EXTRAÇÃO DE DADOS DA WEB POR ROTULAMENTO DE SEQUÊNCIAS

JOÃO MATEUS DE FREITAS VENEROSO

EXTRAÇÃO DE DADOS DA WEB POR ROTULAMENTO DE SEQUÊNCIAS

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação do Instituto de Ciências Exatas da Universidade Federal de Minas Gerais como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação.

Orientador: Berthier Ribeiro-Neto de Araújo

Belo Horizonte
Julho de 2019

JOÃO MATEUS DE FREITAS VENEROSO

WEB DATA EXTRACTION THROUGH NAMED ENTITY LABELING

Dissertation presented to the Graduate Program in Computer Science of the Federal University of Minas Gerais in partial fulfillment of the requirements for the degree of Master in Computer Science.

Advisor: Berthier Ribeiro-Neto de Araújo

Belo Horizonte

July 2019

de Freitas Veneroso, João Mateus

D1234p Web data extraction through named entity labeling / João Mateus de Freitas Veneroso. — Belo Horizonte, 2019

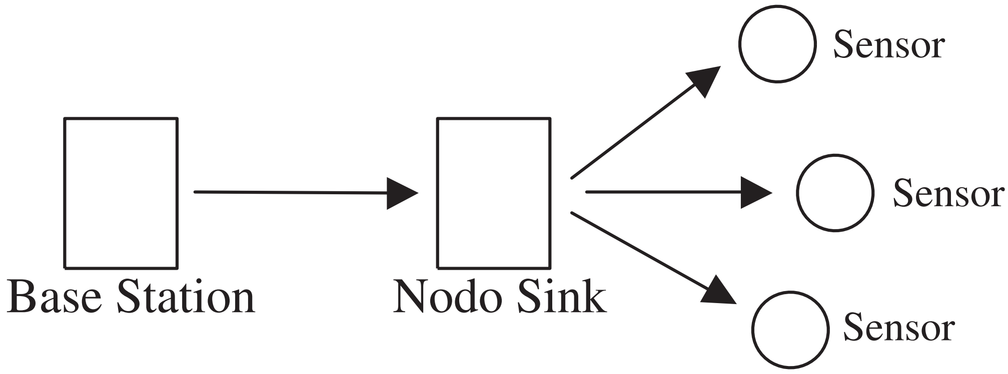
xxiv, 40 f. : il. ; 29cm

Dissertação (mestrado) — Federal University of Minas Gerais

Orientador: Berthier Ribeiro-Neto de Araújo

1. Visão Computacional. 2. Redes. 3. Sabotagens. I. Título.

CDU 519.6*82.10


Sensor
Sensor
Base Station
Nodo Sink
Sensor

Dedicuum cest laborae a quelquis personatum que ajudorat a facirelo.

Acknowledgments

Agradeço aos prótons por serem tão positivos, aos nêutrons pela sua neutralidade e aos elétrons pela sua carga.

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Sed ut perspiciatis unde omnis iste natus error sit voluptatem accusantium doloremque laudantium, totam rem aperiam, eaque ipsa quae ab illo inventore veritatis et quasi architecto beatae vitae dicta sunt explicabo. Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed quia consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt. Neque porro quisquam est, qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit, sed quia non numquam eius modi tempora incidunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim ad minima veniam, quis nostrum exercitationem ullam corporis suscipit laboriosam, nisi ut aliquid ex ea commodi consequatur? Quis autem vel eum iure reprehenderit qui in ea voluptate velit esse quam nihil molestiae consequatur, vel illum qui dolorem eum fugiat quo voluptas nulla pariatur?

At vero eos et accusamus et iusto odio dignissimos ducimus qui blanditiis praesentium voluptatum deleniti atque corrupti quos dolores et quas molestias excepturi sint occaecati cupiditate non provident, similique sunt in culpa qui officia deserunt mollitia animi, id est laborum et dolorum fuga. Et harum quidem rerum facilis est et expedita distinctio. Nam libero tempore, cum soluta nobis est eligendi optio cumque nihil impedit quo minus id quod maxime placeat facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet ut et voluptates repudiandae sint et molestiae non recusandae. Itaque earum rerum hic tenetur a sapiente delectus, ut aut

reiciendis voluptatibus maiores alias consequatur aut perferendis doloribus asperiores repellat.

"Truth and lie are opposite things."
(Unknown)

Resumo

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Sed ut perspiciatis unde omnis iste natus error sit voluptatem accusantium doloremque laudantium, totam rem aperiam, eaque ipsa quae ab illo inventore veritatis et quasi architecto beatae vitae dicta sunt explicabo. Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed quia consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt. Neque porro quisquam est, qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit, sed quia non numquam eius modi tempora incidunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim ad minima veniam, quis nostrum exercitationem ullam corporis suscipit laboriosam, nisi ut aliquid ex ea commodi consequatur? Quis autem vel eum iure reprehenderit qui in ea voluptate velit esse quam nihil molestiae consequatur, vel illum qui dolorem eum fugiat quo voluptas nulla pariatur?

Palavras-chave: Visão Computacional, Redes, Sabotagens.

Abstract

Web data extraction methods often rely on hand-coded rules to identify and extract data from webpages. These methods are usually suited for extracting information from pages within the same website, however they perform poorly on extraction tasks across different websites. Alternatively, statistical and machine-learning-based sequence labeling methods provide a more flexible approach to Web data extraction. Many times, HTML pages are very different from plain text, because sentences are too short to provide adequate context for conventional Named Entity Recognition methods to work properly. Also, the HTML structure may encode information that is not replicated in the text. Nonetheless, these limitations can be overcome by adequate feature engineering, the use of pretrained word embeddings and neural character representations. In this article, we evaluate the performance of different methods of named entity recognition on the task of Web data extraction. In particular, we introduce a novel dataset[1] consisting of faculty listings from university webpages across the world in multiple languages and test the NER models on the task of extracting researcher names from these listings. We found that a neural network architecture that combines a bidirectional LSTM with a Conditional Random Fields output layer and LSTM-based character representations outperforms other methods on the researcher name extraction task, achieving an F1-score of 0.8867 with no feature engineering. With the addition of hand crafted features, the F1-score can be slightly improved to 0.8995.

Palavras-chave: Named entity recognition, information extraction, web data extraction.

[1]The dataset and all models discussed in this article are available in: https://github.com/jmfveneroso/ner-on-html.

List of Figures

List of Tables

Contents

Chapter 1

Introduction

HTML documents most often lie in between the structured/unstructured data paradigm. DOM hierarchy, element disposition, CSS classes, and other features related to the document structure and indirectly associated with the data itself can be valuable information on the task of identifying entities. Yet, we cannot expect these features to be completely constrained by an underlying pattern. Organization patterns tend to follow some guidelines but they are in no way subject to strict rules. That is why classical web data extraction systems such as automatic wrapper generators Kushmerick [2000]; Hsu and Dung [1998]; Muslea et al. [1999] do not translate very well across different websites.

Most existing Web data extraction methods are tailored to extract data from a single webpage, producing different compromises between efficacy and degree of human supervision. Some unsupervised approaches proposed to tackle the problem of data extraction for whole application domains Zhu et al. [2005, 2006]; Abdessalem et al. [2010]; Furche et al. [2012b,a]. Usually, unsupervised WDE methods work in two stages. In the record segmentation stage, WDE systems seek to cluster visually and structurally similar webpage regions and identify repeating data records with heuristics and hand-coded rules. In the attribute labeling stage, WDE systems seek to identify the correct attributes on data records, many times resorting to regular expressions or gazetteer matching strategies. The outcome of each of these stages can aid one another. The inner patterns of data records can help in identifying attributes of other data records. Also, by properly identifying data record attributes, it becomes easier to determine boundaries and perform record segmentation.

While unsupervised approaches can be sometimes adequate to extract information from webpages with similar templates, they usually fail on cross-website data extraction tasks. Also, more sophisticated approaches may be needed when we are not dealing with easily distinguishable attributes such as prices and dates. In this regard, machine-learning-based sequence labeling methods can provide a more flexible approach that works regardless of website structure.

In recent years, we saw an amazing progress in the field of Natural Language Processing, particularly with the introduction of deep recurrent neural network architectures for sequence labeling. However, despite Web data extraction being a closely related field, there is a lack of extraction tools that make use of these recent advancements. The attribute labeling stage in Web data extraction systems is essentially a Named Entity Recognition (NER) problem, the problem of detecting named entities in the text and classifying them into predetermined categories such as person names, locations, dates or organizations.

In many cases, such as when we are extracting researcher names from faculty listings, detecting named entities is sufficient to solve the data extraction task. However, when we are dealing with multi-attribute data records or complex relationships, we may need to perform additional steps. Nevertheless, even in the latter cases, flexible NER methods are still desirable because, while many WDE algorithms can effectively exploit the semi-structured nature of Web documents, too much reliance on structural webpage features often produce poor generalizations on cross website extraction. Also, data records may enclose plain text with relevant named entities.

