Populating Ontologies by Semi-automatically Inducing Information Extraction Wrappers for Lists in OCRed Documents

Thomas L. Packer

December 6, 2012

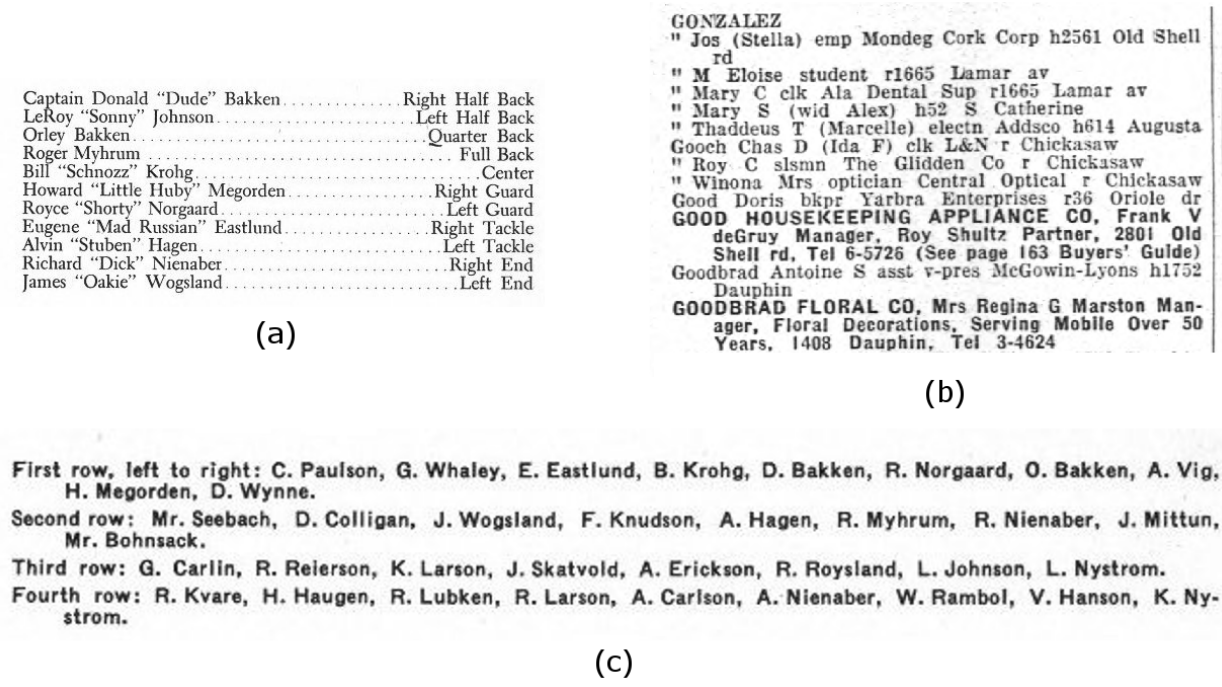
Abstract

A flexible, accurate, and efficient method of extracting facts from lists in OCRed documents and inserting them into an ontology would help make those facts machine queryable, linkable, and editable. But, to work well, such a process must be adaptable to variations in list format, tolerant of OCR errors, and careful in its selection of human guidance. We propose a wrapper-induction solution for information extraction that is specialized for lists in OCRed documents. In this approach, we induce a grammar or model that can infer list structure and field labels in sequences of words in text. Second, we decrease the cost and improve the accuracy of this induction process using semi-supervised machine learning and active learning, allowing induction of a wrapper from a single hand-labeled instance per field per list. We then use the wrappers and data learned from the semi-supervised process to bootstrap an automatic (weakly supervised) wrapper induction process for additional lists in the same domain. In both induction scenarios, we automatically map labeled text to ontologically structured facts. Our implementation induces two kinds of wrappers, namely regular expressions and hidden Markov models. We evaluate our implementation in terms of annotation cost and extraction quality for lists in multiple types of historical documents.

1 Introduction

On smart phones and throughout the Web, a growing body of data is being digitized from printed media. However, the data, itself, is underutilized because of the cost of extracting it from the OCRed text of its document images into an ontology, a machine-readable and mathematically specified conceptualization of a collection of facts.

Lists contain much of the explicit data in the printed texts of some genres and are what we target in the proposed research. Working with lists is already technically challenging, but working with OCRed lists is even more challenging, because OCR errors may be present in the data and because consistent delimiters between records and fields often are not present or are irregular. A few examples of printed lists in historical documents appear in Figure 1.



Captain Donald "Dude" Bakken ..... Right Half Back
LeRoy "Sonny" Johnson ..... Left Half Back
Orley Bakken ..... Quarter Back
Roger Myhrum ..... Full Back
Bill "Schnozz" Krohg ..... Center
Howard "Little Huby" Megorden ..... Right Guard
Royce "Shorty" Norgaard ..... Left Guard
Eugene "Mad Russian" Eastlund ..... Right Tackle
Alvin "Stuben" Hagen ..... Left Tackle
Richard "Dick" Nienaber ..... Right End
James "Oakie" Wogsland ..... Left End

(a)

GONZALEZ
" Jos (Stella) emp Mondeg Cork Corp h2561 Old Shell rd
" M Eloise student r1665 Lamar av
" Mary C clk Ala Dental Sup r1665 Lamar av
" Mary S (wid Alex) h52 S Catherine
" Thaddeus T (Marcelle) electn Addsco h614 Augusta
Gooch Chas D (Ida F) clk L&N r Chickasaw
" Roy C slsmn The Glidden Co r Chickasaw
" Winona Mrs optician Central Optical r Chickasaw
Good Doris bkpr Yarbra Enterprises r36 Oriole dr
GOOD HOUSEKEEPING APPLIANCE CO, Frank V deGruy Manager, Roy Shultz Partner, 2801 Old Shell rd, Tel 6-5726 (See page 163 Buyers' Guide)
Goodbrad Antoine S asst v-pres McGowin-Lyons h1752 Dauphin
GOODBRAD FLORAL CO, Mrs Regina G Marston Manager, Floral Decorations, Serving Mobile Over 50 Years, 1408 Dauphin, Tel 3-4624

(b)

First row, left to right: C. Paulson, G. Whaley, E. Eastlund, B. Krohg, D. Bakken, R. Norgaard, O. Bakken, A. Vig, H. Megorden, D. Wynne.
Second row: Mr. Seebach, D. Colligan, J. Wogsland, F. Knudson, A. Hagen, R. Myhrum, R. Nienaber, J. Mittun, Mr. Bohnsack.
Third row: G. Carlin, R. Reierson, K. Larson, J. Skatvold, A. Erickson, R. Roysland, L. Johnson, L. Nystrom.
Fourth row: R. Kvare, H. Haugen, R. Lubken, R. Larson, A. Carlson, A. Nienaber, W. Rambol, V. Hanson, K. Nystrom.

(c)

Figure 1: Lists in Historical Documents: (a) School Yearbook Football Team, (b) City Directory, (c) School Yearbook Group Photo Caption.

The lists we target have a consistent, regular structure. Each list entry, or record, is a description of a concept or entity. Substrings of these records, called fields, describe attributes, properties, or relationships of the entity. Generally, each record has a small number of fields which are explicitly delimited by text or whitespace. Some of these fields may be optional (not occurring in every record). In Figure 1(a), for example, the nicknames are optional—appearing in some records but not others. Fields may be factored out of the records of a list. The surnames in Figure 1(b) are factored fields that distribute to the ditto marks in each record below them. Figure 1(b) also illustrates what we call a mixed list as it contains records describing different kinds of entities—individuals and businesses in a city directory. Fields may also, themselves, be lists—nested lists. In Figure 1(c), for example, each record in the larger list denotes a row of people in a group photograph in a yearbook and contains a nested sublist—the names of the people in that row.

