**PhD Dissertation**

---

**International Doctorate School in Information and Communication Technologies**

DIT - University of Trento

# A Semi-supervised Approach for Improving Search, Navigation and Data Quality in Autonomous Digital Libraries

Mikalai Krapivin

Advisor:

Prof. Maurizio Marchese

Università degli Studi di Trento

---

February 2010

# Abstract

*Rapid growth of Digital Libraries requires automated procedures for extracting, processing and representing of an information. Moreover, development of WEB 2.0 presents new challenges in web-sites construction and user interfaces creation. New features for users communication are appearing, also new methods of visualization of content like tags are coming. A single quick glance at the "cloud of tags" may hint us a lot about a content. We propose kind of tagging for Autonomous Digital Libraries (ADL), the ones who automatically created. The large dimensionality of ADLs does not allow a manual documents processing, so that, tags or keyphrases must be extracted automatically.*

*The first challenge we met is the absence of a dataset which has proven quality and big size. So we have constructed the one using web-crawling. We restrict ourselves with academic ADLs domain, so the dataset we built consists of 2000 scientific papers, published in the short period from 2003 to 2006 in the Computer Science domain. Dataset has fulltexts of papers, refined from LaTeX rests with help of maximum entropy machine learning, and according keyphrases lists held separately.*

*For keyphrases extraction we use the prepared dataset of 2000 scientific papers. We compared several Machine Learning techniques, namely Random Forest, Support Vector Machines and recently invented Fast Local Kernel Machines with the baseline Naïve Bayes learning-based system KEA. As it has been obtained, Random Forest is the most precise method* *outperforming baseline keyphrases extraction system KEA by 36%, and it is the tradeoff between accuracy and computational speed.*

*The other problem of ADLs is caused by a large dimensionality rather than by automated way of ADL creation. It is the proper ranking problem. Indeed, having huge quantity of documents devoted to a specific topic, it is hard to compile a set of most important up-to-date and relevant papers. Any, even very narrow and specific scientific domain has it's own state-of-the-art with a lot of incremental, duplicated or "noisy" papers. There is a challenge to withdraw the most important set of paper feasible for reading and investigation.*

*For the ranking problem we initially take three main metrics that we believe significant; the standard citation count, the more and more popular Hirsch index, and a variation of PageRank applied to scientific papers (called PaperRank), that is appealing as it mirrors proven and successful algorithms for ranking web pages. As part of analyzing them, we develop generally applicable techniques and metrics for qualitatively and quantitatively analyzing indexes that evaluate content and people, as well as for understanding the causes of their different behaviors. We put the developed ranking techniques at work on a dataset of over 266,000 ACM papers, and discovered that the difference in ranking results is indeed very significant.*

*The mock-up of plug-in for ADL used both tagged search and ranking have been built over 266,000 of papers (ACM storage) and placed in the web. This is going to be a part of the European project LIQUIDPUB.*

**Keywords:**
Page Rank based Indexes, Automated Keyphrases Extraction, Autonomous Digital Libraries, Large Dataset Preparation.

*Conditio sine qua non*

# Acknowledgments

I would like to express my gratitude to people who was supporting me, helping me and making me so sure I will do it and defend my PhD. First of all I would like to thank **Professor Maurizio Marchese** for continuous help and support during all my thesis steps, starting from comprehensive exam and finishing with the structure of the final thesis. I want to thank Professor Marchese for giving me support in different life situations, for helping me to buy a car, translating from Italian and many other things, for his patience and for encouraging me.

My appreciate to Professor Fabio Casati, Professor Fausto Giunchiglia, Professor Enrico Blanzieri, Professor Raymond Ng, Mr. Andrei Yadrantsau, Mr. Nicola Segata, Mr. Aliaksander Autaeu for many useful discussions.

I want to kindly thank my good friend Viktor Pravdin for discussing many non work related situations, drinking wine and successful fishing together!

Huge thanks to my wife Tania and my brother Viktor who have been inspiring me all time since I left my home country. Also great thanks to all russian language speaking people in Trento, specially Uladzimir Kharkevich (my "twin brother"), Andrej Somov, Ivan Minakov, Ivan Tankoev, Max Khadkevich, Alexander Birukou, Volha Bryl and Aliaksandr Autaeu for fruitful and interesting work together.

Talking to Professors Lee Giles, Alexander Smola, Raymond Ng and Fausto Giunchiglia was amazing! I want to thank University of Trento for this opportunity.

