Techniques for Named Entity Recognition: A Survey

Chapter · January 2012

DOI: 10.4018/978-1-4666-3604-0.ch022

Girish Palshikar
Tata Consultancy Services Limited

Collaboration and the Semantic Web:

Social Networks, Knowledge Networks, and Knowledge Resources

Stefan Brüggemann
Astrium Space Transportation, Germany

Claudia d'Amato
Università degli Studi di Bari "A. Moro", Italy

Information Science
REFERENCE

Managing Director:	Lindsay Johnston
Senior Editorial Director:	Heather A. Probst
Book Production Manager:	Sean Woznicki
Development Manager:	Joel Gamon
Development Editor:	Hannah Abelbeck
Acquisitions Editor:	Erika Gallagher
Typesetter:	Nicole Sparano
Cover Design:	Nick Newcomer, Lisandro Gonzalez

Published in the United States of America by
Information Science Reference (an imprint of IGI Global)
701 E. Chocolate Avenue
Hershey PA 17033
Tel: 717-533-8845
Fax: 717-533-8661
E-mail: cust@igi-global.com
Web site: http://www.igi-global.com

Library of Congress Cataloging-in-Publication Data

Collaboration and the Semantic Web: social networks, knowledge networks and knowledge resources / Stefan Bruggemann and Claudia d'Amato, editors.
p. cm.
Includes bibliographical references and index.
Summary: "This book showcases cutting-edge research on the intersections of Semantic Web, collaborative work, and social media research, exploring how the resources of so-called social networking applications, which bring people together to interact and encourage sharing of personal information and ideas, can be tapped by Semantic Web techniques"--Provided by publisher.
ISBN 978-1-4666-0894-8 (hbk.) -- ISBN 978-1-4666-0895-5 (ebook) -- ISBN 978-1-4666-0896-2 (print & perpetual access) 1. Semantic Web. 2. Online social networks. 3. Internet--Social aspects. I. Bruggemann, Stefan, 1978- II. d'Amato, Claudia, 1977-
TK5105.88815.C64 2012
025.04'27--dc23
2012001349

British Cataloguing in Publication Data
A Cataloguing in Publication record for this book is available from the British Library.

All work contributed to this book is new, previously-unpublished material. The views expressed in this book are those of the authors, but not necessarily of the publisher.

Chapter 11
Techniques for Named Entity Recognition:
A Survey

Girish Keshav Palshikar
Tata Research Development and Design Centre, India

ABSTRACT

While building and using a fully semantic understanding of Web contents is a distant goal, named entities (NEs) provide a small, tractable set of elements carrying a well-defined semantics. Generic named entities are names of persons, locations, organizations, phone numbers, and dates, while domain-specific named entities includes names of for example, proteins, enzymes, organisms, genes, cells, et cetera, in the biological domain. An ability to automatically perform named entity recognition (NER) – i.e., identify occurrences of NE in Web contents – can have multiple benefits, such as improving the expressiveness of queries and also improving the quality of the search results. A number of factors make building highly accurate NER a challenging task. Given the importance of NER in semantic processing of text, this chapter presents a detailed survey of NER techniques for English text.

INTRODUCTION

Given the vast amounts of text available on the Web, it is becoming increasingly clear that Internet based tools (e.g., search engines, content creation and management) and applications (e.g., Wikipedia, social networking, blogs) need to understand at least rudimentary semantics of the contents of the Web, which is of course the fundamental motivation for Semantic Web (Shadbolt et al 2006). While semantics is a complex subject and understanding (and using) the complete meaning of a piece of text may well be impossible, it is easy to identify limited types of semantic elements in a text.

DOI: 10.4018/978-1-4666-0894-8.ch011

Named entities (NE) − like names of persons, organizations, locations and dates, times, phone numbers, amounts, zip codes – are just such basic semantic elements of a text that carry a specific and limited kind of meaning. An ability to automatically perform *named entity recognition* (*NER*) – i.e., identify occurrences of NE in Web contents – can have multiple benefits, such as improving the expressiveness of queries and also improving the quality of the search results. Examples where processing queries containing NE as keywords requires NER: Kawasaki the person and Kawasaki the manufacturing company, Jackson the scientist and Jackson the musician, dates in different formats, identifying that Robert Feynman and Dick Feynman are the same persons, Jobs as a person versus jobs as a common noun, high blood pressure and hypertension as synonymous medical terms. As another example, identifying two successive dates in a text can help compute the duration of some event (e.g., of a project). Lack of NER abilities make representation and execution of queries such as "*Find all European physicists who lived for at least 70 years*" difficult for many of today's search engines. Given the frequent use and relatively well-defined semantics of NE, it is possible to use NER to automatically annotate the occurrences of NE in web contents, which can then be used for improving search and other functions. NE are frequently used as sources (origins) of hyperlinks. Further, since NE occur frequently as part of annotations, notes, comments, bookmarks, hyperlinks etc., NE play an important role in collaborative semantic web applications.

Given the importance of NER in semantic processing of text, this paper presents a detailed (but not necessarily exhaustive) survey of NER techniques. We focus on NER in English text, though there is a considerable work for other languages, which presents complex challenges. We focus mainly on NER for generic NE. However, there is a large amount of work on NER for extracting domain-specific NE. NE in the bio-medical domain are the most well-explored, among the various possible domains.

NER is an important sub-problem in text processing − particularly in *information extraction (IE)* − and is useful in many practical applications in the Semantic Web context. The goal of NER is to identify all occurrences of specific types of *named entities* in the given document collection. NE may be divided into several categories.

- **Generic NE** consist of names of persons (PERSON), organizations (ORG), locations (LOCATION), amounts, dates, times, email addresses, URLs, phone numbers etc. Other generic NE include: film title, book title etc. In a richer problem setting (called *fine-gained NER*), the problem is to identify generic NE which are hierarchically organized; e.g., PERSON may be sub-divided into politicians, sports persons, film stars, musicians etc.
- **Domain-specific NE (DSNE)** consist of, for example, names of proteins, enzymes, organisms, genes, cells etc., in the biological domain. As another example, DSNE in the manufacturing domain are: names of manufacturer, product, brand and attributes of the product (Fig. 1).

Often the NER needs to be performed on a fixed type of input text; e.g., news items or research paper abstracts. NER in speech is a much more difficult problem, since information such as capitalization, punctuation etc. is not available in speech data (though some other kinds of information, such as emphasis, may be available). In this paper, we focus on the NER for English text.

Challenges in NER

Several factors make NER a challenging task. First, there is the open nature of the vocabulary; e.g., it is obviously not feasible to maintain a list of all known person names. Clues such as capitalization

Figure 1. Example sentences containing occurrences of generic NE

[J. P. Morgan]$_{ORG}$ strengthens domestic treasury management offering in [Malasia]$_{LOCATION}$.

In a strategic reshuffle at [Bank of America-Merrill Lynch]$_{ORG}$, [Atul Singh]$_{PERSON}$ has taken over as managing director of Global Wealth and Investment Management in [India]$_{LOCATION}$.

are error-prone; words may be wrongly capitalized and capitalization of first words in a sentence may lead to some confusion as in **Jobs said** ... or **Jobs are harder to find** .... Complex techniques such as co-reference resolution are needed to detect indirect occurrences of NE; e.g., through the use of pronouns. In **Prospects of ABC Corp. are looking bright. It has declared a dividend** ..., the pronoun **It** stands for **ABC Corp**. and may need to be identified as an occurrence of ORG. Another problem is the overlap between NE types; e.g., **Washington** can be used both as a PERSON or a LOCATION; **White House** can be used as both a LOCATION and an ORG. Many NE are multi-word (e.g., **Bank of America**). In such cases, identifying the boundary (i.e., the exact sequence of words) of a NE occurrence can be tricky; e.g., is **Boston Gas and Light Company** a single organization or a conjunction of two organizations? Also, NER is (at least partly) a language-specific task. For example, capitalization may be used in English to indicate a PERSON name but such a method is not available in many languages (like Hindi) which do not have capital letters. Alternatively, the form of a verb in a sentence indicates the gender of the subject in many languages (not English), which can help in NER (e.g., to detect PERSON names).

Another difficulty arises due to the different ways of referring to the same NE in the same document: abbreviations (**IBM** versus **International Business Machines**) and shortened names (**Mitsubishi Motor Company** versus **Mitsubishi**). It requires some reasoning to decide that **Boston Gas and Light Company** is a single ORG and does not denote a conjunction of two ORG here, unlike in **Microsoft and Google**). While most NER can be done when processing each individual sentence, sometimes *long distance* clues from other sentences may be helpful. For example, we can decide that **MURDOCH** is a PERSON (though it is in all capitals) in **MURDOCH SATELLITE EXPLODES ON TAKE OFF** if we were able to determine when processing ... **Rupert Murdoch's ambition** ... later in the document that **Murdoch** is a PERSON. Thus a lazy approach to NER – where local decisions or evidences are effectively combined - might be useful, where some NE occurrences may initially be left untagged and which are tagged after sufficient evidence is available later. A one use per document principle might be useful to decide that all occurrences of **Philip Morris** (or **Morris**) are an ORG throughout a document, if even one sentence in the document provides a strong clue that **Philip Morris** is an ORG (... **president of Philip Morris said** ...). An effective NER system would require the use of use of a large amount of prior common-sense knowledge. For example, occurrence of a phrase like **Mitsubishi** and **Nissan** suggests that both these words should be assigned the same NE type. The task of assigning the correct NE type to a group of words in a sentence becomes quite subtle due to the complex nature of interdependencies among various parts of sentences. Overall, it is quite a technical challenge to build NER systems that rival human performance.

Desirable Characteristics of an NER System

Some desirable properties that a good NER system should possess are as follows. First, the system should obviously be highly accurate in its output. Next, it should be as efficient as possible; e.g., in terms of the time taken to tag a set of documents. The system should be robust in the presence of noise such as spelling and grammatical errors, wrong capitalization, incorrect sentence boundaries (e.g., missing periods) etc. The NER system should be as much corpus independent as possible i.e., it should be possible to reuse it on documents in different corpora, such as news items, emails, reports or blogs. Sometimes it is desirable to design an NER system which is portable or language independent and can identify NE in documents written a variety of (related) languages. Further, ideally an NER system should be designed to be largely domain independent, which can be adapted with little or no effort to identify new types of NE, possibly from different domains such as banking or manufacturing. Another aspect of an NER system is extendibility, which is the ability of the expert users to extend the knowledge sources used by the system (e.g., rules and gazetteers of known examples of a NE). Clearly, meeting all (or even most) of these goals is difficult and hence building a good NER system is a challenging task. Evaluation of the accuracy of an NER system is critical for enabling comparison between systems.

Techniques for NER

Approaches to build an NER system can be broadly divided into 4 groups. It should be emphasized that while this grouping helps in understanding the broad category of techniques, many NER systems reported in the literature contain two or more techniques.