We propose to develop and evaluate a novel, general, and effective system for extracting information from lists in images of machine-printed documents. We call our system *ListReader*. ListReader will take the OCR transcriptions of document images as input and produce as output a collection of information about the records in each list stored as a populated ontology. The list in Figure 1(a), for example, may be the input; then ListReader would generate and populate the ontology in Figure 2. The ontology's object and relationship set names provide the field labels that identify the content of each record. One of the object sets of the populated ontology (*FootballPlayer* in Figure 2) contains identifiers of objects denoted by each record in the list. Other object sets contain object identifiers for other objects mentioned in list records (e.g. *Name* in Figure 2), and contain specializations of these object sets when list records mention specialized roles for objects (e.g. *Captain* in Figure 2). Still other object sets contain text objects that tie the text of the record fields directly to content in the ontology in the form of lexical objects. Relationship sets in the populated ontology identify relationships among objects in the ontology. As an example, for the list in Figure 1(a), ListReader populates the ontology in Figure 2 by placing eleven object identifiers, one for each football player in the list, into the *FootballPlayer* object set. It also places the object identifier for the captain Donald Bakken in the *Captain* specialization object set and all the names, nicknames, and positions in their appropriate object sets and populates the relationship sets with relationships that properly relate the object identifiers and text strings.

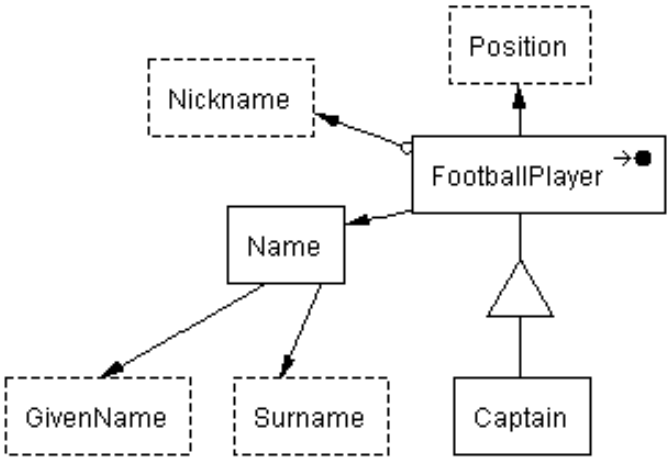


Figure 2: Output Ontology for List in Figure 1(a).

We envision two scenarios: (1) ListReader has no prior knowledge of the corpus domain or genre, and (2) ListReader has prior knowledge derived from having already processed other lists within the same document or domain. We call the first scenario "semi-supervised" because it uses some hand-labeled and some unlabeled text as training data. We call the second scenario "weakly supervised" because it takes as training data text that was automatically and noisily labeled. In the first scenario, a user will hand label one instance of each field in a list, relating it to objects and relationships in the ontology for the list. ListReader will then extract the same kinds of fields from the remaining unlabeled text. In the second setting, the user will not hand label any text. Rather, ListReader will take as input the knowledge it has gained about the format and content of other lists and produce the same kind of output for new lists.

To solve the problem of extracting information from printed lists, partially or fully automatically, we propose to adapt an information extraction technique called wrapper induction (Muslea et al. [1998], Kushmerick [1997]). ListReader will reduce the cost of extracting accurate information by minimizing its need

for human supervision and by leveraging the patterns and knowledge in the text itself. Using a combination of the above two scenarios, ListReader will iteratively improve its ability to extract information and grow its domain knowledge as it runs.

We plan to make the following contributions: (1) show how to perform wrapper induction using only one human-provided label per field per list in Scenario 1, (2) show that cheaper automatic labeling can replace a human labeler as input to wrapper induction in Scenario 2, (3) show how to use facts and wrappers induced from other lists to bootstrap the cheaper approach in Scenario 2, and (4) show that the above approaches can decrease user cost and increase accuracy over a state-of-the-art information-extraction system.

2 Related Work

Although researchers have worked on several topics related to wrapper induction for lists in OCRed documents, it is clear that more work in this area is needed. First we compare our proposal to previous work using wrapper induction and ontology population. Then we compare our proposal to information extraction (IE) research grouped by the kind of data processed.

Wrapper induction. Traditional web wrapper induction techniques (Muslea et al. [1998], Kushmerick [1997]) use both supervised and weakly supervised approaches, but focus on clean, tabular HTML text as input and named entities and their properties as output. Carlson and Schafer [2008] bootstrap the process of weakly supervised wrapper induction using the results of supervised wrapper induction. We believe that a similar approach could be applied to OCRed lists with some important modifications. There is no wrapper induction work that outputs ontological fact predicates beyond named entities and flat relations. There is little wrapper induction work involving lists or noisy OCR text. In an early project that subsumes the goal of wrapper induction for OCRed lists, Adelberg [1998] describes a general-purpose interactive document wrapper learning system for extracting information from a wide variety of text formats, including nested and OCRed lists, but it induces a brittle grammar that assumes constant-valued text used as record and field boundaries, is not tolerant of OCR errors, and is not unsupervised; further, he provides no empirical test results. Heidorn and Wei [2008] focus on extracting information from semi-structured OCRed text. This is the only paper we are aware of that uses an HMM as a wrapper language. Their approach is fully supervised and they do not take advantage of patterns in unlabeled data. Other papers use HMMs for extracting data from a database of isolated records (Grenager et al. [2005], Borkar et al. [2001]), where there is no notion of spatial proximity of records on a page or any other notion of inducing individual wrappers for individual sources.

Ontology population from text. Information extraction is closely related to ontology population from text. In ontology population, the schema of the target facts is in greater focus, which means that the kinds of facts to be extracted are better defined and sometimes more diverse. There is also a preference for unsupervised techniques. Cimiano [2006] surveys about 300 papers on the topics of learning ontology structure and populating ontologies from text. In the chapter about ontology population, Cimiano focuses on large-scale named entity category population and techniques for scaling the processes up to hundreds of ontological classes using Web-search-engine-, weakly-supervised-, and corpus-NLP-based approaches. Similar to much of the work on ontology population from text, our approach targets ontologies with rich conceptualizations, but it differs from previous work in targeting lists in OCRed text.

IE from OCRed lists. Besagni and Belaïd [2004], Besagni et al. [2003], and Belaïd [2001] extract records and fields from lists of citations, but rely primarily on hand-crafted knowledge that is specific to bibliographies. They do not discuss automatically detecting these lists. Belaïd [1998] extracts fields from

pre-segmented, heterogeneous OCRed records and uses a document layout modeling technique based on a complex hand-coded grammar and OCR and visual features of the text.

IE from plain text records. Research in this category assumes the input text contains no OCR errors and consists of pre-segmented records. Like Besagni et al. [2003] mentioned above, Lawrence et al. [1999] and Giles et al. [1998] use heuristic rules to extract fields from bibliographic citations taken from the text of postscript papers. Grenager et al. [2005], Chang et al. [2007], and Peng and McCallum [2004] extract fields from bibliographic citations using statistical sequence modeling, and (in the case of the first two) improve the cost of hand-labeling examples by applying expectation-maximization (EM) over partially unlabeled examples. Michelson and Knoblock [2008] and Agichtein and Ganti [2004] extract fields from text records by performing alignment and record linkage against a large set of pre-assembled reference data. Borkar et al. [2001] train HMMs from labeled examples of postal addresses and bibliographic citations. Pinto et al. [2003] focus on classifying rows in tables within plain text documents using a CRF based on a variety of textual and layout features and extract fields using heuristics. We believe we can build on some of this work, particularly on statistical sequence models like HMMs and unsupervised alignment, by combining these techniques and by taking better advantage of unlabeled examples and additional features in the text.

IE from HTML lists. Elmeleegy et al. [2009] and Gupta and Sarawagi [2009] demonstrate two ways to fully automatically (without manually labeled training examples) segment the fields in lists on the web. They estimate how field-like a substring of a record is with metrics trained on unlabeled data. The approach of Elmeleegy et al. [2009] does not label extracted fields nor handle mixed lists with multiple optional fields. Neither approach addresses the challenge of segmenting records. Embley et al. [1999b] and Embley et al. [1999a] rely on hand-crafted domain knowledge to segment records and to recognize fields in web pages containing lists of semi-structured or unstructured records. Other work by Embley and his colleagues recognizes factored fields and nested lists but only in web pages: Tao et al. [2009] rely on HTML tag trees to interpret nested lists, and Embley and Xu [2000] rely on both HTML tags and a hand-crafted ontology to interpret factored lists. We expect to draw from the unsupervised techniques of the above work on IE from HTML lists. We may also use domain-independent heuristics similar to those of Elmeleegy et al. [2009] and Embley et al. [1999b] in identifying candidate records. We expect that adding field extraction to the process will help find and segment records in OCRed lists as Embley et al. [1999b] showed for web lists.