In this article, we investigate methods of named entity recognition for Web data extraction. Recently proposed neural architectures have achieved exciting results on the NER task in plain text while requiring almost no feature engineering or access to gazetteers Huang et al. [2015]; Lample et al. [2016]; Ma and Hovy [2016]. NER on HTML poses a slightly different type of challenge. Named entities may occur inside tables, lists, or other types of visual elements that provide little to no textual information that could give hints about the semantic category of a word. Still, even with this limitation, we show that it is possible to obtain very good results with a relatively small training set.

By reliably detecting named entities on HTML, we can improve the performance of existing WDE approaches or even construct an end-to-end neural network architecture to solve domain wide data extraction with considerable flexibility. To test different NER approaches to Web data extraction, we explored the task of researcher name extraction from university faculty listings across the world, introducing a novel NER dataset.

Reliable researcher affiliation information is often missing from public researcher databases (especially in departments other than Computer Science). Also, the display of faculty information varies significantly between different university websites, so this task can provide a good measure of the expected performance and data need of NER methods on other Web data extraction tasks.

Chapter 2

Problem Definition

Web data extraction is a task that is constituted of many overlapping problems that demand integrated and sophisticated solutions. Most often, information extraction tools that work exceptionally well on a given task end up performing poorly when given even a slightly different task. Traditional methods of web data extraction were mainly concerned with the extraction of entities described by a simple ontology from web pages generated by the same source (e.g. collecting house prices and addresses from real estate portals). The extraction of complex entities from a plethora of sources presents a different class of problems. The former case could be solved by rigid tools with hard coded rules that many times yielded close to perfect results. The latter case, however, demands far more flexible approaches since the extraction tools are required to deal with a greater variety of page arrangements and also with entities that are ambiguously defined. In many aspects, the flexible extraction of entities from web pages is more similar to the tasks of named entity recognition and relationship extraction from plain text than to the tasks that concerned the early information extraction methods.

Recent advancements in sequence models for natural language led to important breakthroughs in applications such as speech recognition, machine translation, sentence segmentation, and sequence labeling (which concerns us more directly). In fact, if we treat all sentences in a web page as sequences extracted from an underlying presentation graph (i.e. the DOM tree), the problem of information extraction can be solved through the consecutive application of three NLP tasks: sentence segmentation, named entity recognition, and relationship extraction. First, we need to segment the relevant grouping structures (e.g. sentences, rows in table, items in a list). Then, we must identify relevant named entities (e.g. person names, companies, locations). And finally, we have to discover the relationships between those named entities (e.g. person

X works in company Y). The work flow is the same for plain text and web pages, but there are important differences chiefly related to the structure of the data or the lack of it.

In this dissertation, we investigate the best methods for sequence labeling on HTML. To evaluate these methods, we assess their performance on the task of researcher name extraction from faculty directories of universities across the world. We will be mainly concerned with the task of named entity recognition but a brief discussion of the challenges involved in the task of sentence segmentation for HTML will be provided in Chapter 3. The task of relationship extraction, however, is not relevant to the problem of researcher name extraction and will not be discussed.

The remainder of this chapter will describe in more depth the subject of this dissertation. Section 2.1 discusses the task of Information Extraction in the Web. Section 2.2 discusses the importance of Named Entity Recognition for Web Data Extraction. Finally, Section 2.3 describes the specific problem of Researcher Name Extraction.

2.1 Information Extraction in the Web

Information Extraction consists of mapping unstructured or poorly structured data to a semantically well defined structure. "(It) is the process of filling the fields and records of a database from unstructured or loosely formatted text" McCallum [2005]. Usually, the input consists of a corpus containing useful entities that are scattered in the text and the information extraction system is responsible for finding these entities and organizing them according to a rigid hierarchy such as the one defined by the schema of a relational database. It must be stated that it is somewhat misleading to refer to plain text as unstructured data, since prose has a loosely defined structure that ultimately renders it comprehensible. However, in the context of *Information Extraction* we refer to unstructured data in contraposition to tabular data, which are in many cases easier to work with than plain text.

Information Extraction is a multifaceted problem that spans communities of researchers in the fields of *Text Mining*, Information Retrieval, and Natural Language Processing. *Text mining* is the search of patterns in unstructured text that may involve document clustering, document summarization, *Information Extraction*, and other subtasks. *Information Retrieval* is tipically concerned with the parsing, indexing and retrieval of documents and *Information Extraction* can help giving a more precise answer to the user's information needs. Finally, *Natural Language Processing* is a field of Computer Science concerned with how computers process and understand natural language of which two subtasks, namely Named Entity recognition and Relation Extraction, are of special importance for *Information Extraction*.

A popular application of *Information Extraction* is the identification of entities such as people, organizations, or events from news sources and the determination of their relations. For example, one could be interested in determining who is the CEO of a company that was mentioned in the news or which politicians support a bill that is being considered by the Congress. Another interesting news-related application of *Information Extraction* is tracking disease outbreaks through the extraction of disease names and locations from news sources and determining their relation to outline the geographical area affected by an epidemy (cite example). Other *Information Extraction* systems proposed to track disease outbreaks through the analysis of tweets (cite example), which presents a remarkably different challenge since the context of tweets is very limited and the presence of abbreviations and typos is more widespread than in the case of highly curated content such as news.

The field of bio-informatics has also found important applications for *Information Extraction*. The first BioCreAtIvE challenge dealt with extraction of gene or protein names from text, and their mapping into standardized gene identifiers for three model organism databases (fly, mouse, yeast). Hirschman et al. [2005]. In the BioCreative/OHNLP Challenge 2018 Rastegar-Mojarad et al. [2018], researchers were required to investigate methods of family history information extraction. Family history is a critical piece of information in the decision process for diagnosis and treatment of diseases, however the main sources of data are unstructured electronic health records. The task is divided in two subtasks: 1) Entity recognition (family members and disease names); and 2) Relation Extraction (family members and corresponding observations).

Another scientific application for *Information Extraction* methods is the the extraction of information from research papers to populate citation databases and bibliography search engines such as Citeseer Lawrence et al. [1999], DBLP [1], Semantic Scholar [2], and Google Scholar [3]. The vast amount of scientific knowledge produced daily demands automatic methods for extracting authors, titles, affiliations, references, venues and the year of publication from research papers and online resources such as university websites, individual researcher home pages and conference websites. This application is especially important to the evaluation and bibliometrics research communities, that are concerned with the measure of academic impact and researcher productivity through the usage of quantitative indices of academic impact such as the

[1] http://dblp.uni-trier.de/
[2] https://www.semanticscholar.org
[3] https://scholar.google.com/

H-index Hirsch [2005] and P-score Ribas et al. [2015].

Web Data Extraction works a little different from *Information Extraction* in plain text because HTML documents frequently lie in between the structured/unstructured data paradigm. Relevant entities may occur inside tables, lists, or other types of visual elements that provide little to no contextual information that could give hints about their category. Web pages have a two dimensional tabular structure that is usually more similar to a spreadsheet than to text found in news corpora. For this reason, features extracted from the DOM hierarchy such as element disposition, CSS classes, and nesting structure can provide valuable information in identifying entities and extracting their attributes. Yet, we cannot expect these features to be completely constrained by an underlying pattern. Organization patterns tend to follow some guidelines but they are in no way subject to strict rules.

Most existing Web data extraction methods are tailored to extract data from a single webpage, producing different compromises between efficacy and degree of human supervision. Usually, these methods work in two steps. In the record segmentation step, we seek to cluster visually and structurally similar webpage regions and identify repeating data records. In the attribute labeling stage, we seek to identify the correct attributes for each data record, maybe resorting to regular expressions or simple dictionary matching strategies depending on the task at hand. The outcome of each step can aid one another. The inner patterns of data records can help identifying attributes of other data records. Also, by properly identifying attributes, it becomes easier to determine boundaries and perform record segmentation correctly. Figure **??** shows how a hypothetical **??** system that aims to collect book titles and prices from Amazon could perform record segmentation and attribute labeling.

The task of extracting product names and prices from online catalogs can likely be solved by a rather inflexible system that operates mainly with hard coded rules and simple regular expressions, especially if we only consider pages with a very similar template (e.g. Amazon product listings). Classical wrapper generators Kushmerick [2000]; Hsu and Dung [1998]; Muslea et al. [1999] are well suited for this type of task. However, when we need to identify more complex and ambiguous entities such as researcher names in many different contexts such as faculty websites, we might be better off with a more flexible approach. The task of Named Entity Recognition aims to identify named entities (e.g. people, organizations, etc.) in plain text, but we will see in the next chapters that the sequence models that work well in plain text can also be employed succesfully in web extraction tasks, sometimes with a few alterations.

This section gives a brief introduction to *Information Extraction*, but there are many applications that were not discussed here. A more detailed view of the field is given in Sarawagi Sarawagi [2008].

