Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 38 (2014) 116 – 123

Procedia
Computer Science

10th Italian Research Conference on Digital Libraries, IRCDL 2014

Automatic Learning of Linguistic Resources for Stopword Removal and Stemming from Text

Stefano Ferilli[a,b,*], Floriana Esposito[a], and Domenico Grieco[a]

[a]*University of Bari, Via E. Orabona 4, 70126 Bari, Italy*
[b]*Centro Interdipartimentale per la Logica e sue Applicazioni, Via E. Orabona 4, 70126 Bari, Italy*

Abstract

While multimedia digital documents are progressively spreading, most of the content of Digital Libraries is still in the form of text, and this predominance will probably never be questioned. Except pure display of these documents, all other tasks are based on some kind of Natural Language Processing, that must be supported by suitable linguistic resources. Since these resources are clearly language-specific, they might be unavailable for several languages, and manually building them is costly, time-consuming and error-prone. This paper proposes a methodology to automatically learn linguistic resources for a natural language starting from texts written in that language. The learned resources may enable further high-level processing of documents in that language, and/or be taken as a basis for further manual refinements. Experimental results show that its application may effectively provide useful linguistic resources in a fully automatic manner.

Peer-review under responsibility of the Scientific Committee of IRCDL 2014

Keywords: Natural Language Processing; Linguistic Resources; Document Processing; Digital Libraries;

1. Introduction

Although the power and flexibility of new technology (computers, tablets, smartphones, etc.) supports and endorses the spread of multimedia content, text is still the main channel by which information is represented, spread and exchanged by humans. Accordingly, Digital Libraries (DLs for short) have expanded the range of their interests

[*] Corresponding author. Tel.: +39-080-5442293; fax: +39-080-5442031.
E-mail address: stefano.ferilli@uniba.it

Peer-review under responsibility of the Scientific Committee of IRCDL 2014
doi:10.1016/j.procs.2014.10.019

to images, recordings and videos, but the overwhelming majority of their content is still in the form of text. In fact, beyond their perceptual aspects, that are precious for preservation and witnessing purposes, also the meaningful content of sound recordings and images (and sometimes even of portions of images) can be reduced to texts in natural language. It is very likely that this landscape will not change significantly in the future, because natural language is the tool that humans have developed and refined through the millenniums to express their thoughts and exchange notions.

The pervasive presence and use of electronic tools and devices has also caused a dramatic growth in the number of available documents. Efficient and effective management of so many documents is already well beyond human capabilities, and this prevents their meaningful control and exploitation. In a sense, having so large quantities of documents is almost like not having documents at all (a problem known as *information overloading*), unless suitable automatic techniques are developed that can support their organization, indexing and retrieval. Concerning specifically textual content, the Natural Language Processing (NLP) research area aims at developing advanced tools for understanding the components, structure and meaning of sentences in a text, and to properly organize this information as to help human users in satisfying their information needs.

All NLP solutions must in the end rely on the use of linguistic (morphological, lexical, grammatical) resources to carry out the most basic processing steps and build on them higher-level functionalities. Since each language has its own peculiarities, the resources developed for a language are useless for the others. Unfortunately, developing these resources is a critical point, for several reasons. Designing and developing them manually requires linguistic experts, whose activity is costly, time-consuming and error-prone. Once the resources are available, it is very hard to maintain and update them, or to tailor them to specific domains, or to fix possible errors. As a result, for many existing languages linguistic resources are not available, which prevents application of automatic high-level processing techniques to support their users. This is especially true for dialects and jargons, whose spread is so limited that nobody is likely to ever produce these resources. A bad consequence of this is the risk that entire cultures might be lost, just because their underlying documents would not be exploitable. Our efforts were spent in this direction, trying to build a system that carries out fully automatic learning of linguistic information and resources starting from plain texts, and provides the users with tools and facilities to navigate and exploit this information for different purposes. A typical use case would be the following: a collection of documents in a language (for which linguistic resources do not exist) is loaded in a DL; the DL itself automatically develops the linguistic resources needed to index and organize these documents, and to recognize future incoming documents as belonging to that language; the collection is indexed and organized using these resources, and the user can query and browse it as he would do in the case of languages for which resources do exist.

