2013 IEEE 25th International Conference on Tools with Artificial Intelligence

# Semi-automatic Dictionary Curation for Domain-specific Ontologies

Ashish Kulkarni
Indian Institute of Technology, Bombay
Email: kulashish@gmail.com

Chetana Gavankar
Indian Institute of Technology, Bombay
Email: chetanagavankar@gmail.com

Ganesh Ramakrishnan
Indian Institute of Technology, Bombay
Email: ganesh@cse.iitb.ac.in

Sriram Raghavan
IBM Research Lab, India
Email: sriramraghavan@in.ibm.com

***Abstract*—Within the broad area of information extraction, we study the problem of effective dictionary curation in an enterprise setting. Equipped with an ontology, representative of the domain of an enterprise, our approach populates the attributes of leaf nodes of the ontology with instances extracted from the enterprise corpus. For an attribute of interest, given a few seed examples or indicative features for the attribute, we first obtain a ranked list of 'list pages' potentially containing additional dictionary terms. Our ranking model ranks pages from the enterprise corpus based on their 'list' content using several visual and lexical features. We gather users' judgement of the result pages and the model continuously learns from this feedback. We compare different techniques of dictionary curation using rule based extractors and visual features of pages. Based on rule writing exercise, we show the benefit of dictionaries for leaf node attributes, in writing rule based extractors for higher level nodes in an ontology. We have implemented a dictionary curation system based on these ideas. Experimental analysis using academic domain ontology and universities corpora, reveal (in the context of enterprise analytics) (i) the merit of dictionary support in rule based information extraction (ii) the viability and effectiveness of an interactive approach for dictionary creation.**

## I. Introduction

Information search and retrieval today is enriched by the recognition of named entities and relations between them. This has motivated a lot of work [1], [2], [3], [4], [5] in open domain web-scale information extraction. Enterprise search has its own set of unique challenges [6]. Hawking [7], [8] identifies enterprise information complexity as one of the main reasons for poor enterprise search and highlights several research and engineering challenges. It has also been recognized [9], [10], [11], [6], [12], [13] that domain specific IE, to identify and extract entities and concepts relevant to an enterprise, is important for high quality enterprise search.

In this paper we focus on feature engineering, machine learning and algorithmic aspects of enterprise domain information extraction with the immediate end of populating attributes of the leaf level concepts of an academic ontology. This might in turn help in writing precise annotators for higher level concepts in an ontology. In the academic domain for instance, the availability of a dictionary of *professor names* could potentially aid in devising accurate signature for the *professor* concept. A list of such instances is often available on a small and finite set of pages in a corpus. We propose an interactive semi-automated solution to first retrieve 'list pages' from a corpus and then to extract the dictionary of instances for a target concept. We also compare two approaches to extraction - one that is rule based and the other that generalizes over visual features of list pages. With little human assistance, our system is able to curate dictionaries, thereby proving the effectiveness of such an interactive approach in the extractor writing process.

## II. Related Work

There are three broad areas related to our work. We summarize the prior work in each of these areas and relate it with our focus and approach. **Ontology learning and population:** This involves building and populating an ontology from structured, semi-structured and unstructured text. There is a large body of work in ontology population [14], [15], [16] that uses frequency based term extraction along with shallow NLP techniques. Song *et. al.* [17] propose automatic ontology population using web tables. The method computes the topological similarity between the ontology and the web tables by a parse tree kernel since both an ontology and a web table can be expressed as parse trees. The input to their system is web tables obtained through focused web crawl. Given an ontology, specific to the domain of an enterprise, we consider our work as a novel approach to populating attributes of leaf nodes of the ontology.

**Open domain information extraction:** This area concerns itself with web scale extraction of entities and their relations. To the best of our knowledge, most of the techniques in this area exploit the redundancy of information on the web. Systems like Open Information Extraction [18], [19] and Snowball [20] apply NLP based techniques on unstructured text. The approach in Open IE makes use of entity tagging and parse trees of sentences to extract entity relations. The Snowball system follows a bootstrapping approach to extract relations that are similar to the seed examples. Our extraction process partly borrows from this bootstrapping technique. Web Sets [21] presents an approach to open domain information extraction from HTML documents. Their system exploits overlap of content across tables. However, enterprise data usually does not have the luxury of highly redundant data exploited in the above approaches. Thus extraction from enterprise data usually needs the use of high precision features.

**Vision based information extraction:** Vision based page

1082-3409/13 $31.00 © 2013 IEEE
DOI 10.1109/ICTAI.2013.112

segmentation [22] (VIPS) algorithm extracts the content structure of a web page. Page segmentation is done by combining the DOM structure and visual cues like font color, position and separator lines. The defined set of rules are used to judge which elements of a page form visual blocks. Visualized Element Nodes Table extraction (VENTex) [23] uses visual features similar to VIPS. VENTex describes a set of extraction rules and heuristics to extract tables from web pages. Weninger *et. al.* [24] implemented a naive list extraction method which explores the visual alignment of objects in a web page. They define web list to be any set of sibling boxes which are visually aligned on a rendered web page. While these approaches come close to ours, we differ primarily in two aspects. Firstly, we go beyond alignment and use a combination of several other visual and lexical features. Secondly, instead of making hard judgments, we use a learning to rank approach to learn a ranking model based on relevance feedback.

Most of the prior work primarily focuses on information extraction from the Internet. In our current contribution, we study dynamics of this problem in enterprises (or organizations). We show the relevance of dictionaries for attributes of leaf nodes, in writing rule based annotators for higher level nodes in an ontology. We focus on the problem of curating dictionaries from an enterprise corpus. Towards that we propose a semi supervised approach that takes as input a combination of keywords and a seed list of instances for a concept, to first retrieve a ranked list of 'list pages' from corpus and then extract more instances from these pages. To identify potential list pages, we exploit several visual and lexical features and learn a ranking model that continuously updates based on user relevance feedback. We then employ two different techniques to extract instances from the list pages - one that uses rule based annotators and the other that exploits visual features of these pages and their structure.