IE from structured HTML records. Records on the web are often generated from an HTML page template and a database and are therefore consistently structured and delimited, unlike many semi-structured lists in OCRed documents. Dalvi et al. [2010], Álvarez et al. [2008], Etzioni et al. [2005], Carlson and Schafer [2008], Zhai and Liu [2005], Chang et al. [2003], Chidlovskii et al. [2000], and Hsu and Chang [1999] describe scalable and domain-general IE methods for structured web pages based on various forms of unsupervised pattern recognition, such as tree-alignment and repeated substring matching, but they rely on HTML tags which are not present in OCRed list records to delimit fields and records. Tao et al. [2009] use semi-supervised methods that are less scalable but rely less fundamentally on the HTML structure of a page. We plan to take an approach to training a wrapper on noisy labels—without hand-labeled examples—that is similar to a combination of the approaches in Dalvi et al. [2010] and Tao et al. [2009] but adapted to non-HTML wrappers. Kok and Yih [2009] use hand-crafted domain knowledge to recognize fields in semi-structured web pages, an approach we plan to avoid. Etzioni et al. [2005] use an unsupervised method based on seed terms (semantic bootstrapping) which is not well developed for list extraction and requires tuning of several parameters for each domain and corpus. We plan to also use bootstrapping, but applied to OCRed lists in such a way that it decreases manual effort for each new list. Baumgartner et al. [2001a] and Baumgartner et al. [2001b] use a visual labeling tool to train a wrapper for structured web pages. We also target our work to a setting that uses a visual labeling tool, but unlike their approach we plan to use machine

learning techniques to improve the efficiency of wrapper induction. Crescenzi et al. [2001] segment fields without labeling them. We may adapt some techniques from Crescenzi et al. [2001] for recognizing regular-expression patterns, but unlike their work, we will require that our wrappers provide labels for fields, not just boundaries. In addition to structured record IE, Chang et al. [2003] also describe one of the few methods of record detection and splitting that might work for list recognition in OCRed text. However, it is unclear how much their approach relies on identical repeating HTML tags. We believe other novel approaches will work better at list recognition with OCR text where patterns are often implicitly embedded in the semantics, may not be contiguous within the record text, and may not be present in every record of a list.

In summary, we see significant work on which to build as well as opportunities to make contributions in populating ontologies from lists in OCRed text. Efficiently inducing a wrapper that can accurately find, segment, and categorize records and fields in tag-delimited, semi-structured web pages is becoming commonplace. However, wrapper induction is a novel way of looking at the challenge of extracting information from OCRed text—and even from some non-OCRed lists when we consider that existing approaches miss some key steps such as list finding, record segmentation, or field labeling, or are specialized for certain kinds of lists like bibliographies or postal addresses. There is also little research in inducing wrappers for non-HTML lists or in populating ontologies using facts from lists of any kind, but especially for complex lists or in ways that are tolerant of OCR errors.

3 Thesis Statement

It is possible to populate an ontology semi-automatically, with better than state-of-the-art accuracy and cost, by inducing information extraction wrappers to extract the stated facts in the lists of an OCRed document, firstly relying only on a single user-provided field label for each field in each list, and secondly relying on less ongoing user involvement by leveraging the wrappers induced and facts extracted previously from other lists.

4 Project Description

For a list L, we seek to create both an ontology O for the fact schemes of L and to populate O with the facts stated in L. We are interested in being cost-effective, and we wish to automate the creation process to the extent possible. Our tool, ListReader, is a bootstrapping tool that "learns as it goes." ListReader learns by keeping track of the ontologies it has constructed, the facts it has extracted, and the list patterns it has seen and by leveraging them for future list-extraction tasks. We are interested to know (1) the minimal human input required to start the bootstrapping process when ListReader has not yet built any ontologies, has extracted no facts, and has seen no lists, and (2) the minimal bootstrapping required for ListReader to function fully automatically with no human input. Ultimately, we see ListReader as functioning in a partially bootstrapped fashion: for a new list L, ListReader does the best it can to automate the construction of a list ontology O for L and to populate O with the facts stated in L and asks for the minimum amount of human input it needs to complete the task. We explain the details of how we expect ListReader to work through an illustrative running example.

4.1 Scenario 1: Semi-supervised Wrapper Induction

For our running example, we consider as input the list in Figure 1(a), modified slightly to enable us to illustrate all the main points of wrapper induction. Figure 3 shows the user interface for Scenario 1, which

enables a user to initialize an ontology for the list and label the fields of the first record with respect to the ontology. A user loads the document containing a list in the right panel, and, in Figure 3, has toggled the image to view the OCRed text. Looking at the first record, the user employs the GUI to construct a standard data-entry form using a special form builder web application that corresponds to the stated facts of the first record of the list and selects text from the document and inserts it into the form. Figure 3 shows the interface as the selected text, "Right Half Back", is about to be inserted into the highlighted *Position* field.

By construction, the form corresponds to an ontological conceptualization, which ListReader can thus automatically generate. For our running example, the generated ontology is the ontology in Figure 2 except that there is no *Nickname* object set as it is missing from the first record. (Below, we discuss how to add missing field.) Further, ListReader can also label the field values in a copy of the OCRed text with labels that correspond to paths in the form or ontology, which thus provides for extracting the field values with which to populate the ontology. Figure 4 shows the labels applied to the first record. The label "<FootballPlayer.Name.GivenName>", for example, designates that the string "Donald" belongs in the *GivenName* object set and that it links to an object identifier in the *Name* object set which, in turn, links to the object identifier in the *FootballPlayer* object set that identifies the football player for the first record in the list. ListReader can map information among all three representations: a filled-in data entry form, a populated ontology, and labels that sequentially identify the fields in text records.

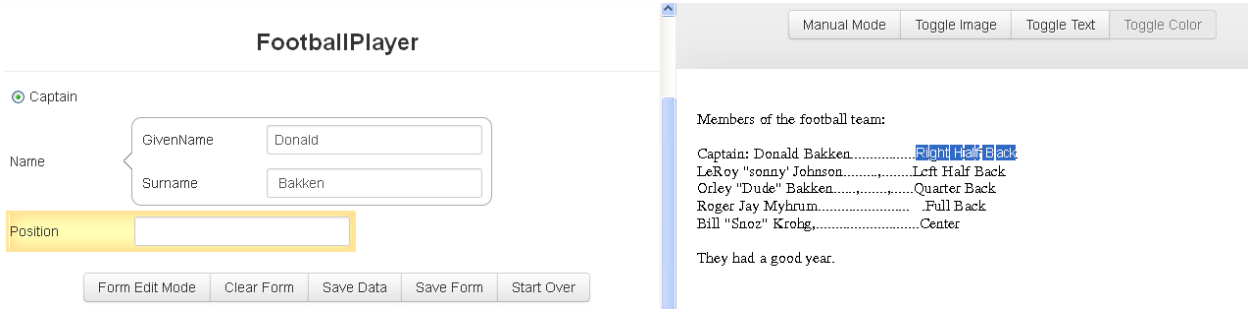


Figure 3: Data Entry Form and Input Text Containing List.

```
<Captain>Captain</Captain>: <FootballPlayer.Name.GivenName>Donald</FootballPlayer.Name.GivenName>
<FootballPlayer.Name.Surname>Bakken</FootballPlayer.Name.Surname>.....<FootballPlayer.Position>
Right Half Back</FootballPlayer.Position>
```

Figure 4: The First Record of Figure 3 with Sequential Labels.

ListReader uses the sequential labeling of fields in the first record to initialize the wrapper-induction process. Figure 5 shows the initial regex wrapper, and Figure 6 shows the initial NB-HMM wrapper (a hidden Markov model with a naive Bayes emission model). The numbered labels below the regular expression in Figure 5 represent the field labels assigned to each capture group in parentheses. To induce the initial regex wrapper in Figure 5, ListReader creates capture groups for each of the four labeled fields. Capture groups for specializations, like *Captain* in our example, contain the literal labeled text, while capture groups for text fields contain regular expressions generalized to loosely identify their field values (e.g., $\backslash w\{6,6\}$ for the six-character token "Donald"). Delimiters between fields become literals, appropriately inserted into the regular expression. In a similar fashion, ListReader initializes the NB-HMM wrapper in Figure 6, establishing the states, transitions, transition probabilities, classes, types, and emission probabilities with respect to a

predefined feature set according to its match with the first record. After the first record, the model is largely constrained to match only that record. ListReader relaxes these constraints below.

```
\r(Captain): (\w{6,6}) (\w{6,6})[\.]{17,17}(\w{5,5} \w{4,4} \w{4,4})
1. Captain,  2. GivenName,  3. Surname,  4. Position
```

Figure 5: Regex Wrapper for First Record of Figure 3.