The BLA-BLA (an acronym for 'Broad-spectrum Language Analysis-Based Learning Application') system currently includes several techniques concerning language identification, stopword removal, term normalization and concept extraction. The learned resources may enable further high-level processing of documents in that language, and/or be taken as a basis for further manual refinements. Whenever more texts become available for the language, it is easy to run again the technique and obtain updated resources. Experimental results show that its application may effectively provide useful linguistic resources in a fully automatic manner. In this paper, we specifically focus on the methodology embedded in BLA-BLA to automatically learn linguistic resources concerning stopword removal and term normalization. These methodologies are general and applicable to any language based on inflection. After recalling some background notions and related work in the next Section, we describe and evaluate the proposed technique in Sections 3 and 4, respectively. Lastly, Section 5 concludes the paper and outlines future work directions.

2. Background and Related Work

NLP typically relies on a number of preliminary steps that provide the matter on which higher-level functions can be built. These steps progressively extract from a given text more and more complex features and components, starting from the morphological level, through the lexical one, up to the syntactic one. Recent research on taxonomies, conceptual graphs and ontologies tries even to approach the semantic level. A selection of general and noteworthy tasks is the following:

- **Language Identification** aims at discovering the language in which the text is written, because depending on the language different tools and resources must be used for the following steps;

- **Stopword Removal** since most state-of-the-art NLP techniques work on the lexical level using statistic approaches, this step removes the terms that are widespread and frequent in any kind of text, and hence are not informative about the specific text content (e.g., articles, prepositions, pronouns);
- **Normalization** since most informative terms usually undergo inflection to be used in the sentences, different occurrences of the same term must be standardized to a single form (*stemming* reports them to their linguistic root, while *lemmatization* reports them to the basic form of the inflection);
- **Part-of-Speech Tagging (POS Tagging)** associates to each term its grammatical function, useful because different functions act as indicators for different aspects of the content in which high-level processing steps may be interested (e.g., techniques concerned with the content focus on nouns and verbs, while those concerned with sentiments focus on adjectives and adverbs);
- **Parsing** returns the syntactic structure of sentences, usually in the form of a (possibly annotated) syntactic tree, that allows to understand the role of different elements of the sentence with respect to the conveyed message;
- **Word Sense Disambiguation** aims at associating each term in the text to the underlying concept, to attack the synonymy and polysemy problems that affect natural language.

As said, each of these steps is typically carried out using suitable (language-specific) linguistic resources. Language Identification often exploits *n*-gram distribution, Stopword Removal exploits lists of frequent terms, Normalization exploits lists of suffixes[16], POS Tagging exploits suffixes and/or grammatical rules (e.g., as in Brill[2]), Parsing uses grammars, Word Sense Disambiguation uses conceptual taxonomies or ontologies. Most works in the literature are concerned with English, probably due to its having a structure which is easier than other languages and to its importance as the standard information interchange language worldwide. Little exists for a few other important languages, and almost nothing for the vast majority of minor languages, especially dialects and jargons.

A few attempts exist to automatically learn resources for Language Identification (in the form of statistics on the distribution of n-grams across the various languages[1,13,143]), POS Tagging (e.g., by learning tagging rules[3,4]), Parsing (with the research stream concerning grammar inference[6]) and Word Sense Disambiguation (with initial attempts to learn concept taxonomies or graphs, or even ontologies, but often based on existing taxonomies/graphs and/or semi-automatically[5,8,11,12,15,18, 19]). Other modules of BLA-BLA have been developed that learn statistics for language identification and conceptual graphs, but are outside the scope of this paper. In particular, the recognition of language relies on statistics about the frequency of *n*-grams, stopwords and suffixes, to be used by bayesian and/or histogram-based recognition approaches.

As regards conceptual graphs, the learning module in charge of this functionality, named ConNeKTion, has already been presented[7,9,10,17]. Most NLP techniques process the text in the form of Bag-of-Words, i.e. considering only the list of terms appearing in the text, possibly associated to weights representing their frequency. Indeed, this setting has proven to be a good trade-off between efficiency and effectiveness in many applications. For instance, document Indexing, which is the foundation for Information Retrieval, can be satisfactorily based on the lexical level alone, and Information Retrieval is one of the most important and urgent needs to organize a DL and make its content available to interested users. For these techniques, the first three steps are sufficient, but unfortunately, except for Language Recognition, nothing can be found in the literature for automatically learning corresponding resources (to the best of our knowledge). This work aims at filling that gap, focusing on the Stopword Removal and Normalization tasks.