1. **Rule-based Approaches**: Here, a set of rules is manually crafted by experts to recognize a particular NE type. The rules are based on syntactic, linguistic and domain knowledge.
2. **Supervised Learning Approaches**: Here, a large hand-tagged corpus is manually created by human experts, where instances of the given NE type are explicitly identified. Supervised learning algorithms from machine learning are used to generalize and discover NER rules from this labeled training dataset.
3. **Unsupervised Approaches**: Here, usually the system is provided a small set of seed instances (or examples) of the NE type; e.g., cities {**'New York', Boston, London, Seoul**}. The system then examines the given document collection and learns some rules from the sentences in which the seed NE examples occur. These rules are applied to identify new examples of NE and then learn a new set of rules. The system continues to learn in this way and stops when no new rules can be discovered.
4. **NE Extraction (NEX)**: The NEX task is quite similar to the unsupervised approaches, except that the goal is not to learn rules for NER but to create a *gazette* (*list* or *gazetteer*) of examples of the NE. Also, NEX is often applied to learn from web pages rather than documents. The idea is that once a comprehensive list of NE examples is created, NER in a given document corresponds to simple look up in this list.

In this paper, we discuss representative work from each of these approaches; see (Nadeau and Sekine 2007) for an excellent survey of NER literature. Detailed guidelines, issues and examples for NER are discussed in (Chinchor 1998), (Sang et al 2003). (Ratinov and Roth 2009) discuss some interesting issues and challenges in NER - particularly, the choice of an inference mechanism and representation of text chunks. Inclusion of the NER task in MUC and coNLL conferences as well as

Table 1. Some Tagged corpora for English NER

Corpus	URL
MUC-7 corpus	http://www.itl.nist.gov/iaui/894.02/relatedprojects/muc/proceedings/muc 7 proceedings
coNLL 2002 shared task corpora	http://cnts.uia.ac.be/conll2002/ner/
coNLL 2003 shared task corpora	http://cnts.uia.ac.be/conll2003/ner/
ACE corpora	http://www.ldc.upenn.edu/Projects/ACE/
GENIA	http://www-tsujii.is.s.u-tokyo.ac.jp/GENIA/home/wiki.cgi

availability of several corpora tagged with named entities has given a boost to the research in NER.

This chapter is organized as follows. First, the tagged datasets available for evaluating accuracy of an NER system are surveyed. Then methods to compute the accuracy of an NER system are presented. Next, specific types of NER techniques are reviewed. Lastly, the techniques are compared and some open problems are discussed.

TAGGED DATASETS FOR NER

Table 1 shows a list of some tagged corpora for NER tasks; see (Fort *et al* 2009) for guiding principles and a methodology for creating effective tagged NE datasets (tagging is also called *NE annotation*); see similar guidelines from LDC (http://projects.ldc.upenn.edu/ace/docs/English-Entities-Guidelinesv6.1.pdf). The Message Understanding Conferences MUC-6 and MUC-7 had a special track for NER tasks. There were a few broad classes of NE. The class Enamex consisted of 3 types of generic NE: PERSON (name of a person), LOCATION (name of a physical location such as city, state, country etc.) and ORG (name of an organization). The class Timex consisted of types of NE such as DATE and TIME. The class Numex consisted of various expressions used to state numeric quantities like state amounts, rates, numbers etc. MUC-7 provided a tagged corpus for these NE. This corpus contains 100 news items containing 4091 Enamex occurrences: 1880 ORG (46%), 1324 LOCATION (32%) and 887 PERSON (22%). The corpus was tagged by several experts and there was substantial (97%) agreement among them. See Fig. 2 for an example.

The CoNLL-2003 shared task corpus for English contains a collection of news wire articles (there is a German corpus as well). The data files contain four columns separated by a single space. Each word is put on a separate line and there is an empty line after each sentence. The first item on each line is a word, the second a POS tag, the third a syntactic chunk tag and the fourth the NE tag. The chunk tags and the NE tags have the format I-TYPE which means that the word is inside a phrase of type TYPE. Only if two phrases of the same type immediately follow each other, the first word of the second phrase will have tag B-TYPE to show that it starts a new phrase. A word with tag O is not part of a phrase. The sentence **U.N. official Ekeus heads for Baghdad**. is tagged as follows:

```
U.N. NNP I-NP I-ORG
official NN I-NP O
Ekeus NNP I-NP I-PER
heads VBZ I-VP O
for IN I-PP O
Baghdad NNP I-NP I-LOC
. . O O
```

EVALUATION OF NER ALGORITHMS

The steps for evaluating any NER system are fairly standardized:

Figure 2. A sample news item in MUC-7 corpus: original and tagged text

```
CAPE CANAVERAL, Fla. &MD; Working in chilly temperatures Wednesday night, NASA
ground crews readied the space shuttle Endeavour for launch on a Japanese
satellite retrieval mission. Endeavour, with an international crew of six, was
set to blast off from the Kennedy Space Center on Thursday at 4:18 a.m. EST, the
start of a 49-minute launching period. The nine day shuttle flight was to be the
12th launched in darkness. <ENAMEX TYPE=''LOCATION''>CAPE CANAVERAL</ENAMEX>,
<ENAMEX TYPE=''LOCATION''>Fla.</ENAMEX> &MD; Working in chilly temperatures
<TIMEX TYPE=''DATE''>Wednesday</TIMEX> <TIMEX TYPE=''TIME''>night</TIMEX>,
<ENAMEX TYPE=''ORGANIZATION''>NASA</ENAMEX> ground crews readied the space
shuttle Endeavour for launch on a Japanese satellite retrieval mission. <p>
Endeavour, with an international crew of six, was set to blast off from the
<ENAMEX TYPE=''ORGANIZATION|LOCATION''>Kennedy Space Center</ENAMEX> on <TIMEX
TYPE=''DATE''>Thursday</TIMEX> at <TIMEX TYPE=''TIME''>4:18 a.m. EST</TIMEX>,
the start of a 49-minute launching period. The <TIMEX  TYPE=''DATE''>nine
```

1. Select a gold standard (or key) tagged corpus containing documents, each of which contains annotated instances of the given types of named entities. This corpus is often tagged manually. Randomly divide the documents in the corpus into training and testing sets (often in 80% − 20% proportion).
2. Use the training set to learn and to tune the NER knowledge base. Usually 10-fold cross validation is used to estimate the training error. No further changes should be made to the knowledge base after the training step is declared complete. Care should be taken to avoid any manual knowledge engineering specifically for improving the accuracy of the proposed algorithm over the given corpus.
3. Use the learned and tuned knowledge base to extract entities from the documents in the test set.
4. For each NE type, compute the precision, recall and an overall accuracy measure such as the F-measure.

Let $A = \{a_1, a_2, \ldots, a_N\}$ denote (multi)set of N occurrences of the chosen type of NE in the test corpus. Let $B = \{b_1, b_2, \ldots, b_M\}$ denote the (multi)set of the M occurrences of the chosen type of NE identified by the algorithm in the test corpus. An occurrence $b_i \in B$ is classified as a *true positive* (*TP*) (as *false positive* (*FP*)) if $b_i \in A$ ($b_i \notin A$ respectively). Thus the number of true positives identified by the algorithm is the number of occurrences which are in both B and A i.e., $\#TP = |A \cap B|$. The number of occurrences which are in B but not in A is the number of false positives: $\#FP = |B - A|$. An occurrence a_i in A is classified as a *false negative* (*FN*) if $a_i \notin B$. The number of occurrences which are in A but not in B is the number of false negatives: $\#FN = |A - B|$. Then the precision P, recall R and F-measure accuracy of the algorithm are:

$$P = \frac{\#TP}{|B|} = \frac{\#TP}{\#TP + \#FP}$$

$$R = \frac{\#TP}{|A|} = \frac{\#TP}{\#TP + \#FN}$$

$$F = \frac{2PR}{P + R}$$

Precision P indicates the fraction of the extracted entities that are correct. Recall R indicates the fraction of the correct entities that are extracted. Low precision indicates high value of *#FP* i.e., lot of noise in extraction. Low recall indicates high value of *#FN* i.e., lots of misses in extraction. The *F*-measure computes an overall accuracy by combining precision and recall. There are other ways of measuring the performance of NER algorithms, though *P*, *R* and *F* are most commonly used.

A delicate issue is how to compare the extracted occurrence and the occurrence in the *gold copy*. For example, suppose the gold copy marks **Bill Clinton** as a PERSON in a sentence and an NER algorithm extracts only **Clinton** as a PERSON in that sentence. Is this a correct match? An exact matching scheme treats such extraction as a mismatch i.e., FP. But often a more forgiving scheme is employed, which accepts partial matches; e.g., left match or right match; see (Tsai *et al* 2006).

RULE-BASED NER

While one can often use lexical (word-level) and syntactic cues for recognizing an NE type (e.g., a person name often begins with a capital letter), these rules are not sufficient; there are many special cases and most rules have exceptions. Following are some example rules to identify the occurrence of a PERSON in a sentence.

- Often consists of a sequence of words each of which begins with a capital letter followed by all lowercase letters (**John Ryder**); the first word is often a known first name (**John**) and the next word is unknown (**Ryder**).
- May contain a prefix title such as **Mr.**, **Dr.** or **Prof.** (**Dr. Enrico Fermi**)
- May contain an initial in the middle or beginning (**Winston S. Churchil**, **A. John Northrop**)
- May contain a suffix such as **Jr.** or **III** (as in **George Bush Sr.**)
- May contain a designation indicator prefix such as **President**, **Justice**, **Sen.**, **Colonel** or **CEO** (**President Clinton**)
- May be followed by an appositive NP suffix whose head word is singular and indicates a profession or a relation (**Malviya, a retired analyst, said ...**; or **Nielson, whose stepfather ...**)
- Does *not* include special characters such as **$**, **&** or **%** (**Johnson & Johnson**)
- Does *not* include prepositions (**Castle Of Windsor**)

Clearly, the rules are not sufficient to identify all occurrences of PERSON in a document. For one, the rules use many hand-crafted lists (e.g., titles, suffixes, designation or profession indicators etc.), which are likely to be incomplete. Also, the rules themselves are incomplete; e.g., they do not cover many examples: **White, 33, was arrested** ... and **Murdoch himself arrived** ... etc. In fact, many more rules can be defined based on a much deeper syntactic or semantic knowledge; e.g., a copula-based rule can be defined to cover examples like **Ryder is a popular juggler**. Another set of rules can be used to identify **Spasky** as a PERSON in **Spasky alighted quickly from the train** ... but this will require deep knowledge of what verbs (e.g., alight) can take only a PERSON as subject and under what situations. Rule-based approaches usually lack robustness and portability. Each new source of text requires significant tweaking of rules to maintain optimal performance and the maintenance efforts tend to be quite steep.

Some well-known rule-based NER systems are Univ. of Sheffield's LaSIEII (Humphreys *et al* 1998), ISOQuest's NetOwl (Krupka and Hausman 1998), Facile (Black *et al* 1998), SRA (Aone *et al* 1998) and Univ. of Edinburgh's LTG system (Mikheev *et al* 1999) and FASTUS (Appelt 1998) for English NER. These systems are mainly based on a set of hand-crafted syntactic and semantic rules for identifying NE instances.

LaSIE-II system (Humphreys *et al* 1998) is a modular IE system and uses a combination of NLP techniques for NER, including word sense disambiguation, co-reference resolution, a shallow semantic representation of the text in quasi-logical form and representation of the domain of discourse as a semantic net. Nodes represent concepts (along with properties) and edges represent hierarchy and support property inheritance; e.g., node *launch_event* has sub-nodes like *vehicle*, *payload*, *astronaut* and has properties such as *launch_date*, *launch_site*, *launch_org* etc. NER is posed essentially as a co-reference task. When processing a sentence, a set of presupposition rules may detect a new instance of a node (e.g., rule if *launch of X* then *X is-a vehicle*) and a set of consequence rules may fill up properties of an already detected instance of a node (e.g., if *launch ... from Y* then *Y is-a launch_site* property).

