Automatic Ontology Population from News

Hernani Costa and Alexandre Miguel Pinto
ATCM Lab work, Technical Report
CISUC, University of Coimbra
Coimbra, Portugal
{hpcosta, ampinto}@dei.uc.pt

Abstract—**Automatic keyphrase extraction is a useful tool in many text related applications, such as clustering and summarisation. Given such importance, systems capable of automatically extracting and representing keyphrases play an important role in Natural Language Processing. In this paper, we present a system with the purpose of extracting keyphrases from heterogeneous Web sources and formally represent them. After exploiting a popular algorithm (KEA), the outcomes revealed to be promising. These keyphrases are then used to populate an Ontology. In the end, a clustering algorithm was used to classify similar objects into different groups.**

I. Introduction

Nowadays we live in a world that is surrounded by information, most of the times provided as natural language text. In order to exploit this written data, many applications are being developed for performing different tasks where understanding the meaning of natural language is critical. Knowledge management [1], exchange of electronic information [2] or the Semantic Web [3] are just some of the areas where we can see this kind of applications. The aforementioned authors demonstrate that Natural Language Processing (NLP) [4] has become more and more dependent on semantic information and so, computational access to such type of knowledge is important and some times indispensable.

In particular, keyphrases provide a brief summary of a document's contents. More specifically, keyphrases are a concise representation of documents and usually are extracted directly from the original text. As large document collections, such as news, become widespread and are created every second, the value of such summary information increases. With this in mind, keyphrases are particularly useful because they can be interpreted individually and independently of each other. They can be used, for example in Information Retrieval (IR) systems as descriptions of the documents (usually, returned by a query, as the basis for search indexes, as a way of browsing a collection, and as a document clustering technique) [5], [6]. Nevertheless, keyphrases are usually chosen manually. Despite being less prone to errors, this task is hardly repeatable, time-consuming and sometimes subjective.

In this paper, we present a system with the purpose of extracting keyphrases from heterogeneous Web sources and, representing them with a graph-based formalism. The goal of this system is to ascertain how well-suited are machine learning approaches for keyphrases extraction task, in a completely automatic fashion. Another goal is to represent the resulting knowledge into a well-defined knowledge base, in this case an ontology, that will provide support to a News Recommender System (RS). In order to send a list of main topics to the user, one of the requirements of the RS is to model the user's preferences These main topics were created through a clustering algorithm that joins similar objects into the same topic.

The remaining of the paper starts with a description of the existing approaches for extract and represent keyphrases (section II). Then, the system's architecture is presented in section III. Finally, before concluding (section V), our experiments are described and their results presented (section IV).

II. Keyphrases Extraction and Representation

In this section, we explain four existing approaches used to extract keyphrases from text (section II-A). Then, some tools and libraries are analysed in order to understand their possible application in this work (section II-B).

A. Existing Approaches

Automatic keyphrases extraction is the task of identifying a small set of words, keywords, keyphrases, or key segments from a document/content that can characterize the meaning of the document [7]. This task should be done repeatedly and with either minimal or no human intervention. The main goal of automatic extraction is to apply the available computational power to the problems of access and discoverability, improving the organisation of information and the task of its retrieval. Thus, it is possible to reduce significantly the costs and drawbacks associated with human indexers [8], such as being a time-consuming, monotonous, repetitive, and hard work, which, most of the times, is subjective as it is dependent on the individual judge's criteria. Therefore, several algorithms and systems to help people perform automatic keyphrase extraction have been proposed. These methods can be divided into four categories: simple statistics, linguistics, machine learning and hybrid approaches (see for instance [8], [9]).

a) ***Simple Statistics Approaches:*** can be considered simple, with limited requirements and do not need training data in the process. These approaches tend to focus on non-linguistic features of the text, such as Term Frequency - Inverse Document Frequency (TF-IDF) [10], and position of a keyword in the sentence. Other statistical methods used to automatically index a document are N-Gram statistical information [11] and word co-occurrences [12]. The benefits of using purely statistical methods are their ease of use and the fact that they do usually return good outputs.