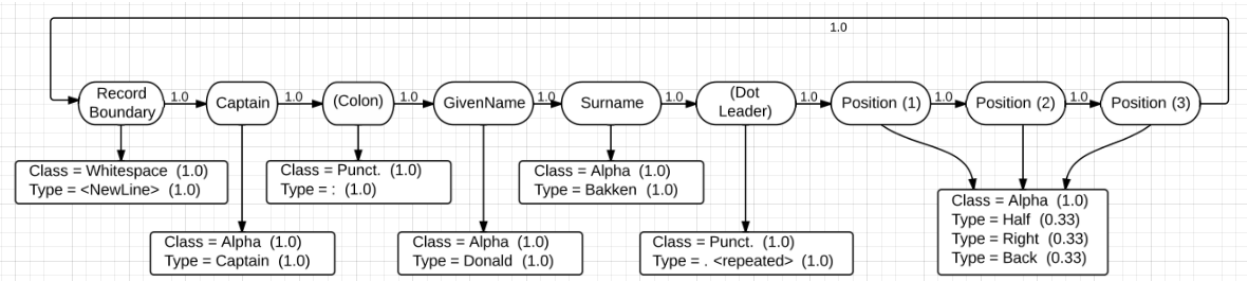


Figure 6: NB-HMM Wrapper for First Record of Figure 3.

ListReader processes subsequent records by finding best possible alignments of delimiters and fields. Based on alignments, ListReader further generalizes field-content regular expressions, adjusts delimiter regular expressions if needed to accommodate OCR errors or formatting variations, makes any missing fields optional, and assesses the possibility of new fields. In our running example, when ListReader aligns the first record with the second, it recognizes that "Captain: " is missing and therefore optional. It also notices that the dot-leader delimiter has some OCR-error commas mixed in with the dots and adjusts its dot-leader delimiter expression to accommodate them. Finally, it recognizes " "sonny' " as a substring that may contain a potential new field and requests user input. In this case the user responds by adding a *Nickname* field to the form and filling in the entry blank with "sonny". This user action also augments the ontology, adding *Nickname* as Figure 2 shows. ListReader processes the third record, adjusting for the close-quote delimiter following the nickname to accommodate the OCR error in the second record and further generalizes the *Position* field recognizer by making one of the three *Position* tokens optional. For the fourth record, ListReader makes a minor adjustment to allow for spaces in the dot-leader. More interestingly, it recognizes " Jay " as a potential new field. Figure 7 is a screenshot of the user interaction needed to provide for a second given name. The user has transformed the *GivenName* entry blank into a multiple-entry blank and is about to click on the selected text, "Jay" to fill in the highlighted field. This adjustment augments the ontology by allowing more than one *GivenName*, leaving *GivenName* dependent on *Name*, but not functionally dependent. ListReader processes the fifth record, needing only to further adjust the *Position* field recognizer to allow it to accept one token. ListReader stops extracting information at the end of a list when updating the wrapper would be too great of a change, as quantified by a wrapper update score. In Figure 3 "They had a good year.", which terminates the list, would have a zero-match wrapper update score. Figures 8 and 9 show the final wrappers induced by ListReader in our running example.

To do the above work systematically, ListReader generalizes a regex by first generating and then testing a pre-defined set of delimiter and field adjustments that it applies to the initial regex. This is a beam search through a wrapper hypothesis space. Adjustments include replacing one character with another based on common OCR errors, extending the length of a field or a word within a field, and generalizing the character set of a word given a predefined character taxonomy. ListReader scores each new candidate regex by a

combination of (1) whether the regex matches text immediately following the last recognized record and (2) how reasonable the adjustment is given how list records vary in structure. ListReader generalizes an NB-HMM in a similar manner, proposing adjustments in the number of hidden states, and transition and emission probabilities, and then scoring candidate NB-HMMs using a combination of likelihood (how well they match subsequent text) and prior probability (how reasonable the adjustments are).

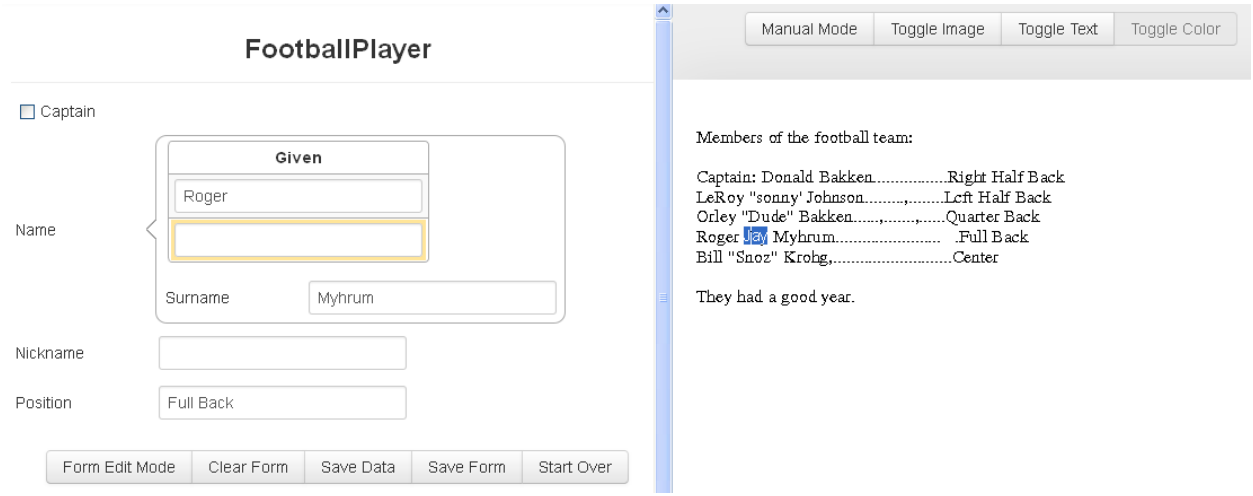


Figure 7: Altered Form to Accommodate a Second Given Name.

```
\r((Captain): )?(\w{4,6}) ((\w{3,3}) )?("(\w{4,5})["'] )?(\w{5,7})[\., ]{17,28}((\w{4,7} )?(\w{4,4} )?\w{4,6})
2. Captain,  3. GivenName,  5. GivenName,  7. Nickname,  8. Surname,  9. Position
```

Figure 8: Regex Wrapper for All Records of Figure 3.

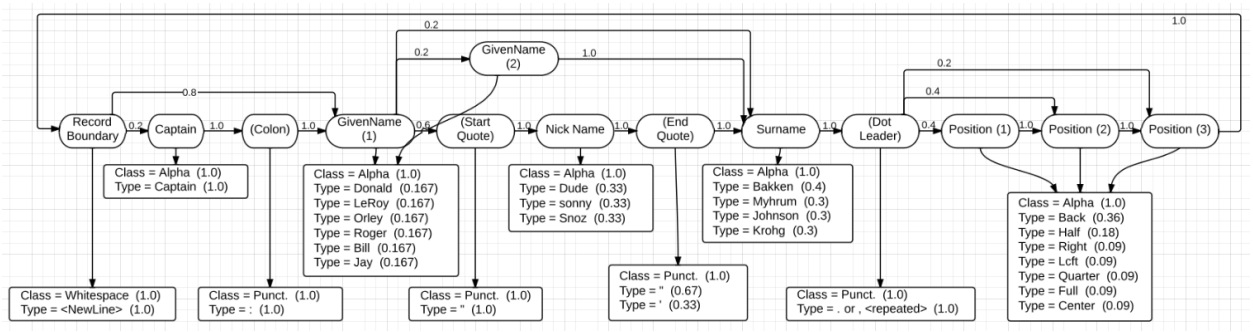


Figure 9: NB-HMM Wrapper for All Records of Figure 3.