2.2 Named Entity Recognition

Named Entity Recognition is a task in Natural Language Processing that aims to identify *Named Entities* in a text. The Named Entity Recognition task first appeared as a subtask of *Information Extraction* in the context of the Message Understanding Conference (MUC) promoted by the american Naval Ocean Systems Center. In MUC-3, a precursor task involved extracting information on terrorist incidents (incident type, date, location, perpetrator, target, instrument, outcome, etc.) from messages disseminated by the Foreign Broadcast Information Service of the U.S. Government Sundheim [1991]. In MUC-6 Grishman and Sundheim [1996] the "named entity" task was created with the goal of identifying the names of people, organizations, and geographic locations from articles of the Wall Street Journal. In MUC-7, the task was expanded to handle multilingual evaluation and "Named Entities (NE) were defined as proper names and quantities of interest. Person, organization, and location names were marked as well as dates, times, percentages, and monetary amounts" Chinchor [1998]. The shared-task at the Conference on Computational Natural Language Learning in 2003 Tjong Kim Sang and De Meulder [2003a] concerned language-independent named entity recognition and it is especially important because it established an enduring data format for NER and it introduced a dataset of articles extracted from news sources that is to this day still employed to evaluate the quality of NER systems.

The task of *Named Entity Recognition* is essentialy a sequence labeling task. That is, given a sequence of tokens we want to attribute labels to each token classifying them in one of a number of predefined classes. Figure **??** describes the process of atributing Named Entity labels to a sentence according to the CONLL format. There are cases where a token may belong to more than one class because of nested named entities (e.g. in "CoNLL 2003", 2003 is both part of a conference name and a date). This complexity is of importance to some applications **?** but it will not be discussed here because nested named entities are absent from our concrete problem of researcher name extraction and most sequence models can be adapted to handle this special case with a few modifications.

A simple approach to *Named Entity Recognition* can be devised through the use of regular expressions, which are search patterns that describe a regular language and can be easily implemented in most programming languages. These programs can be perfectly suited to extract regular entities such as dates and prices with almost perfect accuracy. The extraction of other types of entities that belong to a limited set such as the names of states in a country can be accomplished through dictionary matching. Other types of entities such as the ones investigated in the CoNLL-2003 challenge require more sophisticated methods of sequence labeling.

Some classical statistical methods of *Sequence Labeling* that are able to handle the labeling of complex entities with decent accuracy are Hidden Markov Models, Linear Chain Conditional Random Fields and Maximum Entropy Markov Models. The first two will be explored in further detail in Chapter **??**, and the latter is essentialy a more restrictive discriminative model in comparison with Condition Random Fields. These statistical approaches have had remarkable resiliency in *Sequence Labeling* tasks and still provide a good first approximation to a solution because of their simplicity, speed and accuracy.

As is the case with most NLP tasks, the deep learning revolution has brought huge advancements to *Named Entity Recognition.* Essentialy all the current best scoring models at the Conll-2003 task employ some form of deep neural network, and most often Long Short Term Memory recurrent neural networks with a range of differing characteristics. Combined with pretrained word embeddings these models provide a powerful method of sequence labeling that requires practically no feature engineering or dictionary. However, these models have become quite complex, constrasting with earlier approaches such as HMMs, in the fact that the training time of such networks has increased by many orders of magnitude, and if we take into consideration the training of word embeddings such as Word2Vec's skip gram model in a billion token corpus it is no exaggeration to say that deep learning models increase the training time by a thousandfold when compared to earlier approaches. Additionaly, most deep learning models require expensive hardware in the form of GPUs or more recently TPUs (Google's Tensor Processing Units) to become practical options. That is to say that despite deep learning representing the inquivocal state-of-the-art, classical statistical models still are of significance and one might also risk saying competitive as we will see in later chapters.

Another disadvantage of deep learning in contrast with other models such a HMMs is the lack of interpretability of trained models. The complexity of a neural networks renders impractical the effort of understanding why the model takes the decisions it takes. We also have to discuss the problem of causality

2.3 Researcher Name Extraction

Chapter 3

Dataset

We built a novel dataset to evaluate the performance of NER models on the Web data extraction task. The task consists of finding researcher names in faculty listings from university webpages across the world. This would be a necessary step when linking researcher profiles from university websites to their entries in public databases such as DBLP [1]. Unlike many information extraction datasets, each webpage in the dataset comes from a different website, and therefore has a different format, what makes many information extraction approaches impractical. The idea is to explore systems that are general enough to allow efficient entity extraction from different sources while requiring no supervision between different websites.

This task is similar to labeling authors in comments or articles collected from different publishing platforms. Another similar task is NER on tweets, because of the character limitation that is comparable to what we find in HTML text.

We collected 145 computer science faculty pages from 42 different countries in multiple languages, although the English version was preferred when it was available. We gathered faculty webpages randomly in proportion to the number of universities in each country[2]. Each HTML page was preprocessed and converted to the CoNLL 2003 data format. That is, one word per line with empty lines representing sentence boundaries. Sentence boundaries were determined by line break HTML tags (div, p, table, li, br, etc.) in contrast to inline tags (span, em, a, td, etc.). Sentences that were more than fifty tokens long were also split according to the punctuation.

A proper HTML segmenter poses many challenges by itself. We wanted to evaluate models without relying on any sophisticated data record segmentation system. In many cases, entity annotation may precede the segmentation phase on Web data

[1] http://dblp.uni-trier.de/

[2] A detailed list of universities can be found in https://univ.cc/world.php

extraction methods, so annotators that are able to work with raw HTML data allow for more flexible systems.

Finally, all tokens were tagged using the IOB scheme put forward by Ramshaw and Marcus Ramshaw and Marcus [1999].

3.1 Data

Data file	Documents	Sentences	Tokens	Names
Training	85	24728	110269	5822
Validation	30	8743	36757	1788
Test	30	10399	44795	2708

Table 3.1. Number of HTML pages, sentences and tokens in each data file

The dataset was divided in a training, validation and test set. Table 3.1 contains a description of the data files. The validation set was used in the early stopping validation strategy for the neural networks and CRF training, while the model performance was evaluated by comparing results in the test set.

3.2 Features

Thirteen categorical features were associated with each token in the dataset. They are presented in Table 3.2.

The unaccented lowercase token was used as the key for the GloVe-100 embedding lookup. A gazetteer was constructed from a researcher list extracted from DBLP with 1,595,771 names. Table 3.3 shows the precision, recall and F1 score obtained with an exact gazetteer matching strategy in each data file as a baseline. Feature 2 represents an exact match of a sequence of tokens to any of the 1,595,771 names, and feature 3 represents a partial match. Feature 4 is the rounded logarithm of the frequency of a token in the gazetteer, and feature 5 is the rounded logarithm of the frequency of a token in a word corpus obtained through a random crawl on university websites. Features 6 to 11 represent a simple regular expression match to an email, number, honorific, URL, capitalization or punctuation sign.

Feature 12 represents the HTML enclosing tag and its parent concatenated. Feature 13 represents all CSS classes concatenated. These features are not very useful in a

Feature	Description
1	Unaccented lowercase token
2	Exact gazetteer match
3	Partial gazetteer match
4	Log name gazetteer count
5	Log word gazetteer count
6	Email
7	Number
8	Honorific (Mr., Mrs., Dr., etc.)
9	URL
10	Is capitalized
11	Is a punctuation sign
12	HTML tag + parent
13	CSS class

Table 3.2. Features used in the NER on HTML dataset

Data file	Precision	Recall	F1	Correct names
Training	0.7316	0.2303	0.3504	1341 of 5822
Validation	0.8474	0.2858	0.4274	511 of 1788
Test	0.8717	0.3287	0.4773	890 of 2708

Table 3.3. Gazetteer coverage in each data file

general sense, because every HTML document has a different format, so only because a named entity occurs inside a given HTML tag in a document we cannot say it is more likely to be the case in other documents. However, these features can be useful for the HMM self-training strategy described in section 5.1.1.

3.3 Web Data Extraction

Web data extraction (WDE) is the task of automatically extracting structured information from unstructured or semi-structured Web documents. The input usually consists of Web documents containing a number of predetermined entities organized in a similar manner. The web data extraction task consists of identifying these entities and organizing them according to a template.

Chapter 4

Related Work

In the last 20 years, the astonishing growth of public information in the Web has led to the development of a number of different approaches to the problem of Web data extraction. Traditionally, the task was solved by designing special purpose programs called wrappers to recognize relevant data and store records in a structured format. These early tools varied wildly relative to their degree of automation.

It was readily perceived that manual wrapper generation was a rather tedious and error prone process, unsuited for large scale operations. Wrappers tend to break frequently because they rely heavily on webpage features that can change often. So, in the late nineties, several authors advocated for wrapper induction, a technique that consists of automatically constructing wrappers from a small set of examples by identifying delimiters or context tokens that single out the desired attributes. Some remarkable wrapper induction methods are WIEN Kushmerick [2000], Soft Mealy Hsu and Dung [1998] and STALKER Muslea et al. [1999].