The FACILE system (Black *et al* 1998) consists of a set of hand-crafted patterns for NER in a regular expression like rule notation. The rules include a co-reference mechanism that detects and uses common text from previous occurrence of the same NE (**foreign secretary Robin Cook** and **Mr. Cook**). Interestingly, each rule predicts a NE type, with a certainty factor, if a condition is satisfied. An evidence combination mechanism is used to combine the certainty values of two matched rules for the same NE type.

The LTG system (Mikheev *et al* 1999) uses a lazy approach to NER where the rules are applied in phases, starting with the ”sure fire” rules and proceeding to rules which are more ”relaxed” and tag the NE occurrences left untagged by the previous phases, by making use of the already tagged occurrences of NEs. A grammar-like formalism is used for defining the rules. (Fukuda *et al* 1998) discussed a rule-based NER system for identifying protein names (e.g., p53, interleukin 1 (IL-1)-responsive kinase, insulin) in biomedical documents. (Bellot *et al* 2002) describes an NER system based on hand-crafted patterns (regular expression transducers) and its application to a question-answering system.

SUPERVISED LEARNING APPROACHES

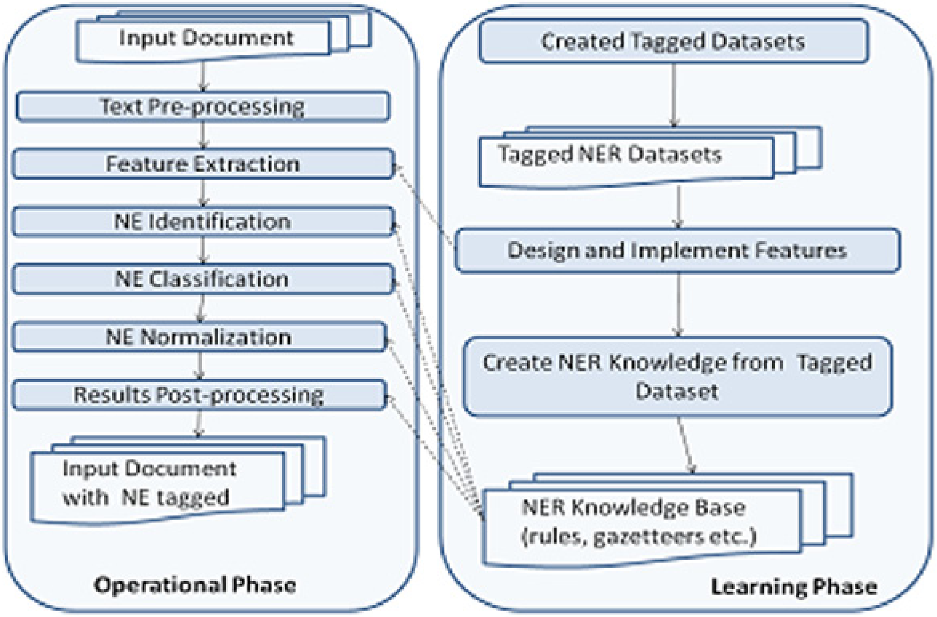
In supervised learning approaches, NER is essentially posed as a classification problem. A set of labeled training dataset is given as input, where instances of the named entities in these documents are identified by human experts. A classification algorithm is used to generalize from these examples and discover a set of rules that can be applied to a new document to identify any instances of the named entities in it. Several distinct approaches to classification algorithms are developed in the machine learning, pattern recognition and statistics literature. Figure. 3 shows the conceptual architecture of an NER system based on supervised learning approach.

Hidden Markov Model Based Approaches

Several authors have used Hidden Markov Models (HMM) for NER; e.g., (Bikel *et al* 1999), (Seymore *et al* 1999), (Collier et al 2000), (Miller et al 1998), (Klein *et al* 2003). HMM approach has also been used for NER in languages other than English. HMM approach has also been used for DSNE; e.g., biomedical domain (Shen et al 2003), (Zhang et al 2002), (Zhao 2004); see also (Liu *et al* 2005) who used HMM for identifying NE such as product names. HMM models the sequence dependencies well and hence are useful for NER because the NE type of a particular word intuitively depends on previous words. The approaches using HMM differ mainly in the structure of the model and in the methods used to compute the associated probabilities.

See (Rabiner 1989) for an overview of the HMM formalism. Briefly, an HMM is like a finite-state automaton, except that each transition

Figure 3. Conceptual architecture of an NER system based on supervised learning



has a probability and transitions are not labeled with any symbol. The HMM also has a set of emission symbols. Each state emits a symbol with a particular probability. Given an observed sequence σ of emitted symbols, the decoding problem for HMM is efficiently solved by the Viterbi algorithm, which identifies a most probable sequence of states through which the HMM may have passed so as to generate σ. Typically, emission symbols are words and σ is the observed sequence of words. The corresponding state sequence obtained by decoding σ is the sequence of NE types, one for each word in σ.

We describe the approach of (Collier *et al* 2000), who used an HMM to detect NEs in the biomedical domain: PROTEIN, DNA, RNA, VIRUS, TISSUE, CELLLINE, CELLTYPE etc. Each state in the HMM represents an NE type. There are special states for start and end of a sentence. Each state emits a word *W*. Each word is represented as a tuple (*W*, F) where *W* is an actual word and F is a vector of features of that word. Features used include DigitNumber (15), SingleCap (M), GreekLetter (alpha), CapsAndDigits (I2), TwoCaps (RalGDS), InitCap (Interleukin), LowCaps (kappaB), lowercase (kinases), Hyphen (-), Fullstop (.) etc. (see Table 2 for more lexical word features). The probability $P(C_i|C_{i-1})$ of transition from the current state (i.e., NE type) C_{i-1} to next state C_i depends on the current and previous word and their features and the current state. It is computed as follows in Exhibit 1.

The λ_I's are constants such that $\sum_{i=1}^{5} \lambda_i = 1$. The authors manually set the values of λ_I's; but techniques such as Baum re-estimation algorithm could be used to estimate them. $f(X|Y)$ denotes the maximum likelihood estimate of the corresponding probability $P(X|Y)$; e.g., $f(X|Y) = \#(X \wedge Y)/\#Y$, where $\#(X \wedge Y)$ and $\#Y$ denote the number of occurrences of event (X and Y) and Y respectively. The above formula employs a progressive back-off to lesser details. For example, suppose C_{i-1} = PERSON, W_{i-1} = **Thomas**, W_i = **Alva**. Suppose we use only 2 word features: InitCap and Lowercase; thus $F_{i-1} = (1, 0)$, $F_i = (1, 0)$. If the sequence

Exhibit 1.

$$
\begin{aligned}
&P(C_i \mid C_{i-1}) = P(C_i \mid< W_i, F_i >, < W_{i-1}, F_{i-1} >, C_{i-1}) \\
&= \lambda_0 f(C_i \mid< W_i, F_i >, < W_{i-1}, F_{i-1} >, C_{i-1}) + \lambda_1 f(C_i \mid< \_, F_i >, < W_{i-1}, F_{i-1} >, C_{i-1}) \\
&+\lambda_2 f(C_i \mid< W_i, F_i >, < \_, F_{i-1} >, C_{i-1}) + \lambda_3 f(C_i \mid< \_, F_i >, < \_, F_{i-1} >, C_{i-1}) + \lambda_4 f(C_i \mid C_{i-1}) \\
&+\lambda_4 f(C_i)
\end{aligned}
$$

Table 2. Orthographic features useful for NER

Feature	Example	Feature	Example	Feature	Example
CapAllLower	**John**	AlphaDigits	**pm50**	AlphaDash	**Hayes-Roth**
CapMixedAlpha	**NFCappaB**	DigitsAlpha	**22A**	DigitsComma	**12,000,000**
AllCaps	**IBM**	SingleCap	**P**	DigitsDot	**12.34**
SingleDigit	**9**	SingleCapDot	**P.**	ContainsAt	**x.y@z.com**
TwoDigits	**99**	LowMixedAlpha	**mRNA**	ContainsDot	**x.y@z.com**
FourDigits	**1999**	DigitAlphaDigit	**32Dc13**	ContainsDash	**12-Dec-1999**
AllDigits	**12345**	AlphaDigitAlpha	**IL23R**	ContainsGreek	**NFkappaB**
RomanNumeral	**IV,xi**	AllCapsDot	**I.B.M.**	AllCapsAnd	**AT&T**

of words **Thomas** followed by **Alva** has not occurred in the corpus, then the transition probability is estimated using only F_i without using the word **Alva**. If the corpus does not contain **Thomas** followed by a word with features (1, 0) then the probability is estimated using only the F_{i-1} and F_i and so on. The probability that the first word in a sentence will belong to some NE type C_j is computed using a similar but separate formula and depends only on the first word and its features, since there is no previous state and previous word.

(Zhou and Su 2002) is also an HMM-based NER system. One distinguishing feature of this system is the use of an HMM-based chunk tagger to identify chunks (i.e., sequences of words) which are candidates for NE. A separate HMM is then used to assign an NE type to each chunk. Another interesting aspect of this work is the use of a feature which checks whether any of the words in the current chunk are part of an NE already identified in the text. The system also integrates various gazetteers (e.g., holidays, cities) and uses match of the current chunk with any of the gazetteers as a feature.

In a different HMM-based approach to NER, called *character-level HMM*, characters (rather than words and phrases) are taken as the primary representation of the text. The idea is to use character-level features to perform NER; e.g., PERSON names often begin with a capital letter, have a mixed case and are preceded by character sequences such as **Mr.**

We describe the model of (Klein *et al* 2003). States encode an NE type and observations are characters (a character is emitted one at a time). Each state is a pair (t, k), where t is an NE type (such as PERSON, LOCATION, ORG etc. including OTHER) and k is the length of the time the model has been in state t; e.g., the state (PERSON, 2) indicates the state reached after processing the second letter in a person's name. The final letter of a word is followed by a space (inserted, if not present in the text) and the model transitions to a special state like (PERSON, F). Also, when k reaches n (the n-gram history order), it is not incremented any further. The transitions are defined in such a way that a state like (PERSON, 2) can transition to (PERSON, 3) or (PERSON, F) only. A final state like (PERSON, F) can only transition a beginning state like (OTHER, 1).

Probability of emitting a particular character in state s depends on the last $n-1$ observed characters: $P(c_0|c_{-5}c_{-4}c_{-3}c_{-2}c_{-1}, s)$, where c_0 is the current character, c_{-1} is the previous character and so on. Empirically, $n = 6$ is recommended. For example, P(s|**Thoma**,PERSON,6) is the probability of emitting character **s** in state (PERSON, 6) when the last 5 observed characters were Thoma. This probability is estimated from the corpus and smoothed using the method of deleted interpolation. The state transition probabilities (e.g., probabilities of going from (PERSON, 2) to either (PERSON, 3) or (PERSON, F)) are also estimated from the corpus. The Viterbi decoding is done in a standard way. For example, given the observed character sequence **I flew from Washington to Denver** the most likely sequence of states can now be found using the Viterbi algorithm.