# Publications list

## Journal papers

1) Mikalai Krapivin and Maurizio Marchese and Fabio Casati, *Exploring and Understanding Citation-based Scientific Metrics*, Advances in Complex Systems (ACS) Journal 2010, accepted for publication Dec 14, 2009.

## Conference papers

1) Mikalai Krapivin and Maurizio Marchese and Fabio Casati, *Exploring and Understanding Scientific Metrics in Citation Networks*, COMPLEX 2009, ps. 1550–1563, [1], Shanghai, China.

2) Mikalai Krapivin and Maurizio Marchese, *Focused Page Rank in Scientific Papers Ranking*, ICADL 2008, ps. 144–153, [2], Bali, Indonesia.

3) M. Krapivin and M. Marchese and A. Yadrantsau and Yanchun Liang", *Automated key-phrases extraction from scientific papers using domain and linguistic knowledge*, Digital Information Management. ICDIM 2008, ps. 105-112, London, GB.

## Workshop papers

1) Mikalai Krapivin, *Focused Page Rank Application for Scientific Papers Ranking in Digital Libraries*[3], ECDL/VLDL 2008, Aarhus, Denmark.

[1]`http://dx.doi.org/10.1007/978-3-642-02469-6_35`
[2]`http://dx.doi.org/10.1007/978-3-540-89533-6_15`
[3]`http://www.nmis.isti.cnr.it/manghi/VLDL2008/Krapivin.pdf`

## Technical reports, DISI University of Trento

1) Mikalai Krapivin, Maurizio Marchese, Fabio Casati, *Exploring and Understanding Citation-based Scientific Metrics*, Technical Report DISI-08-022, DISI, University of Trento, Italy, May 2008[4].

2) Mikalai Krapivin and Aliaksandr Autayeu and Maurizio Marchese and Enrico Blanzieri and Nicola Segata, *Improving Machine Learning Approaches for Keyphrases Extraction from Scientific Documents with Natural Language Knowledge*, Technical Report DISI-10-003, DISI, University of Trento, Italy, 2008[5].

3) Mikalai Krapivin and Aliaksandr Autayeu and Maurizio Marchese, *Large Dataset for Keyphrases Extraction*, Technical Report DISI-09-055, DISI, University of Trento, Italy, 2009[6].

## Submitted papers

1) Mikalai Krapivin, Aliaksandr Autayeu, Maurizio Marchese, Enrico Blanzieri and Nicola Segata, *Keyphrases Extraction from Scientific Documents: Improving Machine Learning Approaches with Natural Language Processing*, ICADL 2010, Gold Coast, Australia.

2) Mikalai Krapivin and Aliaksandr Autayeu and Maurizio Marchese, *Large Dataset for Keyphrase Extraction* ICADL 2010, Gold Coast, Australia.

[4]`http://eprints.biblio.unitn.it/archive/00001426/01/022.pdf`

[5]`http://eprints.biblio.unitn.it/`

[6]`http://eprints.biblio.unitn.it/archive/00001671/01/disi09055-krapivin-autayeu-marchese.pdf`

# Thesis findings

Below we list major finding and innovative aspects of the present Thesis. First, regarding to the data mining for Autonomous Digital Libraries:

1. Innovative method of combination of Machine Learning (in particular Random Forrest [9], Support Vector Machines [5], FaLK-SVM [76, 75]) with Natural Language Processing techniques: POS (Part-of-Speech) tagging [87] and parsing dependencies [64] is proposed.

2. Large Dataset for Keyphrases Extraction is prepared [47] and freely shared in my web-page[7]. This dataset consist of 2000 of scientific papers with fulltexts and according meta data. We believe it may set a ground for the future comparison of different techniques for keyphrases extraction. Since each paper in the dataset is converted from PDF, texts contained a lot of "garbage": pieces of LaTeXformulae, pieces of tables, figures etc. We have developed and implemented text refinement procedure based on Maximum Entropy [71, 70] Machine Learning method. It showed 97% of precision in cleaning of garbage.

3. 50% improvement of precision recall and F-measure [91] comparing with KEA [96] baseline method.

For ranking of items in Autonomous Digital Libraries part we can enumerate the following innovative aspects:

1. We have proposed adaptation of a wellknown PageRank algorithm to the problem of ranking of items in scientific citations graphs. Such adaptation called PaperRank is simpler for computation and easy for embedding into a Digital Library.