Despite being better than constructing wrappers manually, wrapper induction methods still suffered from a lack of expressive power and flexibility. These methods had trouble handling records with missing attributes or unusual structures because patterns could only be identified if they happened at least once in the examples.

Other approaches such as NoDoSE Adelberg [1998] and Debye Laender et al. [2002b] brought greater flexibility to wrapper induction methods by requiring a greater level of human interaction through graphical user interfaces. Web data extraction techniques often require some sort of assistance from human experts to boost accuracy. One of the main challenges in the field lies in determining an adequate trade-off between the degree of automation and the precision and recall of the data extraction tool.

To automate the task of Web data extraction completely some approaches, such as Road Runner Crescenzi et al. [2001], removed entirely the need for data examples.

Road Runner parses documents belonging to a same class (e.g. books on Amazon) and generates wrappers based on their similarities and differences, yielding comparable results to those obtained by wrapper induction methods. However, like previous approaches, it was unsuited for cross site extraction tasks because the learned rules were not general enough.

NLP based approaches aimed at extracting more general rules that could possibly be employed over multiple websites. RAPIER Califf and Mooney [1999] is a method of rule extraction that uses information such as part-of-speech tags and semantic classes from a lexicon to derive patterns from a set of training examples. This approach is more flexible than the wrapper induction methods, however it achieves much lower rates of recall and precision.

In 2002, a survey by Laender et al. Laender et al. [2002a] made a thorough classification of the early approaches with a taxonomy based on their main technology, being them: languages for wrapper development, HTML-aware tools, NLP-based tools, Wrapper Induction Tools, Modeling-based tools and Ontology-based tools. Some noteworthy examples from this era are:

- TSIMMIS Hammer et al. [1997] and WebOQL Arocena and Mendelzon [1999], which are special purpose languages for building wrappers.
- Road Runner Crescenzi et al. [2001], XWRAP Liu et al. [2000] and W4F Sahuguet and Azavant [1999], which are HTML-aware tools that infer meaningful patterns from the HTML structure.
- RAPIER Califf and Mooney [1999], SRV Freitag [1998], WHISK Soderland [1999], which are NLP-based tools.
- WIEN Kushmerick [2000], Soft Mealy Hsu and Dung [1998] and STALKER Muslea et al. [1999] which are wrapper induction methods.
- NoDoSE Adelberg [1998] and Debye Laender et al. [2002b], which are semi supervised modeling based tools that require some interaction with the user by means of a graphical user interface.

In 2006, Chang et al. Chang et al. [2006] complemented the previous surveys with semi-supervised technologies such as Thresher Hogue and Karger [2005], IEPAD Chang et al. [2001] and OLERA Chang and Kuo [2004]. They differed from supervised and unsupervised methods because they either needed only a rough description of data from users for extraction rule generation or some level of post processing that needed user attention. The survey also mentioned newer unsupervised methods such as DeLa Wang and Lochovsky [2003], Exalg Arasu et al. [2003] and Depta Zhai and Liu [2005].

Most of the early information extraction systems were rule-based with either manual rule description or automatic rule learning from examples, thus they suffered from a lack of flexibility when dealing with noisy and unstructured data. Huge progress in the field of statistical learning led to the development of statistical models that tried to solve this problem.

In 2008, Sarawagi Sarawagi [2008] produced a survey that classified wrappers into rule-based methods, statistical methods and hybrid models, bringing together the fields of named entity recognition, relationship extraction and information extraction. The rule based methods encompass most of the previous models. The statistical methods convert the extraction task into a token labeling task, identifying the target entities through the assignment of labels as in a typical Named Entity Recognition task. Traditionally, Hidden Markov Models Leek [1997]; Freitag and Mccallum [1999], Linear Chain Conditional Random Fields Lafferty [2001], and Maximum Entropy Taggers McCallum et al. [2000] have been the usual choice for linear sequence tagging models. More recently, with the advancement of Natural Language Processing and Deep Learning, neural models outperformed previous NER methods for plain text. Huang et. al. Huang et al. [2015] introduced the bidirectional Long Short-Term Memory (LSTM) model with a Conditional Random Field (CRF) output layer for NER. Ma and Hovy Ma and Hovy [2016] incorporated Convolutional Neural Network based character representations on top of the architecture. And Lample et. al. Lample et al. [2016] introduced LSTM based character representations.

Surveys by Ferrara et al. Ferrara et al. [2014], Schulz et al. Schulz et al. [2016] and Varlamov et al. Varlamov and Turdakov [2016] updated the previous surveys on information extraction methods with some interesting innovations. Some examples are: the Visual Box Model Krüpl et al. [2005], a data extraction system that produces a visualization of the webpage to exploit visual cues to identify data presented in a tabular form; automatic wrapper adaptation Ferrara and Baumgartner [2011], a technique that tries to reduce the cost of wrapper maintenance by measuring the similarity of HTML trees and adapting wrappers to the new page structure; AutoRM Shi et al. [2015], a method to mine records from a single webpage by identifying similar data regions through DOM tree analysis; Knowledge Vault Dong et al. [2014], a method that combines different extraction approaches to feed a probabilistic knowledge base.

Most data extraction systems focus on extracting information from single websites and are therefore unsuited for cross website extraction tasks. Even unsupervised approaches that are domain independent, such as RoadRunner Crescenzi et al. [2001]

and EXALG Arasu et al. [2003] only work well for extracting data from pages generated from a same template.

A statistical approach to unsupervised domain independent Web data extraction was described by Zhu et al Zhu et al. [2005]. The 2D CRF model takes a webpage segmented into data blocks and employs a two dimensional conditional random field model to perform attribute labeling. The model was further improved Zhu et al. [2006] to model record segmentation and attribute labeling as a joint task. Some of the limitations of early unsupervised methods were also tackled by ObjectRunner Abdessalem et al. [2010] and AMBER Furche et al. [2012b]. These methods work by annotating webpages automatically with regular expressions, gazetteers and knowledge bases. They can rectify low quality annotations and even improve the annotators by exploring regular structures in the DOM during the record segmentation phase.

Web data extraction methods have undoubtedly improved extraordinarily, but as pointed by Schulz et al. Schulz et al. [2016], it is difficult to compare the results achieved by competing tools, and many seem to rely excessively on heuristic methods. In that regard, the recent advancements in sequence taggers may provide more robust and flexible extraction tools.

Chapter 5

Named Entity Recognition Models

Many Web data extraction systems rely on hand crafted rules or gazetteers to perform attribute annotation. Machine learning approaches to NER can improve annotations of more complex entities and even perform entity detection without any feature engineering. We explored many methods of Named Entity Recognition in the context of a Web data extraction task. First, we discuss two traditional approaches: Hidden Markov Models, and Linear Chain Conditional Random Fields. Then, we discuss neural network architectures.

5.1 Hidden Markov Models

A Markov Model is a stochastic model that computes the most probable sequence of states given a limited set of observable states $S = \{s_1, s_2, ..., s_n\}$. The Hidden Markov Model (HMM) differs from the Markov Model in that it does not observe the states directly, but rather a probabilistic function of those states. For example in NER, the words are observed, however the Named Entity labels associated with these words are not. Formally, we want to compute the most probable sequence of labels $Y = \{y_1, y_2, ..., y_n\}$ for a sequence of observed tokens $X = \{x_1, x_2, ..., x_n\}$.

$$Y^* = \arg\max_Y P(Y|X) \tag{5.1}$$

With Bayes theorem, we can write $P(Y|X)$ as:

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)} \tag{5.2}$$

Since $P(X)$ is the same for all label sequences Y, we can simply maximize the probability $P(X|Y)P(Y)$.

A HMM makes two assumptions. First, the probability of being in a given state depends only on a fixed number of previous states. That is:

$$P(y_i|y_{i-1}x_{i-1}, y_{i-2}x_{i-2}, ..., y_1x_1) = P(y_i|y_{i-1}, y_{i-2}, ..., y_{i-k}) \tag{5.3}$$

In fact, we can get much better results on the NER task by looking at trigrams or quadrigrams ($k = 2$ or $k = 3$) instead of bigrams as with a regular HMM. Some label assignments are highly improbable, such as single token named entities separated by a common word. These kinds of patterns can be perceived by a higher order HMM. Second, the probability of a word depends only on its assigned label. That is:

$$P(x_i|y_{i-1}x_{i-1}, ..., y_1x_1) = P(x_i|y_i) \tag{5.4}$$

With these assumptions, the probability $P(Y|X)$ can be approximated by the expression:

$$P(Y|X) \propto \prod_{i=k+1}^{n} P(y_i|y_{i-1}, y_{i-2}, ..., y_{i-k})P(x_i|y_i) \tag{5.5}$$

All relevant probabilities can be estimated through maximum likelihood estimation from the relative frequencies of labels and features in the corpus. The best sequence of labels can be computed with a variable state Viterbi approach Li and Gray [2000]. However, as we increase k, this computation becomes exponentially more expensive. The beam-search strategy may be employed for a faster search, but we found that for $k \leq 4$, the Viterbi algorithm is still viable.