Maximum Entropy Models

Several authors have used a *maximum entropy* (*ME*) based approach for NER; e.g., (Bender *et al* 2003), (Curran and Clark 2003), (Chieu and Ng 2003). ME based approaches have also been used for languages other than English, such as German, Spanish and Dutch. ME based approaches have been used for detecting DSNE in biological domain; e.g., (Lin *et al* 2004), (Raychaudhuri *et al* 2002). ME approach (which is a supervised approach) is used for NER because of its superior method of estimating a most uniform PDF from training data i.e., one which makes the least additional assumptions. This PDF is then used for predicting the NE type for given sequence of words. The ME approach is often used in conjunction with an HMM approach. The specific works of various authors differ mainly in the features used. See (Berger *et al* 1996) for a good introduction to ME methods for natural language applications.

Given a labeled training dataset (e.g., a corpus containing tagged occurrences of NE), many features (or summary statistics) can be defined, each of which characterizes a particular aspect of the phenomenon (rules used for NE tagging, in our case). Often each feature is modeled as a binary-valued function f_j of arity more than 1, where the first argument denotes the NE type for the i^{th} word in a sentence and the rest of the arguments denote the context of that word. A feature can be thought of as a black box that takes some context of the current word as input and produces the NE tag for the current word as output. Let W_i and C_i denote the *random variables* (*RV*) which take on values as words and NE types respectively (W_i is the i^{th} word in a sentence and C_i is its NE type). Then some examples of features are as follows:

$f_1(C_i, W_i) = 1$ if W_i begins with a capital letter and C_i = PERSON; 0 otherwise

$f_2(C_i, W_{i-1}) = 1$ if POS tag of W_{i-1} is verb and C_i = PERSON; 0 otherwise

$f_3(C_i, W_{i-1}) = 1$ if POS tag of W_{i-1} is preposition and C_i = OTHER; 0 otherwise

Expected value of each feature may be considered as a summary statistic of the training dataset. For example, in a particular corpus, one may find that 37% words that begin with a capital letter were tagged as PERSON; 19% words that follow a verb may be found to be tagged as PERSON; and 88% words that follow a preposition may be found to be tagged as OTHER. The empirically observed expected values of f_1, f_2 and f_3 are $\mathbf{E}_{OBS}(f_1) = 0.37$, $\mathbf{E}_{OBS}(f_2) = 0.19$ and $\mathbf{E}_{OBS}(f_3) = 0.88$ respectively. The expected value of feature f_1 in the training dataset is defined as follows:

$$E_{OBS}(f_1) = \sum_{c \in C,\ \text{words}\ w} P_{OBS}(c, w) f_1(c, w)$$

where $P_{OBS}(c, w)$ is the probability of observing the word w tagged with NE type c in the training dataset; this probability is estimated as relative frequency.

Suppose that we want to identify a *probability distribution function* (*PDF*) $P(C_i|W_i, W_{i-1})$ that predicts the NE type of the current word given its previous word. The observed statistics impose three constraints on the desired PDF: the expected values of features f_1, f_2 and f_3 as computed using P should be the same as those empirically observed in the training dataset.

$\mathbf{E}(f_1) = \mathbf{E}_{OBS}(f_1)$ i.e., $\mathbf{E}(f_1) = 0.37$ and
$\mathbf{E}(f_2) = \mathbf{E}_{OBS}(f_2)$ i.e., $\mathbf{E}(f_2) = 0.19$ and
$\mathbf{E}(f_3) = \mathbf{E}_{OBS}(f_3)$ i.e., $\mathbf{E}(f_3) = 0.88$

In general, an infinite number of PDFs may be consistent with the given constraints. The ancient *principle of insufficient reason* says that one should choose that hypothesis which is consistent with the given facts and which makes the fewest possible additional assumptions. A mathematical formalization of this principle is the *principle of maximum entropy* (*ME*): choose that PDF (among those consistent with the given constraints) which is as uniform as possible i.e., one which has the maximum entropy. It can be shown that such a PDF always exists and is unique. *Entropy* of a PDF $P(X)$ is the expected number of bits required to specify values of RV X drawn according to P.

$$H(P(X)) = \mathrm{E}\left(\frac{1}{\log_2(P(x))}\right) = -\sum_x P(x)\log_2(P(x))$$

Here, x varies over all values of the discrete RV X. Analogously, the entropy of a conditional PDF $P(Y|X)$ is defined as:

$$H(P(Y \mid X)) = -\sum_x P(x)\sum_y P(y \mid x)\log_2(P(y \mid x))$$

In our case, the principle of ME has suggested to identify a conditional PDF (e.g., $P(C_i|W_{i-1}, W_i)$) which satisfies the given constraints and which has the maximum conditional entropy. This is a problem in constrained optimization: maximize a function having the form of conditional entropy under given constraints having the form $\mathbf{E}_{OBS}(f_j) = \mathbf{E}P(f_j)$. The method of Lagrangian multipliers can be applied to find the optimal solution to this constrained optimization problem. It can be shown that the required conditional CDF (which is consistent with the given constraints and has the maximum entropy) has a particular form. In our example, the required conditional PDF $P(C_i|W_{i-1}, W_i)$ has the following form:

$$P(C_i = c \mid W_{i-1} = u, W_i = v) = \frac{e^{\lambda_1 f_1(c,v)+\lambda_2 f_2(c,v)+\lambda_3 f_3(c,v)}}{\sum_{d\in C} e^{\lambda_1 f_1(d,v)+\lambda_2 f_2(d,v)+\lambda_3 f_3(d,v)}}$$

$\lambda_1, \lambda_2, \lambda_3$ are unknown Lagrangian multipliers, one for each constraint. Once their values are determined, we can compute the required probability for any situation; e.g., the probability that the NE type for **John** given its previous word is **to** is computed as:

$$P(C_i = PERSON \mid W_{i-1} = \text{to}, W_i = \text{John}) = \frac{e^{\lambda_1\cdot 1+\lambda_2\cdot 0+\lambda_3\cdot 0}}{e^{\lambda_1\cdot 1+\lambda_2\cdot 0+\lambda_3\cdot \mathbf{0}} + e^{\lambda_1\cdot 0+\lambda_2\cdot 0+\lambda_3\cdot 1}} = \frac{e^{\lambda_1}}{e^{\lambda_1} + e^{\lambda_3}}$$

Assuming that the predicted NE type for **John**, given its previous word is **to** is PERSON, we have f_1(PERSON, **John**) = 1, f_2(PERSON, **to**) = 0, f_3(PERSON, **to**) = 0 and f_3(OTHER, **to**) = 1. In the testing phase, the NE type for **John** would be unknown. The above probability computation can be made for all NE types (e.g., LOCATION, OTHER etc.) and the one having the maximum value is output as the prediction. The optimal values for the Lagrangian multipliers can be found using algorithms such as the Generalized Iterative Scaling.

To summarize, given a labeled training dataset, the principle of ME can be used to estimate a PDF that assigns a NE label to any word in a given sentence, given its context. This is a supervised learning approach, since the constraints are based on a labeled training dataset. The ME approach crucially depends on the feature functions used as summary statistics for the training dataset. The chosen features must be relevant and sufficient for NER. Hundreds of different features can be defined, based on orthographic, lexical and syntactic analysis of the given word and its context. Hence, selection of appropriate features is an important task in the ME approach. Feature selection is a large research area in itself; see (Berger *et al* 1996) for a discussion relevant to the ME approach.

The ME approach has been extended in several ways when used for NER. A common approach (Barthowick 1999) simultaneously assigns NE types to all words in a sentence, rather than one word at a time. Given a sequence of words w_1, …, w_n in a sentence, the corresponding sequence of NE types c_1, …, c_n is one which has the highest posterior probability among all possible NE type sequences:

$$c_1,\dots c_n = \arg\max_{d_1,\dots,d_n} P(d_1,\dots,d_n \mid w_1,\dots,w_n)$$

Such a maximum posterior probability sequence is computed using the Viterbi algorithm. Basically, one defines a Hidden Markov Model, where each state corresponds to an NE type. Transition probability matrix (probability p_{ij} that an NE type i is followed by NE type j) is estimated from the training dataset.

Each state has a probability distribution for emitting a word (or its corresponding feature vector). The Viterbi algorithm efficiently estimates the most likely sequence of states traversed by the HMM in order to generate the given word sequence.

(Chieu and Ng 2003) have proposed the use of global features, based on the entire corpus, in addition to features based on only the local context of a word in a sentence. For example, they compile various lists from the corpus: UNI is a list of words that frequently precede the occurrence of a given NE type; UBI is a list of bigrams that frequently precede the occurrence of a given NE type (e.g., **city of** for LOCATION); SUF is a list of 3-letter suffixes that frequently terminate words in a given NE type (e.g., **inc** for ORG); FUN is a list of function words that frequently occur in occurrences NE type (e.g., **van der** for PERSON). They defined features based on these lists; e.g., whether a word contained a suffix from SUF. They also include features that detect use of acronyms. For example, if **Federal Communications Commission** is detected to be an ORG in a document then the word **FCC** in that document should get tagged as an ORG, since it matches with the sequence of first letters in a known ORG.

(Park *et al* 2006) has used a 2-step approach − *NE identification* (also called *NE boundary detection*) followed by NE classification − for biomedical NER, where they used a separate ME classifier for each step. Mostly lexical features were used in each step and they were divided into groups such as salient words, morphological patterns and collocations.

(Chen and Rosenfeld 1999) proposed a smoothing method to compute the ME model parameters (i.e., the Lagrange multipliers) by assuming a Gaussian prior distribution on their values; all Langrage multipliers have the same Gaussian distribution. This method avoids very large values and also eliminates features that "fire" rarely. (Bender et al 2003) and (Curran and Clark 2003) used this method for their ME-based NER.

Support Vector Machines

Consider a labeled training dataset {($\mathbf{x}_1$, y_1), ($\mathbf{x}_2$, y_2), . . ., ($\mathbf{x}_N$, y_N)} where each $\mathbf{x}_i$ is a point in n-dimensional real space (value of every feature is a real number) and each class label y_i is either +1 or −1. A *support vector machine* (*SVM*) (Vapnik 1998) is a binary classifier that learns the equation $\mathbf{w} \cdot \mathbf{x} + b = 0$ of a separating hyperplane that has the maximum margin. Here, **w** is the vector of feature weights and b is the offset of the hyperplane from the origin. Both **w** and b are learned from the training dataset. Given a query point $\mathbf{q} \in \mathbf{R}^n$, the classification decision is made as follows: if $\mathbf{w} \cdot \mathbf{x} + b > 0$ then predict class label +1 else predict class label −1. *Margin* is defined as the distance between the hyperplane and the *support vectors* (i.e., points from the training dataset which are nearest to the hyperplane). Hyperplane with the maximum margin tends to be more accurate for classifying unseen data i.e., it has a better generalization. Identifying the maximum margin hyperplane can be formulated as a quadratic optimization problem. Its solution (i.e., formulas to compute the values for w and b) requires only computations of the inner products $\mathbf{x}_i \bullet \mathbf{x}_j$ of the feature vectors in the training dataset. Note that a hyperplane is a linear separator for the training dataset.