*b) **Linguistics Approaches:*** take advantage of the linguistic features of the words, sentences and documents. Particularly, linguistic approaches pay attention to linguistic features such as part-of-speech (POS), syntactical structure and semantic and, as a result, they tend to add value, functioning sometimes as filters for bad keywords.

There are some authors that have compared purely statistical methods with linguistic approaches ([13], [14]) and, the results indicate that the use of linguistic features yield remarkable improvements. Nevertheless, most of the linguistic methods in the literature are mixed methods, incorporating linguistic methods with common statistical measures (such as TF-IDF).

*c) **Machine Learning Approaches:*** can be used to automatically extract keyphrases from documents. One of the most used machine learning methods is supervised learning. This approach uses a set of training documents, each of which has a range of human-chosen keywords. Then the gained knowledge is applied to find keywords from new documents.

For example, the Keyphrase Extraction Algorithm (KEA) [15] is the reference for domain-based extraction of technical keyphrases. It takes advantage of machine learning techniques and the Naive Bayes formula to automatically discover keyphrases from new documents. Besides that, there have been other authors (see for instance [16], [17]) who approach the problem of automatically extracting keyphrases from text in a supervised learning fashion.

*d) **Hybrid Approaches:*** mainly combine the methods mentioned above or use some heuristic knowledge in the task of keyword extraction (e.g., the position, length, layout feature of the words, html tags around of the words, etc.) (see [18]). As we know, automatic keyphrase extraction is faster and less expensive than human intervention, and, in many cases, automatic keyphrases algorithms outperforms human indexers [19]. However, the current solutions require either training examples or domain specific knowledge.

B. Keyphrases Search Tools and Libraries

We now present some tools and libraries for keyphrase searching, that will be studied and analysed in order to understand their possible application in this work.

*e) **Weka:*** [1] [20] is a collection of machine learning algorithms for data mining tasks. Weka can either be used in the form of a library (integrated and used in Java), or applied directly to a dataset. Among the possible applications that Weka can perform, we can mention: data pre-processing, classification, regression, clustering, association rules and visualization.

*f) **Kea:*** [2] (Keyword Extraction Algorithm) [15] is an algorithm for extracting keyphrases from documents. Kea includes a cut-down version of the Weka machine learning workbench (more specifically, Kea uses Weka machine learning techniques in the keyphrases extraction process). It is implemented in Java, platform independent and distributed under the GNU (General Public License). Moreover, it can be either used for free indexing or for indexing with a controlled vocabulary.

In free indexing, keyphrases are significant terms that appear in the document. The advantage of this process is that it can be applied to any document. The disadvantages are poor quality of extracted phrases (compared to controlled indexing) and the indexing is not consistent. In contrast, controlled indexing has the advantage that all documents are indexed in a consistent way disregarding their wording. For this purpose, the keyphrases are chosen from a controlled vocabulary (a dictionary, thesaurus, or a list of terms). For example, two documents, one about "personal computer" and another one about "notebooks", would be indexed with the same term, which is the preferred term in the controlled vocabulary to describe this concept.

In addition, we also analysed some Web applications that offer services (in the form of an API – Application Programming Interface) which are useful in the context of this work. These tools are based on knowledge obtained from search engines.

*g) **AlchemyAPI:*** [3] [21] uses NLP technology and machine learning algorithms to analyse content and extracting semantic meta-data (information about people, places, companies, topics, etc.). AlchemyAPI provides both free and commercial support. The service access is done through an API (available to Android OS, Java, Perl, Ruby, Python, PHP-5, C/C++ and C#). Some of the functions available are: named-entity recognition (NER), concept tagging keywords and term extraction, sentimental analysis, topic categorisation and text classification, author extraction, language detection (AlchemyAPI identifies more than 95 languages[4]), text extraction and Web page cleaning, structured data extraction and content scraping, among others features.