Once induced for a list, the wrappers extract all stated facts of the list in a final pass and populate the ontology associated with the wrappers. To enable bootstrapping, ListReader also adds the wrappers and the generated ontologies to its knowledge repository. The ontologies are linguistically grounded extraction ontologies (Embley, et al. [1999]), which are conceptualizations whose object and relationship sets each have recognizers that identify text indicating that an object or literal string belongs to a particular object set or that a relationship among objects and literal strings belongs to a particular relationship set. In general,

the recognizers can be sophisticated regular expressions involving dictionaries, macros, exclusion expressions, and left- and right-context expressions. Although we may initialize some object sets common to many domains, such as *Date*, *Time*, and *Currency*, with sophisticated recognizers, we are also interested in bootstrapping completely from scratch. Thus, when storing an ontology in its knowledge repository, ListReader constructs a dictionary-based recognizer for every lexical object set[1] in the ontology. The dictionary consists of the literal strings recognized for the object set. For the ontology in Figure 2, for example, the dictionary for *GivenName* is {"Donald", "LeRoy", "Orley", "Roger", "Jay", "Bill"} and for *Position* is {"Right Half Back", "Lcft Half Back", "Quarter Back", "Full Back", "Center"} (OCR errors included).

4.2 Scenario 2: Weakly Supervised Wrapper Induction

In Scenario 2, ListReader attempts to process lists fully automatically—without human intervention. Given a bootstrapped knowledge repository and a document page or sequence of pages, ListReader invokes each extraction ontology in its repository to recognize and label text strings in the page(s) for each of the ontologies's lexical object sets. Further, since each extraction ontology has associated with it both an induced regex wrapper and an induced NB-HMM wrapper, ListReader also applies them to find patterns, if any, that correspond to previously seen lists or parts of previously seen lists. Given text-string and text-pattern matches for an ontology, ListReader finds regularities, if any, that indicate the presence of lists and scores the degree of match with potential list records. For fully automatic wrapper induction, ListReader must find a collection of one or more ontologies that together label the fields of the records of the list. Since each label identifies a path in an ontology, ListReader constructs an ontology for the new list by merging the roots of each selected path for the fields. ListReader then finds any one of the records whose fields have a complete labeling and performs wrapper induction just as in Scenario 1 except that (1) ListReader scans both upward and downward from the starting record and (2) if ListReader encounters a new field, it obtains the field's labeling from an extraction-ontology-induced label rather than from a human operator.

To further explain the details of fully automatic wrapper induction, we say how ListReader is able to induce the wrappers in Figures 8 and 9 for the list in Figure 3 without human intervention. We assume that our knowledge repository contains the two extraction ontologies in Figure 10 and that ListReader previously created the *BasketballPlayer* ontology in Figure 10 from wrapping the list in Figure 11 and inducing the regex wrapper in Figure 12. The lists in Figures 11 and 3 are similar; indeed, we are assuming that the two lists come from the same high school yearbook—one for the football team and the other for the basketball team. For our application domain we expect that many lists will be similar to each other (e.g., hundreds of child lists in family-history books all have the same format). We have not provided the details for the *FootballPlayer* ontology in Figure 10 but assume that it comes from some other high school yearbook where each list entry gives the first and last name of a player, a class standing, and the position played.

When ListReader applies the wrapper in Figure 12 to the list in Figure 3, it fails for all records except the last. The main problem is that the *Position* must have only one token. ListReader also applies wrapper fragments—meaningful components of induced wrappers such as *GivenName*-(optional)*NickName*-*Surname* in Figure 12. Applying this wrapper fragment (namely, the regular expression in Figure 12 with the last capture group for *Position* omitted) returns a match for the first, third, and fifth list entries in Figure 3. The second fails to match because of the OCR error in the close-quote delimiter, and the fourth fails because of the extra given name.

[1] An object set is *lexical* if its content is a set of strings and is *non-lexical* if its content is a set of object identifiers. In Figure 2, the object sets *GivenName*, *Surname*, *Nickname*, and *Position* are lexical and *FootballPlayer*, *Captain* and *Name* are non-lexical. Note that object identifiers in *Name* do indeed denote a name since they link the two name components, given name and surname, together.

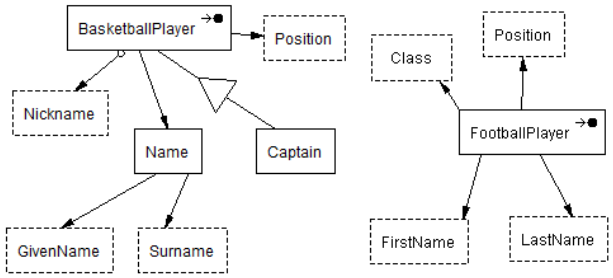


Figure 10: Ontologies in the Knowledge Repository.

```
Members of the basketball team:

Captain: Orley "Dude" Bakken.... .. .. ., ..., ,.Guard
Bill "Stone" Stein.... .. .. ., ..., ,.Guard
Jay Johnson... ., ... ....,. .Forward
LeRoy "Sonny" Johnson....... ........Forward
Stephen "Spike" Wilkinson...., .., ....Center
```

Figure 11: Basketball Player List.

```
\r((Captain): )?(\w{3,7}) ("(\w{4,5})" )?(\w{6,9})[\., ]{14,34}(\w{5,7})
2. Captain, 3. GivenName, 5. Nickname, 6. Surname, 7 Position
```

Figure 12: Regex Wrapper Induced for the List in Figure 11.

Wrapper matching requires that delimiters as well as fields match precisely and sequentially. Extraction ontology matching ignores delimiters and field order, but is stringent in accepting only dictionary matches for field values it has previously encountered. Applying the extraction ontologies learned from processing the list in Figure 11, yields *GivenName* matches for "LeRoy", "Orley", "Jay", and "Bill"; *Nickname* matches for "Dude" (but not "Sonny" because of the OCR capitalization error in Figure 3); and *Surname* matches for "Bakken", "Johnson", and "Bakken" (a second time). For the *FootballPlayer* ontology in Figure 10, we assume *FirstName* matches for the common names "Donald", "Roger", "Jay", and "Bill" and *LastName* matches for the only common surname "Johnson". We also assume matches for all the football-position names except for "Lcft Half Back" since it has an OCR error.

Given the matching results of the extraction ontologies and wrappers in the knowledge repository, ListReader must first locate a list (if any) on the document page(s) being processed. When wrappers or wrapper fragments match multiple successive potential records as they do in our running example, success is highly likely. Otherwise, ListReader must garner enough evidence through repeating field patterns such as football player positions in our example to enable it to locate a list. Once ListReader knows generally where a list is on a page, its next task is to identify any complete record as a starting point. Again, matching wrapper fragments can aid in this record-identification task. Ontology fragment matches can also aid in record identification because each ontology constitutes a group of concepts that typically go together to form a list record. ListReader can identify records for the list in Figure 3 as being composed of (1) the ontology fragment of the *BasketballPlayer* ontology in Figure 10 consisting of all its object and relationship sets except for *Position* and its connecting relationship set and (2) the ontology fragment of the *FootballPlayer* ontology in Figure 10 consisting only of the object sets *FootballPlayer* and *Position* and its connecting relationship set. ListReader stitches ontology fragments together by merging roots (and only roots). Unless the roots of all merged ontologies happen to have the same name, ListReader gives the merged root the name *Thing*. Thus, for our example, the initial induced list ontology is identical to the ontology in Figure 2 except that the name of the root is *Thing* rather than *FootballPlayer*.

After constructing the initial list ontology, ListReader selects one of the identified records as its starting place. In our example, the first, third, and fifth records in Figure 3 are possible starting records. For purposes of illustration, we start with the first. Having a given starting record, ListReader discards all labeling and relabels the starting record with respect to the induced list ontology. In our running example the labeling

is the same as the labeling in Figure 4, except that each instance of *FootballPlayer* is *Thing* instead. Given this new labeling, ListReader proceeds as in Scenario 1, first going backwards to the beginning of the list and then forward to the end. For our running example, ListReader attempts to process "Members of the football team:" as a record, but discovers it must be at the beginning of the list. ListReader next processes the second record. Whereas before, it needed to consult a human operator to resolve the potential new field " "sonny' ", it now has at its disposal the knowledge that "sonny" can be labeled as a *NickName* because the basketball player list wrapper in its repository that extracts a full name (including the nickname) matches two records in the current list; ListReader can make a small alteration to allow it to also match this name. It therefore accepts "sonny" as a Nickname and continues to the third record, which it can immediately process, adjusting for the optional third token in the *Position* name. When processing the fourth record, it recognizes " Jay " as a possible new field. Since "Jay" has already been labeled by a path from the root of an ontology through *Name* to *GivenName*, ListReader accepts it and adjusts the ontology—keeping *GivenName* dependent on *Name* but no longer functionally dependent. ListReader processes the fifth record, adjusting the *Position* field to allow it to be a single token. Finally, ListReader, seeing "They had a good year.", realizes that it has come to the end of the list because nothing matches, even with some small adjustments to the wrapper.