[7] `http://disi.unitn.it/~krapivin`

2. Tradeoff between PageRank and Citation Count named Focused Page Rank has been proposed for scientific citing.

3. We have proposed how to compute analogue of popular author related metrics – Hirsch Index based on PageRank, or PageRankHircsh.

4. New method of comparison between several ranking schemas called divergence has been proposed.

5. New experimental visualization techniques have been created specially to visualize divergence of several ranking schemas. We apply it to put hundreds thousands of ranked scientific papers in one plot.

6. The visual and theoretical explanations of differences between Citation Count and Paper Rank are done.

# Contents

# List of Tables

# List of Figures

# Chapter 1

# Introduction

This thesis has two major parts both related to Autonomous Digital Libraries:

- Ranking of items in Scientific Autonomous Digital Libraries (ADL).
- Keyphrases extraction from research papers and application of ones to the problem of navigation and ranked search through ADL.

This Chapter is the introduction to both problems.

## 1.1 Problems of Modern Digital Libraries

In the present time when information storages are getting cheaper and bigger, it seems that all humankind knowledge is turning into digital format. There are more and more Digital Libraries are arising. This PhD Thesis is dedicated to the problems actual for Scientific Digital Libraries like Citeseer [29], CiteSeerX [30] or Google Scholar [17]. With the continuous growth of Digital Libraries the problem of autonomous knowledge accumulation, warehousing and dissemination is getting more and more sharp and actual. One of the most popular knowledge accumulation methods is web crawling [28, 20, 33], popular public digital libraries CiteseerX, Google Scholar and Rexa [58] have been created exactly in this way. Crawling is a fully automated process which may be powered by some heuristics in crawling mechanism. Crawler or Spider system starts from a bunch of web-cites and then it browses the "adjacent" sites following the links inside web-pages. Getting millions (and we am sure billions in the future) of journal articles, book chapters, proceedings papers, Master and PhD thesis manuscripts, it is impossible to handle all this information manually. That is why we call crawling-based Digital Libraries "Autonomous". This means absence of human supervision (or reducing it as much as possible).

On the one hand there is a way of manual Digital Library correction and contribution, when all the community will be involved in this process, this is the very new approach, currently maturing in Trento University, Italy, called Liquid Publications Project [12][1] (below will be referred as LIQUIDPUB). Being posted to LIQUIDPUB portal a scientific document has different nature than usual publication, it is kind of "versioned" and flexible (alive, continuously changing or "liquid") contribution. This kind of community-based improved documents can be partly handled manually but anyway even in this case some automatic procedures of information retrieval must be applied to improve the quality of Digital Library and to arrange items in it.

### 1.1.1 Samples of Real Autonomous Digital Libraries

Let us briefly consider few samples of real-world public Scholar Autonomous Digital Libraries available in the Web. The biggest one is the GoogleScholar [17], it possesses the largest piece of all available in internet papers due to incredible power of Google crawler, after get crawled, the papers are dissected, understood and indexed by the Google search engine. Google

[1] www.liquidpub.org

Scholar proposes the "user interface" with the ability to search by phrase presence in any part of a paper, by author, venue, date, and subjects fallen into the following major categories:

- Biology, Life Sciences, and Environmental Science
- Business, Administration, Finance, and Economics
- Chemistry and Materials Science
- Engineering, Computer Science, and Mathematics
- Medicine, Pharmacology, and Veterinary Science
- Physics, Astronomy, and Planetary Science
- Social Sciences, Arts, and Humanities

. Second place we would give to CiteseerX [29, 30], public Scientific Digital Library, in the past dedicated just to Computer Science domain only, but in the present time it is quickly growing and now including papers from Chemistry (ChemSeer) and Physics. CiteseerX is not so broad as Google Scholar in terms of quantity of terabytes of information it owes, but it has wider spectrum of services for users, like presence of bibtex files, relative papers, disambiguated search, extraction of information from tables and figures. This is rapidly developing Scientific Autonomous Digital Library with promising future. There are some more ADLs, for instance Rexa [58], the library which is smaller than Google Scholar of CiteseerX in all dimensions, but the one has the simple (and thus attractive) but quite powerful queries mechanism, including the search by topics in general, not just in few certain broad categories like Google Scholar does. From this end Rexa is similar to the idea proposed in this thesis. There are more Digital Libraries, for instance commercial ones who do not crawl for papers, but have content submitted from numerous Conferences, Proceedings, Workshops and Journals. For instance IEEE [39], ACM [25], the piece of content of ACM we used in the present Thesis, DBLP [88], Springer [79] which is the partner of LIQUIDPUB project has been constructing in Trento University.