If the training dataset is not linearly separable, then SVM employs the so-called *kernel trick*. Each n−dimensional point $\mathbf{x}_i$ in the training dataset is transformed into a (usually higher dimensional) point $\Phi(\mathbf{x}_i)$ such that the inner product in the transformed space is computed efficiently using the *kernel function K* i.e., $\Phi(\mathbf{x}_i) \bullet \Phi(\mathbf{x}_j) = K(\mathbf{x}_i, \mathbf{x}_j)$. Hopefully, the training dataset in the transformed space is linearly separable and then the hyperplane in the transformed space can be used to classify query points. A linear separator in the transformed space corresponds to a non-linear separator in the original space. The kernel function K is required to satisfy certain conditions (e.g., K should be positive definite). For example, the transformation

$$\Phi(x_1, x_2) = \left(x_1^2, x_2^2, \sqrt{2}x_1, \sqrt{2}x_2, 1\right)$$

maps a 2-dimensional point (x_1, x_2) to a 5 dimensional point. The polynomial kernel function $K(\mathbf{u}, \mathbf{v}) = (\mathbf{u} \bullet \mathbf{v} + 1)^m$ ($m = 2$ here) satisfies the required property that $\Phi(\mathbf{u}) \bullet \Phi(\mathbf{v}) = K(\mathbf{u}, \mathbf{v})$, as can be checked.

SVM is a binary classifier i.e., it handles only 2 classes. But in general, the number of NE types M is more than 1 (e.g., PERSON, LOCATION, ORG). For this reason, several methods have been proposed to extend SVM for multi-class classification. In *one-vs-rest* method, M SVMs are constructed where the i^{th} SVM predicts class +1 for i^{th} class (NE type) and −1 for the rest of the classes together. The query point **q** is given to all M SVMs and the predicted class for **q** is the one which has the maximum value of $\mathbf{w} \bullet \mathbf{q} + b$. In the *pair-wise* method (Krebel 1999), $M(M-1)/2$ SVMs are constructed, where a particular SVM predicts whether the given point belongs to class i or class j. Treating the decision of each SVM as a *vote*, the predicted class for a given query point **q** is the one that gets the maximum votes. (Isozaki and Kazawa 2002) uses a one-vs-rest approach for multi-class NER using SVM for Japanese; they also propose a feature selection method as well as a method for reducing the dot product computations of the given sample with all support vectors in all the SVMs.

In the training phase, each word in the labeled input text is represented as a vector of numeric features and a class label (NE type). Features for a word can be lexical (AllCap, AllCapDot, InitCap, AllLowerCase, CapDot, ContainsDash, ContainsGreek etc.), POS tag, syntactic tag (NP, VP etc.). Other features can also be used such as whether the word contains a specific prefix, suffix (e.g., **enese**) or substring. Some features for a word may refer to properties of previous or next words.

Several implementations of SVM are available in the public domain such as LIBSVM and SVM^{light}. Several approaches have been developed for using SVM for NER (Yamamoto et al 2003), (Lee et al 2004), (Kazama *et al* 2002), (Song et al 2005), (Takeuchi and Collier 2002) which have more or less followed the above approach, with some modifications for efficiency improvements.

(Lee et al 2004) split the NER task into two parts: *identification* or *boundary detection* (identifying the contiguous sequence of words called region that forms a NE) and *classification* (assigning an NE type such as PERSON to an identified region). Identification is itself a binary classification task: *T* (current word is part of a NE) or *O* (current word not part of an NE). One SVM is created for identification and *M* SVMs are created for classification (*M* = number of NE types). They correct the errors made by the SVM in identifying the boundary using an entity-word dictionary.

Since NE occurrences are relatively less frequent, there is severe class imbalance in the training dataset; e.g., PERSON occurs much less frequently than OTHER NE type. One solution to handle such class imbalance (Kazama *et al* 2002) is to split the OTHER NE type into several sub-types (e.g., one sub-type for each of the 45 POS tags in Penn TreeBank). Other problem is related to data sparseness: creating a large and high quality tagged corpus is difficult. One approach to handle data sparseness is to use more general features (e.g., disjunctions) which apply to more instances. Another approach has used the state of an HMM trained on an untagged corpus as a word feature.

Since a tagged corpus is difficult to create, (Yi *et al* 2004) and (Sasano 2003) have developed techniques to expand the corpus by adding virtual examples using domain knowledge. (Yi *et al* 2004) also proposed the use of *M* edit distances as word features: average edit distance of a given word from each entry in a dictionary for a specific NE type. Edit distance captures structural similarity between two words; if a word is similar (less edit distance) to a known example of an NE type, then it is likely to have the same NE type. *Edit* (or *Levenshtein*) *distance* between two strings *X* and *Y* is defined as the minimum number insert, delete, substitute operations required to transform *X* into *Y*. For example, the edit distance between **kitten** and **sitting** is 3.

Other Approaches

Other supervised learning approaches have been used for NER; e.g., decision trees (Sekine 1998), conditional random fields (CRF) (Lafferty 2001), (Watanabe 2002), (McCallum and Li 2003) and so on. Since the Web is a vast knowledge source, (Kazama and Torisawa 2007) uses category labels from Wikipedia definition of a word as a feature when learning a CRF-based NE classifier. (Settles 2004) uses CRF for biomedical NER, which used, apart from standard features, a number of gazetteers containing semantic knowledge (such as known genes, chromosome locations, viruses, amino acids, proteins and cell-lines etc.). (Gliozzo *et al* 2005) describes some pre-processing techniques that helps in improving classifier performance and accuracy. (Meulder and Daelemans 2003) uses a feature-weighted nearest neighbour classifier: all training examples, along with their features and NE type are stored and when given a new candidate word (and its features), the NE type for it is decided from the NE types of the *k* nearest examples to it. (Krishnan and Manning 2006) uses two coupled CRF classifiers that use non-local (i.e., long-range dependency) features; e.g., *label consistency* or *one use per document* (all occurrences of **New York** should get the same NE type in a document).

(Sun *et al* 2002) used the *n-gram statistical language models* (*LM*) for performing NER in Chinese. Each NE type is a class and class OTHER indicates a non-NE word. In a LM, the probability of n^{th} word is estimated using probabilities of the occurrences of previous $n-1$ words. The context model computes the probability $P(c_3|c_2, c_1)$ for a sequence w_1, w_2, w_3 of words; these probabilities are estimated using a corpus where NE are labeled. The entity model (separate for each NE type) computes the probability of seeing the given sequence of words assuming that each word is labeled with the given NE type; e.g., probability of seeing the sequence of words Air China Corporation assuming that all three words are labeled as ORG. These probabilities are estimated using a corpus and lists. *Deep Belief Net* (*DBN*) have also been used for NER. A DBN is a multi-layer neural network that consists of one or more Restricted Boltzmann Machine layers and a Back Propagation layer. DBN is claimed to have both an efficient learning mechanism and good expressive power.

UNSUPERVISED LEARNING APPROACHES

A basic problem with supervised learning approaches to classification is their crucial dependence on the availability of large, representative and high-quality labeled training datasets. Typically, such labeled datasets are created manually by experts. This makes creation of labeled datasets an expensive, time-consuming and error-prone task. Further, sometimes even the experts disagree among themselves, and this needs to be handled. *Unsupervised learning* approaches work on an unlabeled set of documents to automatically infer occurrences of NE. One typically starts with a small given *seed list* of known NE of a specific NE type (e.g., a small list of known person names) and attempts to discover additions to this list. This is done by detecting common patterns in the usage of given examples and conjecturing that any words that fit into these patterns are candidate NE occurrences. For example, examining the sentences in which cities from the seed list occur, one may find that in several sentences they were preceded by words **city of**. Generalizing this pattern, in another sentence the word following **city of** is a candidate occurrence of NE type CITY. The main question then is: how to discover common patterns in the usage of a given instances of an NE type? We discuss two prominent unsupervised approaches to NER. Such approaches are also called *bootstrapping* or *weakly supervised*.

Unsupervised approaches are somewhat different from *semi-supervised* approaches typically used in classification tasks. Here, in addition to the seed list, a large set of unlabeled examples is also available; e.g., in NER, occurrences of proper nouns may be considered as unlabeled examples and each NE type corresponds to a class. The classifier design takes into account this unlabeled examples.

(Collins and Singer 1999) start with a given list of simple *decision rules*; e.g., *contains*(*Incorporated*) → *ORG* and *full_string* = *Microsoft* → ORG. An untagged corpus is then examined (using syntactic rules) to identify candidate proper names (sequence of consecutive proper nouns). For example, one syntactic rule extracts a proper name if it occurs within an NP and its last word is the head of the NP (e.g., **Al Gore** is extracted from the NP **vice president Al Gore**). Another rule extracts a proper name in an NP *X* if *X* has an appositive modifier NP whose head is a singular noun; e.g., proper name **Maury Cooper** in ..., **says Maury Cooper, a vice president at S&P**. is extracted because the head of the appositive modifier NP is a singular noun (**president**). In each iteration of their DL-coTrain algorithm, first context rules and then spelling rules are discovered as follows. The unlabeled text is labeled using current rules. Then context rules (e.g., *context* = president or *context_type* = *appos*) having precision above a given threshold are extracted. Next, using these new and old rules, new spelling rules (of the form *AllCap1* → *ORG* or *nonalpha* = .) having high precision are extracted in a similar manner. The iterations stop when a limit on the maximum number of new rules is exceeded.

Clearly, a look up approach to NER would be very efficient. It would also be accurate, provided the lists (*gazetteers*) of known NE are guaranteed to be complete. *Named Entity Extraction* (*NEX*) also called *automatic gazetteer construction* consists of a system that automatically constructs a list of the instances of entities of a given type (e.g., CITY) from a given source such as a set of Web pages. The system is typically weakly unsupervised, in the sense that it does not use a large hand-tagged corpus where instances of the given NE type are explicitly identified. It may, however, start with a small set of seed instances of the NE type; e.g., an NEX system to identify all CITY names may start with a seed set of 4 cities {'**New York**', **Berlin**, **London**, **Seoul**}. The NEX task is somewhat different from the NER task because of ambiguity. For example, even if **Jobs** is present in the previously created list of PERSON instances, it still remains a challenge to recognize whether the word **Jobs** in a given document is used as a person name or as a common noun.

KNOWITALL (Etzioni *et a*l 2005) is an unsupervised NEX system that uses the Web as a corpus to create a list of instances of the given NE type (e.g., CITY or FILM). It starts with a set of domain independent patterns instantiated for the given NE type. One such pattern is that if one entry in a list is of the given NE type then all others in the list must also be of the same NE type. This pattern is specified as: NP1 such as ListOfNP2, where head of NP1 is a plural form of the NE label (**cities**) and head of each NP2 is a proper noun (**cities such as London, Paris and Tokyo**). There are additional constraints such as NP1 and each of NP2 must be simple noun phrases and the NE label should be the head of NP1 (which avoids pitfalls such as **city clubs such as** ... where city is not the head of the NP. The proper noun test of NP2 is important to avoid examples such as **Detailed maps for several cities such as street maps, railway maps** .... Whenever an occurrence of an extractor rule is detected in the given text, an assessor module checks whether there is high *point-wise mutual information* (*PMI*) between the occurrence and some automatically generated discriminator phrases. For example, to check whether **Liege** is a CITY, the assessor module computes the PMI between **Liege** and phrases such as **Liege is a city**. PMI between an instance *I* and a discriminator phrase *D* is computed as the ratio of the number of times *I* and *D* occur together (e.g., in the same sentence) to the number of times *I* occurs alone:

$$PMI(I, D) = \frac{\#(D \wedge I)}{\# I}$$

If there are *m* discriminator phrases, then we have *m* PMI values for the given instance. These *m* PMI values are converted to a Boolean feature vector $F_I = (f_1, f_2, \ldots, f_m)$ for *I* where $f_i = 1$ if the i^{th} PMI value is above its threshold and 0 otherwise. KNOWITALL then uses a *Naive Bayes Classifier* (*NBC*) to classify F_I as belonging to the given NE type or not. To estimate the conditional probabilities needed by the NBC, KNOWITALL uses a labeled training dataset of $k = 10$ positive and $k = 10$ negative seed examples for each NE type. KNOWITALL has a simple bootstrapping algorithm to automatically select these 2*k* seed examples (based on their hit counts and PMI scores). Some examples of the discriminator phrases automatically identified by the bootstrapping method are: <I> **is a city**, <I> **and other towns**, **cities** <I> and **cities including**. The threshold for each discriminator phrase is also automatically determined using a separate labeled training dataset of $k = 10$ positive and $k = 10$ negative seed examples for each NE type. KNOWITALL also contains a learning algorithm to automatically discover additional extractor patterns (which are very different from the given "built-in" patterns like NP1 **such as** NP2List discussed above); e.g., **headquartered in** <CITY>. Essentially, this algorithm finds phrases that co-occur frequently with known examples of the given NE type and evaluates them using a metric based on a modified notion of precision and recall.