HMM based taggers have been successfully applied in many NLP and WDE tasks Rabiner [1990]; Leek [1997]; Freitag and McCallum [2000]. They are incredibly fast to train and also they are very interpretable, making them a good choice for a first approximation. However, these models are highly dependent on the right selection of features, what may outweigh the benefit of a small training cost.

5.1.1 Self training

In the task of NER on HTML, there are useful features related to the HTML structure that can help in identifying named entities. In a given website, named entities tend to occur inside the same HTML tags. The HTML tag feature or other HTML features could easily be incorporated in the HMM. However, these features are only useful inside a single website and they cannot be generalized, because different websites use distinct

HTML templates. Therefore, we propose a self-training strategy to obtain probabilities for these HTML features. It is implemented like this:

- Train the HMM without any HTML features.
- Compute labels for a website with the trained HMM.
- Use the computed labels as a proxy for the actual labels in the website and estimate HTML feature frequencies for this website alone.
- Recompute the labels now using the HTML feature probabilities.

In theory, this strategy could be used with any sequence tagger, however retraining a classifier with new features can become prohibitively expensive. This strategy is only possible because the computation of HTML feature frequencies can be performed very quickly. This adds very little overhead to the original HMM and improves precision and recall by a considerable margin.

5.2 Linear Chain Conditional Random Fields

A Linear Chain Conditional Random Field (CRF) is the discriminative analog to the HMM, it was first introduced by Lafferty Lafferty [2001]. It is a distribution $P(Y|X)$ that takes the form:

$$P(Y|X) = \frac{1}{Z(x)} \prod_{t=1}^{T} exp\left(\sum_{k=1}^{K} \theta_k f_k(y_{t-1}, y_t, X)\right) \tag{5.6}$$

where θ is the parameter vector that we are going to learn, $f_k(y_{t-1}, y_t, X)$ are feature functions over the current timestep t_y, the previous timestep y_{t-1}, and the observation vector X. And the partition function $Z(x)$, takes the form:

$$Z(x) = \sum_{Y} \prod_{t=1}^{T} exp\left(\sum_{k=1}^{K} \theta_k f_k(y_{t-1}, y_t, x)\right) \tag{5.7}$$

which is a sum over all possible label assignments Y. The partition function can be efficiently and exactly calculated with the sum-product algorithm. Parameter estimation is usually done through negative log likelihood minimization. The function can be optimized with techniques suitable for other maximum entropy models such as L-BFGS Liu and Nocedal [1989]. The most likely label sequences can be decoded with the Viterbi algorithm, as was the case for HMMs.

CRFs are more general than HMMs because the transitions from y_{t-1} to y_t can depend on the whole vector of observations X. This flexibility of feature functions allows for a wide range of possibilities. Recently, CRFs have been successfully employed as the output layer in complex neural architectures bringing improvements over models that compute labels independently.

5.3 Neural Network Architectures

Recurrent neural networks (RNN) have been successfully employed on numerous NLP tasks such as language modelling, POS tagging, speech recognition and NER. Different from feed-forward neural networks, RNNs can retain information in their internal state, making them more suitable for processing sequences, and consequently for solving text related tasks. Figure 5.1 describes an RNN for sequence labeling unrolled through multiple timesteps.

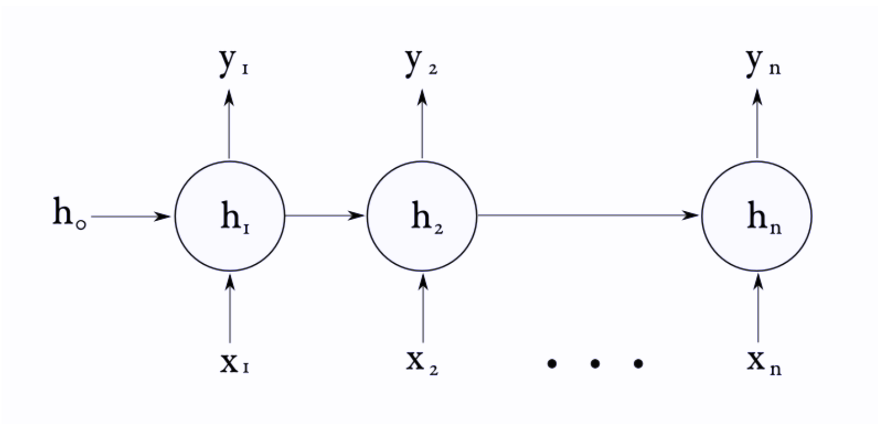


Figure 5.1. RNN for NER

At each timestep, the neural network computes a hidden state h_t using an input vector x_t and the previous hidden state h_{t-1}, that retains information from past iterations. Finally, the RNN produces an output vector y_t representing the label for that timestep. A common definition for an RNN cell is given by the equations:

$h_t = tanh(W_x x_t + W_h h_{t-1})$

$y_t = softmax(W_y h_t)$

Where W_x, W_h and W_y are weight matrices that can be trained with the Backpropagation Through Time (BPTT) algorithm. Theoretically, RNNs are capable of learning and retaining long term dependencies through their internal state h_t. However, in practice, it becomes difficult due to the vanishing gradient problem. Long short term memory networks (LSTM) were introduced by Hochreiter and Schmidhuber Hochreiter and Schmidhuber [1997] with this problem in mind and have been popularized since then.

LSTMs incorporate a memory cell c in the RNN definition and three gates to control the flow of information that comes in and out of the memory cell. The input gate Γ_i controls the amount of new information that will flow into the memory cell, the forget gate Γ_f controls the amount of previous information that will be retained in the memory cell, and the output gate Γ_o controls the amount of information stored in the memory cell that will be used to compute the output activation of the LSTM unit. LSTM cell implementations vary slightly in the literature. A visual description of our LSTM cell is provided in Figure 5.2.

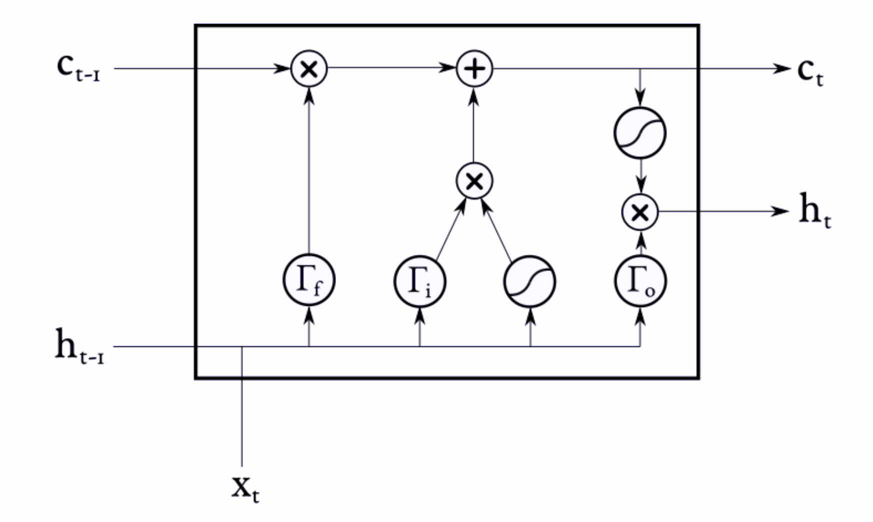


Figure 5.2. LSTM Cell

The equations for the LSTM cell are:

$$\Gamma_i = \sigma(W_i \cdot [x_t, h_{t-1}] + b_i)$$
$$\Gamma_f = \sigma(W_f \cdot [x_t, h_{t-1}] + b_f)$$
$$\Gamma_o = \sigma(W_o \cdot [x_t, h_{t-1}] + b_o)$$
$$c_t = \Gamma_f * c_{t-1} + \Gamma_i * tanh(W_c \cdot [x_t, h_{t-1}] + b_c)$$
$$h_t = \Gamma_o * tanh(c_t)$$

Where σ is the logistic sigmoid function. Γ_i, Γ_f, and Γ_o are the input, forget and output gates, respectively, and W_i, W_f, W_o are the weight matrices corresponding to each gate. c_t is the cell state at time t and h_t is the hidden state at time t. The vector $[x_t, h_{t-1}]$ is formed by concatenating the current input vector x_t and the hidden vector from a previous timestep h_{t-1}. Finally, $A*B$ represents the element-wise multiplication of matrices A and B and $A \cdot B$ represents the dot product of A and B.