The paper is organized as follows. In Section III we discuss the ontology population process using rule based annotators and relevance of dictionaries. Sections IV and V explain our list identification and extraction approach. We illustrate the evaluation of our approach in section VI followed by conclusions in section VII.

## III. Preliminaries

An *ontology* describes entities in a domain and their inter-relationship. We focused on academic domain and built over existing *Benchmark ontology*[1] and *Aisso*[2] ontologies. Ontologies are merged using the *Protege*[3] ontology editor and extended to include several classes like *award*, *project*, etc. and attributes like *professor* has *research-area*, *course* has *prerequisite*,, etc. In addition, we scraped the lists available in Wikipedia to populate class hierarchy rooted at the *concept* class. The ontology[4] that we finally used consists of more than 190 classes, 150 object properties and 150 data properties.

*Ontology population* primarily concerns with the identification of instances and their mapping to classes and their attributes in an ontology. IE and ontology population are closely related in that they share a common goal, namely to enrich a knowledge base with new instances [25]. Various information extraction techniques have been proposed that transform unstructured or semi-structured text to class-instance data. Declarative IE systems like [26], [27] propose to build high precision annotators for every entity to be extracted. When mapped to the ontology population task, this amounts to writing annotators for every node in an ontology. However, given an ontology, it is often not clear where and how to start writing annotators. This can be a tedious and complex task where the complexities arise from interdependencies amongst the concepts and ease (or the lack there of) of writing annotators for a concept before another. With the aim of understanding human judgment behind annotator writing and their ordering, we performed a manual exercise where we analyzed the rule writing process for higher level nodes and their leaf nodes. If a higher order concept has a very precise and obvious signature, then one would rather write that annotator first and perhaps use its output to help write lower-level annotators. An *address* annotator for instance, might aid a *PIN number* annotator in precise extraction of PIN numbers. On the other hand, if such an obvious signature and/or rules are not present, then the composition approach of extracting the properties and then combining them to high order concepts seems easier.

One of the key observations from this exercise was the need for dictionaries. In bottom-up approach, the availability of dictionaries for attributes of leaf concepts in an ontology could help in writing accurate extractors for higher level concepts. A *professor* annotator for instance, will benefit from the availability of dictionaries for *professor-name*, *department-name*, and *course-name*. We describe here the *professor* annotator implemented using AQL [28] to illustrate our point. Assuming that the professor node can be populated using information on professor homepages, we first use a simple regular expression based extractor that looks for occurrence of the *homepage* word. We filter professor homepages using the heuristic that the first name appearing on the homepage is that of its owner. We then use the dictionary of professor names to extract professor names using the AQL `extract` and `dictionary` constructs. The relational operator `select` was used along with `combine spans` construct to identify complete occurrence of professor name entity. The `union all` construct was then used to find combinations of names like *Gene Franklin*, *Gene F.*, *G.Franklin*, and *Franklin Gene*.

Professor's research area was extracted using the *research area* annotator. Here we exploit contextual phrases like "research area", "research interest", "area of interest", *etc.* We can extract the courses taught by the professor, using the dictionary of *course name* and *course id*. We demonstrate in figure 1, the annotations extracted after running the annotator on a sample homepage. Similarly, the dictionary of attributes like *professor-email*, *department-name*, and *phone-number* are also extracted using lexical, contextual and regular expression based features. We can then use these dictionaries to write annotator for the *professor* node.

Creating dictionaries can be a highly time consuming task. In this work we deal with the problem of curating dictionaries and propose an interactive semi-automatic approach that meets the high precision requirement of an enterprise IE system. This

[1] http://swat.cse.lehigh.edu/onto/univ-bench.owl
[2] http://vocab.org/aiiso/schema
[3] http://protege.stanford.edu
[4] available in the left pane of our system


(a) Professor Homepage

```
http://ai.stanford.edu/~ang/index.html

Output View ProfessorName:
5 -14 :Andrew Ng

Output View Profhomepage:
17 -25 :homepage

Output View ProfessorDepartment:
258 -275 :Computer Science

Output View ResearchArea:
393 -524 :
:Artificial Intelligence, Machine learning, Unsupervised feature learning
 and Deep learning, Neuroscience-informed AI, Robotics
```

(b) Professor research area

Fig. 1: Professor homepage annotation

can also be looked at as a method to ontology population limited to the attributes of leaf nodes of a domain ontology. Our approach is based on the thesis that an enterprise-wide corpus often contains list pages that could be exploited for dictionary population. Thus we first identify potential list pages from the corpus and then propose two techniques for extracting instances from these pages.

## IV. Identification of List Pages

A domain corpus is often replete with 'list pages'. In our academic corpus for instance, there are list pages containing list of professor names, course ids, projects, events, research labs and several others. Each of these correspond to an attribute of a leaf node in the academic ontology. The problem here is to locate these list documents for the ontology node of interest. Here we leverage the bootstrapping and learning to rank [29] paradigms in an interactive setting. Posed as a learning to rank problem, the aim is to construct a ranking model that ranks list pages before others. We created the training data as follows. Equipped with the ontology, we seek for search queries in the form of keywords and/or a seed list of instances for the node of interest. The tf-idf feature of these terms (using Lucene) in the corpus is used to obtain a partial ordering over the result set of documents. For each item in the set, we solicit a binary relevance judgement from users to indicate whether or not it is a list document. This feedback is used to obtain pairwise preference (or ranking) constraints as described in [29]. This is then used as the training data to train a rankSVM[5] model. At the time of testing, we use this ranking model to rank the documents in a result set. Next, we describe the features used by our ranking model.

### A. Features used by the ranking model

We studied list pages of various leaf level ontology node attributes like *course id, course name, professor name, event,* and *department name* and for corpora spread across many universities. We made several observations that were consistent across nodes and the corpora.