*h) **Yahoo Content Analysis API:*** [5] is a service provided by Yahoo. The API is able to extract keyphrases from the content (text or a URL) and rank them based on their overall importance to the document. The Yahoo Content Analysis API detects categories, entities/concepts, and relationships within unstructured content. It ranks those detected entities/concepts by their overall relevance, resolves those if possible into Wikipedia pages, and annotates tags with relevant meta-data.

The Content Analysis service is limited to 10,000 queries per IP address per day and 1,000 unsigned calls per hour (as defined in the YQL Terms of Use[6]). Furthermore, Yahoo Content Analysis is available for noncommercial use.

*i) **Summary:*** In this section, we have presented some tools and libraries in order to understand their possible integration in our system. Most of them embrace several NLP tasks, but they are not able to be directly compared. Even so, in table I these tools (**Tool**) are presented in an overview of all of them (e.g., the available programming language (**P.**

[1] http://www.cs.waikato.ac.nz/ml/weka

[2] http://www.nzdl.org/Kea

[3] http://www.alchemyapi.com

[4] http://www.alchemyapi.com/api/lang/langs.html

[5] http://developer.yahoo.com/search/content/V2/contentAnalysis.html

[6] http://info.yahoo.com/legal/us/yahoo/yql/yql-4307.html

Lang.), the languages supported (**Language**), its availability (**Availability**), the most relevant tasks (**Assignment**)), in order to better understand its relevance to this work.

TABLE I
COMPARATIVE ANALYSIS OF SEVERAL TOOLS.

Tool	P. Lang.	Language	Availability	Assignment
Weka	Java	any	Public Domain	Data pre-processing, classification, regression, clustering, association rules and visualization
Kea	Java	any	Public Domain	Keyphrases extraction
AlchemyAPI	HTTP request	any	Academic and Proprietary	Keyphrases extraction, NER, topic categorisation, among other tasks
Yahoo! Content Analysis API	HTTP request	any	Public Domain	Keyphrases extraction

C. Data Storage

In recent years a number of new systems have been designed to provide better horizontal scalability for simple read and write database operations, distributed over many servers. In contrast, traditional databases have comparatively little or no ability to scale horizontally on these applications.

Unlike relational databases (such as MySQL and SQLite), document-oriented databases[7] do not store data and relationships in tables. Instead, each database is a collection of independent documents. For example, graph databases uses graph structures with nodes, edges, and properties to represent and store data (e.g., Neo4j [8]). By definition, a graph database is any storage system that provides index-free adjacency. This means that every element contains a direct pointer to its adjacent element and no index lookups are necessary. Moreover, compared with relational databases, graph databases are often faster for associative data sets, and map more directly to the structure of object-oriented applications.

Thus, graph databases are a powerful tool for graph-like queries, for example computing the shortest path between two nodes in the graph. Other graph-like queries can be performed over a graph database in a natural way (e.g., graph's diameter computations or community detection).

In short, this kind of database is designed for data whose relations are well represented as a graph (elements interconnected with an undetermined number of relations between them). The kind of data could be social relations, public transport links, road maps, network topologies, among others. Graph databases are excellent for storing and analysing social network information.

D. Graph Clustering

Graphs are structures formed by a set of vertices (also called nodes) and a set of edges that are connections between pairs of vertices. Any nonuniform data contains underlying structure due to the heterogeneity of the data. The process of identifying this structure in terms of grouping the data elements is called clustering, and the resulting groups are called clusters. Moreover, the task of grouping the vertices of the graph into clusters, taking into consideration the edge structure of the graph, is called graph clustering [22]. The main goal of graph clustering is to partition vertices in a large graph into different clusters, based on various criteria such as vertex connectivity or neighbourhood similarity. Graph clustering techniques are very useful for detecting connections between groups in a large graph [23].