3. Learning Approach

Our learning procedure processes a set of input training documents in pure text, each of which is associated to the corresponding language. The pre-processing step proceeds by scanning each document in turn character by character, and skipping all character sequences except *words*, where a word is defined as a sequence of alphabetic characters only, delimited by blank spaces. Between the initial blank and the first character, and/or between the last character and the final blank, punctuation symbols are allowed (not considered as belonging to the word). The case of an apostrophe joining two words was considered as well. More formally, the linear expression pattern is:

$$\lambda P\{W'\}^{*}WP\lambda$$

where

- λ is the blank symbol;
- ' is the apostrophe;
- $P = \{.|,|;|:|?|!|"|'\}^*$ is a (possibly empty) sequence of punctuation marks;
- $W = \{a \mid b \mid \ldots \mid z\}^+$ is the word (hypothesizing a latin alphabet).

Our assumption is that each document belongs exactly to one language. This does not mean, of course, that it cannot include words or expressions from other languages, but these are to be considered as noise, and suitably handled by the normal learning approach.

3.1. Stopword Removal

Stopwords are terms in a language that appear so often and pervasively in the documents as to make them irrelevant to distinguish documents with respect to their content. For this reason, they can be safely ignored by all NLP techniques that work at the lexical level. The removal task is simply carried out by lookup in a pre-determined list of keywords. The usual way by which such a list is prepared is including all *function words*, i.e. terms associated to invariant Parts-of-Speech of the language (usually articles, pronouns and prepositions). However, for domain-specific applications, also other terms that are insignificant in the particular context (e.g., the word '*computer*' in a DL specialized in Computer Science) can be added to the list.

Of course, in our setting no linguistic information whatever is available for the language. So, we resort to purely statistical considerations to identify stopwords as those terms that appear with a significantly higher frequency than the other words in the training documents. In BLA-BLA, this is obtained by setting a frequency threshold σ, and in the preliminary prototype this threshold was simply set as the average frequency of all terms collected for the language:

$$\sigma = \frac{\alpha}{n} \sum_{i=1}^{n} t_i \tag{1}$$

where n is the number of terms in the language, t_i is the frequency of the i-th term and α is an adjustment factor used to smooth the effect of the average. In the current prototype, $\alpha = 1.05$ was used.

3.2. Normalization

Normalization is usually carried out by finding and removing word suffixes that are connected to inflection, in order to identify and consider different occurrences of the same term independently of their role in the context of the specific sentences in which they are used (e.g., '*computer*' and '*computers*' convey the same meaning, as well as '*computing*' and '*computed*'). Again, the suffixes to be searched for must be known *a priori* and, of course, this implies that linguistic knowledge must be available to compile the list of suffixes. Two kinds of normalization are available: *stemming* reduces a term to its root, which is not necessarily a meaningful word in the language (e.g., '*computer*', '*computers*', '*computing*' and '*computed*' would all be reduced to the same stem '*comput*'); *lemmatization* transforms the term to its basic form, depending on its grammatical type (so, e.g., '*computer*' and '*computers*' would be changed to '*computer*', while '*computing*' and '*computed*' would be changed to '*compute*'). Clearly, lemmatization once again requires linguistic competence to know which is the basic suffix to be used for replacing the inflection suffix. As said, we cannot exploit any a priori linguistic knowledge. So, we focus on stemming, and again use statistical considerations to identify possible stems and suffixes of the language. We first find the stems, i.e. initial parts of the words that are frequently found in the term collection, and then obtain the suffixes by difference. Once learned, these suffixes can be provided to a general suffix stripping algorithm which will apply them to new documents to obtain their normalized version. To find the stems, the list of words is considered and sorted alphabetically. This brings close to each other the words that have the same initial characters. Then the similarity of adjacent pairs of words is computed, and may be graphically represented by a histogram. Note that the presence of prefixes causes different facets of the same meaning (e.g., ‘pre-processing’ and ‘post-processing’) to be considered as completely different stems, due to their initial part being very different.