- List items are usually aligned vertically on a page
- While there might be other lists (for menu items *etc.*), the list of interest occupies the major screen real estate
- Each list item is bounded in a well defined block
- The text accompanying each list item consists of short sentences usually limited to some keywords
- Number of items in a list is usually 10 or more
- Lists longer than 10, are often paginated
- The label of the ontology node (or its synonym) that the list instantiates, often appears in the page URL, title or content

Inspired by our observations, we identified several document features that are broadly classified as either visual [30] or lexical (Refer table I). The computation of values for visual features is based on the page rendering by an open source rendering engine called *CSSBox*[6]. CSSBox loads the page HTML code along with its CSS style sheets to create a tree of *boxes*. A box is a rectangular area on a rendered page roughly mapping to each DOM element, text strings and other objects on that page. We traversed the tree and extracted features that include: number of sub-boxes in each box, CSS class of each box, coordinates and area occupied by a box, and text(if the box is a textbox). For the binary features, we then applied appropriate thresholds arrived at based on our observations with several list documents. *E.g.*, we use a lower bound of 10 for number of items in a list. That corresponds to a box with 10 or more sub-boxes. For computation of lexical features, we consider match between the keywords (after their wordnet expansion to include synonyms) with document title, URL and content. In addition, we also consider the size of list items and the presence of stop-words around list items.

We then used the learning to rank approach as described earlier to train a ranking model based on relevance judgments. Our system presents a continuous learning setting where the model keeps updating itself based on new relevance feedback.

### B. List search system

Figure 2 captures the design of our *List search system*. The system presents a user with an interface (refer figure 3) that shows the academic ontology in the left pane and a search textbox in the right. The user browses through the ontology to select a target node and provides in the search box, a short seed list of instances for an attribute of interest. This input is converted into a fielded query that is fired against a backend index. Results are ranked using our ranking model before presenting them to the user. The top 20 results are

[5]http://www.cs.cornell.edu/people/tj/svm_light/svm_rank.html

[6]http://cssbox.sourceforge.net/

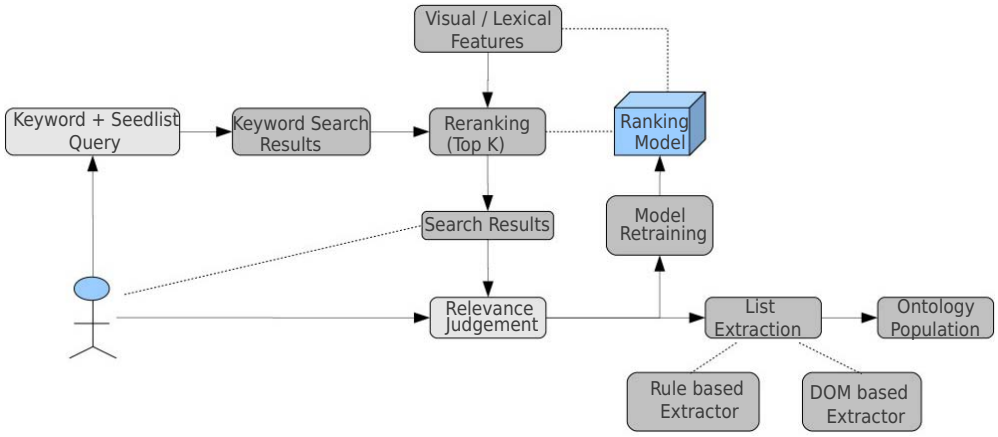



Fig. 2: List Extraction Flow

TABLE I: Features used by the ranking model

| | |
|---|---|
| Visual | Number of items in the list more than some threshold |
| | Ratio of area of list box to that of the viewport greater than some threshold |
| | Is the document paginated? |
| | Is the width of each item box in the list same? |
| | Are the list items aligned? |
| | Is the HTML tag-path of list items the same? |
| | Do the list item boxes belong to the same css class? |
| Lexical | Match between the query keywords (wordnet expanded) and the document title |
| | Match between the query keywords (wordnet expanded) and the document content |
| | Match between the query keywords (wordnet expanded) and the document URL |
| | Number of words in a list item |
| | Number of stop-words in a list item |
| | Seed word in separate list item |

presented to the user with a checkbox alongside to gather binary relevance. The relevance judgments are then used to re-train our ranking model. Model re-training is an expensive operation with its cost proportional to the number of ranking constraints present in the training data. Thus we prefer re-training after gathering relevance judgements for a batch of queries. Initial size of this batch for the first model training and the step size for subsequent training can be configured through the configuration page in the system. Setting both the sizes to 1 amounts to model re-training after every relevance gathering step.

The results page also contains a link that allows users to view the list items extracted from a document. This serves as evidence to a user while making a relevance judgement. More importantly, for documents finally marked as relevant by the user, the corresponding extractions are promoted to the dictionary for that node attribute in the ontology. Currently the system does not do any post processing like de-duplication that might be required before the items are promoted. Next section describes our list extraction approaches.

## V. List extraction

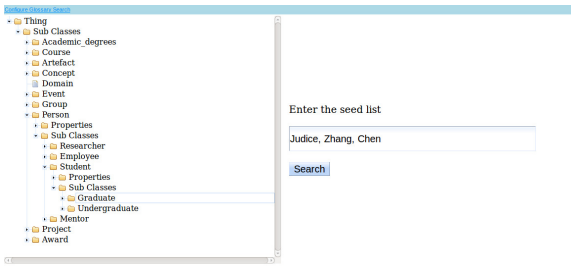
Having identified the list documents, the next task is to extract the list items from these. Towards this, we propose two techniques - one that exploits rule based annotators and other that generalizes the DOM path feature of the list items.