### 1.1.2 Information mining in ADL

Having millions of documents librarians should develop automated procedures of texts refinements, text meta information (like authorship, affiliations etc.) collecting, indexing and disambiguation. Meta information retrieval may be done automatically using various machine learning techniques [76, 95, 67, 66]. From the variety of possible problems of information extraction from scientific texts we concentrated on keyphrases extraction problem. Keyphrases may be partly interpreted as as *tags* as it is common for www community. Typically web-site user put some tags when posting a piece of text content. Clouds of such tags may be used as an additional guide through many texts. To be more concrete we have considered the problem of keyphrases extraction from scientific papers and from parts of scientific papers.

### 1.1.3 Ranking Schemas for Items in DL

The “curse of dimensionality” causes another interesting problem. There are a lot of scientific domains which hold many thousands of documents (for instance Google Scholar proposes 21,500 of papers for the ”PageRank computation” query). How to recognize the most important papers? How to cut the long tail in this “crowd” of papers? This question leads us to the problem of applying of different ranking schemas to score research papers or other scientific contributions according to their impact. The area of scientific metrics (metrics that assess the quality of scientific productions) is an emerging area of research aiming at the following two objectives:

- measuring scientific papers, so that "good" papers can be identified and so that researchers can quickly find useful contributions when studying a given field, as opposed to browsing a sea of papers, and
- measuring individual contributions, to determine the impact of a scientist and to help screen and identify candidates for hiring and promotions in industry and academia.

Proposed ranking and information extraction solutions are to be included (enhanced) in mockup of the system which is built over ACM meta information storage available through internet. In the future different parts of such system will be included into LIQUIDPUB project as a part of functionality of LIRUIDPUB portal.

## 1.2 Structure of the Thesis

The Thesis is organized in six chapters.

- Chapter 1 is the introduction with major results briefly deiscussed.
- Chapter 2 summarizes the problem to solve.
- Chapter 5 is one of two big sub-problems resolved in the present Thesis, namely it is the problem of ranking of scientific documents in the citation networks. State-of-the-art and detailed description of proposed approach are located in this Chapter.
- The second major part is about supervised keyphrases (which may be treated as *tags*) extraction from scientific documents. Chapter 4 defines the state-of-the-art in keyphrases extraction and presents the

methodology, including detailed description of Natural Language Processing, text processing and machine learning parts.

- Chapter 6 describes the prototype system developed on top of relational database system and providing interface for querying for scientific publications ranked and restricted by certain keyphase related to found documents. We used indexer provided by Lucene[32] to index scientific papers using keyphrases and full texts.
- And the last part is Chapter 7, with major results summary and discussion.

# Chapter 2

# The Problem: Improving Search, Navigation and Data Quality in ADL

This chapter briefly describes the challenges to overcome.

## 2.1 Dataset creation

State-of-the-art in keyphrases extraction is quite extensive, but it is very hard to compete with other approaches because of the following reason: there are no publicly available proven quality datasets, most of papers tell about manually collected papers/news with author-assigned keyphrases (see Chapter 3). Some people use annotators, hired experts to annotate texts with the keyphrases, but most of researches use author-assigned keyphrases. There are several types of content valid for keyphrases extraction, namely: web pages, news, scientific papers, mails, meeting records. And a size of a dataset usually varies from 10 to 500 documents. Supervised or unsupervised Machine Learning is very set-dependent task. Thus publicly available dataset like TREC [16] in Information Retrieval community is required for fair competition. That is why we prepared dataset and publish one [47] before doing the keyphrases extraction.

Summarizing from the text above we can elicit the following major problems to solve:

- Propose sufficiently large dataset for keyphrases extraction problem.
- Refine such dataset and make it as linguistically clear as possible.
- Develop new, more precise than state-of-the-art methods of keyphrases extraction.
- Compare the best approaches, choose the best and fastest.