Now, we need a way to assess how similar the initial parts of two character sequences are. After trying different solutions, we decided to assess the similarity between two sequences of characters simply based on their longest common initial subsequences. Formally, given two sequences $W' = < w_1', w_2', \ldots, w_n' >$ and $W'' = < w_1'', w_2'', \ldots, w_n'' >$, their *Prefix Similarity* is computed as:

$$ps(W', W'') = \min(\{i \mid 1 \le i \le \min(n, m) \wedge \delta(w'_i, w''_i) = 0\}) - 1 \qquad (2)$$

where $\delta(w', w'')$ is the usual Kronecker function that is 1 if w' and w'' are equal, 0 otherwise. $ps(\cdot,\cdot)$ takes the least index for which the characters are different, and then subtracts 1 to obtain the greatest index at which the prefix is the same. For instance:

ps(decido,decidere) = ps(decidere,decido) = 5
ps(decido,disporre) = ps(disporre,decido) = 1

Then, we need to determine which terms have an ‘interesting’ initial sequence that might be a candidate prefix, stem or suffix. Let $\mathsf{W} = < W_1', W_2', \ldots, W_n' >$ a list of sequences of characters, where W is sorted by ascending alphabetic ordering and does not include duplicates. We define a *group* as a subsequence of W having high prefix similarity, and low prefix similarity with the items in W immediately preceding or following the sequence:

$$G = \langle W_i, \ldots, W_h \rangle \ with \ 1 \le i \le h \le n \qquad (3)$$

such that:

1. $\forall j = i+1, \ldots, h : ps(W_{j-1}, W_j) \ge \theta$;
2. If $i > 1$ then $ps(W_i, W_{i-1}) < \theta$
3. If $h < n$ then $ps(W_{h+1}, W_h) < \theta$

where the threshold is currently computed as the average of all similarities between adjacent elements in W, suitably smoothed or amplified by a factor β.

$$\theta = \beta \frac{\sum_{k=2}^{n} ps(W_{k-1}, W_k)}{n-1}$$

In the current prototype, we empirically used $\beta = 1.0$, obtaining a standard average. Note that, by construction, all the groups that can be found in *W* are disjoint, because conditions 2 and 3 ensure that the elements in the list immediately preceding and following each group cannot be included in the group. They might be included in a previous or next group, respectively, but they could also be not included in any group, when their similarity with their immediately preceding (respectively, following) element does not fulfill condition 1. So, there can be gaps of unused subsequences that do not concur to extract any group.

Each group returns a stem, as the longest subsequence common to all of its elements (see Figure 1 for a sample application of the technique). Suffixes can be then obtained from the words by removing the corresponding stem found in the previous step. Each suffix is associated to a weight given by the product of the number of terms that end with that suffix times the suffix length. This rewards longer suffixes or very frequent ones. Finally, suffixes are ranked by decreasing weight, and only those passing a given threshold are selected. The returned suffixes include the morphological changes due to word inflection. These are the typical suffixes included in the suffix lists commonly used for stemming by suffix stripping. To obtain ‘pure’ suffixes, which may be of interest for linguistic analysis purposes, the same technique can be applied to the ‘extended’ suffixes obtained in the previous step as follows. Each suffix is reversed and the list of reversed suffixes is sorted. This brings close to each other the suffixes that have the same final characters. Then the similarity of adjacent pairs of reversed suffixes is computed, and the grouping technique is applied.