### A. List extraction using Rule Based Annotators

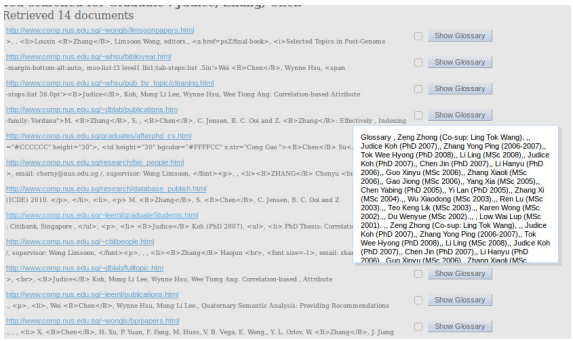
We had motivated the relevance of dictionaries by highlighting their use in writing high precision annotators for non-leaf nodes in an ontology. However these dictionaries themselves could be created using rule based annotators for leaf-level nodes. In general this could be a highly tedious task perhaps requiring several person-days of effort to come up with a precise collection of rules to compose an entity annotator. The complexity arises from the need to identify accurate signals for the node of interest. By locating potential list pages, our list search system identifies the subset of documents in the corpus that is rich in signals for a target concept attribute. A relatively simpler rule based annotator exploiting these signals can now extract instances with high precision. Thus our system meets the precision requirement by limiting the collection of documents on which annotators are executed. For example, a simple regular expression based named entity annotator might result in a lot of noise when run over all documents in a corpus, while it is intuitive that the same annotator might be highly accurate if run on a list page of, say, *student names*. We support this argument empirically in the experiments section.

### B. List extraction using DOM path

In a completely automated approach, we studied how well the visual features generalize in the list extraction process. List items usually follow a consistent visual placement on a document that is reflected by the underlying markup language. This placement could be obtained using our visual features along with DOM path. Specifically, given a seed list of instances and list documents identified using our interactive system, we trace the document DOM path of the seed instances on those documents. We then use that DOM path to further extract other instances from the list page. The approach works very well achieving close to perfect precision and recall especially in vertically aligned lists.




(a) Select ontology node and provide seed-list query




(b) Extracted list items

Fig. 3: List Search System

## VI. Experiments and Results

We generated our experimental corpus by crawling following university websites - MIT, Stanford University, Indian Institute of Technology Bombay (IITB), National University of Singapore (NUS), and Monash University - spanning different geographies. The evaluation is two pronged. We evaluate our ranking model for identification of list pages. We also evaluate the two list extraction techniques - one that uses hand crafted rule based annotators and the other that generalizes over the document DOM path feature.

TABLE II: Evaluation of list document identification: The model was trained using the corpus of over 400,000 documents obtained from three universities with relevance feedback resulting in about 50,000 ranking constraints. It was tested on the training corpus and on a test corpus of over 300,000 documents

| | Train | | Test | | Across Nodes | |
|---|---|---|---|---|---|---|
| | MAP | NDCG | MAP | NDCG | MAP | NDCG |
| Baseline (Lucene) | 0.314 | 0.537 | 0.42 | 0.58 | 0.398 | 0.54 |
| Lexical | 0.53 | 0.639 | 0.75 | 0.82 | 0.492 | 0.61 |
| Visual | 0.477 | 0.617 | 0.63 | 0.76 | 0.6 | 0.73 |
| Lexical + Visual | 0.693 | 0.775 | 0.73 | 0.81 | 0.75 | 0.81 |

### A. Identification of list documents

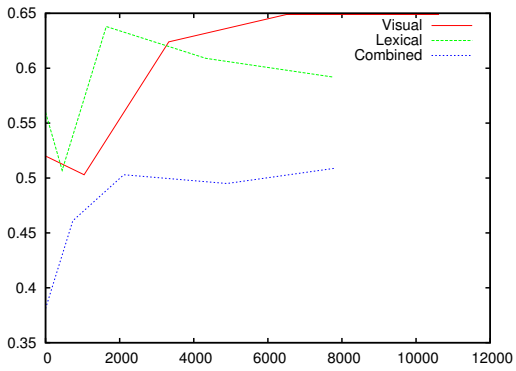
We trained a RankSVM model using the visual and lexical features to search and rank list pages from the university corpus. The model was trained over about 50,000 ranking constraints obtained by collecting binary relevance feedback from users. The training corpus consisted of over 400,000 documents crawled from three different universities - IITB, Stanford and Monash. Testing was done first on the same corpus and then on a corpus of about 300,000 documents crawled from two other universities - NUS and MIT. Top 20 results were shown to users to gather their judgement. The experiment was repeated first using the default Lucene ranking and then with the SVMRank models trained on different combinations of our features. We computed the MAP and NDCG metric while using the default Lucene ranking as baseline. The experiment results are reported in table II (Refer Train and Test results). Our ranking model shows significant improvement over the baseline. The results are consistent across train and test suggesting that our model with its visual and lexical features generalizes well across the university corpora.

*Feature generalization across nodes:* We performed another evaluation to check if the features generalize well across the ontology nodes. We trained our model by gathering relevance feedback on results for a specific set of leaf node attributes and then used an independent set of attributes for testing. Results for *professor name*, *student name*, *course id*, *course name*, *department name*, and *staff name* were used for training and the test set comprised of *event*, *project*, and *award* attributes. The results are documented in table II (Refer Across Nodes column). Again, the features were shown to generalize well across the node attributes. This aligns with our expectation as the visual features exploit the visual placement of instances and are independent of ontology nodes and their attributes. In some cases, the lexical features were observed to be affected by noisy URLs and titles where the words or their wordnet expansions do not overlap with the query keywords. In all our experiments, we used the label of the ontology node as the query keyword for computing the lexical features.

