Entity name extraction from faculty directories

João Mateus de Freitas Veneroso
Universidade Federal de Minas Gerais
Belo Horizonte, MG, Brazil
jmfveneroso@gmail.com

Berthier Ribeiro-Neto
Universidade Federal de Minas Gerais & Google Engineering
Belo Horizonte, MG, Brazil
berthier@dcc.ufmg.br
berthier@google.com

ABSTRACT

Public bibliographic databases hold invaluable data about the academic environment. However, researcher affiliation information is frequently missing or outdated. We propose a statistical data extraction method to acquire affiliation information directly from university websites and solve the name extraction task in general. Previous approaches to web data extraction either lack in flexibility, because wrappers do not generalize well across websites, or they lack in precision, because domain agnostic methods neglect useful properties of this particular application domain. Our statistical approach solves the name extraction task with a good tradeoff between generality and precision. We conducted experiments over a collection of 152 faculty web pages in multiple languages from universities in 49 countries and obtained 94.37 % precision, 97.61% recall and 0.9596 F-measure.

1 INTRODUCTION

Tipically, Information Extraction tasks consist of mapping unstructured or poorly structured data to a semantically well defined structure. The input is most commonly composed of a set of documents that describe a group of entities in a similar manner. The Information Extraction task consists of identifying these entities and organizing them according to a template. The most successful approaches at solving the problem of information extraction are documented in several surveys [1–5].

We are interested in the task of name extraction, particularly extracting researcher names from university websites to complement data from public databases such as the DBLP repository (http://dblp.uni-trier.de/) [1]. We propose a probabilistic method to handle the name extraction problem in general.

The name extraction problem is not different from a Named Entity Recognition problem, because if we are able to recognize entities we may also map them to a structured format. However, state-of-the-art named entity recognition approaches such as Conditional Random Fields do not perform so well in information extraction scenarios such as the name extraction problem, because the text in web pages is usually insufficient to provide proper contextual information about the semantic category of a word. For instance, tabular data such as the one found in most faculty directories provides minimal textual context. Our method of label assignment resembles a Naive Bayesian classifier without the assumption of token independence. We also rely as little as possible on contextual information because in the web extraction task it is frequently missing, noisy or inadequate.

[1] This is useful, for instance, if one needs to compare the research output of deparments in Germany, or study international publication patterns.

2 IMPLEMENTATION

Before applying the classifier over the web page content, we must run the input web page through a series of pre-processing steps to clean the data as much as possible. At this stage we remove useless tags, and account for malformed HTML. Spaces, tabs, newlines and special characters are used as separators in a tokenization stage, producing a list of tokens. Each token is saved as an object that holds the token value and a pointer to its position in the DOM tree. Through the DOM Element pointer we can figure out the parent elements, nesting depth, siblings and other information that can be used to improve estimates in the probabilistic method we adopt.

2.1 Model

Let $t = (t_1, t_2, ..., t_n)$ be a sequence of token objects obtained on the pre-processing stage, and $y = (y_1, y_2, ..., y_n)$ be a sequence of labels attributed to these tokens where y_i can be either a "Name Label" (N), meaning that token t_i is a person's name, or a "Word Label" (W), meaning that token t_i is a common word (not a person's name). Then, the problem of extracting names from a sequence of tokens is just a series of binary classification problems. Considering that each token t_i has a probability $P(t_i = y_i)$ of having label y_i, the problem of finding an optimal sequence of labels y^* for a sequence of tokens t can be written as:

$$y^* = \underset{y}{\operatorname{argmax}} P(t_1 = y_1, t_2 = y_2, ..., t_n = y_n) \tag{1}$$

We may employ the chain rule to explore the relationship between joint and conditional probabilities. For ease of exposure, consider that $P(Y_i) \equiv P(t_i = y_i)$ yielding:

$$P(Y_1, Y_2, ..., Y_n) = P(Y_1)P(Y_2|Y_1)...P(Y_n|Y_{n-1}, Y_{n-2}, ...) \tag{2}$$

The conditional probabilities $P(Y_i|Y_{i-1}, ...)$ are hard to estimate, because the joint distribution depends both on the previous labels and the previous tokens. To simplify our modeling, let us assume that the probability that token t_i has label y_i depends on the values of previous labels but is independent of the previous tokens. That is, we assume that, given a sequence of tokens $\{"John", "Smith"\}$, the conditional probability $P("Smith"|"John" = Name)$ is equivalent to $P("Smith"|Any\ name)$. In other words, this means that the probability of Smith being a last name is the same regardless of a person's first name, as long as we can make sure that the previous token is a name. By taking this assumption, we get:

$$P(Y_i|Y_{i-1}, Y_{i-2}, ...) \approx P(t_i|y_i, y_{i-1}, ...)P(y_i|y_{i-1}, y_{i-2}, ...) \tag{3}$$

In Equation 3, the probability $P(t_i|y_i, y_{i-1}, ...)$ depends on the current and previous labels. We make a simplifying assumption that $P(t_i|y_i, y_{i-1}, ...)$ can be approximated by $P(t_i|y_i)$. With this assumption, Equation 3 becomes:

$$P(Y_i|Y_{i-1}, Y_{i-2}, ...) \approx P(t_i|y_i)P(y_i|y_{i-1}, y_{i-2}, ...) \quad (4)$$

Finally, by replacing Equation 4 into Equation 2 and grouping together the second terms of Equation 4 to form the joint probability $P(y_1, y_2, \ldots, y_n)$, we obtain:

$$P(Y_1, \ldots, Y_n) \approx P(y_1, \ldots, y_n)P(t_1|y_1)P(t_2|y_2)\ldots P(t_n|y_n) \quad (5)$$

Equation 5 can be split into two parts: the prior: $P(y_1, y_2, \ldots, y_n)$, and the conditional token probabilities: $P(t_1|y_1)P(t_2|y_2)\ldots P(t_n|y_n)$.