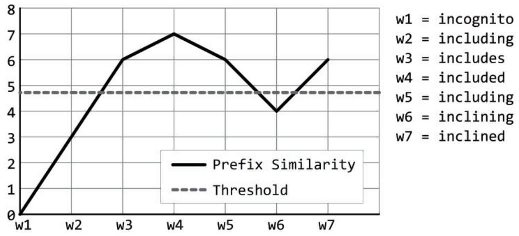

Fig. 1. Example of group identification

Summing up, the basic idea is to find in the training corpus many words having the same sequence of initial or final characters. Such sequences are likely to be prefixes/stems or standard suffixes of the language. For instance, one group returned stem ‘decid’ and suffixes ‘-e, -er’; another returned stem ‘declar’ and suffixes ‘-amus, -ation, -e’. A few considerations are worth concerning the identification of typical prefixes in a language. Differently from suffixes, prefixes do not change the function of a word, rather its meaning (e.g., ‘post-processing’ means ‘processing after’, which is a quite different intuition than just ‘processing’). So, they are often not stripped by NLP techniques. Nevertheless, identifying prefixes may be useful in some cases: first, obtaining a list of prefixes that are typical for a given language is another useful linguistic resource in its own right; second, because sometimes the NLP technique actually needs to abstract from the specific perspective with which a term is used, and consider only its fundamental meaning (e.g., ‘pre-processing’, ‘processing’ and ‘post-processing’ all basically express the same function that is carried out, albeit in different moments of a procedure). To identify frequent prefixes in a language we apply the same technique as before, working on the list of stems identified in the first step.

4. Evaluation

The proposed approach was tested on texts taken from several languages. In addition to official languages, having well-defined grammar (English, Italian, French, Latin), also a dialect from a southern Italy town (Squinzano, LE) was considered, for which no grammar, nor (of course) linguistic resource have ever been compiled. All experiments were run on a PC equipped with an Intel Q9450 2.66 GHz 64 bit processor and 2 GB RAM. Two documents were loaded for each language, taken from the Project Gutenberg repository (http://www.gutenberg.com), as reported in the following (along with the associated number of characters/bytes and time to load, including pre-processing):

1. *The picture of Dorian Gray* (O. Wilde) - English (308,067 B, 11 min)
2. *Gulliver's Travels* (J. Swift) - English (590,390 B, 8 min)
3. *Le portrait de Dorian Gray* (O. Wilde) - French (452,167 B, 17 min)
4. *Notre Dame de Paris* (V. Hugo) - French (1,025,165 B, 38 min)
5. *I Promessi Sposi* (A. Manzoni) - Italian (1,306,223 B, 40 min)
6. *La compagnia dell'anello* (Tolkien) - Italian (554,758 B, 19 min)
7. *De bello gallico* (Caesar) + Orationes (Seneca) - Latin (234,607 B, 9 min)
8. *Carmina* (Catullus) - Latin (108,258 B, 5 min)

For the dialect, a series of old stories from past centuries for children were loaded, taken from the book ‘Cuntame nnu cuntu!’ by Anna Messito (unpublished manuscript). The lexical analysis lasted 75 minutes overall, yielding the results shown in Table 1. The learned stopwords were as follows:

- **English** the, of, and, to, i, a, in, that, was, he, it, my, his, with, me, as, had, for, you, is, be, at, by, not, which, but, they, or, have, their, him, on, were, this, from, all, so, would, are, one, them, some, an, could, what, upon, her, who, about, when, there, we, any, into
- **French** de, la, et, le, a, l, il, un, les, que, d, en, une, qui, est, vous, je, des, qu, dans, du, ne, etait, ce, se, pas, elle, s, sur, n, son, avait, c, au, sa, lui, plus, pour, avec, tout, comme, ses, on, cette, y, ou, dit, mais, par, si, nous, j, me, cela, bien, m, aux, deux, fait
- **Italian** e, di, che, a, il, la, un, non, in, per, si, una, l, con, le, piu, era, ma, da, d, se, del, gli, i, come, della, al, lo, quel, ne, disse, alla, anche, s, aveva, o, quella, cosi, poi, loro, io, suo, cosa, sua, lui, nel, tutto, ci, quando, questo, altro, mi, qualche, renzo, chi, de, ora, all, dell, due, senza, ho, c, ha, uno, tutti, frodo, questa, perche, casa, tempo, ogni

- **Latin** et, in, non, ad, cum, est, quod, ut, qui, atque, esse, ex, se, a, quae, si, ac, quam, neque, sed, ab, aut, de, te, iam, me, his, mihi, quid, caesar, etiam, ne, hoc, sunt, id
- **Dialect** e, te, la, lu, cu, ca, a, se, nu, li, ia, lla, nne, tisse, comu, nnu, llu, pe, nna, le, quandu, ma, rre, ci, era, me, shta, maria, alla, allu, sse, poi, iddhra, lli, jou, puru, tuttu, nthra

