Shen, Jianyong Wang, Ping Luo, and Min Wang. Linking named entities in tweets with knowledge base via user interest modeling. In *Proc. of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 68–76. ACM, 2013. DOI: 10.1145/2487575.2487686. 59, 96

[140] Hongzhao Huang, Yunbo Cao, Xiaojiang Huang, Heng Ji, and Chin-Yew Lin. Collective tweet wikification based on semi-supervised graph regularization. In *Proc. of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 380–390, 2014. DOI: 10.3115/v1/p14-1036. 59, 96

[141] Abhishek Gattani, Digvijay S Lamba, Nikesh Garera, Mitul Tiwari, Xiaoyong Chai, Sanjib Das, Sri Subramaniam, Anand Rajaraman, Venky Harinarayan, and AnHai Doan. Entity extraction, linking, classification, and tagging for social media: A wikipedia-based approach. *Proc. of the VLDB Endowment*, 6(11), pages 1126–1137, 2013. DOI: 10.14778/2536222.2536237. 59, 96

[142] Jens Lehmann and Johanna Völker. *Perspectives on Ontology Learning*, volume 18. IOS Press, 2014. 61

[143] Paul Buitelaar and Philipp Cimiano. *Ontology Learning and Population: Bridging the Gap Between Text and Knowledge*, volume 167. IOS Press, 2008.

[144] P. Buitelaar, P. Cimiano, and B. Magnini. *Ontology Learning from Text: Methods, Applications and Evaluation*. IOS Press, 2005. 61

[145] T. Berners-Lee, D. Connolly, and R. R. Swick. Web architecture: Describing and exchanging data. Technical report, W3C Consortium, `http://www.w3.org/\protect\discretionary{\char\hyphenchar\font}{}{}1999/04/WebData`, 1999. 62

[146] Nitin Indurkhya and Fred J. Damerau. *Handbook of Natural Language Processing*, volume 2. CRC Press, 2010. 63

[147] G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, 1983. 64

[148] W. Bosma and P. Vossen. Bootstrapping language-neutral term extraction. In *7th Language Resources and Evaluation Conference (LREC)*, Valletta, Malta, 2010. 65

[149] K. T. Frantzi and S. Ananiadou. The C-Value/NC-Value domain independent method for multi-word term extraction. *Journal of Natural Language Processing*, 6(3), pages 145–179, 1999. DOI: 10.5715/jnlp.6.3_145. 65

[150] D. G. Maynard and S. Ananiadou. Identifying terms by their family and friends. In *Proc. of 18th International Conference on Computational Linguistics (COLING)*, Saarbrücken, Germany, 2000. DOI: 10.3115/990820.990897. 65

[151] D. Bourigault. Surface grammatical analysis for the extraction of terminological noun phrases. In *Proc. of 14th International Conference on Computational Linguistics (COLING)*, pages 977–981, Nantes, France, 1992. DOI: 10.3115/992383.992415. 65

[152] S. J. Nelson, N. E. Olson, L. Fuller, M. S. Tuttle, W. G. Cole, and D. D. Sherertz. Identifying concepts in medical knowledge. In *Proc. of 8th World Congress on Medical Informatics (MEDINFO)*, pages 33–36, 1995. 65

[153] Alexander Maedche and Steffen Staab. Ontology learning. In *Handbook on Ontologies*, pages 173–190. Springer, 2004. DOI: 10.1007/978-3-540-24750-0_9. 66

[154] Gerard Salton, Anita Wong, and Chung-Shu Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18(11), pages 613–620, 1975. DOI: 10.1145/361219.361220. 66

[155] G. Heyer and H. F. Witschel. Terminology and metadata—on how to efficiently build an ontology. *TermNet News—Newsletter of International Cooperation in Terminology*, 87, 2005. 66

[156] Philipp Cimiano, Andreas Hotho, and Steffen Staab. Comparing conceptual, divise and agglomerative clustering for learning taxonomies from text. In *Proc. of the 16th Eureopean Conference on Artificial Intelligence, ECAI'2004, Including Prestigious Applicants of Intelligent Systems, PAIS*, 2004. 66

[157] Diana Maynard. *Term Recognition Using Combined Knowledge Sources*. Ph.D. thesis, Department of Computing and Mathematics, Manchester Metropolitan University, UK, 1999. 66, 67

[158] G. Grefenstette. *Explorations in Automatic Thesaurus Discovery*. Kluwer Academic Publishers, 1994. DOI: 10.1007/978-1-4615-2710-7. 66

[159] G. Rigau, J. Atserias, and E. Agirre. Combining unsupervised lexical knowledge methods for word sense disambiguation. In *Proc. of ACL/EACL*, pages 48–55, Madrid, Spain, 1997. DOI: 10.3115/976909.979624. 67

[160] A. Smeaton and I. Quigley. Experiments on using semantic distances between words in image caption retrieval. In *Proc. of 19th International Conference on Research and Development in Information Retrieval*, Zurich, Switzerland, 1996. DOI: 10.1145/243199.243261. 67

[161] Gang Zhao. *Analogical Translator: Experience-Guided Transfer in Machine Translation*. Ph.D. thesis, Department of Language Engineering, UMIST, Manchester, England, 1996. 67

[162] T. Tsutsumi. Natural language processing: The PLNLP approach. In K. Jenon, G. E Heidhorn, and S. D. Richardson, Eds., *Word Sense Disambiguation by Examples*, pages 263–272. Kluwer Academic Publishers, Dordrecht, 1993. 67

[163] N. Uramoto. A best-match algorithm for broad-coverage example-based disambiguation. In *Proc. of 15th International Conference on Computational Linguistics*, volume 2, pages 717–721, Kyoto, Japan, 1994. DOI: 10.3115/991250.991261. 67

[164] E. Sumita and H. Iida. Experiments and prospects of example-based machine translation. In *Proc. of 29th Annual Meeting of the Association for Computational Linguistics*, pages 185–192, Berkeley, California, 1991. DOI: 10.3115/981344.981368. 67

[165] M. A. Hearst. Automatic acquisition of hyponyms from large text corpora. In *Conference on Computational Linguistics (COLING'92)*, Nantes, France, 1992. Association for Computational Linguistics. DOI: 10.3115/992133.992154. 68

[166] M. Berland and E. Charniak. Finding parts in very large corpora. In *Proc. of ACL-99*, pages 57–64, College Park, MD, 1999. DOI: 10.3115/1034678.1034697. 68

[167] Z. S. Harris. *Mathematical Structures of Language*. Wiley (Interscience), New York, 1968. 68, 69

[168] Frank Smadja. Retrieving collocations from text: Xtract. *Computational Linguistics*, 19(1), pages 143–177, 1993. 68

[169] Wim Peters. Text-based legal ontology enrichment. *Proc. of LOAIT*, pages 55–66, 2009. 68

[170] Naomi Sager. Syntactic formatting of scientific information. In *Proc. of 1972 Fall Joint Computer Conference*, volume 41 of *AFIPS Conf. Proc.*, pages 791–800, Montvale, NJ, 1972. DOI: 10.1145/1480083.1480101. 69

[171] L. Hirschman, R. Grishman, and N. Sager. Grammatically based automatic word class formation. *Information Processing and Retrieval*, 11, pages 39–57, 1975. DOI: 10.1016/0306-4573(75)90033-3. 69

[172] L. Hirschman and N. Sager. Automatic information formatting of a medical sublanguage. In Kittredge and Lehrberger, Eds., *Sublanguage: Studies of Language in Restricted Semantic Domains*, pages 27–69. Walter de Gruyter, 1982. DOI: 10.1515/9783110844818. 69

[173] R. A. Rocha, B. Rocha, and S. M. Huff. Automated translation between medical vocabularies using a frame-based interlingua. In *Proc. of SCAMC'94*, pages 690–694, 1994. 70

[174] P. Cimiano and J. Voelker. Text2Onto—A framework for ontology learning and data-driven change discovery. In *Proc. of the 10th International Conference on Applications of Natural Language to Information Systems (NLDB)*, Alicante, Spain, 2005. 70

[175] D. Maynard, A. Funk, and W. Peters. SPRAT: A tool for automatic semantic pattern-based ontology population. In *International Conference for Digital Libraries and the Semantic Web*, Trento, Italy, 2009. 70