5 Validation

We aim to demonstrate our thesis by testing four hypotheses (5.1–4 below) with empirical measurements, evaluating against a gold standard corpus. This corpus will consist of a variety of historical documents provided by Ancestry.com and FamilySearch.org that include document images and OCRed text for each page. We will hand annotate the facts in separate development and blind test sets, consisting of 100 and 400 pages respectively. We will evaluate precision and recall in the usual way. We will combine accuracy with cost in a single measurement, namely, number of user-specified labels, which accounts for both training data and extraction mistakes (Heidorn and Wei [2008], Kristjansson et al. [2004]).

5.1 Hypothesis 1: Semi-supervised Fact Extraction

Hypothesis 1: To induce a wrapper for a given list L that can extract all stated facts from L and populate an ontology O with these facts, ListReader requires only a single user labeling of each field of L with respect to O.

To test Hypothesis 1, we will see how well semi-supervised wrapper induction works for lists in our corpus and see whether ListReader can generate a wrapper using only one labeled instance per field. We will also see how well ListReader can detect new optional fields in unlabeled records.

5.2 Hypothesis 2: Weakly Supervised Fact Extraction

Hypothesis 2: Given a page or sequence of pages of a document OCRed with reasonable quality that contains a list L, and given a knowledge repository of populated ontologies and wrappers induced for previously processed lists, then, assuming that the ontologies and wrappers in the repository recognize at least $R\%$ of the field strings in L belonging to each target lexical field scheme, along with perhaps some false positives (less than $(100-P)\%$), ListReader can fully automatically find L and generate a list ontology and wrapper for L and extract all its stated facts. Finding the exact recall R and precision P sufficient for full automation is not the objective, rather finding some R and P that work. We expect both P and R to be around 50%. What

"reasonable OCR quality" is also depents on the required P and R, so we will specify OCR word error rates at the same time.

To test Hypothesis 2, we will see how well weakly supervised wrapper induction works for each list in a corpus by determining a minimum level of P and R for successful wrapper generation. We will artificially add noise to the labels of each list in our corpus at varying levels of P and R and induce a wrapper for each list, as in Scenario 2.

5.3 Hypothesis 3: Bootstrapping

Hypothesis 3: A wrapper fragment or ontology fragment with n fields selected as a match of a record fragment of a list reduces the number of user-specified labels needed to extract all stated facts from L by n.

To test Hypothesis 3, we will see how well bootstrapping works using two kinds of experiments. First, we will show that bootstrapping beginning with Scenario 1 can produce P- and R-level noisy labelers. Second, we will determine how much user-specified labeling decreases in Scenario 2 using noisy recognizers induced in the bootstrapping process.

To show that bootstrapping works outside the laboratory, we will apply ListReader to multiple random page samples from our corpus and plot accuracy and the number of user-specified labels as the number of processed lists increases. As part of this, we will vary the number of pages assigned to Scenario 1 and Scenario 2 to show ListReader's sensitivity to the number of lists wrapped with the help of a completely manually labeled record.

5.4 Hypothesis 4: Comparisons with Other Systems

Hypothesis 4: ListReader can label fields in, and extract facts from, lists in OCRed documents with less cost (human effort) and higher accuracy (precision, recall, and F-measure) compared to a representative state-of-the-art information extraction system.

Reducing fact extraction to sequence labeling allows us to compare ListReader not only systems like OntoES (Embley et al. [2011]), which extract fact predicates natively, but also to information extraction approaches that apply labels to sequences of character or word tokens, such as CRFs (Sutton and McCallum [2010]). For Hypothesis 4, we will compare the precision, recall, F-measure, overall accuracy, and number of user labels required of ListReader, with the supervised linear-chain CRF in the MALLET toolkit (McCallum [2002]). We will also compare ListReader to OntoES and the manual work it requires.

6 Dissertation Schedule

1. Prepare datasets	Incremental
2. Semi-supervision and label mapping	Fall 2012
3. ICDAR conference paper "Semi-supervised Wrapper Induction for OCRed Lists"	Feb. 1 2013
4. Journal paper "Semi-supervised Wrapper Induction for OCRed Lists"	Winter 2013
5. Weak supervision	Winter 2013
6. Journal paper "Weakly-supervised Wrapper Induction for OCRed Lists"	Winter 2013
7. Dissertation	Summer 2013
8. Dissertation defense	Fall 2013

(Journals we are considering: IJDAR first; JASIST, PAMI, PR, TKDE, DKE? second)

7 Appendix: Research Area

Work on machine-printed list recognition and information extraction lies at the intersection of three large areas:

1. Document image analysis,
2. Information extraction from semi-structured text, and
3. Semi-supervised machine learning within natural language processing.

7.1 Document Image Analysis

Information extraction (IE) from printed document images depends on processes that convert images of text into editable, searchable digital text. These processes are part of the area called document image analysis. Nagy [2000] reviews 99 document image analysis papers.

The conversion of a document image into text is composed of two processes: (1) document structure and page layout recognition, and (2) optical character recognition (OCR). Document structure recognition is divided into two steps: physical and logical analysis. Physical structure analysis identifies the hierarchical composition of geometric document objects, e.g., characters, words, text lines, and text blocks or zones. Logical structure analysis, a process related to information extraction, groups and labels physical structures with categories or roles. Mao et al. [2003] survey many ways of doing these two steps and their deficiencies. These methods take the output of physical layout analysis as input and label those text blocks using rules sensitive to location, format, and textual content of those blocks. Logical layout analysis may also reconstruct paragraphs broken during formatting and determine their reading order.

OCR converts images of printed text into digital character codes such as ASCII that we can more conveniently process electronically. Fujisawa [2008] reviews OCR research starting in the 1950's and Mori et al. [1992] review the historical evolution of OCR methods showing a confluence of two research perspectives: template matching and structural analysis. To make the most of OCR output which often contains errors, some researchers correct OCR errors in a post-processing step. Kukich [1992] surveys and unifies error-detection and -correction techniques within the framework of generating, ranking, and selecting candidate corrections.

Some document image analysis researchers are interested in IE as a means of extracting document metadata, such as bibliographic fields, from the printed text itself. The metadata extraction perspective of IE differs from other forms of IE (such as website wrapper induction) in that it must deal with OCR errors (Ishitani [2001]) and must sometimes integrate language and spatial clues (Zhu et al. [2007]). It is also not applied to as broad a range of information.

7.2 Information Extraction, Especially from Semi-structured Text

Semi-structured text is a continuum between structured text in which tabular, database-like structure dominates and unstructured text in which natural language grammatical structure dominates. Because semi-structured text has characteristics of both, there is a confluence of research methodologies and ideas from both natural language processing and database systems.

We list three surveys of the broad area of IE. Sarawagi [2008] surveys IE in general. Turmo et al. [2006] also survey IE emphasizing adaptive machine learning methods. Laender et al. [2002] discuss and categorize information extraction tools and emphasize the database perspective.

Kushmeric made a foundational contribution to the area of information extraction from tabular web pages in his dissertation (Kushmerick [1997]). In it, he framed the problem in terms of inductive learning of a wrapper for individual websites using both weakly supervised and supervised learning, and assesses learnability both empirically and analytically. Wrapper induction is the building of an information extraction model that is specific to each sources (pages) format using some kind of automated or machine learning technique and has reduced the requirement for hand-crafted knowledge and human supervision. His work is in contrast to the two more traditional alternatives to wrapper induction, which are (1) to build information extraction models that are not specific to each page (no *wrapper*), e.g. to build a single NLP-based approach to cover all types of pages, or (2) to not use any machine learning approach (no *induction*), e.g. to hand-craft extraction rules.