Almost all stopwords were correct, as confirmed by computing the precision with respect to existing standard stopword lists for those languages (except for dialect, where such a resource is not available). A few exceptions concern common or proper nouns such as: 'frodo', 'renzo', 'casa' in Italian; 'caesar' in Latin; 'tisse', 'rre', 'maria' for dialect[†]. These cases lead precision to 1.0 for English and French, (72 - 3)/72 = 0.96 for Italian, (35 - 1)/35 = 0.97 for Latin and (38 - 3)/38 = 0.92 for dialect (it is 1.0 for the other languages). While these may seem errors, there is both an explanation and a justification for their presence. The explanation is that there were few documents for each language, and many of them were novels, so the names of the characters (e.g., Frodo and Renzo in Italian, or Maria and 'rre' [= 'king'] in the dialect) are more frequent than expected in general in the language. As regards the dialect, the training texts were stories for children, hence the high frequency of the term 'tisse' [= 'said']. It is expected that adding more (and more varied) training texts will fix these exceptions. The justification is that this behavior is not fully undesirable. Indeed, just because of the same reason expressed above, these terms can be considered domain-specific, and hence so widespread in the documents that their information content is almost null. Thus, it may be considered correct to take them as stopwords.

Table 1. Results of linguistic analysis

	English	French	Italian	Latin	Dialect
Terms	11 053	20 099	24 008	13 788	5 174
Stopwords	54	59	72	35	38
Suffixes	188	275	187	220	86
Stems	976	2 103	1 770	996	413
Reduction (%)	91.17	89.54	92.63	92.78	92.02
Prefixes	33	34	69	29	16

As to word normalization, using the learned suffixes to normalize the terms in the collection, the reduction of the set of words to about 1/10 of the original size confirms that the proposed learning technique is actually effective.

As regards prefixes, since they are not fundamental for the purpose of pre-processing the text in order to apply higher-level NLP techniques, we did not run thorough experiments, but just applied the same setting as for the stems. This resulted in a sensible number of prefixes, but a varied quality level depending on the language. English was the best one, but in general the returned suffixes tend to be too long (e.g., des-, del- should be just de-). This can be probably solved by acting on the thresholds, because prefixes must have a much higher frequency than stems.

To assess the performance, it would be unfair to compare the performance of our approach to that of grammar-based techniques, since our fundamental assumption is that no knowledge at all is available about the language to be processed. So, we compared the suffixes returned by the system to those in a publicly available resource for English[‡], including both a general list and a list restricted to most common suffixes. Results for all different thresholds $t \in [0,707]$ are very good in terms of Area-Under-Curve: 0.58 AUC-PR and 0.90 AUC-ROC. The average error rate[§] over all thresholds is 0.03. The best performance was obtained for $t = 42$, where both Precision and Recall are 0.61. Specifically, for $t = 42$, 30 suffixes are found, of which 19 appear in the 31-suffixes restricted list.

[†] Standard stopword lists include 'deux' for French, and 'cosa', 'tutto', 'tutti', 'ora','tempo' for Italian.

[‡] http://www.darke.k12.oh.us/curriculum/la/suffixes.pdf

[§] ER= 1 - (TruePositives + TrueNegatives)

5. Conclusions and Future Work

While multimedia digital documents are progressively spreading, most of the content of Digital Libraries is still in the form of text, and this predominance will probably never be questioned. Except pure display of these documents, all other tasks are based on some kind of Natural Language Processing, that must be supported by suitable linguistic resources. Since these resources are clearly language-specific, they might be unavailable for several languages, and manually building them is costly, time-consuming and error-prone.

This paper proposes a methodology to automatically learn linguistic resources for a natural language starting from texts written in that language. The learned resources may enable further high-level processing of documents in that language, and/or be taken as a basis for further manual refinements. Experimental results show that its application may effectively provide useful linguistic resources in a fully automatic manner. Future work will extend and refine the proposed techniques. More experiments will be run to evaluate the performance of high-level NLP tasks based on the learned resources, and to identify more effective parameter settings.