[176] Francesco Draicchio, Aldo Gangemi, Valentina Presutti, and Andrea Giovanni Nuzzolese. FRED: From natural language text to RDF and owl in one click. In *Extended Semantic Web Conference*, pages 263–267. Springer, 2013. DOI: 10.1007/978-3-642-41242-4_36. 70

[177] Johan Bos. Wide-coverage semantic analysis with boxer. In *Proc. of the Conference on Semantics in Text Processing*, pages 277–286. Association for Computational Linguistics, 2008. DOI: 10.3115/1626481.1626503. 70

[178] Aldo Gangemi. Ontology design patterns for semantic web content. In *The Semantic Web-ISWC*, pages 262–276. Springer, 2005. DOI: 10.1007/11574620_21. 71

[179] G. Aguade de Cea, A. Gómez-Pérez, E. Montiel Ponsoda, and M-C. Suárez-Figueroa. Natural language-based approach for helping in the reuse of ontology design patterns. In *Proc. of the 16th International Conference on Knowledge Engineering and Knowledge Management Knowledge Patterns (EKAW)*, Acitrezza, Italy, 2008. DOI: 10.1007/978-3-540-87696-0_6. 71

[180] Kaarel Kaljurand and Norbert E Fuchs. Verbalizing OWL in attempto controlled english. In *OWLED*, volume 258, 2007. DOI: 10.5167/uzh-33256. 71

[181] Cathy Dolbear, Glen Hart, Katalin Kovacs, John Goodwin, and Sheng Zhou. The rabbit language: Description, syntax and conversion to OWL. *Ordinance Survey Research Labs Technical Report*, 2007. 71

[182] Anne Cregan, Rolf Schwitter, Thomas Meyer, et al. Sydney owl syntax-towards a controlled natural language syntax for owl 1.1. In *OWLED*, volume 258, 2007. 71

[183] Adam Funk, Valentin Tablan, Kalina Bontcheva, Hamish Cunningham, Brian Davis, and Siegfried Handschuh. CLOnE: Controlled language for ontology editing. In *Proc. of the 6th International Semantic Web Conference (ISWC)*, Busan, Korea, 2007. DOI: 10.1007/978-3-540-76298-0_11. 71

[184] Diana Maynard and Mark A. Greenwood. Who cares about sarcastic tweets? Investigating the impact of sarcasm on sentiment analysis. In *Proc. of LREC*, Reykjavik, Iceland, 2014. 75, 78, 79

[185] Diana Maynard and Jonathon Hare. Entity-based opinion mining from text and multimedia. In *Advances in Social Media Analysis*, pages 65–86. Springer, 2015. DOI: 10.1007/978-3-319-18458-6_4. 77

[186] Xiaowen Ding, Bing Liu, and Lei Zhang. Entity discovery and assignment for opinion mining applications. In *Proc. of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1125–1134. ACM, 2009. DOI: 10.1145/1557019.1557141. 77

[187] Saif M. Mohammad, Svetlana Kiritchenko, Parinaz Sobhani, Xiaodan Zhu, and Colin Cherry. SemEval-2016 task 6: Detecting stance in tweets. In *Proc. of the International Workshop on Semantic Evaluation*, SemEval'16, San Diego, California, 2016. DOI: 10.18653/v1/s16-1003. 77

[188] Isabelle Augenstein, Tim Rocktäschel, Andreas Vlachos, and Kalina Bontcheva. Stance detection with bidirectional conditional encoding. In *Proc. of EMLNP*, 2016. 77

[189] Leonardo Rocha, Fernando Mourão, Thiago Silveira, Rodrigo Chaves, Giovanni Sá, Felipe Teixeira, Ramon Vieira, and Renato Ferreira. Saci: Sentiment analysis by collective

inspection on social media content. *Web Semantics: Science, Services and Agents on the World Wide Web*, 2015. DOI: 10.1016/j.websem.2015.05.006. 78

[190] Diana Maynard, Gerhard Gossen, Marco Fisichella, and Adam Funk. Should I care about your opinion? Detection of opinion interestingness and dynamics in social media. *Journal of Future Internet*, 2015. DOI: 10.3390/fi6030457. 78

[191] Christine Liebrecht, Florian Kunneman, and Antal van den Bosch. The perfect solution for detecting sarcasm in tweets# not. *WASSA*, page 29, 2013. 78

[192] David Bamman and Noah A Smith. Contextualized sarcasm detection on twitter. In *9th International AAAI Conference on Web and Social Media*, 2015. 79

[193] Ameeta Agrawal and Aijun An. Unsupervised emotion detection from text using semantic and syntactic relations. In *Proc. of the IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technology*, Volume 01, WI-IAT'12, pages 346–353, Washington, DC, 2012. IEEE Computer Society. DOI: 10.1109/wi-iat.2012.170. 79

[194] J. Bollen, A. Pepe, and H. Mao. Modeling public mood and emotion: Twitter sentiment and socio-economic phenomena. `http://arxiv.org/abs/0911.1583`, 2009. 79, 100

[195] Marc Schröder, Paolo Baggia, Felix Burkhardt, Catherine Pelachaud, Christian Peter, and Enrico Zovato. EmotionML—an upcoming standard for representing emotions and related states. In *Affective Computing and Intelligent Interaction*, pages 316–325. Springer, 2011. DOI: 10.1007/978-3-642-24600-5_35. 79

[196] W. Gerrod Parrott. *Emotions in Social Psychology: Essential Readings*. Psychology Press, 2001. 79

[197] Phillip Shaver, Judith Schwartz, Donald Kirson, and Cary O'connor. Emotion knowledge: further exploration of a prototype approach. *Journal of Personality and Social Psychology*, 52(6), page 1061, 1987. DOI: 10.1037//0022-3514.52.6.1061. 79

[198] B. Pang and L. Lee. Opinion mining and sentiment analysis. *Information Retrieval*, 2(1), 2008. DOI: 10.1561/1500000011. 81

[199] S. Moghaddam and F. Popowich. Opinion polarity identification through adjectives. *CoRR*, abs/1011.4623, 2010. 82

[200] A. C. Mullaly, C. L. Gagné, T. L. Spalding, and K. A. Marchak. Examining ambiguous adjectives in adjective-noun phrases: Evidence for representation as a shared core-meaning. *The Mental Lexicon*, 5(1), pages 87–114, 2010. DOI: 10.1075/ml.5.1.04mul. 82

[201] A. Weichselbraun, S. Gindl, and A. Scharl. A context-dependent supervised learning approach to sentiment detection in large textual databases. *Journal of Information and Data Management*, 1(3), pages 329–342, 2010. 83

[202] T. Wilson, J. Wiebe, and P. Hoffmann. Recognizing contextual polarity: An exploration of features for phrase-level sentiment analysis. *Computational Linguistics*, 35(3), pages 399–433, 2009. DOI: 10.1162/coli.08-012-r1-06-90. 83

[203] S. Gindl, A. Weichselbraun, and A. Scharl. Cross-domain contextualisation of sentiment lexicons. In *Proc. of 19th European Conference on Artificial Intelligence (ECAI)*, pages 771–776, 2010. DOI: 10.3233/978-1-60750-606-5-771. 83

[204] A. Aue and M. Gamon. Customizing sentiment classifiers to new domains: A case study. In *Proc. of the International Conference on Recent Advances in Natural Language Processing*, Borovetz, Bulgaria, 2005. 83

[205] Krisztian Balog, Gilad Mishne, and Maarten De Rijke. Why are they excited? Identifying and explaining spikes in blog mood levels. In *Proc. of the 11th Conference of the European Chapter of the Association for Computational Linguistics: Posters and Demonstrations*, pages 207–210. Association for Computational Linguistics, 2006. DOI: 10.3115/1608974.1609010. 83

[206] C. J. Hutto and Eric Gilbert. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *8th International AAAI Conference on Weblogs and Social Media*, 2014. 83

[207] M. Taboada, J. Brooke, M. Tofiloski, K. Voll, and M. Stede. Lexicon-based methods for sentiment analysis. *Computational Linguistics*, 1(September 2010), pages 1–41, 2011. DOI: 10.1162/coli_a_00049. 83

