On the Use of Correspondence Analysis to Learn Seed Ontologies from Text

Davide Eynard davide.eynard@polimi.it
Fabio Marfia fabio.marfia@stud.polimi.it
Matteo Matteucci matteo.mattucci@polimi.it

Politecnico di Milano
Dipartimento di Elettronica e Informazione

Motivations and Objectives

Background
As the Web becomes easily accessible as a "read-write" medium to more and more users, we can assist to an **information overload**:

- on the one hand, constantly updated information is available in the form of collection of Web pages, large document corpora, databases, and so on;
- on the other hand, this information is mostly published by humans for humans, it is unstructured and cannot be automatically consumed by a machine.

=> Knowledge Acquisition Bottleneck
Creating large, usable, expandable and valid representations of semantics about a specific domain of interest represents the most time consuming task of a KM project.

Ontology Learning from Text
A discipline whose objective is to subsume ontological proposition from collections of natural language propositions, relying on some characteristics which are intrinsic to text:

- the set of descriptive rules represented by the *grammar* and the *syntax* of the language (NLP approach)
- the domain specificity of a particular corpus of documents (ontological approach)
- the distribution of terms across all the documents in the corpus (statistical approach)

Distributional Hypothesis
"Words are similar to the extent that they share similar context"
(Harris, 1968)

Objectives
Being Ontology Learning for Text an approximate activity, users can hardly foresee what is definitely the best solution that will extract the most usable and valid ontology from their document corpora.

The main objective of this work is to offer a *valid* and *versatile* alternative to other approaches, allowing the user to try different possibilities and find between them the best solution, according to his needs.

This is done by developing a *modular application* which:
- analyzes a corpus of documents and extracts its most relevant concepts;
- uses Correspondence Analysis to calculate similarity between these concepts and show this relation in a 2D space;
- applies different techniques to derive a taxonomy from similarity relationships between concepts

Our approach

1) Input
Two different corpora are passed as an input to the system: a **training set**, which is used as a reference to what is considered as common knowledge, and a **test set**, the one from which we want to extract the concept hierarchy.

2) Indexing
Documents are indexed using **Apache Lucene**, which takes care of tokenization, indexing, and stopwords filtering. As the indexing process is the more time-consuming, indices (such as the one for the training set) can be saved and reused.

3) Data Filtering
Three different filters can be applied: a **Wordnet-based** noun detector, an **NLP-based** one, and a **short-terms discarder**.

4) Relevant term identification
For every term *t* in the test corpus we calculate the **Information Gain:**

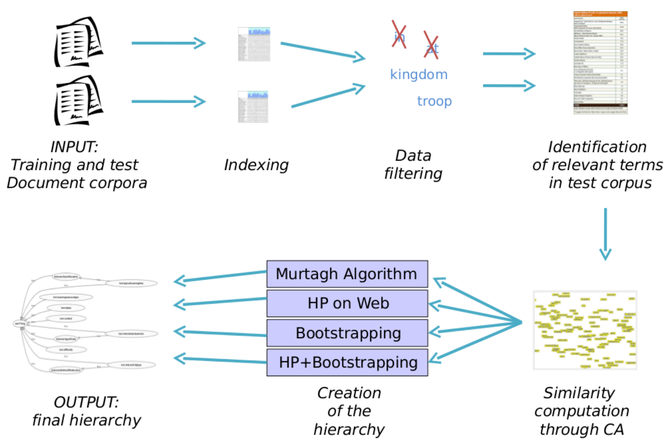
$$IG(t) = H_{D_{total}} - \frac{|D_t|}{|D_{total}|}H_{D_t} - \frac{|D_{\neg t}|}{|D_{total}|}H_{D_{\neg t}}$$

where the entropy H for a generic document set D_g is calculated as:

$$H_{D_g} = -\sum_i p(i) \cdot log(p(i))$$

Terms are then ordered by their IG and the top ones are taken as potentially relevant concepts.

5) Similarity computation
Similarity between terms is calculated by applying the framework of **Correspondence Analysis.** CA allows for a compact representation of term similarities by projecting them in a multidimensional Euclidean space. Term distances in this space are the χ^2 distances between their **column profiles** (i.e. the distribution of documents in which the terms to be matched appear).

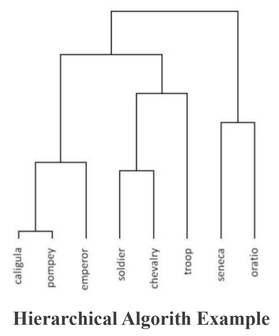


$M_{colprof}$	caligula	city	group	F_i
D1	53%	17%	31%	35%
D2	18%	76%	3%	21%
D3	28%	4%	1%	10%
D4	1%	3%	65%	34%

TFIDF — Document Weight (%observations)

$$\chi^2(l,k) = \sqrt{\sum_i \frac{(n_{l,i} - n_{k,i})^2}{F_i}}$$

6) Hierarchy creation
The tool allows one to choose among four different options: **Murtagh's Algorithm** [4], **Hearst Patterns on Web** [1,2], Maedche and Staab's **Bootstrapping** [3], and an original **combination** of these last two.