Among other work on unsupervised NER, (Watanabe et al 2003) uses CRF to create gazetteers from Wikipedia. (Jimeno *et al* 2008) compares various NER methods for automatically creating a gazetteer as well as an annotated NER corpus for disease names in medicine. Given a seed list of NE type examples, (Talukdar *et al* 2006) learns a pattern (as an automaton) from their contexts (*k* words before and after). The contexts are pruned using the IDF measure and then an automaton is induced from the contexts using a grammatical induction algorithm. Each transition in the induced automaton is given a probability and transitions with weak probability are pruned. (Meulder and Daelemans 2003) use a simple conjunction-based generalization to construct a gazetteer from a seed list (if a word in a conjunction is a known NE type then the other words in the conjunction also have the same NE type). (Liao and Veeramachaneni 2009) is an iterative unsupervised algorithm that starts by learning a CRF NE classifier from seed examples, and then identifies NE occurrences from the text that the classifier classifies with low confidence but for which strong independent evidence is available; e.g., if **Safeway Inc.** is known to be ORG then **Safeway** in **Safeway has recently opened** ... is also likely to be ORG with high confidence, even though the classifier may have low confidence on it. The CRF classifier is retrained after adding such examples to the training dataset. The process stops when no new examples are found. (Kim *et al* 2002) use a similar iterative approach (seed examples, train classifier, add new examples re-train) for NER in Korean text, except that they used an ensemble of 3 machine learning methods (nearest-neighbour, network of Winnows and ME). Final NE type is selected by a voting mechanism based on the probability of correct decision for each of the 3 classifiers. (Shinyama and Sekine2004) propose an interesting method to identify NE from a set of comparable news articles (e.g., news articles reporting the same events but from different newspapers and on different days). The idea is that a general word like **killed** will have a very different distribution (in the corpus of comparable articles) than a NE word such as **Yitzhak** (e.g., diffuse versus spiky).

Unsupervised approaches have also been applied to the task of *fine-grained NER*, where the goal is to assign an appropriate sub-class from a given ontology *T* to each occurrence of a NE; e.g., each PERSON may be assigned a sub-class such as *politician*, *scientist*, *sports-person*, *film-star* or *musician*. The ontology *T* is generally organized as a hierarchy; e.g., a *scientist* may be *physicist*, *chemist* or *biologist*. Due to the large and changing nature of the ontology, it is very effort-intensive to create a manually tagged corpus with sufficient examples of each class. Hence, unsupervised approaches are attractive for fine-grained NER. Fine-grained NER is likely to be more useful for tasks related to semantic web.

(Fleischman and Hovy 2002) use supervised classification methods (decision trees, neural network, nearest neighbour, SVM and Naive Bayes) to classify PERSON NE into 8 subclasses (*athlete*, *politician*, *clergy*, *businessperson*, *artist*, *lawyer*, *scientist*, *police*) based on local context, topic-specific terms and as WordNet hypernyms for the context words. (Tanev and Magnini 2008) combines dependency-analysis results obtained from all the contexts in which a given seed example occurs into a single graph. Context of a candidate NE is then compared (for similarity) with each of these syntactic models (graphs) to decide its class. In a similar manner, (Ganti *et al* 2008) take the union of the words present around each occurrence of the given NEs and use frequently occurring *n*-grams in these aggregated contexts as features. For example, **painted** by may frequently occur around a given NE **Picasso**. They also use memberships of the context words in given lists of known NEs as features; e.g., if **NBA** is known to be a sport ORG then **Ming is drafted by NBA** gives a clue that **Ming** may be a sports-person. A separate classifier is then built for each subclass using this training data. (Ekbal et al 2010) develop an unsupervised method for acquiring a comprehensive dataset for fine-grained NER (for PERSON) by applying linguistic patterns (and filtering rules) to a corpus acquired from the Web; e.g., a pattern like [the|The]? [JJ|NN]* [NN] [NP] matches ... **writings of the abstract painter Kandinsky frequently explored similarities between** ... and can extract NE **Kandinsky** with fine-grained class label *painter*. They also develop an ME classifier for fine-grained NER.

NER IN OTHER LANGUAGES

Along with English, NER techniques have been developed for many other languages. These techniques can be broadly understood as either (i) language-specific NER techniques designed to use characteristics and linguistic knowledge of a particular language; or (ii) application of a language-independent NER technique across a class of related languages. Conferences CoNLL-2002 and coNLL2003 included a shared task for language independent NER. Here, we review only a scattering of the work in this area, as a more complete review of NER in other languages needs a separate paper. NER techniques have been applied to European languages (French, German, Spanish, Greek etc.), Asian languages (Arabic, Chinese, Japanese, Korean, Vietnamese) and Indic languages (Hindi, Urdu, Bengali, Tamil, Marathi, Oriya etc.). Several differences - such as richer morphology, gender sensitive word-forms, different word ordering and lack of capitalization - between English and many of the other languages make the NER task different (and sometimes harder). As an example, detecting whether a word (e.g., **Ganga**) is a proper or a common noun (POS tag is an important feature for NER) is difficult in Indic languages, because there is no capitalization - a problem that may also occur in noisy English texts such as blogs. One may have to refer to a lexicon to make such a decision (proper nouns are generally not present in a lexicon). Word segmentation itself is a major problem in many languages including Chinese and Japanese.

CONCLUSION AND FUTURE RESEARCH DIRECTIONS

We have reviewed (in a far from exhaustive manner) some major approaches to English NER. We now discuss some open areas in NER research.

One critical issue that has received less attention is that of post-processing of results produced by an NER system. NER is often accompanied by some post-processing to correct classification errors that may have occurred. (Lin *et al* 2004) propose a simple method to correct classification errors. They also propose a method for correcting NE boundary errors when only part of the NE has been detected correctly (e.g., rules for extending the detected NE to right or left). Combining the outputs of several NER systems, in the spirit of classifier ensembles, also has not received as much attention as it should have. Such classifier

ensemble methods have shown promise in that the overall accuracy is better than that of the constituent classifiers, in standard statistical classification tasks (not necessarily NER). (Florian *et al* 2003) uses a class-error based voting scheme to combine the outputs of NER classifiers based on ME, HMM, Robust risk minimization and transformation-based learning. (Thao *et al* 2007) compares 3 voting mechanisms (majority, total accuracy, class-wise accuracy) to combine CRF, SVM, Naive Bayes and decision tree based NER classifiers for Vietnamese (see also (Tsai *et al* 2006)). (Wang and Patrick 2009) reports a combination scheme to combine SVM, ME and CRF classifiers and its application to perform NER from clinical notes. (Ekbal and Bandyopadhyay 2010) use a majority voting approach to combine NER classifiers for Bengali based on ME, CRF and SVM and demonstrate an increase of about 11% over the best performing SVM classifier for this task.

Systematic comparison of various NER techniques, particularly for different languages, over different domains and across varied and unseen corpora, is an important issue. (Krishnarao *et al* 2009) compare CRF and SVM based NER systems for Hindi. (Petasis et al 2004) compares the performance of different NER systems on English, French, Greek and Italian web-pages. (Sekine and Eriguchi 2000) compare various techniques (ME, Decision Tree, HMM as well as hand-crafted patterns) for Japanese NER. NER from sources other than plain text (e.g., news articles) has received less attention, except possibly for HTML web-pages.

Building an NER system that works smoothly (with little or no tuning) on multiple types of text sources is a difficult task. It is often observed that an NER system trained on one type of text source (e.g., news articles) does not work well on other text sources (e.g., Web pages). (Maynard *et al* 2001) describes a system called Muse (based on the open GATE architecture framework for NLP systems), which is capable of NER from diverse types of text sources. A separate set of resources (patterns, gazetteers etc.) are developed for each text type (like emails, spoken text, scientific text, religious text). (Balasuriya *et al* 2009) evaluate an NER system trained on manually tagged Wikipedia pages.

We have barely mentioned in this paper other important problems in NER. First, there is *NE disambiguation*, where the task is to identify correct NE type for an identified NE instance; (e.g., does **Washington went ahead** mention a PERSON, a LOCATION, or an ORG?). Among much work done for NE disambiguation, we mention (Cucerzan 2007), (Bunescu and Pasca 2006) and (Han and Zhao 2009) which use Wikipedia as a knowledge source to perform NE disambiguation. Most pages in Wikipedia are associated with an entity or concept, along with NE type (PERSON, LOCATION, ORG etc.) and category/topic tags. In addition, much knowledge can be derived about the entity by analyzing the content of the associated page. For example, a document that contains the surface forms **Columbia** and **Discovery** is likely to refer to the Space Shuttle Columbia and the Space Shuttle Discovery because these candidate entities share the category tags *LIST_astronomical_topics*, *CAT_Manned_spacecraft*, *CAT_Space_Shuttles*, while alternative entity disambiguations, such as **Columbia Pictures** and **Space Shuttle Discovery**, do not share any common category tags.

(Nadeau *et al* 2006) uses the web as a source for NE disambiguation. (Mikheev 1999) proposes heuristics for NE-noun disambiguity (**Jobs** as a PERSON or noun). For example, in a given document, assume that a word or phrase with initial capitals (e.g., **Jobs**) is a NE unless (1) it sometimes appears in the document without initial capitals (e.g., **jobs**), (2) it only appears at the start of a sentence or at the start of a quotation (e.g., "**Jobs that pay well** ..." or (3) it only appears inside a sentence in which all words with more than three characters start with a capital letter (e.g., a title or section heading).

NE-NE ambiguity is harder to resolve; e.g., is **France** a LOCATION or PERSON? (Nguyen and Cao 2008) proposes a hybrid methodology for NE disambiguation that uses a both statistical and rule-based steps in an iterative manner. Knowledge from WordNet ontologies can also be used for NER; e.g., see (Negri and Magnini 2004). For example, WordNet hypernym tree for **Mississippi** includes location. Further, WordNet gloss and relations can provide trigger words that can be used for NER.

In *NE boundary detection*, the task is to identify the correct NE boundary (e.g., does the **Alliance for Democracy in Mali** mention one, two, or three entities?). (Palmer and Day 2006) and (Nadeau *et al* 2006) prescribe heuristics; e.g., merge all consecutive words of the same NE type and every NE type occurrence with any adjacent capitalized words. As an example, if **Jean** and **Smith** are both marked as PERSON then mark Jean Smith as PERSON and if **Red Sox** is an ORG then tag **Boston Red Sox** also as ORG.