[208] Diego Reforgiato Recupero, Mauro Dragoni, and Valentina Presutti. Eswc'15 challenge on concept-level sentiment analysis. In *Semantic Web Evaluation Challenge*, pages 211–222. Springer, 2015. DOI: 10.1007/978-3-319-12024-9_1. 84

[209] Diego Reforgiato Recupero and Erik Cambria. Eswc'14 challenge on concept-level sentiment analysis. In *Semantic Web Evaluation Challenge*, pages 3–20. Springer, 2015. DOI: 10.1007/978-3-319-12024-9_1. 84

[210] Anni Coden, Dan Gruhl, Neal Lewis, Pablo N. Mendes, Meena Nagarajan, Cartic Ramakrishnan, and Steve Welch. Semantic lexicon expansion for concept-based aspect-aware sentiment analysis. In *Semantic Web Evaluation Challenge*, pages 34–40. Springer, 2014. DOI: 10.1007/978-3-319-12024-9_4. 84

[211] Pablo N. Mendes, Max Jakob, and Christian Bizer. Dbpedia: A multilingual cross-domain knowledge base. In *LREC*, pages 1813–1817, 2012. 84

[212] Pei Yin, Hongwei Wang, and Kaiqiang Guo. Feature—opinion pair identification of product reviews in chinese: A domain ontology modeling method. *New Review of Hypermedia and Multimedia*, 19(1), pages 3–24, 2013. DOI: 10.1080/13614568.2013.766266. 84

[213] Samaneh Moghaddam and Martin Ester. The flda model for aspect-based opinion mining: addressing the cold start problem. In *Proc. of the 22nd international conference on World Wide Web*, pages 909–918. International World Wide Web Conferences Steering Committee, 2013. DOI: 10.1145/2488388.2488467. 84

[214] Mike Thelwall, Kevan Buckley, Georgios Paltoglou, Di Cai, and Arvid Kappas. Sentiment strength detection in short informal text. *Journal of the American Society for Information Science and Technology*, 61(12), pages 2544–2558, 2010. DOI: 10.1002/asi.21416. 85, 119

[215] Peter Pirolli. Powers of 10: Modeling complex information-seeking systems at multiple scales. *IEEE Computer*, 42(3), pages 33–40, 2009. DOI: 10.1109/mc.2009.94. 87

[216] N. Ravikant and A. Rifkin. Why twitter is massively undervalued compared to facebook. *TechCrunch*, 2010. `http://techcrunch.com/2010/10/16/why-twitter-is-massively-undervalued-compared-to-facebook/` 88

[217] Meredith M. Skeels and Jonathan Grudin. When social networks cross boundaries: A case study of workplace use of facebook and linkedIn. In *Proc. of the ACM International Conference on Supporting Group Work*, GROUP'09, pages 95–104, New York, NY, 2009. DOI: 10.1145/1531674.1531689. 88

[218] K. Bontcheva and H. Cunningham. Semantic annotation and retrieval: Manual, semi-automatic and automatic generation. In J. Domingue, D. Fensel, and J. A. Hendler, Eds., *Handbook of Semantic Web Technologies*. Springer, 2011. DOI: 10.1007/978-3-540-92913-0. 88

[219] X. Liu, S. Zhang, F. Wei, and M. Zhou. Recognizing named entities in tweets. In *Proc. of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 359–367, 2011. 88, 96

[220] A. Ritter, S. Clark, Mausam, and O. Etzioni. Named entity recognition in tweets: An experimental study. In *Proc. of Empirical Methods for Natural Language Processing (EMNLP)*, Edinburgh, UK, 2011. 88, 93, 95, 97, 98

[221] P. N. Mendes, A. Passant, P. Kapanipathi, and A. P. Sheth. Linked open social signals. In *Proc. of the IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, WI-IAT'10, pages 224–231, Washington, DC, 2010. IEEE Computer Society. DOI: 10.1109/wi-iat.2010.314. 89, 95, 119

[222] B. Han and T. Baldwin. Lexical normalisation of short text messages: Makn sens a #twitter. In *Proc. of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, HLT'11, pages 368–378, 2011. 89

[223] S. Gouws, D. Metzler, C. Cai, and E. Hovy. Contextual bearing on linguistic variation in social media. In *Proc. of the Workshop on Languages in Social Media*, LSM'11, pages 20–29, 2011. 89

[224] T. Baldwin and M. Lui. Language identification: The long and the short of the matter. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 229–237, Los Angeles, California, 2010. 89

[225] I. Celino, D. Dell'Aglio, E. Della Valle, Y. Huang, T. Lee, S. Park, and V. Tresp. Making sense of location-based micro-posts using stream reasoning. In *Proc. of the Making Sense of Microposts Workshop (#MSM2011), Collocated with the 8th Extended Semantic Web Conference*, Heraklion, Crete, Greece, 2011. 91

[226] S. Scerri, K. Cortis, I. Rivera, and S. Handschuh. Knowledge Discovery in Distributed Social Web Sharing Activities. In *Proc. of the #MSM2012 Workshop, CEUR*, volume 838, 2012. 91

[227] T. Plumbaum, S. Wu, E. W. De Luca, and S. Albayrak. User Modeling for the Social Semantic Web. In *2nd Workshop on Semantic Personalized Information Management: Retrieval and Recommendation, in conjunction with ISWC*, 2011. 91

[228] A. Passant and P. Laublet. Meaning of a tag: A collaborative approach to bridge the gap between tagging and linked data. In *Proc. of the Linked Data on the Web Workshop (LDOW)*, Beijing, China, 2008. 91

[229] A. Passant, J. G. Breslin, and S. Decker. Rethinking microblogging: Open, distributed, semantic. In *Proc. of the 10th International Conference on Web Engineering*, pages 263–277, 2010. DOI: 10.1007/978-3-642-13911-6_18. 91

[230] Dominik Heckmann, Tim Schwartz, Boris Brandherm, Michael Schmitz, and Margeritta von Wilamowitz-Moellendorff. GUMO—the general user model ontology. In *Proc. of the 10th International Conference on User Modeling*, pages 428–432, 2005. DOI: 10.1007/11527886_58. 91

[231] S. Angeletou, M. Rowe, and H. Alani. Modelling and analysis of user behaviour in online communities. In *Proc. of the 10th International Conference on the Semantic Web*, ISWC'11, pages 35–50. Springer-Verlag, 2011. DOI: 10.1007/978-3-642-25073-6_3. 92, 123, 125

[232] M. Rowe, S. Angeletou, and H. Alani. Predicting discussions on the social semantic web. In *Proc. of the 8th Extended Semantic Web Conference on the Semanic Web*, ESWC'11, pages 405–420. Springer-Verlag, 2011. DOI: 10.1007/978-3-642-21064-8_28. 92, 123

[233] R. Mihalcea and P. Tarau. TextRank: Bringing order into text. In *Proc. of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 404–411, 2004. 93

[234] W. Wu, B. Zhang, and M. Ostendorf. Automatic generation of personalized annotation tags for twitter users. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 689–692, 2010. 93, 125

[235] F. Zanzotto, M. Pennaccchiotti, and K. Tsioutsiouliklis. Linguistic Redundancy in Twitter. In *Proc. of the Conference on Empirical Methods in Natural Language Processing*, pages 659–669, Edinburgh, UK, 2011. Association for Computational Linguistics. 93

[236] B. Sharifi, M. A. Hutton, and J. Kalita. Summarizing microblogs automatically. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 685–688, Los Angeles, California, 2010. 93

[237] W. Xin, Z. Jing, J. Jing, H. Yang, S. Palakorn, W. X. Zhao, J. Jiang, J. He, Y. Song, P. Achananuparp, E. P. Lim, and X. Li. Topical keyphrase extraction from twitter. In *Proc. of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, HLT'11, pages 379–388, 2011. 93, 125

[238] Daniel Ramage, Susan Dumais, and Dan Liebling. Characterizing microblogs with topic models. In *Proc. of the 4th International Conference on Weblogs and Social Media (ICWSM)*, 2010. 93, 132