However, not all graphs have a structure with natural clusters. Nonetheless, a clustering algorithm outputs a clustering for any input graph. For example, if the structure of the graph is completely uniform, with the edges likewise distributed over the set of vertices, the clustering computed by any algorithm will be rather arbitrary. Quality measures and visualizations will help to determine whether there are significant clusters present in the graph and whether a given clustering reveals them or not.

III. System Architecture

As referred in section I, the first goal of our work is the creation of an ontology from Portuguese news, in an automatic fashion. To do that, we have designed a modular system (see figure 1) to: gather news from heterogeneous Web sources (Aggregator module); save this information to a database (Database module); extract keyphrases from the news (KEA module); represent keyphrases into a triple store (Triple Store module); and identify clusters of keyphrases (Graph-Clustering module) class-candidates in the ontology.

The Aggregator module, as it name suggests, is responsible for the aggregation process and also for extracting all the news' properties. The KEA module extracts keyphrases from the news' description. The Database module is used to store both the news' properties and the output keyphrases extracted with KEA. The Triple Store module stores the keyphrases extracted by KEA into an ontology. And finally, these clusters can then be used to induce other classes in a different, automatically constructed, ontology (Graph-Clustering).

A. Main algorithm

The process work-flow is simple, first we give some input documents, in this case *RSS feeds*, gathered from different sources, to the Aggregator module. This module is responsible, not only for the aggregation process, but also for extracting all the news' properties (e.g., title, description, link and source). Then, these properties are stored into a traditional database (in this work, we used MySQL, an open source RDBMS[9]). In this project, we exploit seven different news topics: *Science*, *World*, *Economy*, *Technology*, *Macintosh*, *Video-DVD* and *Cinema*, retrieved from the search RSS engine Yahoo[10]. All the text is written in Portuguese.

[7] The term "document-oriented database" has grown with the use of the term NoSQL itself.

[8] http://neo4j.org

[9] RDBMS, acronym for Relational Database Management System.

[10] http://br.noticias.yahoo.com/mapa

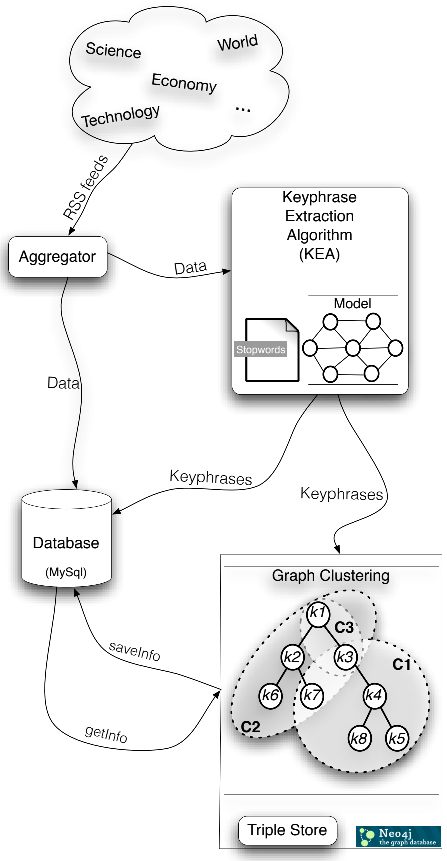

Fig. 1. System's Architecture.

The KEA module uses the title and the description of the news to extract keyphrases, in a completely automatic fashion (see section II-B and IV-A). In detail, KEA takes advantage of the previously trained model and a list of stopwords[11], see section IV-A for more details about the results obtained. Then, the outputs produced by the KEA module are stored not only in the MySQL database, but also sent to the Triple Store module.

The triple store chosen to store the keyphrases was the Neo4j [8] , a graph database (see section II-C). Similar to a triple store, Neo4j uses nodes to represent classes and the edges to describe the object properties.

Another system requirement is the use of a cluster (C) algorithm to group sets of similar keyphrases (k), represented in figure 1 as C_n and k_n, respectively, where $_n \in N$. To identify these domains, it was necessary to create associations between "close in context" objects, i.e., clusters (see section IV-C). These associations will allow other systems to easily find related keyphrases by browsing or searching.