This implementation differs from the LSTM cell described in Huang et al. Huang et al. [2015] in that the gates Γ_i and Γ_f do not receive inputs from the previous cell state c_{t-1} and the output gate Γ_o does not receive inputs from the current cell state c_t. This variation produces little difference in terms of model accuracy on the performed task, but it reduces model complexity.

5.3.1 BI-LSTM-CRF

On named entity recognition tasks, both past and future words are important to attribute a label at time t, however a regular LSTM network only takes past states into consideration. A bidirectional LSTM solves this problem by stacking two regular LSTMs, and feeding them with observations in opposite directions. The first LSTM receives forward states and the second LSTM receives backward states. The hidden states from both networks can then be concatenated at each timestep to produce output labels. With this architecture, LSTM cells may use information from past and future timesteps to decide the label at time t.

Huang et al. Huang et al. [2015] proposed a bidirectional LSTM with a CRF layer (BI-LSTM-CRF) on the output to tackle the sequence tagging problem. The main benefit of adding a CRF layer in the neural sequence model is that the labels are jointly decoded for a whole sentence instead of being predicted individually. Another possibility would be to use a beam search decoder to find an optimal sequence of labels. Predicted tags should be highly correlated in a named entity recognition task, so it is desirable to predict sequences conjointly. The BI-LSTM-CRF is described in Figure 5.3.

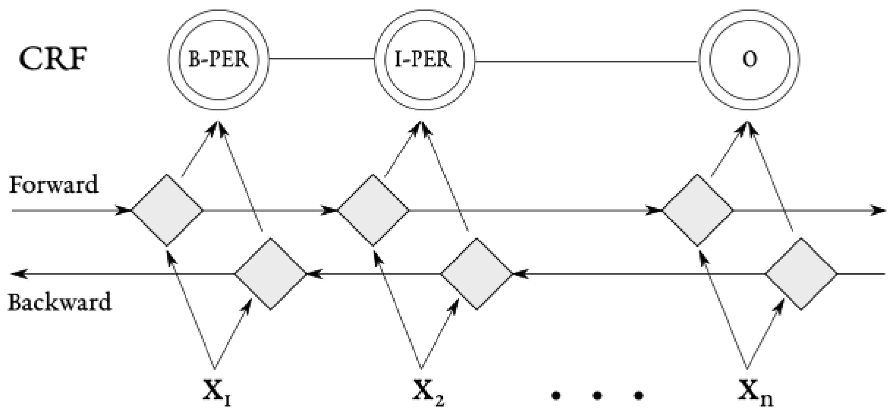


Figure 5.3. Bidirectional LSTM-CRF

This architecture achieved an F1 score of 90.10 on the English data from the CoNLL-2003 NER shared task Tjong Kim Sang and De Meulder [2003b], in contrast to 85.17 for a bidirectional LSTM without a CRF layer. In our experiments, the LSTM-CRF architecture uses a bidirectional LSTM with 100 hidden states, no peepholes and input and output dropout layers with a dropout rate of 0.5. The dropout layers have proven to be very important to prevent overfitting and allow better generalization.

5.3.2 CNN character representations

Ma and Hovy Ma and Hovy [2016] proposed to add a convolutional neural network (CNN) layer on top of a bidirectional LSTM-CRF to encode character-level information. The CNN layer is described visually in Figure 5.4.

The convolutional neural network receives character embeddings as inputs. The character representations generated by the CNN are combined with word level representations and fed to the BI-LSTM-CRF described in section 5.3.1. This architecture can learn morphological features that are very useful in the NER task, since similar named entities often present morphological similarities. This architecture obtained an F1 score of 91.21 in the CoNLL2003 dataset. In our experiments, the LSTM-CRF architecture with CNN character representations uses a one dimensional convolutional neural network with 30 filters and a window size of three characters on top of the LSTM-CRF architecture. The character embeddings fed to the CNN have 30 dimensions that are randomly initialized.

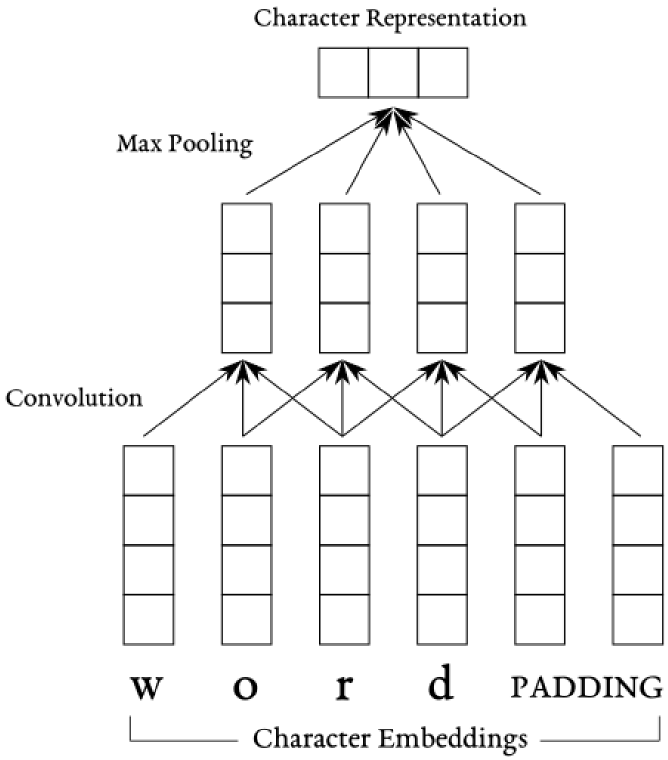


Figure 5.4. CNN based character representations

5.3.3 LSTM character representations

Lample et al Lample et al. [2016] proposed to use a bidirectional LSTM to model character-level representations on top of a BI-LSTM-CRF. Combining the forward and backward LSTM hidden states to form the character representation, as described in Figure 5.5.

This character representation is also combined with a word representation and fed to a BI-LSTM-CRF network. The forward state is expected to be a better representation of the suffix of a token, and the backward state is expected to be a better representation of the prefix of a token. This differentiates the architecture from the CNN based approach described in Section 5.3.2, because CNN filters discover positional invariant features, while LSTMs can better represent suffixes and prefixes. In our experiments, the LSTM-CRF architecture with LSTM character representations was implemented with a bidirectional LSTM with 25 hidden states, producing character representations of size 50. The character embeddings have 30 dimensions that are randomly initialized.

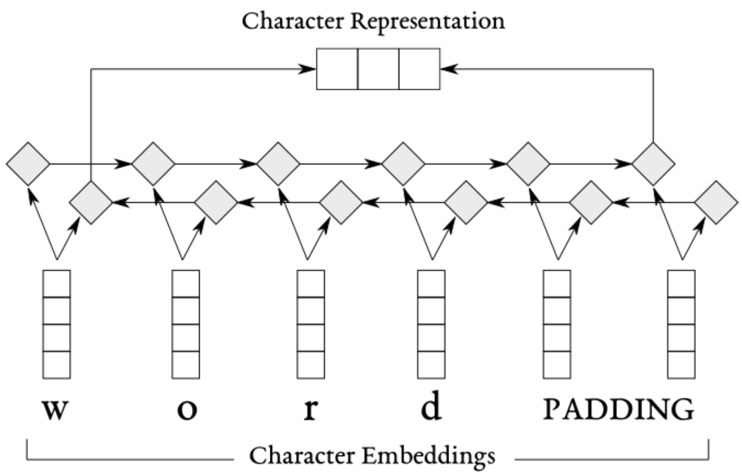


Figure 5.5. LSTM based character representations

5.3.4 Network training

All neural models were trained using mini batch Stochastic Gradient Descent over 50 epochs with batch size 10, learning rate 0.01, momentum 0.9 and decay rate 0.05. We used early stopping Caruana et al. [2000] to select the best parameters, considering the F1 measure in the validation set. All neural models used GloVe 100-dimensional word embeddings Pennington et al. [2014] that were fine tuned during training. In the case of NER on HTML, word embeddings work similarly to a gazetteer. Named entities with the same type have similar embeddings, so good word embeddings can achieve exceptional performance with little training and without a gazetteer.

Chapter 6

Experiments

We conducted experiments to evaluate sequence labeling methods of named entity recognition on HTML in the context of Web data extraction using the dataset described in Section **??**. The tested models are described in Table 6.1.

Model	Description
hmm-1	Regular HMM
hmm-2	HMM with $k = 2$
hmm-3	HMM with $k = 3$
crf	Linear chain conditional random fields
bi-lstm-crf	BI-LSTM-CRF model
bi-lstm-crf-cnn	BI-LSTM-CRF with CNN character representations
bi-lstm-crf-lstm	BI-LSTM-CRF with LSTM character representations

Table 6.1. Model descriptions

The evaluation of model performance was done through the precision, recall and F1 scores Rijsbergen [1979]. Precision is the percentage of named entities found by the model that are correct. Recall is the percentage of named entities that are present in the corpus and were found by the model. The F1 score is a composite measure that combines precision and recall with the formula:

$$F1 = \frac{2 * precision * recall}{precision + recall} \tag{6.1}$$

Named entities were only considered to be correct if they were a complete match of the corresponding entity in the dataset.