[239] G. Mishne. AutoTag: A collaborative approach to automated tag assignment for weblog posts. In *Proc. of the 15th International Conference on World Wide Web*, pages 953–954, 2006. DOI: 10.1145/1135777.1135961. 93

[240] L. Qu, C. Müller, and I. Gurevych. Using tag semantic network for keyphrase extraction in blogs. In *Proc. of the 17th Conference on Information and Knowledge Management*, pages 1381–1382, 2008. DOI: 10.1145/1458082.1458290. 93

[241] G. Solskinnsbakk and J. A. Gulla. Semantic annotation from social data. In *Proc. of the 4th International Workshop on Social Data on the Web Workshop*, 2011. 93

[242] N. Ireson and F. Ciravegna. Toponym resolution in social media. In *Proc. of the 9th International Semantic Web Conference (ISWC)*, pages 370–385, 2010. DOI: 10.1007/978-3-642-17746-0_24. 95

[243] David Laniado and Peter Mika. Making sense of twitter. In Peter Patel-Schneider, Yue Pan, Pascal Hitzler, Peter Mika, Lei Zhang, Jeff Pan, Ian Horrocks, and Birte Glimm, Eds., *The Semantic Web (ISWC)*, volume 6496 of *Lecture Notes in Computer Science*, pages 470–485. Springer Berlin/Heidelberg, 2010. 95

[244] D. Gruhl, M. Nagarajan, J. Pieper, C. Robson, and A. Sheth. Context and domain knowledge enhanced entity spotting in informal text. In *Proc. of the 8th International Semantic Web Conference (ISWC)*, 2009. DOI: 10.1007/978-3-642-04930-9_17. 95, 104

[245] S. Choudhury and J. Breslin. Extracting semantic entities and events from sports tweets. In *Proc. of the 1st Workshop on Making Sense of Microposts (MSM): Big Things Come in Small Packages*, pages 22–32, 2011. 100

[246] Elizabeth L. Murnane, Bernhard Haslhofer, and Carl Lagoze. Reslve: Leveraging user interest to improve entity disambiguation on short text. In *Proc. of the 22nd International Conference on World Wide Web*, pages 1275–1284, 2013. DOI: 10.1145/2487788.2488162. 96

[247] Chenliang Li, Jianshu Weng, Qi He, Yuxia Yao, Anwitaman Datta, Aixin Sun, and Bu-Sung Lee. Twiner: Named entity recognition in targeted twitter stream. In *Proc. of the 35th ACM Conference on Research and Development in Information Retrieval*, pages 721–730. ACM, 2012. DOI: 10.1145/2348283.2348380. 96

[248] Kalina Bontcheva, Leon Derczynski, Adam Funk, Mark A. Greenwood, Diana Maynard, and Niraj Aswani. TwitIE: An open-source information extraction pipeline for microblog text. In *Proc. of the International Conference on Recent Advances in Natural Language Processing*. Association for Computational Linguistics, 2013. 97, 134

[249] H. Cunningham, D. Maynard, K. Bontcheva, V. Tablan, N. Aswani, I. Roberts, G. Gorrell, A. Funk, A. Roberts, D. Damljanovic, T. Heitz, M. A. Greenwood, H. Saggion, J. Petrak, Y. Li, and W. Peters. *Text Processing with GATE (Version 6)*. 2011. 97

[250] B. Han, P. Cook, and T. Baldwin. Automatically constructing a normalisation dictionary for microblogs. In *Proc. of the Conference on Empirical Methods in Natural Language Processing*, pages 421–432. ACL, 2012. 98

[251] L. Derczynski, D. Maynard, N. Aswani, and K. Bontcheva. Microblog-genre noise and impact on semantic annotation accuracy. In *Proc. of the 24th ACM Conference on Hypertext and Social Media*, 2013. DOI: 10.1145/2481492.2481495. 98

[252] E. Forsyth and C. Martell. Lexical and discourse analysis of online chat dialog. In *International Conference on Semantic Computing*, pages 19–26. IEEE, 2007. DOI: 10.1109/icosc.2007.4338328. 99

[253] M. P. Marcus, B. Santorini, and M. A. Marcinkiewicz. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2), pages 313–330, 1993. 99

[254] M. Dork, D. Gruen, C. Williamson, and S. Carpendale. A visual backchannel for large-scale events. *IEEE Transactions on Visualization and Computer Graphics*, 16(6), pages 1129–1138, 2010. DOI: 10.1109/tvcg.2010.129. 99, 127, 128, 131, 132

[255] S. Petrović, M. Osborne, and V. Lavrenko. Streaming first story detection with application to Twitter. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 181–189, 2010. 99

[256] H. Becker, M. Naaman, and L. Gravano. Learning similarity metrics for event identification in social media. In *Proc. of the 3rd International Conference on Web Search and Web Data Mining*, pages 291–300, 2010. DOI: 10.1145/1718487.1718524.

[257] Hila Becker, Mor Naaman, and Luis Gravano. Beyond trending topics: Real-world event identification on twitter. In *Proc. of the 5th International Conference on Weblogs and Social Media (ICWSM)*, 2011. 99

[258] H. Sayyadi, M. Hurst, and A. Maykov. Event detection and tracking in social streams. In *Proc. of the 3rd International ICWSM Conference*, pages 311–314, 2009. 99

[259] Takeshi Sakaki, M. Okazaki, and Y. Matsuo. Earthquake shakes twitter users: Real-time event detection by social sensors. In *Proc. of the 19th International Conference on World Wide Web (WWW)*, pages 851–860. ACM, 2010. DOI: 10.1145/1772690.1772777. 99

[260] J. Y. Weng, C. L. Yang, B. N. Chen, Y. K. Wang, and S. D. Lin. IMASS: An intelligent microblog analysis and summarization system. In *Proc. of the ACL-HLT System Demonstrations*, pages 133–138, Portland, Oregon, 2011. 99, 127, 131

[261] M. Nagarajan, K. Gomadam, A. Sheth, A. Ranabahu, R. Mutharaju, and A. Jadhav. Spatio-temporal-thematic analysis of citizen sensor data: Challenges and experiences. In *Web Information Systems Engineering*, pages 539–553, 2009. DOI: 10.1007/978-3-642-04409-0_52. 100, 127, 128, 129

[262] A. Marcus, M. S. Bernstein, O. Badar, D. R. Karger, S. Madden, and R. C. Miller. TwitInfo: Aggregating and visualizing microblogs for event exploration. In *Proc. of the Conference on Human Factors in Computing Systems (CHI)*, pages 227–236, 2011. DOI: 10.1145/1978942.1978975. 100, 127, 128, 131

[263] M. Naaman, J. Boase, and C. Lai. Is it really about me? Message content in social awareness streams. In *Proc. of the ACM Conference on Computer Supported Cooperative Work*, pages 189–192. ACM, 2010. DOI: 10.1145/1718918.1718953. 100, 102, 124, 125, 126

[264] B. J. Jansen, M. Zhang, K. Sobel, and A. Chowdury. Twitter power: Tweets as electronic word of mouth. *JASIST*, 60(11), pages 2169–2188, 2009. DOI: 10.1002/asi.21149. 100

[265] Patrick Lai. Extracting strong sentiment trends from twitter. http://nlp.stanford.edu/courses/cs224n/2011/reports/patlai.pdf, 2010. 100

[266] Diana Maynard, Kalina Bontcheva, and Dominic Rout. Challenges in developing opinion mining tools for social media. In *Proc. of @NLP can u tag #usergeneratedcontent?! Workshop at LREC 2012*, Turkey, 2012. 100

[267] A. Pak and P. Paroubek. Twitter based system: Using twitter for disambiguating sentiment ambiguous adjectives. In *Proc. of the 5th International Workshop on Semantic Evaluation*, pages 436–439, 2010. 101

[268] A. Go, R. Bhayani, and L. Huang. Twitter sentiment classification using distant supervision. Technical Report CS224N Project Report, Stanford University, 2009. 101

[269] N. Diakopoulos, M. Naaman, and F. Kivran-Swaine. Diamonds in the rough: Social media visual analytics for journalistic inquiry. In *Proc. of the IEEE Conference on Visual Analytics Science and Technology*, pages 115–122, 2010. DOI: 10.1109/vast.2010.5652922. 101, 127