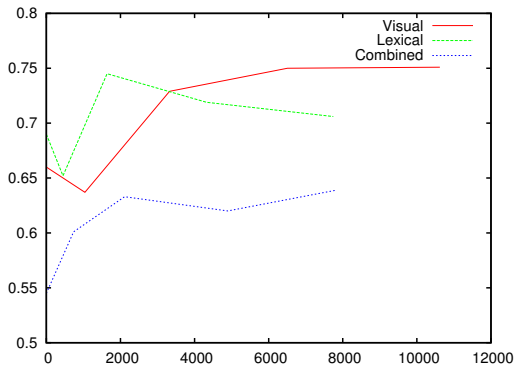
*Learning rate of features:* Our system provided a continuous learning setting where our RankSVM model continuously learned based on relevance judgments by users. We initially trained the model using the training data obtained from the user feedback on results for a batch of 5 queries. We then re-trained the model after every subsequent batch of 5 queries. At every training step we recorded the number of ranking constraints used for training and computed the MAP and NDCG for results obtained using that model (Refer figure 4). The absolute MAP and NDCG numbers vary due to the use of different queries in the experiments with visual, lexical and combined features. However the rate of learning reveals their relative weight in our ranking model. The visual features show a strong distinguishing ability and the model learns well with about 12-15 thousand ranking constraints. The lexical features are relatively slow in learning and were observed to suffer due to noise in page titles and URLs.

### B. List instances extraction techniques

Our interactive system with the ranking model along with user judgements helps in identifying the list pages. We then used two different techniques to extract list items from these pages. In one case, we ran our rule based annotators (for various leaf node attributes) on these pages.




(a) MAP plotted against number of ranking constraints




(b) NDCG plotted against number of ranking constraints

Fig. 4: Rate of learning of visual and lexical features: Initial training size = 5 queries, Step size = 5 queries.

TABLE IV: Evaluation of rule based list extraction from list and non-list pages

| | List pages | | | | | | Non-list pages | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Micro | | | Macro | | | Micro | | | Macro | | |
| | P | R | F | P | R | F | P | R | F | P | R | F |
| IITB | | | | | | | | | | | | |
| Professor Name | 1 | .93 | .96 | 1 | .94 | .97 | .03 | 1 | .58 | .09 | 1 | .17 |
| Research Area | .97 | .97 | .97 | .98 | .98 | .98 | 1 | 1 | 1 | 1 | 1 | 1 |
| Professor Email | 1 | 1 | 1 | 1 | 1 | 1 | .16 | 1 | .28 | .20 | 1 | .33 |
| Department Name | .94 | 1 | .97 | .94 | 1 | .97 | 0 | 0 | 0 | 0 | 0 | 0 |
| Academic Degree | .64 | .7 | .67 | .64 | .7 | .67 | .40 | 1 | .57 | .47 | 1 | .62 |
| Event | 1 | .95 | .97 | 1 | .96 | .98 | 0 | 0 | 0 | 0 | 0 | 0 |
| Research Lab | .86 | 1 | .92 | .87 | 1 | .93 | .45 | 1 | .87 | .47 | 1 | .89 |
| Course ID | 1 | 1 | 1 | 1 | 1 | 1 | .50 | 1 | .67 | .50 | 1 | .67 |
| Course Name | 1 | 1 | 1 | 1 | 1 | 1 | .50 | 1 | .67 | .50 | 1 | .67 |
| Student Name | .78 | 1 | .87 | .80 | 1 | .89 | .20 | .96 | .33 | .26 | .97 | .4 |
| Student Email | 1 | .99 | .99 | 1 | .99 | .99 | .69 | 1 | .81 | .46 | 1 | .56 |
| Stanford | | | | | | | | | | | | |
| Professor Name | .86 | 1 | .92 | .87 | 1 | .93 | .15 | .62 | .22 | .25 | .87 | .39 |
| Professor Email | 1 | 1 | 1 | 1 | 1 | 1 | .6 | 1 | .67 | .33 | 1 | .49 |
| Department Name | .87 | 1 | .93 | .87 | 1 | .93 | 0 | 0 | 0 | 0 | 0 | 0 |
| Event | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Research Lab | .75 | .92 | .83 | .80 | .95 | .87 | .06 | .83 | .1 | .03 | .75 | .06 |
| Course ID | .89 | 1 | .94 | .84 | 1 | .91 | .58 | 1 | .71 | .47 | 1 | .64 |
| Course Name | .89 | 1 | .94 | .84 | 1 | .91 | .46 | 1 | .55 | .27 | 1 | .42 |
| Student Name | .90 | 1 | .94 | .90 | 1 | .94 | .12 | 1 | .21 | .11 | 1 | .21 |
| Monash | | | | | | | | | | | | |
| Professor Name | .80 | 1 | .89 | .80 | 1 | .89 | .28 | 1 | .44 | .30 | 1 | .36 |
| Department Name | .87 | 1 | .93 | .87 | 1 | .93 | 0 | 0 | 0 | 0 | 0 | 0 |
| Event | 1 | 1 | 1 | 1 | 1 | 1 | .42 | 1 | .58 | .4 | 1 | .57 |
| Research Lab | .75 | .92 | .83 | .80 | .95 | .87 | .02 | .67 | .04 | .01 | .5 | .02 |
| Course ID | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Student Name | 1 | .97 | .98 | 1 | .97 | .98 | .08 | 1 | .14 | .08 | 1 | .14 |
| MIT | | | | | | | | | | | | |
| Professor Name | .89 | .96 | .92 | .85 | .94 | .89 | .33 | 1 | .5 | .33 | 1 | .49 |
| Event | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Academic Degree | .71 | .91 | .79 | .83 | .94 | .88 | .6 | .61 | .6 | .3 | .27 | .31 |
| Course ID | 1 | 1 | 1 | 1 | 1 | 1 | .5 | 1 | .66 | .21 | 1 | .34 |
| Department Name | .79 | 1 | .88 | .78 | 1 | .87 | 0 | 0 | 0 | 0 | 0 | 0 |
| Research Lab | .89 | .87 | .88 | .64 | .57 | .56 | - | - | - | - | - | - |
| Student Name | .71 | .85 | .78 | .75 | .89 | .81 | .57 | 1 | .66 | .22 | 1 | .36 |
| Course Name | .93 | .75 | .83 | .86 | .74 | .79 | .18 | 1 | .3 | .36 | 1 | .38 |
| Student Email | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Research Area | 1 | .96 | .98 | 1 | .87 | .93 | - | - | - | - | - | - |
| Professor Email | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| NUS | | | | | | | | | | | | |
| Professor Name | .94 | .71 | .81 | .94 | .71 | .81 | .2 | 1 | .39 | .13 | 1 | .23 |
| Student Name | .95 | .94 | .95 | .94 | .94 | .94 | .11 | 1 | .2 | .1 | 1 | .2 |
| Academic Degree | .93 | .87 | .89 | .95 | .89 | .92 | .58 | 1 | .73 | .6 | 1 | .75 |
| Course ID | 1 | 1 | 1 | 1 | 1 | 1 | .57 | 1 | .72 | .57 | 1 | .72 |
| Research Lab | 0.19 | 1 | .32 | .27 | 1 | .37 | - | - | - | - | - | - |
| Student Email | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Course Name | 1 | 1 | 1 | 1 | 1 | 1 | .57 | 1 | .72 | .57 | 1 | .72 |