IV. Experiment

In this section, we describe the experiment carried out to populate an ontology from Portuguese news. Section IV-A describes the algorithm used to extract keyphrases and also the obtained results. In section IV-B, it is presented how the ontology was created and populated. Finally, section IV-C shows the clustering process used to identify main topics in the ontology.

A. KEA's Model Evaluation

Due to the limited time, we only obtained 840 news after running our system for about 8 days (see table II). From these, we selected $\approx$ 17.7% random news, in order to created the KEA dataset, i.e., 149 news. This sample size[12] was determined by using a confidence level of 95% and a confidence interval of 7.3%. From these 149 random news, we manually assign keyphrases to them. Then, we used 104 news ($\approx$70%) to train the KEA algorithm. After the training process, the remaining 45 news ($\approx$30%) were used to test the algorithm's precision, recall and F_1. Table II presents the resulted obtained. In the first column it is presented the number of news in the system for all the topics (label **S**). The second column presents the number of news manually evaluated (**E**). Then, the percentage of these used in the training process (**D**) and in the test (**T**), are presented in the third and fourth column. The remaining columns describe the obtained values for precision (**P**), recall (**R**) and F_1 measures.

TABLE II
KEA's Evaluation.

	S	**E**	**D**(%)	**T**(%)	**P**(%)	**R**(%)	F_1(%)
Science	85	20	70.0	30.0	85.4	57.1	66.5
World	632	41	70.7	29.3	66.6	63.0	61.3
Economy	35	24	70.8	29.2	60.7	48.7	52.4
Technology	30	22	72.3	27.3	47.9	72.2	53.5
Macintosh	13	10	70.0	30.0	46.8	68.3	54.4
Video-DVD	26	16	68.8	31.2	37.5	56.4	43.3
Cinema	19	16	68.8	31.2	27.5	39.1	28.9
	Total		***Average***				
	840	***149***	***70.3***	***29.7***	***53.2***	***57.8***	***51.3***

As we can see the categories *Video-DVD* and *Cinema* do not have high results (see for instance F_1 equal to 43.3% and 28.9%, respectively). However, this is due to the fact that we have only a few set of news for these categories, at our disposal (see column **S**). Consequently, it was only used a small set of news in the training process. Nevertheless, we consider the results promising for these categories, see for example the recall value 56.4% for *Video-DVD*. In the future, it is imperative a deep study and sensitivity analysis for the evolution of the precision, recall and F_1 measures by increasing the sets of news.

On one hand we have lower accuracy, e.g., *Cinema*, on the other hand the category *Science* and *World* have higher accuracy. This can be easily explained by the fact that the number of news in the systems for the categories *Science* and *World* are greater than *Cinema*, for example. Additionally, the text inside each category is different, for instance the category *Cinema* contains mostly named-entities such as the names of

[11] Stopwords are general and very frequent words, usually functional, like prepositions, determiners or pronouns.

[12] http://www.macorr.com/sample-size-calculator.htm

persons, directors, movies, etc., and KEA do not perform well for this kind of information, see for example the following news: *"O diretor Alain Resnais, autor de filmes que entraram para a história do cinema como "Noite e Neblina", "Meu Tio da América" ou "La guerre est finie", completa neste domingo 90 anos em plena atividade, tendo apresentado seu novo filme, "Vous n'avez encore rien vu", na mais recente edição do Festival de Cannes, em maio."* Nonetheless, the *Science* category contains completely different text, see for example: *"Os adolescentes americanos fumam e bebem menos do que os jovens europeus, mas drogam-se mais, revelou um estudo realizado nos Estados Unidos e em 36 países europeus, divulgado nesta sexta-feira pela Universidade de Michigan".*

A possible improvement in the extraction process is the use of an automatic identification of noun compounds and named entities, as demonstrated by István Nagy et al. [24], where they proved that the integration of noun compound and named entity related features into a keyphrase extractor is more effective than the model that not include them.