Hierarchical Algorith Example

HP on Web algorithm

1. build five pre-defined HP strings:
hp_1 : pluralize(t_i) such as t_n
hp_2 : pluralize(t_i) including t_n
hp_3 : pluralize(t_i) especially t_n
hp_4 : pluralize(t_i) like t_n
hp_5 : t_n is a/an t_i

2. execute six Google queries (hp_1 to hp_3 plus t_n) and get number of Google hits for each query

3. the score of every string is defined as

$$score_{hp_i} = \frac{googleHits(hp_i)}{googleHits(t_n)}$$

4. the total score is obtained as the sum of the five different scores

Maedche and Staab's algorithm

Expands a concept hierarchy by adding a new t_n term according to the hypernyms of its m nearest neighbors in the Euclidean space generated with CA.
1. Least common superconcept is defined as:

$$lcs(a,b) = \underset{c}{\operatorname{argmin}}\ \delta(a,c) + \delta(b,c) + \delta(root,c)$$

2. Taxonomic similarity is defined as:

$$\sigma(a,b) = \frac{\delta(root,c)+1}{\delta(root,c)+\delta(a,c)+\delta(b,c)+1}$$

where c=lcs(a,b).

3. The score for a candidate hypernym is computed as:

$$W(f) = \sum_{h \in H(f)} sim(t_n,h) \cdot \sigma(n,h)$$

Applications and results

Input Data
Three different document corpora were selected:

- a set of 847 Wikipedia articles about **Artificial Intelligence**;
- a set of 1464 Wikipedia articles about **Roman Empire** and related historical articles;
- a set of 1364 Wikipedia articles about **Biology.**

As a referential Training corpus we have used a collection of 1414 Wikipedia articles *about Wikipedia itself*.

Data analysis and results evaluation
Analysis has been performed in three main steps:

- *concept extraction*: we have indexed the training and test set, filtered them to get only the *names*, and calculated information gain to extract the most relevant concepts;
- *correspondence analysis*: we have generated a 2-dimensional representation of distributional similarity of the extracted terms;
- *hierarchy generation*: the different algorithms have been applied and compared. Results are shown in the table below.

Algorithm	AI	Rome	Biology	Average
Murtagh	6.6%	5.22%	1.87%	**4.56%**
Hearst Patterns on Web	60.00%	75.00%	37.50%	**57.50%**
Hearst Patterns on Web (assisted)	90.00%	94.52%	85.07%	**89.86%**

Table 1: Precision measures obtained from the evaluation of the different ontology learning algorithms, obtained by comparing the tool's results with relations proposed by a human judge.

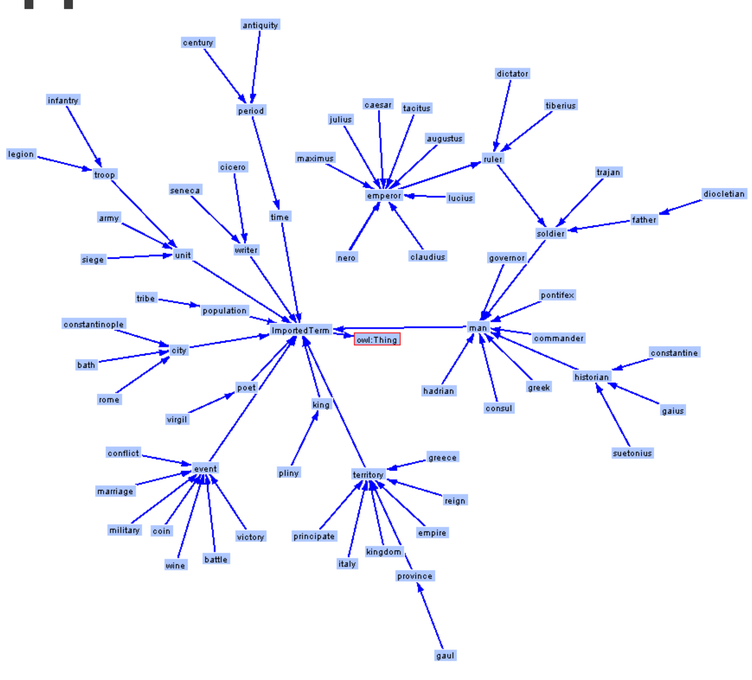


Figure1: A hierarchy generated by the Hearst Patterns on Web algorithm in its semi-automatic form, from 100 terms of a corpus about the Roman Empire.

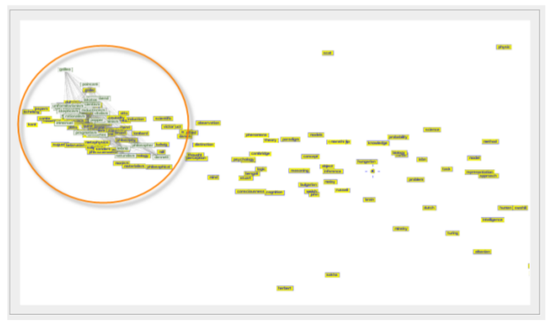


Figure2: The possibility to plot extracted concepts in the 2-D Euclidean space offers the possibility to graphically evaluate the quality of results. In the current example, the semantic area of philosophers names and movements is highlighted as a well defined part of the corpus about Artificial Intelligence.

Related Work:

[1] **Cimiano P., Handschuh S., S. S. (2004).** *Towards the self-annotating web*. In Proceedings of the 13th WWW Conference.
[2] **Hearst, M. (1992).** *Automatic acquisition of hyponymsfrom a large text corpora*. In Proceedings of the 14thInternational Conference of Computational Linguistics.
[3] **Maedche A., Pekar V., S. S. (2003).** *On discovering taxo-nomic relations from the web*. Technical report, Insti-tute AIFB - University of Karlsruhe, Germany.
[4] **Murtagh, F. (2007).** *Ontology from hierarchical structure intext*. Technical report, University of London Egham.