## 2.2 Improving data mining for ADLs

### 2.2.1 Keyphrases extraction

How can we cope with the exponential increase in the number of digital documents and artifacts? How do we find the relevant and related documents? These issues are central to current information age and have a central role in the Digital Libraries domain. Classifications have been used for centuries with the goal of cataloguing and searching large sets of objects. Before some document can be classified, a set of keywords/key-phrases must be retrieved from the ACM collection[1] available also at authors homepage [45]. However the definition of natural language keyphrases is a time-consuming task for human experts and show its limitations when one tries to scale the process to the very large number of current digital objects. Machine learning methods are commonly and successfully used to support unsupervised or semi-supervised such information retrieval (IR) tasks. The large majority of research work in IR domain is dedicated to the extraction of

[1]http://portal.acm.org

information from web pages, mails, news and typically short and unstructured type of digital content (see for instance [86]). A specific challenge lies in the domain of scholarly papers [36]. This challenge is related to the current development of autonomous digital libraries in academia domain [20]. Spider systems like Citeseer [29], Google Scholar [17] or Rexa [58] crawl the web seeking for scientific papers. To achieve unsupervised (or at least semi-supervised) information extraction, classification and categorization processes, machine learning techniques are often used. The task of information extraction from scholarly papers can be separated into two broad cases:

- recognition of information which is structurally present inside the body of the scientific paper (e.g. authors, mails, institutions, venues, title of a paper, keywords and/or key-phrases assigned by authors, abstract etc.);
- (ii) extraction of information which is implicitly present in the paper but there is no guarantee that (this information) is located inside a document: for example the extraction of a number of generic topics - not explicitly assigned by the authors - from a full text of a document. In this Thesis, we focus on the second, more challenging type of information extraction, i.e. extraction of implicit information.

## 2.3 Improving ranking of items in ADL

Until only 20 years ago, the number of researchers and of conferences was relatively small, and it was relatively easy to assess papers and people by looking at papers published in international journals. With small numbers, the evaluation was essentially based on looking at the paper themselves. In terms of quantitative and measurable indexes, the number of publication was the key metric (if used at all). With the explosion of the number of researchers, journals, and conferences, the "number of publications" metric progressively lost meaning. On the other hand, this same explosion increased the need for quantitative metrics at least to "Filter the noise". For example, a detailed, individual, qualitative analysis of hundreds of applications typically received today for any job postings becomes hard without quantitative measures for at least a significant preliminary filtering. Recently, the availability of online databases and Web crawling made it possible to introduce and compute indexes based on the number of citations of papers (citation count and its variations or aggregations, such as the impact factor and the h and g indexes [35]) to understand the impact of papers and scientists on the scientific community. More and more, Universities (including ours) are using these indexes as a way to filter or even decide how to fill positions by "plotting" candidates on charts based on several such indexes. The challenges that I see here are:

- Understand **why** one index is superior than another?
- Understand the true **meaning** of each new index.
- Plot them together, or represent the rankings visually for better understanding.
- Understand how to exploit the indexes, how to optimally and easily compute them.
- Invent new, more appropriate indexes, ones that figure out the spirit of modern research.

# Chapter 3

# Preparation of Large Dataset for Keyphrases Extraction

This chapter is devoted to a large dataset for machine learning-based automatic keyphrase extraction construction. The constructed dataset has a high quality and contains 2,000 scientific papers from computer science domain published by ACM. Each paper has its keyphrases assigned by the authors and verified by the reviewers. Different parts of papers, such as title and abstract, are separated, enabling extraction based on a part of text. The content of each paper is converted from PDF to plain text. The pieces of formulae, tables, figures and LaTeX mark up were removed automatically. For removal we have used Maximum Entropy Model based machine learning and achieved 97.04% precision. Preliminary investigation with help of the state-of-the-art keyphrase extraction system KEA shows keyphrases recognition accuracy improvement for refined texts.

We hope it will establish a ground for fair evaluation and comparison of different keyphrase extraction systems.

## 3.1 Introduction

Modern Digital Libraries like CiteSeerX[30] or Google Scholar[17] contain millions of documents. Typically, the crawler downloads a document, converts it to a plain text format and then extracts all necessary information. For instance automatic extraction of keyphrases is one of the challenges in information extraction task. The state-of-the-art contains complaints about absence of standard benchmarking sets for keyphrase extraction validation and methodology proof [63].

We claim that our dataset is characterized by the following features, necessary for any good *automatically crawled* dataset [31]:

- Correctness, that is the dataset should be correct and of high quality;
- Complexity or “hardness”, which addresses the fact that state-of-the-art mining systems mine differently.

Below we will argue in favor of these points regarding to our dataset.