B. *Using Neo4j as a triplestore*

In this topic, we describe how we defined and used Neo4j to populate our news ontology. As we previously mentioned, Neo4j can be used as a triple RDF store, and has SPARQL[13] implementation.

Firstly, we have created the RDF schema in the Protégé[14], a free open source ontology editor and knowledge-base framework (figure 2 presents all the classes, objects and relations used). As we can see in figure 2 every keyphrase occurs in news (resulting in the triple t_n=$\{keyphrase_n$, occursIn, $news_n\}$, where $_n \in N$) and belongs to a cluster (see section IV-C for more details). After importing the RDF schema into Neo4j, the algorithm associates all the keyphrases extracted from news, and inserts the resulting triples into the ontology. In order to analyse the keyphrases importance in the ontology, the system has been running for one week (i.e., 7 days × 24 hours), in order to gather all the daily news. As a result, we obtained 4683 nodes (keyphrases), 18569 edges (connections between the keyphrases and the news). In detail, 156 nodes only have more then three occurrences, 352 two and 4175 only one occurrence. This means that our graph is very dispersed, but for this work this is not a problem because we only want to do a proof of concept, as we describe in the next topic (section IV-C).

C. *Graph-Clustering*

In this topic, we describe our first experiment to identify clusters in the ontology. The clustering process was performed using the Gephi[15] platform. Gephi can be described as a powerful interactive visualisation and exploration platform for all kinds of networks and complex systems, dynamic and hierarchical graphs.

A deep study about the appropriate clustering algorithm and parameter settings (including values such as the distance function to use, a density threshold or the number of expected clusters) need to be done in the future. Nevertheless, due to the limited time, we used the modularity algorithm[16] [25] as a proof of concept (already available through the Gephi platform). This algorithm uses a simple method to extract the community structure of large networks, using a heuristic method that is based on modularity optimisation.

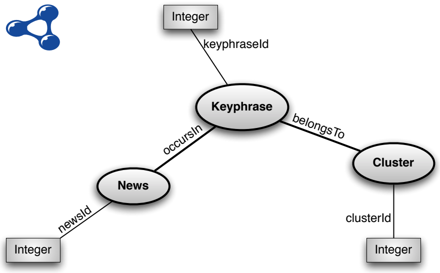

Fig. 2. RDF Schema used to create the ontotlogy.

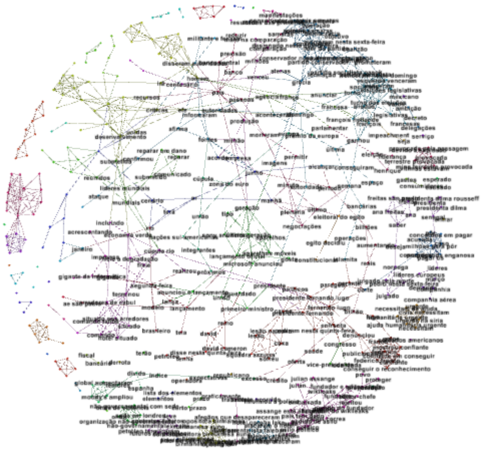

Fig. 3. Overview of the resulted clusters using colors.

Figure 3 shows the obtained clusters, and figure 4 presents a sample of one of the resulted clusters. As we can see in figure 4, there are some problems in the extraction process, for instance the keyphrases "perante um partido" or "presidente fernando" are not named-entities as expected. A correct keyphrases for "presidente fernando" can be found in the same cluster as "presidente fernando lugo". Still, these keyphrases can not be considered completely incorrect. Nevertheless, the keyphrase extraction process need to be improved, for example, through the application of a pre-filter and a post-filter as we explain in section V.

V. Concluding Remarks and Future Work

Although the limited time, we have created a system capable of extracting keyphrases from heterogeneous Web sources, formally represent them into an ontology and applied a clustering algorithm in the ontology to identify main topics.