[270] D. Maynard and A. Funk. Automatic detection of political opinions in tweets. In Dieter Fensel Raúl García-Castro and Grigoris Antoniou, Eds., *The Semantic Web: ESWC 2011 Selected Workshop Papers, Lecture Notes in Computer Science*. Springer, 2011. 101

[271] D. Maynard, M. A. Greenwood, I. Roberts, G. Windsor, and K. Bontcheva. Real-time social media analytics through semantic annotation and linked open data. In *Proc. of WebSci*, Oxford, UK, 2015. DOI: 10.1145/2786451.2786500. 101

[272] M. Dowman, V. Tablan, H. Cunningham, and B. Popov. Web-assisted annotation, semantic indexing and search of television and radio news. In *Proc. of the 14th International World Wide Web Conference*, Chiba, Japan, 2005. DOI: 10.1145/1060745.1060781. 102

[273] A. Hubmann-Haidvogel, A. M. P. Brasoveanu, A. Scharl, M. Sabou, and S. Gindl. Visualizing contextual and dynamic features of micropost streams. In *Proc. of the #MSM2012 Workshop, CEUR*, volume 838, 2012. 102, 127, 128, 131

[274] W. X. Zhao, J. Jiang, J. Weng, J. He, E. Lim, H. Yan, and X. Li. Comparing twitter and traditional media using topic models. In *Proc. of the 33rd European Conference on Advances in Information Retrieval (ECIR)*, pages 338–349, 2011. DOI: 10.1007/978-3-642-20161-5_34. 102

[275] Stephan Lewandowsky, Ullrich KH Ecker, Colleen M Seifert, Norbert Schwarz, and John Cook. Misinformation and its correction continued influence and successful debiasing. *Psychological Science in the Public Interest*, 13(3), pages 106–131, 2012. DOI: 10.1177/1529100612451018. 102

[276] Rob Procter, Jeremy Crump, Susanne Karstedt, Alex Voss, and Marta Cantijoch. Reading the riots: What were the police doing on twitter? *Policing and Society*, 23(4), pages 413–436, 2013. DOI: 10.1080/10439463.2013.780223. 102

[277] Mendoza Marcelo, Poblete Barbara, and Castillo Carlos. Twitter under crisis: Can we trust what we are? In *1st Workshop on Social Media Analytics (SOMA)*, 2010. DOI: 10.1145/1964858.1964869. 102

[278] Rob Procter, Farida Vis, and Alex Voss. Reading the riots on twitter: Methodological innovation for the analysis of big data. *International Journal of Social Research Methodology*, 16(3), pages 197–214, 2013. DOI: 10.1080/13645579.2013.774172. 102

[279] Zhe Zhao, Paul Resnick, and Qiaozhu Mei. Early detection of rumors in social media from enquiry posts. In *International World Wide Web Conference Committee (IW3C2)*, 2015. 102, 103

[280] Jing Ma, Wei Gao, Zhongyu Wei, Yueming Lu, and Kam-Fai Wong. Detect rumors using time series of social context information on microblogging websites. In *Proc. of the 24th ACM International on Conference on Information and Knowledge Management*, CIKM'15, pages 1751–1754, New York, NY, 2015. ACM. 102

[281] Vahed Qazvinian, Emily Rosengren, Dragomir R. Radev, and Qiaozhu Mei. Rumor has it: Identifying misinformation in microblogs. In *Proc. of the Conference on Empirical Methods in Natural Language Processing*, EMNLP'11, pages 1589–1599, 2011. 103

[282] Sardar Hamidian and Mona T Diab. Rumor identification and belief investigation on twitter. In *Proc. of NAACL-HLT*, pages 3–8, 2016. DOI: 10.18653/v1/w16-0403. 103

[283] Michal Lukasik, Kalina Bontcheva, Trevor Cohn, Arkaitz Zubiaga, Maria Liakata, and Rob Procter. Using Gaussian processes for rumour stance classification in social media. *CoRR*, abs/1609.01962, 2016. 103

[284] Xiaomo Liu, Armineh Nourbakhsh, Quanzhi Li, Rui Fang, and Sameena Shah. Real-time rumor debunking on twitter. In *Proc. of the 24th ACM International on Conference on Information and Knowledge Management*, CIKM'15, pages 1867–1870, New York, NY, 2015. ACM. 103

[285] Michal Lukasik, Trevor Cohn, and Kalina Bontcheva. Classifying tweet level judgements of rumours in social media. In *Proc. of the Conference on Empirical Methods in Natural Language Processing, EMNLP* Lisbon, Portugal, pages 2590–2595, 2015. DOI: 10.18653/v1/d15-1311. 103

[286] Li Zeng, Kate Starbird, and Emma S. Spiro. # unconfirmed: Classifying rumor stance in crisis-related social media messages. In *10th International AAAI Conference on Web and Social Media*, 2016. 103

[287] Arkaitz Zubiaga, Maria Liakata, Rob Procter, Geraldine Wong Sak Hoi, and Peter Tolmie. Analysing how people orient to and spread rumours in social media by looking at conversational threads. *PLoS ONE*, 11(3), pages 1–29, 2016. DOI: 10.1371/journal.pone.0150989. 103

[288] G. Demartini, D. E. Difallah, and P. Cudré-Mauroux. ZenCrowd: Leveraging probabilistic reasoning and crowdsourcing techniques for large-scale entity linking. In *Proc. of the 21st Conference on World Wide Web*, pages 469–478, 2012. DOI: 10.1145/2187836.2187900. 103

[289] Christian M. Meyer and Iryna Gurevych. Wiktionary: A new rival for expert-built lexicons? Exploring the possibilities of collaborative lexicography. In *Electronic Lexicography*. Oxford University Press, 2012. DOI: 10.1093/acprof:oso/9780199654864.003.0013. 104, 136

[290] Iryna Gurevych, Judith Eckle-Kohler, Silvana Hartmann, Michael Matuschek, Christian M. Meyer, and Christian Wirth. UBY: A large-scale unified lexical-semantic resource based on LMF. In *13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 580–590, 2012. 104, 136

[291] Paul Buitelaar, Philipp Cimiano, Peter Haase, and Michael Sintek. Towards linguistically grounded ontologies. In *Proc. of the European Semantic Web Conference (ESWC'09), LNCS 5554*, pages 111—125, 2009. DOI: 10.1007/978-3-642-02121-3_12. 104, 136

[292] T. Finin, W. Murnane, A. Karandikar, N. Keller, J. Martineau, and M. Dredze. Annotating named entities in twitter data with crowdsourcing. In *Proc. of the NAACL HLT Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 80–88, 2010. 104, 139

[293] D. Maynard and M. A. Greenwood. Large scale semantic annotation, indexing and search at the national archives. In *Proc. of LREC 2012*, Turkey, 2012. 105, 113, 138

[294] Kalina Bontcheva, Valentin Tablan, and Hamish Cunningham. Semantic search over documents and ontologies. In *Bridging Between Information Retrieval and Databases*, volume 8173, pages 31–53. Springer Verlag, 2014. DOI: 10.1007/978-3-642-54798-0_2. 107

[295] V. Tablan, K. Bontcheva, I. Roberts, and H. Cunningham. Mímir: An open-source semantic search framework for interactive information seeking and discovery. *Journal of Web Semantics*, 30, pages 52–68, 2015. DOI: 10.1016/j.websem.2014.10.002. 107, 110, 111, 113, 120

[296] A. Kiryakov, B. Popov, D. Ognyanoff, D. Manov, A. Kirilov, and M. Goranov. Semantic annotation, indexing and retrieval. *Journal of Web Semantics*, 1(2), pages 671–680, 2004. DOI: 10.1016/j.websem.2004.07.005. 107, 109, 112

[297] Hamish Cunningham, Valentin Tablan, Ian Roberts, Mark A. Greenwood, and Niraj Aswani. Information extraction and semantic annotation for multi-paradigm information management. In Mihai Lupu, Katja Mayer, John Tait, and Anthony J. Trippe, Eds., *Current Challenges in Patent Information Retrieval*, volume 29 of *The Information Retrieval Series*, pages 307–327. Springer Berlin Heidelberg, 2011. DOI: 10.1007/978-3-642-19231-9. 107, 109, 138