Here we list three representatives of IE methods applied particularly to semi-structured text. Zhai and Liu [2005] use HTML tag tree alignment to automatically find and segment records and fields in semi-structured web pages without training data or domain knowledge. Embley et al. [1999a] make use of domain ontologies to extract schema and data from web pages in a way that is generally insensitive to changes in page format. McCallum et al. [2000] demonstrate the benefits of one popular statistical sequence model, the maximum entropy Markov model, by segmenting a semi-structured corpus of FAQs.

7.3 Semi-supervised Machine Learning

In the proposed research, we aim to contribute to machine learning (ML) as applied to IE. Supervised ML relies on large amounts of hand-labeled examples to produce a target model, a function that maps instances in some domain to their class labels. More recently, semi-supervised techniques accomplish the same tasks with less hand-labeled training data by taking advantage of unlabeled or partially labeled data. Zhu [2005] provides a general survey of semi-supervised ML. The following semi-supervised techniques are particularly relevant for IE.

Transductive inference (Vapnik [1998]) is the process of inferring something about instances directly from knowledge of other instances. Transduction is in contrast to typical ML which uses an inductive step (learning a general model from instances) followed by a deductive step (inferring instance labels using the general model). Transduction can be more effective because it solves a simpler problem: it makes inferences about selected instances, not about all possible instances. It is common to transfer labels from labeled examples through clusters of unlabeled examples.

Self-training is perhaps the most basic way to use unlabeled data in typical, inductive ML. In self-training, a learning algorithm learns from a small amount of labeled instances, makes predictions about unlabeled instances and then learns from the most confident self-labeled data. Some special cases are equivalent to the popular Expectation-Maximization algorithm (Dempster et al. [1977]).

Co-training (Blum and Mitchell [1998], Yarowsky [1995]) relies on two separate, ideally conditionally independent and sufficient, instance views composed of two distinct sets of features. Using a small set of labeled examples, two classifiers are trained, each on a different view of the data. Confident predictions by one classifier create training data for the other classifier.

Active learning (Settles [2010]), a form of lightly-supervised ML, reduces the cost of labeled training data by automatically selecting the most beneficial instances to be labeled by a human. Most authors view active learning as distinct from, and complementary to, semi-supervised ML. In this regard, semi-supervised learning focuses on leveraging what the learner thinks it knows about the data while active learning focuses on what it does not know.

Transfer learning and domain adaptation (Pan and Yang [2010]) are techniques for learning more efficiently when labeled training data are not perfectly representative of the data on which a trained classifier will be applied.

Looking at IE applications of ML, Nadeau [2007] surveys semi-supervised approaches to named entity recognition, an important kind of IE. Iterative bootstrapping and grammar induction are two important approaches.

Iterative bootstrapping is a semi-supervised technique that is especially relevant to IE and similar NLP tasks. It is a process of iteratively refining both world knowledge (e.g., an ontology) and an extraction model (e.g., a grammar). The term comes from the usual practice of initializing either the ontology or the grammar with a small amount of hand-crafted information. The bootstrapping process imposes a mutual dependence somewhat similar to co-training that allows one type of knowledge to grow as the other grows. Seminal works in bootstrapping include the following: Yarowsky [1995] sense-tags ambiguous words by iteratively learning rules based on (1) neighboring words and (2) the single sense of the ambiguous word used throughout each document. Hearst [1992] and Riloff and Jones [1999] extract relations between entities by iteratively learning rules based on (1) surrounding words and (2) known entities that participate in the target relation.

Grammar induction, also called grammatical inference, is the process of learning a grammar for a language from strings in that language. Grammar induction is typified by unsupervised context-free grammar induction although other learning methods and grammar categories are possible including the wrappers in Kushmerick [1997]. Two important lines of research in grammar induction are the following: Wolff [2003] shows how to induce a grammar using sequence alignment and rule search guided by the principle of minimum description length. Goldwater [2007] induces grammars using non-parametric Bayesian statistical inference. If we consider hidden Markov models (HMMs) as a type of stochastic regular grammar, then we must also include HMM structure learning as a type of grammar induction. The seminal work in this area, Stolcke and Omohundro [1993], starts with an HMM topology that is maximally specific with respect to the training strings and then merges states that improve Bayesian posterior probability.

References

Brad Adelberg. NoDoSE — a tool for semi-automatically extracting structured and semistructured data from text documents. *ACM SIGMOD Record*, 27:283–294, 1998.

Eugene Agichtein and Venkatesh Ganti. Mining reference tables for automatic text segmentation. In *Proceedings of the 2004 ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 20–29, Seattle, Washington, USA, 2004.

Manuel Álvarez, Alberto Pan, Juan Raposo, Fernando Bellas, and Fidel Cacheda. Extracting lists of data records from semi-structured web pages. *Data & Knowledge Engineering*, 64:491–509, 2008.

R. Baumgartner, S. Flesca, and G. Gottlob. Visual web information extraction with lixto. In *Proceedings of the International Conference on Very Large Data Bases*, page 119–128, Rome, Italy, 2001a.

Robert Baumgartner, Sergio Flesca, and Georg Gottlob. Supervised wrapper generation with lixto. In *Proceedings of the International Conference on Very Large Data Bases*, pages 715–716, Rome, Italy, 2001b.

Abdel Belaïd. Retrospective document conversion: application to the library domain. *International Journal on Document Analysis and Recognition*, 1:125–146, 1998.

Abdel Belaïd. Recognition of table of contents for electronic library consulting. *International Journal on Document Analysis and Recognition*, 4:35–45, 2001.

Dominique Besagni and Abdel Belaïd. Citation recognition for scientific publications in digital libraries. In *Proceedings of the First International Workshop on Document Image Analysis for Libraries*, pages 244–252, Palo Alto, California, USA, 2004.

Dominique Besagni, Abdel Belaïd, and Nelly Benet. A segmentation method for bibliographic references by contextual tagging of fields. In *Proceedings of the Seventh International Conference on Document Analysis and Recognition*, pages 384–388, Edinburgh, Scotland, 2003.

Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, pages 92–100, Madison, Wisconsin, USA, 1998.

Vinayak Borkar, Kaustubh Deshmukh, and Sunita Sarawagi. Automatic segmentation of text into structured records. In *Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data*, pages 175–186, Santa Barbara, California, USA, 2001.

Andrew Carlson and Charles Schafer. Bootstrapping information extraction from semi-structured web pages. In *Proceedings of European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, pages 195–210, Antwerp, Belgium, 2008.

Chia-Hui Chang, Chun-Nan Hsu, and Shao-Cheng Lui. Automatic information extraction from semi-structured web pages by pattern discovery. *Decision Support Systems*, 35:129–147, 2003.

Ming-Wei Chang, Lev Ratinov, and Dan Roth. Guiding semi-supervision with constraint-driven learning. In *Proceedings of the Forty-fifth Anual Meeting of the Association of Computational Linguistics*, volume 51, pages 280–287, Antwerp, Belgium, 2007.

Boris Chidlovskii, Jon Ragetli, and Maarten de Rijke. Wrapper generation via grammar induction. In *Proceedings of the Eleventh European Conference on Machine Learning*, pages 96–108, Barcelona, Catalonia, Spain, 2000.

Philipp Cimiano. *Ontology Learning and Population from Text*. 2006.

Valter Crescenzi, Giansalvatore Mecca, and Paolo Merialdo. Roadrunner: Towards automatic data extraction from large web sites. In *Proceedings of the International Conference on Very Large Data Bases*, pages 109–118, Rome, Italy, 2001.

Nilesh Dalvi, Ravi Kumar, and Mohamed Soliman. Automatic wrappers for large scale web extraction. *Proceedings of the VLDB Endowment*, 4:219–230, 2010.

Arthur P. Dempster, Nan M. Laird, and Donal B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39:1–38, 1977.

Hazem Elmeleegy, Jayant Madhavan, and Alon Halevy. Harvesting relational tables from lists on the web. *Proceedings of the VLDB Endowment*, 2:1078–1089, 2009.

David W. Embley and Li Xu. Record location and reconfiguration in unstructured multiple-record web documents. In *Proceedings of the Third International Workshop on the Web and Databases*, pages 123–128, Dallas, Texas, USA, 2000.