## 3.2 Datasets used in State-of-the-art

Let us briefly mention some previous works about keyphrases extraction from the point of view of benchmarking set usage. Chronologically the pioneer in successful keyphrase extraction was Peter Tourney [89]. He proposed very detailed investigation of decision trees based algorithms and several links to freely available datasets. For instance, NEXOR[1], FIPS[2] and others [89]. However, that was more than decade ago and *all* those links are no longer available and we have failed to find any of the proposed

[1]http://www.nexor.com/public/aliweb/search/doc/form.html
[2]http://www.itl.nist.gov/div897/pubs/

datasets in internet. Later work which is one of the most valuable in the domain is KEA[3] [96]. Its algorithm is based on Naive Bayes classifier. KEA is a free software and can be downloaded through KEA website, but there are no standard datasets in the download package. KEA inventors mention that they obtained Tourney dataset directly from the author. Nguen *et al* [63] directly pointed out to the impossibility to find any proper datasets and used their own dataset constructed from **250** crawled documents.

We emphasize that most of the datasets used in state of the art belong to the area of scientific papers. For instance, Tourney [89] used **75** scientific papers from different domains: Neuro Science, Behavorial and Brain Science and Chemistry on one hand. He also used **311** email messages and up to **140** of web pages from different domains. Dataset of a similar size was mentioned in [23]. In the more recent work Tourney proposed **500** of scientific papers from Physics domain taken from arXiv.org e-Print archive[4]. Papers were taken in PostScript (PS) format and author did not mention neither how they converted them to text nor what is the conversion quality. In the [93] authors proposed a dataset consisting of **160** scientific papers without mentioning particular domains. Annette Hulth [37] took **198** pieces of short Swedish texts related to social activities. In previous work she proposed commercial dataset from Inspec[5] [38].

We have recently proposed a novel method combining state of the art Support Vector Machines learning in combination with Stanford NLP Parser [44] upon **400** of scientific papers in Computer science domain published in ACM.

[3]http://www.nzdl.org/Kea/
[4]http://arxiv.org/
[5]http://www.theiet.org/publishing/inspec/

### 3.2.1 Problems with existing datasets

Let us summarize major dataset related problems in the state-of-the-art. Apart from already mentioned complexity and correctness, we point out the following:

- **Dataset size** is one of the biggest problems of any keyphrase extraction research papers. To the best of our knowledge no one used dataset containing more than 500 scientific papers. We think this is caused by the difficulties in dataset construction and further results evaluation. Most of the trials were done *manually*, which is extremely time consuming. However, increasing the dataset size may lead to significant improvements in precision and recall of tasks based on supervised machine learning methods.

- **Availability and sustainability** are the main problems for most of the datasets considered in the state of the art. It is really hard to compare new algorithms and methodologies with previous work because the results may vary from dataset to dataset drastically. Even taking papers from the same storage and nearly same domain may change the results depending on machine learning method.

In the present chapter we address all these issues:

- we propose the largest dataset in the state-of-the-art (about 2000 documents with full texts);

- the dataset has high quality;

- the dataset is freely available through internet for further competition.

### 3.2.2 Related work

Creation of benchmarking sets is not a new field. There are some datasets well-known in Information Retrieval. For example, Reuters Dataset[6], prepared by David Lewis. This dataset carries thousands of short news texts with labels and helps to evaluate classification algorithms. Another example is a large dataset called TREC [16] [7]. TREC collection is dedicated to web mining, indexing and query answering. It fits well to semantic search community tasks and has been used in different semantics and Natural Language Processing-based evaluations. There is the dataset constructed by Giunchiglia *et. al.* [31] by crawling (as we do here) for ontology matching. And there are many more samples of datasets available in web.

## 3.3 Dataset description

The dataset we present contains papers from Computer Science domain published by ACM [25] in the period from 2003 to 2005. All these papers are written in English and stored in UTF-8 text encoding. Each text has clearly indicated:

- Title.
- Abstract.
- Body.
- References (recognized by our method [40]).
- References crawled from ACM portal.
- References to citing papers (also taken from ACM).

[6]http://www.daviddlewis.com/resources/testcollections/reuters21578/
[7]http://trec.nist.gov/

The separation of the parts enables to use them as an additional training material for training text part recognition. Moreover, they can be used to restrict search for a keyphrase to a part of the text. For example, search can be restricted to abstract and references only [44, 93]. This is convenient for computationally expensive methods like SVM [93].