In many domains (e.g., biomedical), there are several names for the same conceptual entity. In that case, a normalization step is required, where two different NE occurrences are mapped to a unique conceptual NE; e.g., (Cohen 2005) uses techniques such as removing noise words and identifying orthographic variants (e.g., **IL-10** and **IL 10**) to perform normalization.

With steady progress in NER, it has now become possible to look at another important problem, which may be called *NE Relation Rec-*

ognition (*NERR*): that of identifying a semantic relation (if any) that connects NE instances; e.g., an occurrence of a PERSON and an ORG may be connected through relations such as JOINED, LEFT and HOLDS-POSITION. Similarly, several relations are possible between PERSON and PERSON (PARENT, SIBLING, GRANDPARENT, ASSOCIATE, BOSS etc.). Conversely, detection of a particular relation among candidate NE instances may itself help in NER. Note that relation extraction is a more general problem than NERR in the sense that relations may exist between non-NE words as well; e.g., **car** and **wheel** have a relation PART-OF. Special kinds of relation features can be devised to perform NERR, either in a supervised or unsupervised manner. A good starting point for this work is the collection of papers in the special shared task track on NERR in ACE04 conference (Doddington *et al* 2004). Kernel-based approaches are being explored for NERR in particular and relation extraction in general; see, for example, (Zhao and Grishman 2005) and (Culotta and Sorensen 2004).

While the output of an NER system over a given set of documents is useful in itself, other uses for NER have not been widely explored; see (Bellot et al 2002) for question-answering, (Montalvo et al 2006) for document clustering and (Aramaki *et al* 2009) for document summarization. In particular, the use of NER in semantic web tools and applications needs to be explored more extensively.

It is quite clear from the literature that there is a need for a systematic linguistic theory of named entities, both generic and domain-specific. While we have a number of operational features, rules etc. to identify named entities, we need a more linguistic (or more semantic) theory, which we can use to answer basic questions like the following. What is a named entity, really? What are the characteristics of a NE? What are the relationships between NE? For example, what is the semantic difference between a domain-specific NE and a generic NE? It is clear that for outperforming human experts, the next generation NER system would need to incorporate a considerable amount of linguistic, domain and common sense knowledge. Automatically creating such NER-related knowledge (particularly, linguistic knowledge) in a form that can be reused, edited and understood by human experts, is a challenging task. Many researchers have proposed a "look up" approach to NER based on large gazetteers of known NE. Hence, creating such gazetteers for each type of NE is an important task. We have already discussed various NEX systems for the purpose of automatically creating gazetteers or annotated NER corpus.

Designing techniques for automatically discovering the features relevant for NER, particularly in a language independent manner, is also helpful, since identifying the right features required a lot of linguistic knowledge. (Li and McCallum 2003) use CRF for performing NER in Hindi, where the features are not hard-coded into the system but are induced from the labeled training data using an automatic feature induction technique. CRFs are undirected graphical models used to calculate the conditional probability of values on designated output nodes given values on other designated input nodes. Features are arbitrary (typically Boolean) functions about the two consecutive states, any part of the observation sequence and the current position; e.g., a conjunctive feature may ask whether a word is a known ORG and is followed by the word **spokesman**.

REFERENCES

Aone, C., Halverson, L., Hampton, T., & Ramos-Santacruz, M. (1998). SRA: Description of the IE2 system used for MUC-7. In *Proceedings of the 7th Message Understanding Conference*.

Appelt, D., Hobbs, J., Bear, J., Israel, D., Kameyama, M., & Martin, D. … Tyson M. (1995). SRI international FASTUS system: MUC-6 test results and analysis. In *Proceedings of the 6th Message Understanding Conference*, (pp. 237–248).

Aramaki, E., Miura, Y., Tonoike, M., Ohkuma, T., Mashuichi, H., & Ohe, K. (2009). Text2table: Medical text summarization system based on named entity recognition and modality identification. In *Proceedings of the BioNLP-2009 Workshop*, (pp. 185–192).

Balasuriya, D., Ringland, N., Nothman, J., Murphy, T., & Curran, J. (2009). Named entity recognition in wikipedia. In *Proceedings of the 2009 Workshop on the Peoples Web Meets NLP* (*ACL IJCNLP 2009*), (pp. 10–18).

Barthowick, A. (1999). *A maximum entropy approach to named entity recognition*. Unpublished doctoral dissertation, New York University.

Bellot, P., Crestan, E., El-Beze, M., Gillard, L., & de Loupy, C. (2002). Coupling named entity recognition, vector-space model and knowledge bases for TREC 11 question answering track. In *Proceedings of 11th Text Retrieval Conference* (*TREC-2002*).

Bender, O., Och, F. J., & Ney, H. (2003). Maximum entropy models for named entity recognition. In *Proceedings of Conference on Computational Natural Language Learning* (*CoNLL-2003*), (pp. 148–151).

Berger, L., Pietra, S. A. D., & Pietra, V. J. D. (1996). A maximum entropy approach to natural language processing. *Computational Linguistics*, *22*(1), 39–72.

Bikel, D. M., Schwartz, R., & Weischedel, R. M. (1999). An algorithm that learns what's in a name. *Machine Learning*, *34*, 211–231. doi:10.1023/A:1007558221122

Black, W., Rinaldi, F., & Mowatt, D. (1998). FACILE: Description of the NE system used for MUC-7. In *Proceedings of the 7th Message Understanding Conference*.

Bunescu, R., & Pasca, M. (2006). Using encyclopedic knowledge for named entity disambiguation. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics* (*EACL-2006*), (pp. 9–16).

Chen, S., & Rosenfeld, R. (1999). *A Gaussian prior for smoothing maximum entropy models*. Unpublished technical report CMUCS-99-108, Carnegie Mellon University.

Chieu, H., & Ng, H. (2003). Named entity recognition with a maximum entropy approach. In *Proceedings of Conference on Computational Natural Language Learning* (*CoNLL-2003*), (pp. 160–163).

Chinchor, N. (1998). MUC-7 named entity task definition, v3.5. In *Proceedings of the 7th Message Understanding Conference*.

Cohen, A. (2005). Unsupervised gene/protein named entity normalization using automatically extracted dictionaries. In *Proceedings of the ACL-ISMB Workshop on Linking Biological Literature, Ontologies and Databases: Mining Biological Semantics*, (pp. 17–24).

Collier, N., Nobata, C., & Tsujii, J. (2000). Extracting the names of genes and gene products with a hidden Markov model. In *Proceedings of the Conference on Computational Linguistics* (*COLING-2000*), (pp. 201–207).

Collins, M., & Singer, Y. (1999). Unsupervised models for named entity classification. In *Proceedings of Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora* (*EMNLP-1999*).

Cucerzan, S. (2007). Large-scale named entity disambiguation based on Wikipedia data. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning* (*EMNLP-CoNLL-2007*), (pp. 708–716).

Culotta, A., & Sorensen, J. (2004). Dependency tree kernels for relation extraction. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics* (*ACL-2004*), (pp. 423–429).

Curran, J., & Clark, S. (2003). Language independent NER using a maximum entropy tagger. In *Proceedings of Conference on Computational Natural Language Learning* (*CoNLL-2003*), (pp. 164–167).

Doddington, G., Mitchell, A., Przybocki, M., Ramshaw, L., Strassel, S., & Weischedel, R. (2004). The automatic content extraction (ace) program: Tasks, data and evaluation. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation* (*LREC-2004*), (pp. 837–840).

Ekbal, A., & Bandyopadhyay, S. (2010). Improving the performance of a NER system by post-processing and voting. In *Structural, Syntactic, and Statistical Pattern Recognition, LNCS 5342* (pp. 831−841). Springer.

Ekbal, A., Sourjikova, E., Frank, A., & Ponzetto, S. P. (2010). Assessing the challenge of fine-grained named entity recognition and classification. In *Proceedings of the 2010 Named Entities Workshop*, (pp. 93–101).

Etzioni, O., Cafarella, M., Downey, D., Popescu, A.-M., Shaked, T., Soderland, S., & Weld, D. (2005). Unsupervised named-entity extraction from the web: an experimental study. *Artificial Intelligence*, *165*, 91–134. doi:10.1016/j.artint.2005.03.001

Fleischman, M., & Hovy, E. (2002). Fine grained classification of named entities. In *Proceedings of Conference on Computational Linguistics* (*COLING-2002*).

Florian, R., Ittycheriah, A., Jing, H., & Zhang, T. (2003). Named entity recognition through classifier combination. In *Proceedings of 7th Conference on Natural Language Learning at HLT-NAACL 2003*, (pp. 168–171).

Fort, K., Ehrmann, M., & Nazarenko, A. (2009). Towards a methodology for named entities annotation. In *Proceedings of the Third Linguistic Annotation Workshop*, (pp. 142–145).

Fukuda, K., Tsunoda, T., Tamura, A., & Takagi, T. (1998). Towards information extraction: identifying protein names from biological papers. In *Proceedings of the Pacific Symposium on Biocomputing* (*PSB-98*).

Funayama, H., Shibata, T., & Kurohashi, S. (2009). Bottom-up named entity recognition using a two-stage machine learning method. In *Proceedings of the 2009 Workshop on Multiword Expressions* (*ACL-IJCNLP-2009*), (pp. 55–62).

Ganti, V., Konig, A., & Vernica, R. (2008). Entity categorization over large document collections. In *Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (*KDD-2008*), (pp. 274–282).

Gliozzo, A., Giuliano, C., & Rinaldi, R. (2005). Instance filtering for entity recognition. *SIGKDD Explorations Newsletter*, *7*, 11–18. doi:10.1145/1089815.1089818

Han, X., & Zhao, J. (2009). Named entity disambiguation by leveraging Wikipedia semantic knowledge. In *Proceeding of the 18th ACM Conference on Information and Knowledge Management* (*CIKM-2009*), (pp. 215–224).

Humphreys, K., Gaizauskas, R., Azzam, S., Huyck, C., Mitchell, B., Cunningham, H., & Wilks, Y. (1998). Univ. of Sheffield: Description of the LaSIE-II system as used for MUC-7. In *Proceedings of the 7th Message Understanding Conference*.

Isozaki, H., & Kazawa, H. (2002). Efficient support vector classifiers for named entity recognition. In *Proceedings of the Conference on Computational Linguistics* (*COLING-2002*).

Jimeno, A., Jimenez-Ruiz, E., Lee, V., Gaudan, S., Berlanga, R., & Rebholz-Schuhmann, D. (2008). Assessment of disease named entity recognition on a corpus of annotated sentences. *BMC Bioinformatics*, *9*(3), 1–10. doi:10.1186/1471-2105-9-S3-S3

Kazama, J., Makino, T., Ohta, Y., & Tsujii, J. (2002). Tuning support vector machines for biomedical named entity recognition. In *Proceedings of ACL 2003 Workshop on Natural Language Processing in the Biomedical Domain*, (pp. 1–8).

Kazama, J., & Torisawa, K. (2007). Exploiting Wikipedia as external knowledge for named entity recognition. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning* (*EMNLP-CoNLL-2007*), (pp. 698–707).