David W. Embley, Douglas M. Campbell, Y. S. Jiang, Stephen W. Liddle, Deryle W. Lonsdale, Yiu-Kai Ng, and Randy D. Smith. Conceptual-model-based data extraction from multiple-record web pages. *Data & Knowledge Engineering*, 31:227–251, 1999a.

David W. Embley, Y. S. Jiang, and Yiu-Kai Ng. Record-boundary discovery in web documents. In *Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data*, pages 467–478, Philadelphia, Pennsylvania, USA, 1999b.

David W. Embley, Stephen W. Liddle, and Deryle W. Lonsdale. Conceptual modeling foundations for a web of knowledge. In David W. Embley and Bernhard Thalheim, editors, *Handbook of Conceptual Modeling: Theory, Practice, and Research Challenges*, pages 477–516. Springer, Heidelberg, Germany, 2011.

Oren Etzioni, Michael Cafarella, Doug Downey, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S. Weld, and Alexander Yates. Unsupervised named-entity extraction from the web: An experimental study. *Artificial Intelligence*, 165:91–134, 2005.

Hiromichi Fujisawa. Forty years of research in character and document recognition — an industrial perspective. *Pattern Recognition*, 41:2435–2446, 2008.

C. Lee Giles, Kurt D. Bollacker, and Steve Lawrence. CiteSeer: an automatic citation indexing system. In *Proceedings of the Third ACM Conference on Digital Libraries*, pages 89–98, Pittsburgh, Pennsylvania, USA, 1998.

Sharon J. Goldwater. *Nonparametric Bayesian Models of Lexical Acquisition*. PhD thesis, Brown University, Providence, Rhode Island, 2007.

Trond Grenager, Dan Klein, and Christopher D. Manning. Unsupervised learning of field segmentation models for information extraction. In *Proceedings of the Forty-third Annual Meeting on Association for Computational Linguistics*, pages 371–378, Ann Arbor, Michigan, USA, 2005.

Rahul Gupta and Sunita Sarawagi. Answering table augmentation queries from unstructured lists on the web. *Proceedings of the VLDB Endowment*, 2:289–300, 2009.

Marti A. Hearst. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the Fourteenth International Conference on Computational Linguistics*, pages 539–545, Nantes, France, 1992.

P. Bryan Heidorn and Qin Wei. Automatic metadata extraction from museum specimen labels. In *Proceedings of the 2008 International Conference on Dublin Core and Metadata Applications*, pages 57–68, Berlin, Germany, 2008.

Chun-nan Hsu and Chien-chi Chang. Finite-state transducers for semi-structured text mining. In *Proceedings of IJCAI-99 Workshop on Text Mining: Foundations, Techniques and Applications*, page 38–49, Stockholm, Sweden, 1999.

Yasuto Ishitani. Model-based information extraction method tolerant of OCR errors for document images. In *International Conference on Document Analysis and Recognition*, pages 908–915, Los Alamitos, California, USA, 2001.

Stanley Kok and Wentau Yih. Extracting product information from email receipts using markov logic. In *Proceedings of the Sixth Conference on Email and Anti-Spam*, Mountain View, California, USA, 2009.

Trausti Kristjansson, Aron Culotta, Paul Viola, and Andrew McCallum. Interactive information extraction with constrained conditional random fields. In *Proceedings of the National Conference on Artificial Intelligence*, page 412–418, San Jose, California, USA, 2004.

Karen Kukich. Techniques for automatically correcting words in text. *ACM Computing Surveys*, 24:377–439, 1992.

Nicholas Kushmerick. *Wrapper induction for information extraction*. PhD thesis, University of Washington, Seattle, Washington, USA, 1997.

Alberto H. F. Laender, Berthier A. Ribeiro-Neto, Altigran S. da Silva, and Juliana S. Teixeira. A brief survey of web data extraction tools. *ACM SIGMOD Record*, 31:84–93, 2002.

Steve Lawrence, C. Lee Giles, and Kurt Bollacker. Digital libraries and autonomous citation indexing. *IEEE Computer*, 32:67–71, 1999.

Song Mao, Azriel Rosenfeld, and Tapas Kanungo. Document structure analysis algorithms: A literature survey. In *Proceedings of the SPIE Electronic Imaging Conference*, pages 197–207, Santa Clara, California, USA, 2003.

Andrew McCallum, Dayne Freitag, and Fernando Pereira. Maximum entropy markov models for information extraction and segmentation. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 591–598, Stanford, California, USA, 2000.

Andrew Kachites McCallum. MALLET: a machine learning for language toolkit. http://mallet.cs.umass.edu/, 2002. URL `http://mallet.cs.umass.edu/`.

Matthew Michelson and Craig A. Knoblock. Creating relational data from unstructured and ungrammatical data sources. *Journal of Artificial Intelligence Research*, 31:543–590, 2008.

Shunji Mori, Ching Y. Suen, and Kazuhiko Yamamoto. Historical review of OCR research and development. *Proceedings of the IEEE*, 80:1029–1058, 1992.

I. Muslea, S. Minton, and C. Knoblock. Stalker: Learning extraction rules for semistructured, web-based information sources. In *Proceedings of AAAI-98 Workshop on AI and Information Integration*, page 74–81, 1998. URL `http://www.aaai.org/Papers/Workshops/1998/WS-98-14/WS98-14-011.pdf`.

David Nadeau. *Semi-Supervised Named Entity Recognition: Learning to Recognize 100 Entity Types with Little Supervision*. Thesis, University of Ottawa, Ottawa, Canada, 2007.

George Nagy. Twenty years of document image analysis in PAMI. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:38–62, 2000.

Thomas L. Packer, Joshua F. Lutes, Aaron P. Stewart, David W. Embley, Eric K. Ringger, Kevin D. Seppi, and Lee S. Jensen. Extracting person names from diverse and noisy OCR text. In *Proceedings of the CIKM Workshop on the Analysis of Noisy Documents*, pages 19–26, Toronto, Ontario, Canada, 2010.

Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22:1345–1359, 2010.

Fuchun Peng and Andrew McCallum. Accurate information extraction from research papers using conditional random fields. In *In HLT-NAACL*, pages 329–336, Boston, Massachusetts, USA, 2004.

David Pinto, Andrew McCallum, Xing Wei, and W. Bruce Croft. Table extraction using conditional random fields. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 235–242, Toronto, Canada, 2003. ACM.

Ellen Riloff and Rosie Jones. Learning dictionaries for information extraction by multi-level bootstrapping. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence*, pages 474–479, Orlando, Florida, USA, 1999.

Sunita Sarawagi. Information extraction. *Foundations and Trends in Databases*, 1:261–377, 2008.

Burr Settles. Active learning literature survey. Technical report, University of Wisconsin-Madison, 2010.

Andreas Stolcke and Stephen Omohundro. Hidden markov model induction by bayesian model merging. *Advances in Neural Information Processing Systems*, pages 11–18, 1993.

Charles Sutton and Andrew McCallum. An introduction to conditional random fields. Technical report, 2010.

Cui Tao, David W. Embley, and David W. Liddle. FOCIH: form-based ontology creation and information harvesting. In *Proceedings of the Twenty-eighth International Conference on Conceptual Modeling*, pages 346–359, Gramado, Brazil, 2009.

Jordi Turmo, Alicia Ageno, and Neus Català. Adaptive information extraction. *ACM Computing Surveys*, 38:1–47, 2006.

Vladimir N. Vapnik. *Statistical Learning Theory*. Wiley New York, 1998.

J. Gerard Wolff. Information compression by multiple alignment, unification and search as a unifying principle in computing and cognition. *Artificial Intelligence Review*, 19:193–230, 2003.

David Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the Thirty-third Annual Meeting of the Association for Computational Linguistics*, pages 189–196, Cambridge, Massachusetts, USA, 1995.

Yanhong Zhai and Bing Liu. Web data extraction based on partial tree alignment. In *Proceedings of the 14th International Conference on World Wide Web*, pages 76–85, Chiba, Japan, 2005.

Guangyu Zhu, Timothy J. Bethea, and Vikas Krishna. Extracting relevant named entities for automated expense reimbursement. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1004–1012, San Jose, California, USA, 2007.

Xiaojin Zhu. *Semi-supervised Learning with Graphs*. PhD thesis, Carnegie Mellon University, 2005.