The obtained results from the extraction process revealed F_1 scores higher than 60%. Moreover, this system can be seen

[13] http://sparql.org

[14] http://protege.stanford.edu

[15] http://gephi.org

[16] https://sites.google.com/site/findcommunities

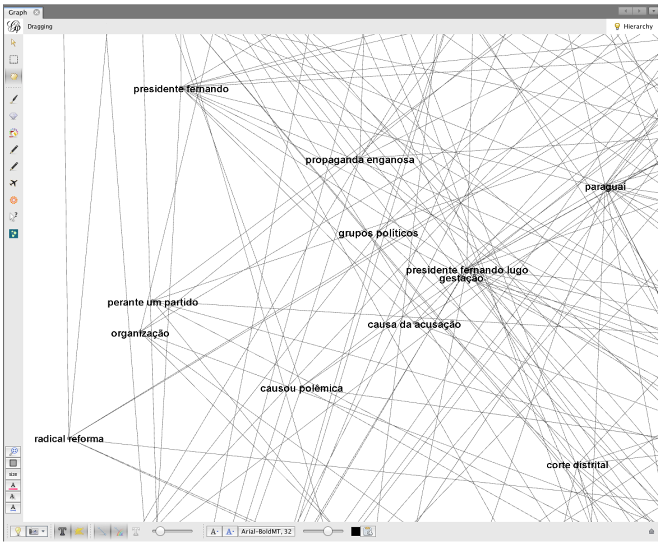

Fig. 4. Graph clustering example.

as a prototype capable of provide support to other systems, such as a News Recommender Systems. Even though this work is made for Portuguese, it can be easily adapted to other languages.

Nevertheless, the extraction algorithm can be improved by increasing the stopword list and and the usage of a POS tagger or even a grammar to create rules before the extraction process (see for instance [26]). In addition, we believe that a post-filtering appliance to identify the keyphrases POS (for example by discarding verbs) and rate the keyphrases (like done in [27]), will improve the algorithm accuracy. The rating process can be done, for instance by taking advantage of other resources, such as Onto.PT[17] or BDpedia[18]. Another study we intend to explore in the future is the comparison between other algorithms capable of extracting keyphrases from text, such as those presented in section II-B. Furthermore, we will use more sources and consequently more news to perform these experiences in order to observe the effect in the system accuracy.

In the end of this work, important contributions for the computational processing of Portuguese language are provided, such as computational tools capable of aggregating news from different sources, extracting keyphrases from text, the methodology to automatically populate an ontology, and also the appliance of clustering algorithms in the news ontology.

References

[1] B. R. Gaines and M. L. Shaw, "Knowledge Acquisition, Modeling and Inference through the World Wide Web," *Int. Journal of Human-Computer Studies*, vol. 46, pp. 729–759, 1997.

[2] A. H. Boss and J. B. Ritter, *Electronic Data Interchange Agreements - A Guide and Sourcebook*. Publication CCI No. 517, Paris, 1993.

[3] T. Berners-Lee, J. Hendler, and O. Lassila, "The Semantic Web: Scientific American," *Scientific American*, 2001.

[4] D. Jurafsky and J. H. Martin, *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition (Prentice Hall Series in Artificial Intelligence)*. Prentice Hall, 2000.

[5] C. Gutwin, G. Paynter, I. Witten, C. Nevill-Manning, and E. Frank, "Improving Browsing in Digital Libraries with Keyphrase Indexes," Dept. of Computer Science, University of Saskatchewan, Canada, Tech. Rep., 1998.

[6] I. H. Witten, "Browsing around a digital library," in *SODA*, 2003, pp. 99–99.

[7] V. Hristidis, Y. Papakonstantinou, and A. Balmin, "Keyword Proximity Search on XML Graphs," in *ICDE*. CA, USA: IEEE Computer Society, 2003, pp. 367–378.

[8] M. J. Giarlo, "A Comparative Analysis of Keyword Extraction Techniques," Rutgers, The State University of New Jersey, 2005.