[298] K. Mahesh, J. Kud, and P. Dixon. Oracle at TREC8: A lexical approach. In *Proc. of the 8th Text Retrieval Conference (TREC-8)*, 1999. 107

[299] E. Voorhees. Using WordNet for text retrieval. In C. Fellbaum, Ed., *WordNet: An Electronic Lexical Database*. MIT Press, 1998. 107

[300] Li Ding, Tim Finin, Anupam Joshi, Rong Pan, R. Scott Cost, Yun Peng, Pavan Reddivari, Vishal C. Doshi, and Joel Sachs. Swoogle: A search and metadata engine for the semantic web. In *Proc. of the 13th ACM Conference on Information and Knowledge Management*, 2004. DOI: 10.1145/1031171.1031289. 108

[301] M. Hildebrand, J. van Ossenbruggen, and J. Hardman. Facet: A browser for heterogeneous semantic web repositories. In *Proc. of the 5th International Semantic Web Conference*, 2006. DOI: 10.1007/11926078_20. 108

[302] G. Klyne and J. Carroll. Resource description framework (RDF): Concepts and abstract syntax. W3C recommendation, W3C, 2004. `http://www.w3.org/TR/rdf-concepts/` 108

[303] Sean Bechhofer, Frank van Harmelen, Jim Hendler, Ian Horrocks, Deborah L. McGuinness, Peter F. Patel-Schneider, and Lynn Andrea Stein. OWL web ontology language reference. W3C recommendation, W3C, `http://www.w3.org/`, 2004. 108

[304] Eric Prud'hommeaux and Andy Seaborne. SPARQL Query language for RDF. W3C recommendation, W3C, `http://www.w3.org/TR/rdf-sparql-query/`, 2008. 108

[305] Hannah Bast, Florian Bäurle, Björn Buchhold, and Elmar Haussmann. A case for semantic full-text search. In *Proc. of the 1st Joint International Workshop on Entity-Oriented and Semantic Search*, JIWES'12, pages 4:1–4:3. ACM, 2012. DOI: 10.1145/2379307.2379311. 109

[306] Hannah Bast, Florian Bäurle, Björn Buchhold, and Elmar Haussmann. Broccoli: Semantic full-text search at your fingertips. *CoRR*, abs/1207.2615, 2012. 109, 110, 111

[307] Amit Singhal. Introducing the knowledge graph: Things, not strings. http://googleblog.blogspot.it/2012/05/introducing-knowledge-graph-things-not.html, 2012. 109

[308] K. Bontcheva, J. Kieniewicz, S. Andrews, and M. Wallis. Semantic enrichment and search: A case study on environmental science literature. *D-Lib Magazine*, 21(1/2), 2015. DOI: 10.1045/january2015-bontcheva. 110, 116

[309] J. Kieniewicz, A. Sudlow, and E. Newbold. Coordinating improved environmental information access and discovery: Innovations in sharing environmental observations and information. In W. Pillman, S. Schade, and P.Smits, Eds., *Proc. of the 25th International EnviroInfo Conference*, 2011. 110

[310] J. Kieniewicz and M. Wallis. User requirements. Technical Report http://gate.ac.uk/projects/envilod/EnviLOD-WP2-User-Requirements.pdf, EnviLOD project deliverable, 2012. 110

[311] Mihai Lupu and Allan Hanbury. Patent retrieval. *Foundations and Trends in Information Retrieval*, 7(1), pages 1–97, 2013. DOI: 10.1561/1500000027. 110

[312] Lei Zhang, Qiaoling Liu, Jie Zhang, Haofen Wang, Yue Pan, and Yong Yu. Semplore: An ir approach to scalable hybrid query of semantic web data. In *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC-ASWC*, pages 652–665, 2007. DOI: 10.1007/978-3-540-76298-0_47. 110, 111

[313] Miriam Fernández, Iván Cantador, Vanesa López, David Vallet, Pablo Castells, and Enrico Motta. Semantically enhanced information retrieval: An ontology-based approach. *Web Semantics*, 9(4), pages 434–452, 2011. DOI: 10.1016/j.websem.2010.11.003. 111

[314] Haofen Wang, Thanh Tran, Chang Liu, and Linyun Fu. Lightweight integration of ir and db for scalable hybrid search with integrated ranking support. *Web Semantics: Science, Services and Agents on the World Wide Web*, 9(4), 2011. DOI: 10.1016/j.websem.2011.08.002. 111

[315] Bettina Fazzinga, Giorgio Gianforme, Georg Gottlob, and Thomas Lukasiewicz. Semantic web search based on ontological conjunctive queries. *Web Semantics: Science, Services and Agents on the World Wide Web*, 9(4), 2011. DOI: 10.1016/j.websem.2011.08.003. 111

[316] Nikos Bikakis, Giorgos Giannopoulos, Theodore Dalamagas, and Timos Sellis. Integrating keywords and semantics on document annotation and search. In Robert Meersman, Tharam Dillon, and Pilar Herrero, Eds., *On the Move to Meaningful Internet Systems*, volume 6427, pages 921–938. Springer, 2010. DOI: 10.1007/978-3-540-88871-0. 111

[317] Borislav Popov, Atanas Kiryakov, Angel Kirilov, Dimitar Manov, Damyan Ognyanoff, and Miroslav Goranov. KIM—Semantic annotation platform. In *2nd International Semantic Web Conference (ISWC)*, pages 484–499, Berlin, 2003. Springer. DOI: 10.1007/978-3-540-39718-2_53. 112, 132

[318] Atanas Kiryakov. OWLIM: Balancing between scalable repository and light-weight reasoner. In *Proc. of the 15th International World Wide Web Conference (WWW), 2010*, Edinburgh, Scotland, 2006. 113

[319] Paolo Boldi and Sebastiano Vigna. MG4J at TREC 2005. In Ellen M. Voorhees and Lori P. Buckland, Eds., *Proc. of the 14th Text REtrieval Conference (TREC)*, volume 500 of *Special Publications*, pages 266–271. NIST, 2005. `http://mg4j.dsi.unimi.it/` 113

[320] V. Tablan, I. Roberts, H. Cunningham, and K. Bontcheva. Gatecloud.net: A platform for large-scale, open-source text processing on the cloud. *Philosophical Transactions of the Royal Society A*, 371(1983), 2013. DOI: 10.1098/rsta.2012.0071. 113, 138

[321] J. Teevan, D. Ramage, and M. R. Morris. #twittersearch: A comparison of microblog search and web search. In *Proc. of the 4th ACM International Conference on Web Search and Data Mining (WSDM)*, pages 35–44, 2011. DOI: 10.1145/1935826.1935842. 118

[322] K. Bontcheva and D. Rout. Making sense of social media through semantics: A survey. *Semantic Web—Interoperability, Usability, Applicability*, 5(5), pages 373—403, 2014. 119

[323] K. Holmberg and I. Hellsten. Analyzing the climate change debate on twitter—content and differences between genders. In *Proc. of the ACM WebScience Conference*, pages 287–288, Bloomington, IN, 2014. DOI: 10.1145/2615569.2615638.

[324] René Pfitzner, Antonios Garas, and Frank Schweitzer. Emotional divergence influences information spreading in twitter. *ICWSM*, 12, pages 2–5, 2012.

[325] C. Meili, R. Hess, M. Fernandez, and G. Burel. Earth hour report. Technical Report D6.2.1, DecarboNet Project Deliverable, 2014.