Kim, J.-H., Kang, I.-H., & Choi, K.-S. (2002). Unsupervised named entity classification models and their ensembles. In *Proceedings of the Conference on Computational Linguistics* (*COLING-2002*).

Klein, D., Smarr, J., Nguyen, H., & Manning, C. (2003). Named entity recognition with character-level models. In *Proceedings of 7th Conference on Natural Language Learning* (*HLT-NAACL-2003*), (pp. 180–183).

Krebel, U. H.-G. (1999). Pairwise classification and support vector machines. In Scholkopf, B., Burges, C., & Smola, A. (Eds.), *Advances in kernel methods - Support vector learning. MIT Press, 1999*.

Krishnan, V., & Manning, C. (2006). An effective two-stage model for exploiting non-local dependencies in named entity recognition. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, (pp. 1121–1128).

Krishnarao, A., Gahlot, H., Srinet, A., & Kushwaha, D. (2009). A comparison of performance of sequential learning algorithms on the task of named entity recognition for Indian languages. In *Proceedings of the International Conference on Computational Science* (*ICCS 2009*), *LNCS 5544,* (pp. 123–132). Springer.

Krupka, G. R., & Hausman, K. (1998). IsoQuest Inc.: Description of the NetOwl^{TM} extractor system as used for MUC-7. In *Proceedings of the 7th Message Understanding Conference*.

Lafferty, J., McCallum, A., & Pereira, F. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the International Conference Machine Learning* (*ICML-2001*).

Lee, K.-J., Hwang, Y.-S., Kim, S., & Rim, H.-C. (2004). Biomedical named entity recognition using two-phase model based on SVMs. *Journal of Biomedical Informatics*, *37*(6), 393–428. doi:10.1016/j.jbi.2004.08.012

Li, W., & McCallum, A. (2003). Rapid development of Hindi named entity recognition using conditional random fields and feature induction. *ACM Transactions on Asian Language Information Processing*, *2*(3), 290–294. doi:10.1145/979872.979879

Liao, W., & Veeramachaneni, S. (2009). A simple semi-supervised algorithm for named entity recognition. In *Proceedings of the NAACL HLT-2009 Workshop on Semi-Supervised Learning for Natural Language Processing*, (pp. 58–65).

Lin, Y.-F., Tsai, T.-H., Chou, W.-C., Wu, K.-P., Sung, T.-Y., & Hsu, W.-L. (2004). A maximum entropy approach to biomedical named entity recognition. In *Proceedings of Workshop on Data Mining in Bioinformatics* (*BIOKDD04*), (pp. 56–61).

Liu, F., Zhao, J., Lv, B., Xu, B., & Yu, H. (2005). Product named entity recognition based on hierarchical hidden Markov model. In *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing*, (pp. 40–47).

Maynard, D., Tablan, V., Ursu, C., Cunningham, H., & Wilks, Y. (2001). Named entity recognition from diverse text types. In *Proceedings of the Conference on Recent Advances in Natural Language Processing* (*RANLP-2001*).

McCallum, A., & Li, W. (2003). Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *Proceedings of the Seventh Conference on Natural language learning at HLT-NAACL 2003 - Volume 4*, (pp. 188–191).

Meulder, F., & Daelemans, W. (2003). Memory-based named entity recognition using unannotated data. In *Proceedings of the Seventh Conference on Natural language learning at HLT-NAACL-2003 - Volume 4*, (pp. 208–211).

Mikheev, A. (1999). A knowledge-free method for capitalized word disambiguation. In *Proceedings of the Conference of Association for Computational Linguistics* (*ACL-1999*).

Mikheev, A., Moens, M., & Grover, C. (1999). Named entity recognition without gazetteers. In *Proceedings of 9th Conference of the European Chapter of the Association for Computational Linguistics* (*EACL-1999*), (pp. 1–8).

Miller, S., Crystal, M., Fox, H., Ramshaw, L., Schwartz, R., Stone, R., & Weischedel, R., & the Annotation Group. (1998). BBN: Description of the SIFT system as used for MUC-7. In *Proceedings of the 7th Message Understanding Conference*.

Montalvo, S., Martınez, R., Casillas, A., & Fresno, V. (2006). Multilingual document clustering: An heuristic approach based on cognate named entities. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, (pp. 1145–1152).

Nadeau, D., & Sekine, S. (2007). A survey of named entity recognition and classification. *Lingvisticae Investigationes*, *30*, 3–26. doi:10.1075/li.30.1.03nad

Nadeau, D., Turney, P., & Matwin, S. (2006). Unsupervised named-entity recognition: Generating gazetteers and resolving ambiguity. In *Proceedings of the 19th Canadian Conference on Artificial Intelligence*.

Negri, M., & Magnini, B. (2004). Using WordNet predicates for multilingual named entity recognition. In *Proceedings of Global WordNet Conference* (pp. 169–174). GWC.

Nguyen, H., & Cao, T. (2008). Named entity disambiguation: A hybrid statistical and rule-based incremental approach. In *Proceedings of the 3rd Asian Semantic Web Conference on The Semantic Web* (*ASWC-2008*), *LNCS 5367,* (pp. 420–433). Springer-Verlag.

Palmer, D., & Day, D. (2006). A statistical profile of the named entity task. In *Proceedings of ACL Conference for Applied Natural Language Processing* (*ANLP-1997*).

Park, K.-M., Kim, S.-H., Rim, H.-C., & Hwang, Y.-S. (2006). ME-based biomedical named entity recognition using lexical knowledge. *ACM Transactions on Asian Language Information Processing*, *5*(1), 4–21. doi:10.1145/1131348.1131350

Petasis, G., Karkaletsis, V., Grover, C., Hachey, B., Pazienza, M.-T., Vindigni, M., & Coch, J. (2004). Adaptive, multilingual named entity recognition in web pages. In *Proceedings of 2004 European Conference on Artificial Intelligence* (*ECAI-2004*), (pp. 1073–1074).

Rabiner, L. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, *77*(2), 257–286. doi:10.1109/5.18626

Ratinov, L., & Roth, D. (2009). Design challenges and misconceptions in named entity recognition. In *Proceedings of the 13th Conference on Computational Natural Language Learning* (*CoNLL-2009*), (pp. 147–155).

Raychaudhuri, S., Chang, J., Sutphin, P., & Altman, R. (2002). Associating genes with gene ontology codes using a maximum entropy analysis of biomedical literature. *Genome Research*. (n.d)., 37.

Sang, T. K., Erik, F., & de Meulder, F. (2003). Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Conference on Computational Natural Language Learning* (*coNLL-2003*), (pp. 142–147.

Sasano, M. (2003). Virtual examples for text classification with support vector machines. In *Proceedings of Conference on Empirical Methods in Natural Language Processing* (*EMNLP-2003*).

Sekine, S. (1998). Description of the Japanese NE system used for MET-2. In *Proceedings of the 7th Message Understanding Conference*.

Sekine, S., & Eriguchi, Y. (2000). Japanese named entity extraction evaluation - Analysis of results. In *Proceedings of the Conference on Computational Linguistics* (*COLING-2000*), (pp. 1106–1110).

Settles, B. (2004). Biomedical named entity recognition using conditional random fields and novel feature sets. In *Proceedings of Joint Workshop on Natural Language Processing in Biomedicine and its Applications* (*JNLPBA-2004*), (pp. 104–107).

Seymore, K., McCallum, A., & Rosenfeld, R. (1999). Learning hidden Markov structure for information extraction. In *Proceedings of AAAI'99 Workshop on Machine Learning for Information Extraction*.

Shadbolt, N., Hall, W., & Berners-Lee, T. (2006). The Semantic Web revisited. *IEEE Intelligent Systems*, (May/June): 96–101. doi:10.1109/MIS.2006.62

Shen, D., Zhang, J., Zhou, G., Su, J., & Tan, C.-L. (2003). Effective adaptation of a hidden Markov model-based named entity recognizer for biomedical domain. In *Proceedings of the Meeting of Association for Computational Linguistics* (*ACL-2003*).

Shinyama, Y., & Sekine, S. (2004). Named entity discovery using comparable news articles. In *Proceedings of the Conference on Computational Linguistics* (*COLING-2004*), (pp. 848–853).

Song, Y., Yi, E., Kim, E., & Lee, G. (2005). POSBIOTM-NER: A machine learning approach for bio-named entity recognition. *Bioinformatics (Oxford, England)*, *21*(11), 2784–2796.

Sun, J., Gao, J., Zhang, L., Zhou, M., & Huang, C. (2002). Chinese named entity identification using class-based language model. In *Proceedings of the Conference on Computational Linguistics* (*COLING-2002*).

Takeuchi, K., & Collier, N. (2002). Use of support vector machines in extended named entity recognition. In *Proceedings of 2002 Conference on Natural Language Learning* (*coNLL-2002*), (pp. 119–125).

Talukdar, P., Brants, T., Liberman, M., & Pereira, F. (2006). A context pattern induction method for named entity extraction. In *Proceedings of the 10th Conference on Computational Natural Language Learning* (*CoNLL-2006*), (pp. 141–148).

Tanev, H., & Magnini, B. (2008). Weakly supervised approaches for ontology population. In *Proceeding of the 2008 Conference on Ontology Learning and Population: Bridging the Gap between Text and Knowledge*, (pp. 129–143).

Thao, P., Tri, T., Dien, D., & Collier, N. (2007). Named entity recognition in Vietnamese using classifier voting. *ACM Transactions on Asian Language Information Processing*, *6*(4), 1–18. doi:10.1145/1316457.1316460

Tsai, R., Wu, S., Chou, W., Lin, Y., He, D., & Hsiang, J. (2006). Various criteria in the evaluation of biomedical named entity recognition. *BMC Bioinformatics*, *7*(92).

Vapnik, V. (1998). *Statistical learning theory*. Wiley Interscience.

Wang, Y., & Patrick, J. (2009). Cascading classifiers for named entity recognition in clinical notes. In *Proceedings of the Workshop on Biomedical Information Extraction*, (pp. 42–49).

Watanabe, Y., Asahara, M., & Matsumoto, Y. (2007). A graph-based approach to named entity categorization in Wikipedia using conditional random fields. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning* (*EMNLP-CoNLL-2007*), (pp. 649–657).

Yamamoto, K., Kudo, T., Konagaya, A., & Matusmoto, Y. (2003). Protein name tagging for biomedical annotation in text. In *Proceedings of ACL 2003 Workshop on Natural Language Processing in Biomedicine*.

Yi, E., Lee, G. G., Song, Y., & Park, S.-J. (2004). SVM-based biological named entity recognition using minimum edit-distance feature boosted by virtual examples. In *Proceedings of International Joint Conference on Natural Language Processing* (*IJCNLP-2004*), *LNCS 3248*, (pp. 807–814).

Zhang, J., Shen, D., Zhou, G., Su, J., & Tan, C.-L. (2002). Enhancing HMM-based biomedical named entity recognition by studying special phenomena. *Journal of Biomedical Informatics*, *12*(6), 411–422.

Zhao, S. (2004). Named entity recognition in biomedical texts using an HMM model. In *Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and Its Applications*, (pp. 84–87).

Zhao, S., & Grishman, R. (2005). Extracting relations with integrated information using kernel methods. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics* (*ACL-2005*), (pp. 419–426).

Zhou, G., & Su, J. (2002). Named entity recognition using an HMM-based chunk tagger. In *Proceedings of 40th Meeting of Association of Computational Linguistics* (*ACL-2002*), (pp. 473–480).