TABLE III: Comparison of list extraction techniques

| | Micro | | | Macro | | |
|---|---|---|---|---|---|---|
| | P | R | F | P | R | F |
| IITB | | | | | | |
| SEAL | 1 | 0.79 | 0.89 | 1 | 0.58 | 0.73 |
| Our Method (AQL) | 0.94 | 0.99 | 0.96 | 0.95 | 0.99 | 0.97 |
| Our Method (DOM) | 0.98 | 0.89 | 0.93 | 0.97 | 0.96 | 0.96 |
| Stanford | | | | | | |
| SEAL | 1 | 0.21 | 0.35 | 1 | 0.25 | 0.4 |
| Our Method (AQL) | 0.88 | 0.98 | 0.94 | 0.77 | 0.99 | 0.87 |
| Our Method (DOM) | 0.97 | 0.90 | 0.93 | 0.96 | 0.94 | 0.95 |
| Monash | | | | | | |
| SEAL | 1 | 0.75 | 0.86 | 1 | 0.56 | 0.72 |
| Our Method (AQL) | 0.96 | 0.99 | 0.97 | 0.93 | 0.99 | 0.96 |
| Our Method (DOM) | 0.96 | 1 | 0.98 | 0.88 | 1 | 0.94 |
| MIT | | | | | | |
| SEAL | 0.85 | 0.66 | 0.74 | 0.91 | 0.59 | 0.71 |
| Our Method (AQL) | 0.88 | 0.96 | 0.92 | 0.89 | 0.97 | 0.9 |
| Our Method (DOM) | 0.68 | 0.96 | 0.8 | 0.8 | 0.96 | 0.87 |
| NUS | | | | | | |
| SEAL | 0.57 | 0.27 | 0.36 | 0.74 | 0.63 | 0.68 |
| Our Method (AQL) | 0.8 | 0.95 | 0.87 | 0.81 | 0.93 | 0.86 |
| Our Method (DOM) | 0.99 | 1 | 0.99 | 0.99 | 1 | 0.99 |

*Rule based list extraction from list and non-list pages:* The results in table IV show that highly precise extraction can be achieved from list pages that primarily contain instances of an attribute of an ontology leaf node. Instances of nodes that have a well defined signature like *email*, and *course-id* show close to $100\%$ extraction accuracy. Others like *department-name*, *events*, and *research-lab* that exploit document features like page title, and contextual phrases, *etc.* also achieve high precision when run on list pages. The approach achieved relatively low precision on few nodes like *academic-degrees* where the features did not generalize well even on list pages. The extractors for *person-name* and *research-area* additionally make use of negative word dictionaries comprising of common nouns, articles and conjunctions. To check how poorly the extractors can perform without the knowledge of list pages, we executed the annotators on non-list pages. The precision was nearly $50\%$ lower with some annotators returning no results due to absence of document level features. Other annotators that relied on contextual clues also suffered from highly noisy extractions.

*List extraction using DOM paths:* We verified how well the visual placement of list items generalizes across list pages.The use of DOM path is motivated by the observation that all list items usually follow a similar DOM path within a document. We reused the potential list boxes as identified by cssbox and traced DOM paths within these that led to the seed list instances. We then extracted other instances by following the same path within these list boxes. We compared the extraction results with ground truth obtained manually to compute *micro* and *macro* averaged precision (P), recall (R) and fscore (F). While *macro*-average averages the per-document metrics over the complete set, *micro*-average sums up the per-document statistics and then computes effective measures. Results are documented in table III. Unlike the rule based approach, this approach is purely based on visual features and thus independent of any node dependent features. We therefore report results over the set of all list pages across various node attributes. The results for rule based extractors is similarly micro and macro averaged across nodes. Our evaluation on corpus of different universities shows that this approach is able to extract list items with precision that is comparable to that of the hand crafted rule based annotators. The precision is only affected by the presence of non-list elements like list headers *etc.* along the same DOM path. The recall on the other hand varies significantly as list items in a document might be visually spread across multiple lists either in more than one columns or divided alphabetically or in a tiled fashion. In such cases, the DOM path for instances from the seed list only covers other instances in the same list and fails to generalize. The results reported here are for the case when seed list is representative of all sub-lists in the document. As expected, the recall improved and was close to $92\%$.