6.1 Experiment 1: No features

Experiment 1 aimed to evaluate the performance of sequence model with no features besides GloVe-100 embeddings. In the case of HMMs, only the lowercase unaccented token was used as a feature. Table 6.2 shows the Precision (P), Recall (R), and F1-scores (F1) for this experiment.

Model	Validation			Test		
	P	R	F1	P	R	F1
hmm-1	0.6965	0.5749	0.6299	0.6263	0.4431	0.5190
hmm-2	0.7047	0.6286	0.6645	0.6480	0.5222	0.5783
hmm-3	0.6127	0.6141	0.6134	0.5471	0.4634	0.5018
crf	0.7173	0.6683	0.6920	0.6671	0.5868	0.6244
bi-lstm-crf	0.8484	0.9044	0.8755	0.8358	0.8497	0.8427
bi-lstm-crf-cnn	0.9058	0.9575	0.9309	0.8779	0.8737	0.8758
bi-lstm-crf-lstm	0.9134	0.9435	0.9282	**0.8920**	**0.8815**	**0.8867**

Table 6.2. Precision, recall and F1 in the NER on HTML dataset for models that incorporate no features

Without carefully designed features or gazetteers, HMMs and CRFs have a very poor performance, achieving an F1-score of only 0.5783 for HMM-2 and 0.6244 for CRF at the test set. This is expected, since these models rely on good feature selections.

The neural models achieved high F1-scores in the test set even with the absence of features. The plain BI-LSTM-CRF architecture improved performance significantly in comparison with the conventional CRF (0.8427 against 0.6244). Also, neural character representations boosted performance by a significant margin reaching an F1-score of 0.8758 for CNN-based representations and 0.8867 for LSTM-based representations. LSTM based representations were superior in modelling morphological features, perhaps because they are able to differentiate suffixes and prefixes, while CNN filters are position invariant.

The results in Experiment 1 also show that pretrained word embeddings can work as a kind of universal gazetteer. Words with similar embeddings are likely to belong to the same class. This knowledge combined with the ability to learn morphological features can make up for the scarcity of textual data on some webpages.

6.2 Experiment 2: All features

Experiment 2 aimed to evaluate the performance of sequence model with all the features described in Table 3.2. In this experiment, we also evaluate the self-training strategy for HMMs described in Section 5.1.1. The self trained HMMs are described with the suffix "+ST". Table 6.3 shows the results for Experiment 2.

Model	Validation			Test		
	P	R	F1	P	R	F1
hmm-1	0.6061	0.7282	0.6616	0.7106	0.7633	0.7360
hmm-2	0.6279	0.7550	0.6856	0.7521	0.7810	0.7663
hmm-3	0.6573	0.7819	0.7142	0.7523	0.7795	0.7657
hmm-1+ST	0.7032	0.9077	0.7925	0.7522	0.8663	0.8052
hmm-2+ST	0.7321	0.9172	0.8143	0.7737	0.8789	0.8230
hmm-3+ST	0.7551	0.9172	0.8283	0.7961	0.8534	0.8237
crf	0.9024	0.9049	0.9037	0.8751	0.8227	0.8481
bi-lstm-crf	0.9430	0.9530	0.9480	0.8998	0.8527	0.8756
bi-lstm-crf-cnn	0.9244	0.9715	0.9474	0.9017	**0.8973**	**0.8995**
bi-lstm-crf-lstm	0.9465	0.9692	0.9577	**0.9108**	0.8715	0.8907

Table 6.3. Precision, recall and F1 in the NER on HTML dataset for models that incorporate all features

Conventional models like HMMs, and CRFs can become competitive with the right selection of features and a good gazetteer, however they still lose to the best neural model without features, demonstrating their inherent limitations. HMMs that employed trigrams or quadrigrams (hmm-2, hmm-3) performed better than regular HMMs. Also, we can see that the self-training strategy for HMMs improved the quality of the models significantly in all cases, boosting both precision and recall. This hints at the possibility to adapt this strategy to neural networks and boost the performance of neural models on the NER on HTML task.

The neural models also improved a little with the addition of features. The plain BI-LSTM-CRF model gets a closer performance to the models that employed neural character representations. It suggests that the LSTM and CNN character representations were able to learn at least part of the morphological features automatically in the first experiment. So, when these features are added explicitly, the differences in performance between different neural models become less noticeable.

Chapter 7

Conclusion

Machine-learning-based sequence labeling models provide a flexible approach to Web data extraction, in contrast to more traditional methods. In simple cases, a neural named entity tagger may be sufficient to solve the entire data extraction task. In other cases, the sequence tagger remains an important part of the web data extraction system, as it performs attribute labeling on data records with accuracy and flexibility.

In this article, we compared the performance of different sequence models on the task of named entity recognition on HTML, introducing a novel dataset that is publicly available. We found that there are two components to the most successful models: neural based character representations that extract morphological features automatically, and the joint modelling of output labels.

We showed that a BI-LSTM-CRF neural network with LSTM-based character representations can be employed effectively to solve a web data extraction task, achieving an F1-score of 0.8867 with no feature engineering on the faculty listings dataset.

The effective recognition of named entities on HTML is an essential step in most general Web data extraction methods. The accuracy achieved by deep neural architectures even on webpages that are very different from the plain text for which these architectures were initially designed shows the potential for a truly flexible approach to cross domain web data extraction.

Bibliography

Abdessalem, T., Cautis, B., and Derouiche, N. (2010). ObjectRunner: lightweight, targeted extraction and querying of structured web data. *Proceedings of the VLDB ...*, 3(2):1585--1588. ISSN 21508097.

Adelberg, B. (1998). NoDoSE—a tool for semi-automatically extracting structured and semistructured data from text documents. *ACM SIGMOD Record*, 27(2):283--294. ISSN 01635808.

Arasu, A., Garcia-Molina, H., Arasu, A., and Garcia-Molina, H. (2003). Extracting structured data from Web pages. *2003 ACM SIGMOD International Conference on Management of Data*, pages 337 -- 348.

Arocena, G. O. and Mendelzon, A. O. (1999). WebOQL: Restructuring documents, databases, and webs. *Theory and Practice of Object Systems*, 5(3):127--141. ISSN 10743227.

Califf, M. E. and Mooney, R. J. (1999). Relational learning of pattern-match rules for information extraction. *Computational Linguistics*, 4:9--15. ISSN 15324435.

Caruana, R., Lawrence, S., and Giles, L. (2000). Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping. In *Proceedings of the 13th International Conference on Neural Information Processing Systems*, NIPS'00, pages 381--387, Cambridge, MA, USA. MIT Press.

Chang, C., Chang, C., Lui, S., and Lui, S. (2001). IEPAD: information extraction based on pattern discovery. *Proceedings of the 10th international conference on World Wide Web*, pages 681--688.

Chang, C.-H., Kayed, M., Girgis, M. R., and Shaalan, K. F. (2006). A Survey of Web Information Extraction Systems. *IEEE Transactions on Knowledge and Data Engineering*, 18(10):1411--1428. ISSN 1041-4347.

Chang, C. H. and Kuo, S. C. (2004). OLERA: Semisupervised Web-data extraction with visual support. *IEEE Intelligent Systems*, 19(6):56--64. ISSN 15411672.

Chinchor, N. (1998). Overview of muc-7. In *Seventh Message Understanding Conference (MUC-7): Proceedings of a Conference Held in Fairfax, Virginia, April 29 - May 1, 1998*.

Crescenzi, V., Mecca, G., and Merialdo, P. (2001). Roadrunner: Towards automatic data extraction from large web sites. *Proceedings of the 27th International Conference on Very Large Data Bases*, pages 109--118. ISSN 10477349.

Dong, X., Gabrilovich, E., Heitz, G., Horn, W., Lao, N., Murphy, K., Strohmann, T., Sun, S., and Zhang, W. (2014). Knowledge vault: a web-scale approach to probabilistic knowledge fusion. *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '14*, pages 601--610. ISSN 0893-6080.

Ferrara, E. and Baumgartner, R. (2011). Automatic wrapper adaptation by tree edit distance matching. *Smart Innovation, Systems and Technologies*, 8:41--54. ISSN 21903018.

Ferrara, E., De Meo, P., Fiumara, G., and Baumgartner, R. (2014). Web data extraction, applications and techniques: A survey. *Knowledge-Based Systems*, 70:301--323. ISSN 09507051.

Freitag, D. (1998). Information Extraction from HTML: Application of a General Machine Learning Approach. *Proceedings of the Fifteenth National/Tenth Conference on Artificial Intelligence/Innovative Applications of Artificial Intelligence*, pages 517--523.