[326] Matthew Rowe and Harith Alani. Mining and comparing engagement dynamics across multiple social media platforms. In *Proc. of the ACM conference on Web science*, pages 229–238, 2014. DOI: 10.1145/2615569.2615677. 119

[327] A. Esuli and F. Sebastiani. SentiWordNet: A publicly available lexical resource for opinion mining. In *Proc. of LREC*, 2006. 119

[328] Hamish Cunningham, Diana Maynard, Kalina Bontcheva, and Valentin Tablan. GATE: An architecture for development of robust HLT applications. In *Proc. of the 40th Annual Meeting on Association for Computational Linguistics*, ACL'02, pages 168–175, Stroudsburg,

PA, 2002. Association for Computational Linguistics. DOI: 10.3115/1073083.1073112. 119

[329] K. Tao, F. Abel, C. Hauff, and G.-J. Houben. What makes a tweet relevant to a topic. In *Proc. of the #MSM2012 Workshop, CEUR*, volume 838, 2012. 119

[330] P. N. Mendes, A. Passant, and P. Kapanipathi. Twarql: Tapping into the wisdom of the crowd. In *Proc. of the 6th International Conference on Semantic Systems*, I-SEMANTICS'10, pages 45:1–45:3, 2010. DOI: 10.1145/1839707.1839762. 119

[331] F. Abel, I. Celik, G.-J. Houben, and P. Siehndel. Leveraging the semantics of tweets for adaptive faceted search on Twitter. In *Proc. of the 10th International Conference on the Semantic Web—Volume Part I*, ISWC'11, pages 1–17, Berlin, Heidelberg, 2011. Springer-Verlag. DOI: 10.1007/978-3-642-25073-6_1. 120

[332] Miriam Fernandez, Arno Scharl, Kalina Bontcheva, and Harith Alani. User profile modelling in online communities. In *Proc. of the 3rd International Conference on Semantic Web Collaborative Spaces—Volume 1275*, pages 1–15. CEUR-WS. org, 2014. 122

[333] L. Aroyo and G.-J. Houben. User modeling and adaptive semantic web. *Semantic Web*, 1(1,2), pages 105–110, 2010. DOI: 10.3233/SW-2010-0006. 122

[334] S. Decker and M. Frank. The Social Semantic Desktop. Technical report, DERI Technical Report 2004-05-02, 2004. 122

[335] P. Mika. Ontologies are us: A unified model of social networks and semantics. *Journal of Web Semantics*, 5(1), pages 5–15, 2007. DOI: 10.1007/11574620_38. 122

[336] J. Chen, R. Nairn, L. Nelson, M. Bernstein, and E. Chi. Short and tweet: Experiments on recommending content from information streams. In *Proc. of the 28th International Conference on Human Factors in Computing Systems*, CHI'10, pages 1185–1194, 2010. DOI: 10.1145/1753326.1753503. 123, 126

[337] P. Kapanipathi, F. Orlandi, A. Sheth, and A. Passant. Personalized filtering of the twitter stream. In *2nd workshop on Semantic Personalized Information Management at ISWC*, 2011. 123, 124

[338] M. Szomszor, H. Alani, I. Cantador, K. O'Hara, and N. Shadbolt. Semantic modelling of user interests based on cross-folksonomy analysis. In *Proc. of the 7th International Conference on The Semantic Web (ISWC)*, pages 632–648. Springer-Verlag, 2008. DOI: 10.1007/978-3-540-88564-1_40. 123

[339] E. Zavitsanos, G. A. Vouros, and G. Paliouras. Classifying users and identifying user interests in folksonomies. In *Proc. of the 2nd Workshop on Semantic Personalized Information Management: Retrieval and Recommendation*, 2011. 123

[340] M. Zhou, S. Bao, X. Wu, and Y. Yu. An unsupervised model for exploring hierarchical semantics from social annotations. In *Proc. of the 6th International Semantic Web Conference*, ISWC'07, pages 680–693. Springer-Verlag, 2007. DOI: 10.1007/978-3-540-76298-0_49. 123

[341] Ciro Cattuto, Dominik Benz, Andreas Hotho, and Gerd Stumme. Semantic grounding of tag relatedness in social bookmarking systems. In *Proc. of the 7th International Conference on The Semantic Web*, pages 615–631, 2008. DOI: 10.1007/978-3-540-88564-1_39. 123

[342] Zhiyuan Cheng, James Caverlee, and Kyumin Lee. You are where you tweet: A content-based approach to geo-locating twitter users. In *Proc. of the 19th ACM International Conference on Information and Knowledge Management*, CIKM'10, pages 759–768, New York, NY, 2010. ACM. DOI: 10.1145/1871437.1871535. 123

[343] S. Yardi and D. Boyd. Tweeting from the town square: Measuring geographic local networks. In *Proc. of ICWSM*, 2010. 123

[344] J. Burger, J. Henderson, G. Kim, and G. Zarrella. Discriminating Gender on Twitter. In *Proc. of the Conference on Empirical Methods in Natural Language Processing*, EMNLP'11, pages 1301–1309, 2011. 123

[345] M. Pennacchiotti and A. M. Popescu. A machine learning approach to twitter user classification. In *Proc. of ICWSM*, pages 281–288, 2011. 123

[346] Clay Fink, Christine Piatko, James Mayfield, Tim Finin, and Justin Martineau. Geolocating blogs from their textual content. In *Working Notes of the AAAI Spring Symposium on Social Semantic Web: Where Web 2.0 Meets Web 3.0*, pages 1–2. AAAI Press, 2008. 123

[347] J. Eisenstein, B. O'Connor, N. A. Smith, and E. P. Xing. A latent variable model for geographic lexical variation. In *Proc. of the Conference on Empirical Methods in Natural Language Processing*, pages 1277–1287, 2010. 123

[348] Jalal Mahmud, Jeffrey Nichols, and Clemens Drews. Where is this tweet from? Inferring home locations of twitter users. In *Proc. of the 6th International AAAI Conference on Weblogs and Social Media*, Dublin, Ireland, 2012. 123

[349] J. Chan, C. Hayes, and E. Daly. Decomposing discussion forums using common user roles. In *Proc. of WebSci10: Extending the Frontiers of Society On-Line*, 2010. 124

[350] Markus Strohmaier, Christian Koerner, and Roman Kern. Why do users tag? detecting users' motivation for tagging in social tagging systems. 2010. 124

[351] A. L. Gentile, V. Lanfranchi, S. Mazumdar, and F. Ciravegna. Extracting semantic user networks from informal communication exchanges. In *Proc. of the 10th International*

Conference on the Semantic Web, ISWC'11, pages 209–224. Springer-Verlag, 2011. DOI: 10.1007/978-3-642-25073-6_14. 125

[352] C. Beaudoin. Explaining the relationship between internet use and interpersonal trust: Taking into account motivation and information overload. *Journal of Computer Mediated Communication*, 13, pages 550—568, 2008. DOI: 10.1111/j.1083-6101.2008.00410.x. 125

[353] Fabian Abel, Qi Gao, Geert-Jan Houben, and Ke Tao. Analyzing temporal dynamics in twitter profiles for personalized recommendations in the social web. In *Proc. of the 3rd International Web Science Conference*, WebSci'11, pages 2:1–2:8, New York, NY, 2011. ACM. DOI: 10.1145/2527031.2527040. 126

[354] J. Chen, R. Nairn, and E. Chi. Speak little and well: Recommending conversations in online social streams. In *Proc. of the Annual Conference on Human Factors in Computing Systems*, CHI'11, pages 217–226, 2011. DOI: 10.1145/1978942.1978974. 126, 132

[355] N. Bansal and N. Koudas. Blogscope: Spatio-temporal analysis of the blogosphere. In *Proc. of the 16th International Conference on World Wide Web*, WWW'07, pages 1269–1270, 2007. DOI: 10.1145/1242572.1242802. 127, 128, 132

[356] D. A. Shamma, L. Kennedy, and E. F. Churchill. Tweetgeist: Can the twitter timeline reveal the structure of broadcast events? In *Proc. of CSCW*, 2010. 127, 131

[357] O. Phelan, K. McCarthy, and B. Smyth. Using twitter to recommend real-time topical news. In *Proc. of the ACM Conference on Recommender Systems*, pages 385–388, 2009. DOI: 10.1145/1639714.1639794. 127

[358] M. S. Bernstein, B. Suh, L. Hong, J. Chen, S. Kairam, and E. H. Chi. EDDI: Interactive topic-based browsing of social status streams. In *Proc. of the 23nd ACM Symposium on User Interface Software and Technology (UIST)*, pages 303–312, 2010. DOI: 10.1145/1866029.1866077. 127

[359] B. Adams, D. Phung, and S. Venkatesh. Eventscapes: Visualizing events over time with emotive facets. In *Proc. of the 19th ACM International Conference on Multimedia*, pages 1477–1480, 2011. DOI: 10.1145/2072298.2072044. 127, 131