*Comparison with other systems:* SEAL (Set Expander for Any Language) [31] is a set expansion system that takes as input a few seed instances of a target concept and then discovers other similar instances from semi-structured documents like web pages. We implemented SEAL and used it to populate the leaf node attributes from our academic ontology. Surprisingly, SEAL did not return any results for any of the seed instances that we tried. We then gave SEAL the benefit of knowing the list pages in the corpus and used it to extract instances from individual list page URLs. We report the results in table III[7]. While SEAL achieves $100\%$ precision on most of the list pages, its recall is significantly lower than that of both our extraction techniques. Our system consistently performed better than SEAL on all the universities corpora.

## VII. Conclusion

Inspired by the prior research on information extraction at web scale, in this paper we studied aspects of the dynamics of information extraction in an enterprise (or organization). In the context of ontology population, we motivated the relevance of dictionaries for leaf node attributes, in writing annotators for the higher level nodes. We showed that a combination of visual and lexical features is indeed capable of identifying list documents from a corpus with high precision. We proposed two techniques for list extraction from these documents - one based on rule based annotators and the other with little manual intervention exploiting the DOM structure of a document. The semi-automated approach showed good precision and we made several observations motivating further generalized rule induction. Our interactive approach shows that it is indeed a viable and an effective way for information extraction in the context of enterprise analytics.

## References


[1] S. Chakrabarti, S. Kasturi, B. Balakrishnan, G. Ramakrishnan, and R. Saraf, "Compressed data structures for annotated web search," in *Proceedings of the 21st international conference on World Wide Web*, ser. WWW '12. New York, NY, USA: ACM, 2012, pp. 121–130. [Online]. Available: http://doi.acm.org/10.1145/2187836.2187854

[2] T. Cheng, X. Yan, and K. C.-C. Chang, "Entityrank: searching entities directly and holistically," in *Proceedings of the 33rd international conference on Very large data bases*, ser. VLDB '07. VLDB Endowment, 2007, pp. 387–398. [Online]. Available: http://dl.acm.org/citation.cfm?id=1325851.1325898

[3] G. Kasneci, F. M. Suchanek, G. Ifrim, S. Elbassuoni, M. Ramanath, and G. Weikum, "Naga: harvesting, searching and ranking knowledge," in *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, ser. SIGMOD '08. New York, NY, USA: ACM, 2008, pp. 1285–1288. [Online]. Available: http://doi.acm.org/10.1145/1376616.1376756


[7]SEAL failed to extract instances even from some of these list pages

[4] X. Li, C. Li, and C. Yu, "Entityengine: answering entity-relationship queries using shallow semantics," in *Proceedings of the 19th ACM international conference on Information and knowledge management*, ser. CIKM '10. New York, NY, USA: ACM, 2010, pp. 1925–1926. [Online]. Available: http://doi.acm.org/10.1145/1871437.1871766

[5] G. Ramakrishnan, S. Chakrabarti, D. Paranjpe, and P. Bhattacharya, "Is question answering an acquired skill?" in *Proceedings of the 13th international conference on World Wide Web*, ser. WWW '04. New York, NY, USA: ACM, 2004, pp. 111–120. [Online]. Available: http://doi.acm.org/10.1145/988672.988688

[6] R. Fagin, R. Kumar, K. S. McCurley, J. Novak, D. Sivakumar, J. A. Tomlin, and D. P. Williamson, "Searching the workplace web," in *Proceedings of the 12th international conference on World Wide Web*, ser. WWW '03. New York, NY, USA: ACM, 2003, pp. 366–375. [Online]. Available: http://doi.acm.org/10.1145/775152.775204

[7] D. Hawking, "Enterprise search - the new frontier?" in *ECIR'06*, 2006, pp. –1–1.

[8] ——, "Challenges in enterprise search," in *Proceedings of the 15th Australasian database conference - Volume 27*, ser. ADC '04. Darlinghurst, Australia, Australia: Australian Computer Society, Inc., 2004, pp. 15–24. [Online]. Available: http://dl.acm.org/citation.cfm?id=1012294.1012297

[9] Z. Bao, B. Kimelfeld, and Y. Li, "Automatic suggestion of query-rewrite rules for enterprise search," in *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, ser. SIGIR '12. New York, NY, USA: ACM, 2012, pp. 591–600. [Online]. Available: http://doi.acm.org/10.1145/2348283.2348363

[10] P. A. Dmitriev, N. Eiron, M. Fontoura, and E. Shekita, "Using annotations in enterprise search," in *Proceedings of the 15th international conference on World Wide Web*, ser. WWW '06. New York, NY, USA: ACM, 2006, pp. 811–817. [Online]. Available: http://doi.acm.org/10.1145/1135777.1135900

[11] A. Doan, R. Ramakrishnan, and S. Vaithyanathan, "Managing information extraction: state of the art and research directions," in *Proceedings of the 2006 ACM SIGMOD international conference on Management of data*, ser. SIGMOD '06. New York, NY, USA: ACM, 2006, pp. 799–800. [Online]. Available: http://doi.acm.org/10.1145/1142473.1142595

[12] H. Zhu, S. Raghavan, S. Vaithyanathan, and A. Löser, "Navigating the intranet with high precision," in *Proceedings of the 16th international conference on World Wide Web*, ser. WWW '07. New York, NY, USA: ACM, 2007, pp. 491–500. [Online]. Available: http://doi.acm.org/10.1145/1242572.1242639

[13] A. Z. Broder and A. C. Ciccolo, "Towards the next generation of enterprise search technology," *IBM Systems Journal*, vol. 43, no. 3, pp. 451 –454, 2004.