Freitag, D. and McCallum, A. (2000). Information extraction with hmm structures learned by stochastic optimization. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*, pages 584--589. AAAI Press.

Freitag, D. and Mccallum, A. K. (1999). Information extraction with hmms and shrinkage. In *In Proceedings of the AAAI-99 Workshop on Machine Learning for Information Extraction*, pages 31--36.

Furche, T., Gottlob, G., Grasso, G., Gunes, O., Guo, X., Kravchenko, A., Orsi, G., Schallhart, C., Sellers, A., and Wang, C. (2012a). Diadem: Domain-centric, intelligent, automated data extraction methodology. In *Proceedings of the 21st International Conference on World Wide Web*, WWW '12 Companion, pages 267--270, New York, NY, USA. ACM.

Furche, T., Gottlob, G., Grasso, G., Orsi, G., Schallhart, C., and Wang, C. (2012b). AMBER: Automatic Supervision for Multi-Attribute Extraction. *arXiv preprint*, 1210(5984):1--22.

Grishman, R. and Sundheim, B. (1996). Message understanding conference-6: A brief history. In *Proceedings of the 16th Conference on Computational Linguistics - Volume 1*, COLING '96, pages 466--471, Stroudsburg, PA, USA. Association for Computational Linguistics.

Hammer, J., Mchugh, J., and Garcia-molina, H. (1997). Semistructured Data : The TSIMMIS Experience. *Proceedings of the First East-European Symposium on Advances in Databases and Information Systems*, pages 1--8.

Hirsch, J. E. (2005). An index to quantify an individual's scientific research output. *Proceedings of the National Academy of Sciences*, 102(46):16569--16572.

Hirschman, L., Yeh, A. S., Blaschke, C., and Valencia, A. (2005). Overview of biocreative: critical assessment of information extraction for biology. *BMC Bioinformatics*, 6:S1 – S1.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Comput.*, 9(8):1735--1780. ISSN 0899-7667.

Hogue, A. and Karger, D. (2005). Thresher : Automating the Unwrapping of Semantic Content from the World Wide Web. *WWW '05: Proceedings of the 14th international conference on World Wide Web*, pages 86--95.

Hsu, C. N. and Dung, M. T. (1998). Generating finite-state transducers for semistructured data extraction from the Web. *Information Systems*, 23(8):521--538. ISSN 03064379.

Huang, Z., Xu, W., and Yu, K. (2015). Bidirectional LSTM-CRF models for sequence tagging. *CoRR*, abs/1508.01991.

Krüpl, B., Herzog, M., and Gatterbauer, W. (2005). Using visual cues for extraction of tabular data from arbitrary HTML documents. *Special interest tracks and posters of the 14th international conference on World Wide Web - WWW '05*, pages 1000----1001.

Kushmerick, N. (2000). Wrapper induction: efficiency and expressiveness. *Artificial Intelligence*, 118(1-2):15--68. ISSN 00043702.

Laender, A., Ribeiro-Neto, B. A., and S.Teixeria, J. (2002a). A brief survey of web data extraction tools. *ACM SIGMOD Record 31(2)*, pages 84--93.

Laender, A. H. F., Ribeiro-Neto, B., and da Silva, A. S. (2002b). DEByE - Date extraction by example. *Data Knowl. Eng.*, 40(2):121--154. ISSN 0169-023X.

Lafferty, J. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. pages 282--289. Morgan Kaufmann.

Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., and Dyer, C. (2016). Neural architectures for named entity recognition. *CoRR*, abs/1603.01360.

Lawrence, S., Lee Giles, C., and Bollacker, K. (1999). Digital libraries and autonomous citation indexing. *Computer*, 32(6):67–71.

Leek, T. R. (1997). Information extraction using hidden markov models.

Li, J. and Gray, R. M. (2000). *Image Segmentation and Compression Using Hidden Markov Models*. Kluwer Academic Publishers, Norwell, MA, USA. ISBN 0792378997.

Liu, D. C. and Nocedal, J. (1989). On the limited memory bfgs method for large scale optimization. *Mathematical Programming*, 45(1):503--528.

Liu, L., Pu, C., and Han, W. (2000). XWRAP: an XML-enabled wrapper construction system for Web information sources. *Proceedings of 16th International Conference on Data Engineering*, pages 611--621. ISSN 1063-6382.

Ma, X. and Hovy, E. H. (2016). End-to-end sequence labeling via bi-directional lstm-cnns-crf. *CoRR*, abs/1603.01354.

McCallum, A. (2005). Information extraction: Distilling structured data from unstructured text. *Queue*, 3(9):4:48--4:57. ISSN 1542-7730.

McCallum, A., Freitag, D., and Pereira, F. C. N. (2000). Maximum entropy markov models for information extraction and segmentation. In *Proceedings of the Seventeenth International Conference on Machine Learning*, ICML '00, pages 591--598, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Muslea, I., Minton, S., and Knoblock, C. (1999). A Hierarchical Approach to Wrapper Induction. *Proc. of the Third Annual Conf. on Autonomous Agents, ACM*, pages 190--197.

Pennington, J., Socher, R., and Manning, C. D. (2014). Glove: Global vectors for word representation. In *In EMNLP*.

Rabiner, L. R. (1990). Readings in speech recognition. chapter A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition, pages 267--296. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Ramshaw, L. A. and Marcus, M. P. (1999). *Text Chunking Using Transformation-Based Learning*, pages 157--176. Springer Netherlands, Dordrecht.

Rastegar-Mojarad, M., Liu, S., Wang, Y., Afzal, N., Wang, L., Shen, F., Fu, S., and Liu, H. (2018). Biocreative/ohnlp challenge 2018. In *Proceedings of the 2018 ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics*, BCB '18, pages 575--575, New York, NY, USA. ACM.

Ribas, S., Ribeiro-Neto, B., de Souza e Silva, E., Ueda, A. H., and Ziviani, N. (2015). Using reference groups to assess academic productivity in computer science. In *Proceedings of the 24th International Conference on World Wide Web*, WWW '15 Companion, pages 603--608, New York, NY, USA. ACM.

Rijsbergen, C. J. V. (1979). *Information Retrieval*. Butterworth-Heinemann, Newton, MA, USA, 2nd edition. ISBN 0408709294.

Sahuguet, A. and Azavant, F. (1999). Building light-weight wrappers for legacy Web data-sources using W4F. *Proceedings of the 25th VLDB Conference*, 99:738--741.

Sarawagi, S. (2008). Information extraction. *Foundations and Trends in Databases*, 1(3):261--377. ISSN 1931-7883.

Schulz, A., Lässig, J., and Gaedke, M. (2016). Practical web data extraction: Are we there yet? — A short survey. *IEEE/WIC/ACM International Conference on Web Intelligence (WI), 2016*, pages 562----567.

Shi, S., Liu, C., Shen, Y., Yuan, C., and Huang, Y. (2015). AutoRM: An effective approach for automatic Web data record mining. *Knowledge-Based Systems*, 89:314--331. ISSN 09507051.

Soderland, S. (1999). Learning Information Extraction Rules for Semi-Structured and Free Text. *Machine Learning*, 34(1):233--272. ISSN 0885-6125.

Sundheim, B. M. (1991). Overview of the third message understanding evaluation and conference. In *Proceedings of the 3rd Conference on Message Understanding*, MUC3 '91, pages 3--16, Stroudsburg, PA, USA. Association for Computational Linguistics.

Tjong Kim Sang, E. F. and De Meulder, F. (2003a). Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003 - Volume 4*, CONLL '03, pages 142--147, Stroudsburg, PA, USA. Association for Computational Linguistics.

Tjong Kim Sang, E. F. and De Meulder, F. (2003b). Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003 - Volume 4*, CONLL '03, pages 142--147, Stroudsburg, PA, USA. Association for Computational Linguistics.

Varlamov, M. I. and Turdakov, D. Y. (2016). A survey of methods for the extraction of information from Web resources. *Programming and Computer Software*, 42(5):279--291. ISSN 0361-7688.

Wang, J. and Lochovsky, F. H. (2003). Data extraction and label assignment for web databases. *Proceedings of the twelfth international conference on World Wide Web - WWW '03*, page 187. ISSN 00320862.

Zhai, Y. and Liu, B. (2005). Web data extraction based on partial tree alignment. *Proceedings of the 14th international conference on World Wide Web - WWW '05*, page 76. ISSN 10414347.

Zhu, J., Nie, Z., Wen, J.-R., Zhang, B., and Ma, W.-Y. (2005). 2d conditional random fields for web information extraction. In *Proceedings of the 22Nd International Conference on Machine Learning*, ICML '05, pages 1044--1051, New York, NY, USA. ACM.

Zhu, J., Nie, Z., Wen, J.-R., Zhang, B., and Ma, W.-Y. (2006). Simultaneous record detection and attribute labeling in web data extraction. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, pages 494--503, New York, NY, USA. ACM.