[360] J. Eisenstein, D. H. P. Chau, A. Kittur, and E. Xing. Topicviz: Semantic navigation of document collections. In *CHI Work-in-Progress Paper (Supplemental Proceedings)*, 2012. 128

[361] D. Archambault, D. Greene, P. Cunningham, and N. J. Hurley. ThemeCrowds: Multiresolution summaries of twitter usage. In *Workshop on Search and Mining User-generated Contents (SMUC)*, pages 77–84, 2011. DOI: 10.1145/2065023.2065041. 128, 132

[362] B. Meyer, K. Bryan, Y. Santos, and B. Kim. TwitterReporter: Breaking news detection and visualization through the geo-tagged twitter network. In *Proc. of the ISCA 26th International Conference on Computers and Their Applications*, pages 84–89, 2011. 128, 131

[363] S. Faridani, E. Bitton, K. Ryokai, and K. Goldberg. Opinion space: A scalable tool for browsing online comments. In *Proc. of the 28th International Conference on Human Factors in Computing Systems (CHI)*, pages 1175–1184, 2010. DOI: 10.1145/1753326.1753502. 131

[364] Omar F. Zaidan and Chris Callison-Burch. Crowdsourcing translation: Professional quality from non-professionals. In *Proc. of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1220–1229, 2011. 136

[365] Mahmoud El-Haj, Udo Kruschwitz, and Chris Fox. Using mechanical turk to create a corpus of Arabic summaries. In *Proc. of the 7th Conference on International Language Resources and Evaluation*, 2010. 136

[366] Vamshi Ambati and Stephan Vogel. Can crowds build parallel corpora for machine translation systems? In *Proc. of the NAACL HLT Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 62–65, 2010. 136

[367] Chris Callison-Burch. Fast, cheap, and creative: Evaluating translation quality using amazon's mechanical turk. In *Proc. of the Conference on Empirical Methods in Natural Language Processing*, pages 286–295, 2009. DOI: 10.3115/1699510.1699548. 139

[368] Ann Irvine and Alexandre Klementiev. Using mechanical turk to annotate lexicons for less commonly used languages. In *Proc. of the NAACL HLT Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 108–113, 2010. 136

[369] A. Weichselbraun, S. Gindl, and A. Scharl. Using games with a purpose and bootstrapping to create domain-specific sentiment lexicons. In *Proc. of the 20th ACM Conference on Information and Knowledge Management (CIKM)*, pages 1053–1060, 2011. DOI: 10.1145/2063576.2063729. 136

[370] Nitin Madnani, Jordan Boyd-Graber, and Philip Resnik. Measuring transitivity using untrained annotators. In *Proc. of the NAACL HLT Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 188–194, 2010. 136

[371] Massimo Poesio, Nils Diewald, Maik Stührenberg, Jon Chamberlain, Daniel Jettka, Daniela Goecke, and Udo Kruschwitz. Markup infrastructure for the anaphoric bank: Supporting web collaboration. In Alexander Mehler, Kai-Uwe Kühnberger, Henning Lobin, Harald Lüngen, Angelika Storrer, and Andreas Witt, Eds., *Modeling, Learning, and Processing of Text Technological Data Structures*, volume 370 of *Studies in Computational*

Intelligence, pages 175–195. Springer Berlin/Heidelberg, 2012. DOI: 10.1007/978-3-642-22613-7. 136, 138, 139

[372] W. Rafelsberger and A. Scharl. Games with a purpose for social networking platforms. In *Proc. of the 20th ACM Conference on Hypertext and hypermedia*, HT'09, pages 193–198, 2009. DOI: 10.1145/1557914.1557948. 136

[373] Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati, and Christopher D. Manning. Multi-instance multi-label learning for relation extraction. In Jun'ichi Tsujii, James Henderson, and Marius Pasça, Eds., *Proc. of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 455–465, Jeju Island, Korea, 2012. Association for Computational Linguistics. 137

[374] Vidhoon Viswanathan, Nazneen Fatema Rajani, Yinon Bentor, and Raymond Mooney. Stacked ensembles of information extractors for knowledge-base population. In Chengqing Zong and Michael Strube, Eds., *Proc. of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 177–187, Beijing, China, 2015. Association for Computational Linguistics. DOI: 10.3115/v1/p15-1. 137

[375] Alon Halevy, Peter Norvig, and Fernando Pereira. The unreasonable effectiveness of data. *IEEE Intelligent Systems*, 24(2), pages 8–12, 2009. DOI: 10.1109/mis.2009.36. 137

[376] Roger Barga, Dennis Gannon, and Daniel Reed. The client and the cloud: Democratizing research computing. *IEEE Internet Computing*, 15(1), pages 72–75, 2011. DOI: 10.1109/mic.2011.20. 137

[377] Marios D. Dikaiakos, Dimitrios Katsaros, Pankaj Mehra, George Pallis, and Athena Vakali. Cloud computing: Distributed internet computing for IT and scientific research. *IEEE Internet Computing*, 13(5), pages 10–13, 2009. DOI: 10.1109/mic.2009.103. 137

[378] Kalina Bontcheva, Hamish Cunningham, Ian Roberts, Angus Roberts, Valentin Tablan, Niraj Aswani, and Genevieve Gorrell. GATE Teamware: A web-based, collaborative text annotation framework. *Language Resources and Evaluation*, 47, pages 1007—1029, 2013. DOI: 10.1007/s10579-013-9215-6. 139

[379] Seid Muhie Yimam, Chris Biemann, Richard Eckart de Castilho, and Iryna Gurevych. Automatic annotation suggestions and custom annotation layers in webanno. In *Proc. of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 91–96, Baltimore, Maryland, 2014. Association for Computational Linguistics. DOI: 10.3115/v1/p14-5016. 139

[380] Leah Hoffmann. Crowd control. *Communications of the ACM*, 52(3), pages 16–17, 2009. DOI: 10.1145/1467247.1467254. 139

[381] Sharoda A. Paul, Lichan Hong, and Ed H. Chi. What is a question? Crowdsourcing tweet categorization. In *CHI'2011 Worshop on Crowdsourcing and Human Computation*, 2011. 139

[382] K. Siorpaes and M. Hepp. Games with a purpose for the semantic web. *Intelligent Systems, IEEE*, 23(3), pages 50–60, 2008. DOI: 10.1109/mis.2008.45. 139

[383] S. Thaler, K. S. E. Simperl, and C. Hofer. A survey on games for knowledge acquisition. Technical Report Tech. Rep. STI TR 2011-05-01, Semantic Technology Institute, 2011. 139

[384] J. Waitelonis, N. Ludwig, M. Knuth, and H. Sack. WhoKnows? Evaluating linked data heuristics with a quiz that cleans up DBpedia. *Interactive Technology and Smart Education*, 8(4), pages 236–248, 2011. DOI: 10.1108/17415651111189478. 139

[385] Richard McCreadie, Craig Macdonald, and Iadh Ounis. Identifying top news using crowdsourcing. *Information Retrieval*, pages 1–31, 2012. 10.1007/s10791-012-9186-z. DOI: 10.1007/s10791-012-9186-z. 139

[386] Dan Gillick and Yang Liu. Non-expert evaluation of summarization systems is risky. In *Proc. of the NAACL HLT Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 148–151, 2010. 139

[387] David Inouye and Jugal K. Kalita. Comparing twitter summarization algorithms for multiple post summaries. In *SocialCom/PASSAT*, pages 298–306, 2011. DOI: 10.1109/passat/socialcom.2011.31. 139

[388] Andrea Glaser and Hinrich Schütze. Automatic generation of short informative sentiment summaries. In *Proc. of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 276–285, Avignon, France, 2012. 139

[389] Kalina Bontcheva, Ian Roberts, Leon Derczynski, and Dominic Rout. The GATE crowdsourcing plugin: Crowdsourcing annotated corpora made easy. In *Proc. of the 14th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*. Association for Computational Linguistics, 2014. DOI: 10.3115/v1/e14-2025. 139

[390] G.W. Allport and L. Postman. The psychology of rumor. *Journal of Clinical Psychology*, 1947. 102