Acknowledgements

This work was partially funded by the Italian PON 2007-2013 project PON02 00563 3489339 'Puglia@Service'.

References

1. Ahmed B, Cha SH, Tappert C. *Language identification from text using n-gram based cumulative frequency addition*. Proceedings of Student/Faculty Research Day, CSIS, Pace University, pages 12-1, 2004.
2. Brill E. *A simple rule-based Part of Speech tagger*. In HLT '91: Proceedings of the workshop on Speech and Natural Language, pages 112-116, 1992.
3. Brill E. *Some advances in transformation-based Part of Speech tagging*. In Proceedings of the 12th National Conference on Artificial Intelligence (AAAI) vol. 1, pages 722-727, 1994.
4. Brill E. *Unsupervised learning of disambiguation rules for Part of Speech tagging*. In Natural Language Processing Using Very Large Corpora Workshop, pages 1-13. Kluwer, 1995.
5. Cimiano P, Hotho A, Staab S. *Learning concept hierarchies from text corpora using formal concept analysis*. J. Artif. Int. Res., 24(1):305-339, August 2005.
6. D'Ulizia A, Ferri F, Grifoni P. *A survey of grammatical inference methods for natural language learning*. Artificial Intelligence Review, 36(1):1-27, april 2012.
7. Rotella F, Leuzzi F, Ferilli S. *Learning and exploiting concept networks with ConNeKTion*. Applied Intelligence, Springer, 2014. DOI: 10.1007/s10489-014-0543-z. (Currently available electronically)
8. Hensman S. *Construction of conceptual graph representation of texts*. In Proceedings of the Student Research Workshop at HLT-NAACL 2004, HLT-SRWS '04, pages 49-54, Stroudsburg, PA, USA, 2004. Association for Computational Linguistics.
9. Leuzzi F, Ferilli S, Rotella F. *ConNeKTion: A tool for handling conceptual graphs automatically extracted from text*. Proceedings of the 9th Italian Research Conference on Digital Libraries (IRCDL 2013), volume 385 of CCIS. Springer-Verlag Berlin Heidelberg, 2013.
10. Leuzzi F, Ferilli S, Rotella F. *Improving robustness and flexibility of concept taxonomy learning from text*. Revised Selected Papers, volume 7765 of CCIS, pages 232-244. Springer-Verlag Berlin Heidelberg, April 2013.
11. Maedche A, Staab S. *Mining ontologies from text*. In EKAW, pages 189-202, 2000.
12. Maedche A, Staab S. *The text-to-onto ontology learning environment*. In ICCS-2000 - Eight International Conference on Conceptual Structures, Software Demonstration, 2000.
13. Martins B, Silva MJ. *Language identification in web pages*. In Proceedings of the 2005 ACM symposium on Applied computing, pages 764-768. ACM, 2005.
14. Nagarajan T, Murthy HA. *Language identification using parallel syllable like unit recognition*. In Acoustics, Speech, and Signal Processing, 2004. Proceedings.(ICASSP'04). IEEE International Conference on, volume 1, pages I-401. IEEE, 2004.
15. Ogata N. *A formal ontology discovery from web documents*. In Web Intelligence: Research and Development, First Asia-Pacific Conference (WI 2001), number 2198 in Lecture Notes on Artificial Intelligence, pages 514-519. Springer-Verlag, 2001.
16. Porter MF. *An algorithm for suffix stripping*. Program, 14(3):130-137, 1980.
17. Rotella F, Ferilli S, Leuzzi F. *An approach to automated learning of conceptual graphs from text*. Proceedings, volume 7906 of Lecture Notes in Computer Science, pages 341-350. Springer, 2013.
18. Shamsfard M, Barforoush AA. *Learning ontologies from natural language texts*. Int. J. Hum.-Comput. Stud., 60(1):17-63, jan 2004.
19. Velardi P, Navigli R, Cucchiarelli A, Neri F. *Evaluation of OntoLearn, a methodology for automatic population of domain ontologies*. In Ontology Learning from Text: Methods, Applications and Evaluation. IOS Press, 2006.