[14] M. Brunzel, "The xtreem methods for ontology learning from web documents," in *Proceedings of the 2008 conference on Ontology Learning and Population: Bridging the Gap between Text and Knowledge*. Amsterdam, The Netherlands, The Netherlands: IOS Press, 2008, pp. 3–26. [Online]. Available: http://dl.acm.org/citation.cfm?id=1563823.1563826

[15] M. Poesio and A. Almuhareb, "Extracting concept descriptions from the web: the importance of attributes and values," in *Proceedings of the 2008 conference on Ontology Learning and Population: Bridging the Gap between Text and Knowledge*. Amsterdam, The Netherlands, The Netherlands: IOS Press, 2008, pp. 29–44. [Online]. Available: http://dl.acm.org/citation.cfm?id=1563823.1563828

[16] D. Maynard, Y. Li, and W. Peters, "Nlp techniques for term extraction and ontology population," in *Proceedings of the 2008 conference on Ontology Learning and Population: Bridging the Gap between Text and Knowledge*. Amsterdam, The Netherlands, The Netherlands: IOS Press, 2008, pp. 107–127. [Online]. Available: http://dl.acm.org/citation.cfm?id=1563823.1563834

[17] H.-J. Song, S.-B. Park, and S.-Y. Park, "An automatic ontology population with a machine learning technique from semi-structured documents," in *Information and Automation, 2009. ICIA '09. International Conference on*, june 2009, pp. 534 –539.

[18] O. Etzioni, M. Banko, S. Soderland, and D. S. Weld, "Open information extraction from the web," *Commun. ACM*, vol. 51, no. 12, pp. 68–74, Dec. 2008. [Online]. Available: http://doi.acm.org/10.1145/1409360.1409378

[19] A. Fader, S. Soderland, and O. Etzioni, "Identifying relations for open information extraction," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, ser. EMNLP '11. Stroudsburg, PA, USA: Association for Computational Linguistics, 2011, pp. 1535–1545. [Online]. Available: http://dl.acm.org/citation.cfm?id=2145432.2145596

[20] E. Agichtein and L. Gravano, "Snowball: extracting relations from large plain-text collections," in *Proceedings of the fifth ACM conference on Digital libraries*, ser. DL '00. New York, NY, USA: ACM, 2000, pp. 85–94. [Online]. Available: http://doi.acm.org/10.1145/336597.336644

[21] B. B. Dalvi, W. W. Cohen, and J. Callan, "Websets: extracting sets of entities from the web using unsupervised information extraction," in *Proceedings of the fifth ACM international conference on Web search and data mining*, ser. WSDM '12. New York, NY, USA: ACM, 2012, pp. 243–252. [Online]. Available: http://doi.acm.org/10.1145/2124295.2124327

[22] D. Cai, S. Yu, J.-R. Wen, and W.-Y. Ma, "Extracting content structure for web pages based on visual representation," in *Proceedings of the 5th Asia-Pacific web conference on Web technologies and applications*, ser. APWeb'03. Berlin, Heidelberg: Springer-Verlag, 2003, pp. 406–417. [Online]. Available: http://dl.acm.org/citation.cfm?id=1766091.1766143

[23] W. Gatterbauer, P. Bohunsky, M. Herzog, B. Krüpl, and B. Pollak, "Towards domain-independent information extraction from web tables," in *Proceedings of the 16th international conference on World Wide Web*, ser. WWW '07. New York, NY, USA: ACM, 2007, pp. 71–80. [Online]. Available: http://doi.acm.org/10.1145/1242572.1242583

[24] T. Weninger, F. Fumarola, R. Barber, J. Han, and D. Malerba, "Unexpected results in automatic list extraction on the web," *SIGKDD Explor. Newsl.*, vol. 12, no. 2, pp. 26–30, Mar. 2011. [Online]. Available: http://doi.acm.org/10.1145/1964897.1964904

[25] A.-P. Manine, E. Alphonse, and P. Bessières, "Information extraction as an ontology population task and its application to genic interactions," in *Proceedings of the 2008 20th IEEE International Conference on Tools with Artificial Intelligence - Volume 02*, ser. ICTAI '08. Washington, DC, USA: IEEE Computer Society, 2008, pp. 74–81. [Online]. Available: http://dx.doi.org/10.1109/ICTAI.2008.117

[26] L. Chiticariu, R. Krishnamurthy, Y. Li, S. Raghavan, F. R. Reiss, and S. Vaithyanathan, "Systemt: an algebraic approach to declarative information extraction," in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ser. ACL '10. Stroudsburg, PA, USA: Association for Computational Linguistics, 2010, pp. 128–137. [Online]. Available: http://dl.acm.org/citation.cfm?id=1858681.1858695

[27] W. Shen, A. Doan, J. F. Naughton, and R. Ramakrishnan, "Declarative information extraction using datalog with embedded extraction predicates," in *Proceedings of the 33rd international conference on Very large data bases*, ser. VLDB '07. VLDB Endowment, 2007, pp. 1033–1044. [Online]. Available: http://dl.acm.org/citation.cfm?id=1325851.1325968

[28] L. Chiticariu, V. Chu, S. Dasgupta, T. W. Goetz, H. Ho, R. Krishnamurthy, A. Lang, Y. Li, B. Liu, S. Raghavan, F. R. Reiss, S. Vaithyanathan, and H. Zhu, "The systemt ide: an integrated development environment for information extraction rules," in *Proceedings of the 2011 international conference on Management of data*, ser. SIGMOD '11. New York, NY, USA: ACM, 2011, pp. 1291–1294.

[29] T. Joachims, "Optimizing search engines using clickthrough data," in *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, ser. KDD '02. New York, NY, USA: ACM, 2002, pp. 133–142. [Online]. Available: http://doi.acm.org/10.1145/775047.775067

[30] R. Burget and I. Rudolfová, "Web page element classification based on visual features," in *Intelligent Information and Database Systems, 2009. ACIIDS 2009. First Asian Conference on*. IEEE, 2009, pp. 67–72.

[31] R. C. Wang and W. W. Cohen, "Language-independent set expansion of named entities using the web," in *Proceedings of the 2007 Seventh IEEE International Conference on Data Mining*, ser. ICDM '07. Washington, DC, USA: IEEE Computer Society, 2007, pp. 342–350. [Online]. Available: http://dx.doi.org/10.1109/ICDM.2007.104