2.1.1 Prior probabilities. By assuming a window of size $k \leq n$ we may approximate the priors by calculating the joint probability $P(y_1, y_2, \ldots, y_k)$. We observed empirically that when a name token is found, it is very likely that the next token will also be a name. However, arbitrary sequences of non-name tokens do not alter significantly the probabilities for the next token's label. We can use this property to slide the window of k tokens over the entire sequence of tokens and estimate probabilities without deviating too much from the real distribution. We start at the beginning of the token list and calculate probabilities for each possible sequence of labels, taking the most likely one. If the first token is not a name we slide the window right by one token and repeat the process. If the first token is a name, then we extract all the tokens that are part of that name starting from the first one and slide the window right by the number of name tokens extracted.

2.1.2 Conditional token probabilities. For our experiments, the conditional token probabilities were obtained by maximum likelihood estimation with Laplace smoothing to account for tokens that did not occur in the corpus. Conditional token probabilities can be made more precise by incorporating features in equation 5. We do that by changing the token conditional probabilities to:

$$P(t_i, f_1, f_2, \ldots, f_n|y_i) \approx P(t_i|y_i)P(f_1|y_i)\ldots P(f_n|y_i) \quad (6)$$

where f_i are features that are assumed to be independent. Textual features take textual clues from context words and the current token value. Structural features infer token probabilities based on the surrounding HTML structure. The textual features found to be useful in our model were: token value, token incidence and token length. The structural features found to be useful in our model were: first and second HTML parents, CSS class name and nesting depth.

2.1.3 Secondary estimates. HTML structure varies wildly between different websites, so we cannot extract useful probabilities looking at the entire collection. For example, if all names appear inside a <tr> tag in a given document it does not mean that names tend to appear inside <tr> tags rather than any other HTML tag in other documents. However for that particular document we may be able to identify other names and exclude non name tokens by knowing that tokens that occur inside <tr> tags have a higher probability of being names.

Given that the basic algorithm (without structural features) was able to extract a satisfactory number of names from a web page with sufficient precision on a first run, we may estimate probabilities for a structural feature $P(f_j|y_i)$ by looking at the extracted names. In fact, since we already attributed a label to every token in the web page, we can access the originating DOM element through the token object and estimate feature probabilities by maximum likelihood, as we did for the token conditional probabilities.

2.2 Experiments

Features	Precision	Recall	F-measure
None	91.71%	91.39%	0.9155
Textual	91.83% (0.46)	93.90% (0.15)	0.9285 (0.23)
Full	94.37% (0.05)	**97.61% (0.01)**	**0.9596 (0.01)**

Table 1: Name extraction experiment

The test collection was a set of 152 manually labeled faculty directory web pages from 49 different countries with 11,782 researcher names. These names are the target of our extraction procedure. We calculated the precision, recall and f-measures for a base model with no additional features besides conditional token probabilities and priors, a model with only textual features and a full model with textual and structural features. All measures were tested for statistical significance with a one tailed paired T-test in comparison with the base model, the t-values are presented inside parenthesis next to each measure. The measures were obtained by the averaged results of a 5 fold cross validation run. Extracted names were only considered to be correct when they resulted on an exact match with the test data.

3 CONCLUSION

Our name extraction method achieved 94.37% precision, 97.61% recall and 0.9596 F-measure over the test corpus. The model is tuned for the particular problem of name extraction, but we believe this result can be generalized to solve other data extraction problems. The algorithm is general enough to handle other types of information extraction tasks without the need for too much tinkering. The secondary estimates strategy was very successful in further improving the base model. It can also be remodeled to fit other statistical classifiers since it does not rely on any particular implementation details of our strategy.

REFERENCES

[1] A.H.F. Laender, B. A. Ribeiro-Neto, and Juliana S.Teixeria. A brief survey of web data extraction tools. *ACM SIGMOD Record 31(2)*, pages 84–93, 2002.

[2] Sunita Sarawagi. Information extraction. *Foundations and Trends in Databases*, 1(3):261–377, 2008.

[3] Emilio Ferrara, Pasquale De Meo, Giacomo Fiumara, and Robert Baumgartner. Web data extraction, applications and techniques: A survey. *Knowledge-Based Systems*, 70:301–323, 2014.

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

[5] M. I. Varlamov and D. Yu. Turdakov. A survey of methods for the extraction of information from Web resources. *Programming and Computer Software*, 42(5):279–291, 2016.