[9] C. Zhang, "Automatic Keyword Extraction from Documents Using Conditional Random Fields," *Journal of Computational Information Systems*, vol. 4, no. 3, pp. 1169–1180, 2008.

[10] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Information Processing and Management: an Int. Journal*, vol. 24, no. 5, pp. 513–523, 1988.

[11] L. Guo, F. Shao, C. Botev, and J. Shanmugasundaram, "XRANK: ranked keyword search over XML documents," in *Proc. ACM SIGMOD Int. Conf. on Management of Data*, ser. SIGMOD'03. NY, USA: ACM, 2003, pp. 16–27.

[12] Y. Matsuo and M. Ishizuka, "Keyword Extraction from a Single Document using Word Co-occurrence Statistical Information," *Int. Journal on Artificial Intelligence Tools*, vol. 13, no. 1, pp. 157–169, 2004.

[13] L. van der Plas, V. Pallotta, M. Rajman, and H. Ghorbel, "Automatic Keyword Extraction from Spoken Text. A Comparison of two Lexical Resources: the EDR and WordNet," *CoRR*, vol. cs.CL/0410062, 2004.

[14] A. Hulth, "Improved automatic keyword extraction given more linguistic knowledge," in *Proc. 2003 Conf. on Empirical Methods in Natural Language Processing*, ser. EMNLP'03. PA, USA: ACL, 2003, pp. 216–223.

[15] I. H. Witten, G. W. Paynter, E. Frank, C. Gutwin, and C. G. Nevill-Manning, "KEA: practical automatic keyphrase extraction," in *Proc. 4^{th} ACM Conf. on Digital Libraries*, ser. DL'99. NY, USA: ACM, 1999, pp. 254–255.

[16] P. D. Turney, "Learning algorithms for keyphrase extraction," *Information Retrieval*, vol. 2, no. 4, pp. 303–336, 2000.

[17] F. Fukumoto and Y. Sekiguchi, "Keyword extraction of radio news using term weighting with an encyclopedia and newspaper articles," in *Proc. 21^{st} Annual Inter. ACM SIGIR Conf. on Research and Development in Information Retrieval*, ser. SIGIR'98, Y. Suzuki, Ed. NY, USA: ACM, 1998, pp. 373–374.

[18] J. B. K. Humphreys, "PhraseRate: An HTML Keyphrase Extractor," Dept. of Computer Science, University of California, Riverside, California, USA, Tech. Rep., 2002.

[19] O. Medelyan, "Human-competitive automatic topic indexing," Ph.D. dissertation, University of Waikato, New Zealand, Waikato, New Zealand, 2009.

[20] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: an update," *SIGKDD Explor. Newsl.*, vol. 11, no. 1, pp. 10–18, 2009.

[21] Orchestr8 LLC, "AlchemyAPI," http://www.alchemyapi.com, retrieved on 2012-05-20, 2012.

[22] S. Schaeffer, "Graph clustering," *Computer Science Review*, vol. 1, no. 1, pp. 27–64, 2007.

[23] Y. Zhou, H. Cheng, and J. X. Yu, "Graph clustering based on structural/attribute similarities," *VLDB Endowment*, vol. 2, no. 1, pp. 718–729, 2009.

[24] T. István Nagy, G. Berend, and V. Vincze, "Noun Compound and Named Entity Recognition and their Usability in Keyphrase Extraction," in *RANLP*, 2011, pp. 162–169.

[25] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 10, pp. 1–8, 2008.

[26] H. Costa, *Automatic Extraction and Validation of Lexical Ontologies from text: Creating Lexical Ontologies from text*. LAP LAMBERT Academic Publishing, 2011.

[27] H. Costa, H. Gonçalo Oliveira, and P. Gomes, "The Impact of Distributional Metrics in the Quality of Relational Triples," in *Proc. ECAI 2010, Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities (LaTeCH)*, 2010.

[17] http://ontopt.dei.uc.pt

[18] http://dbpedia.org