Each file holds full text of a paper and has the name like "[id].txt" where "[id]" is a valid ACM[8] document id, for instance "1005858.txt" corresponds to a real paper with id "1005858". One may find this paper at http://portal.acm.org and make sure it is a paper "A framework for architecting peer-to-peer receiver-driven overlays" with attached keyphrases "congestion control, peer-to-peer streaming". Keyphrases for particular file are located in file "[id].key". This format is common for machine learning community and used in KEA [96]. Dataset contains 2304 papers freely available in internet[9]. It is not separated into a training set and a test set, so we presume applying of cross-validating procedure (see for example [65]). The papers full texts were downloaded from CiteSeerX Autonomous Digital Library.

## 3.4 Dataset preparation

We took the papers in PDF format from CiteseerX, skipping all corrupted or "unconvertible" PDFs (such as PDF stored as image). Metainformation like titles, references and abstracts was taken from ACM portal. We have mapped ACM metainformation to Citeseer texts on the bases of crawled id mappings and information kindly shared with us by professor Lee Giles, creator of Citeseer and CiteseerX digital libraries. Then we converted PDF to plain text using a commercial system, then information was processed step by step as described in [40]. While doing this we have found some

[8]http://www.acm.org/
[9]http://disi.unitn.it/ krapivin/

"garbage", or lexically meaningless pieces of information, which trapped into texts as a result of double conversion: from LaTeX to PDF and then from PDF to text. While using Natural Language Processing tools may improve keyphrase recognition rate [44, 38] this "garbage" descreases the precision of Natural Language Processing tools. We have used Maximum Entropy Model based tool to eliminate the "garbage".

### 3.4.1 Garbage cleaning

PostScript and PDF formats are current standard of presenting scientific papers. While they have many advantages of allowing rich formatting, complex formulas and figures to be used, for many tasks requiring natural language processing this presents an additional challenge of extracting plain text out of a PDF document.

Many tools address the issue of PDF to plain text conversion. However, the resulting plain text document often contains remains of LaTeX markup, various extra punctuation symbols, clusters of brackets. For example, Figure 3.1 shows the example of a "garbage" remaining in plain text after the conversion from PDF.

Figure 3.1: PDF converted to plain text.

```
a linear system Ax = b, in which
satisfy k = (M \Gamma1 N), so the iteration
```

These markup and punctuation pieces restrain modern NLP tools from achieving maximum performance and even cause failures in less robust tools. Therefore, it is desirable to clean up this "garbage" from the text.

Figure 3.2 shows how cleaned text looks like. Cleaned in this way text eliminates failures in NLP tools and allows them to achieve better results.

Figure 3.2: Cleaned plain text.

```
a linear system b, in which
satisfy, so the iteration
```

Due to the large size of the dataset, manual cleaning will take a lot of time and is unfeasible. Our approach is to use supervised machine learning. The task of identifying the garbage in a text could be seen as deciding for each token its category, which could be either “text” or “garbage”.

We annotate a small sample of the dataset, consisting of 6 documents containing together about 53,000 tokens. To each token we attach a tag identifying whether it is a “text” token or a “garbage”. The task of classifying text tokens into different categories is well-known in NLP as part-of-speech tagging.

We train and evaluate two state-of-the art part-of-speech taggers, Stanford POS tagger [87] and OpenNLP tools [62] POS tagger on our annotated dataset. Both of them are based on Maximum Entropy Models [71]. We tried several combinations of options available in taggers, however the best performance was achieved using default settings.

For tagging we use approach described in [70]. We use our own very small tag set of 2 tags, namely `T` for text and `G` for garbage. We extract and use the following features to make tagging decisions:

- up to 4 prefixes made of first 4 characters
- up to 4 suffixes made of last 4 characters
- presence of punctuation characters inside a token
- presence of initial capital letter in a token
- presence of digits inside a token

- 2 previous words and their tags
- 2 successive words and their tags

We evaluate both taggers using 10-fold cross-validation on our annotated sample. Table 3.1 summarizes taggers performance.

Table 3.1: POS taggers performance, precision per token, %.

| | Overall | Garbage |
|---|---|---|
| Stanford | 95.55 | 83.19 |
| OpenNLP | 97.04 | 87.21 |

For the better performing OpenNLP POS tagger Figure 3.3 shows precision improvement during incremental training.

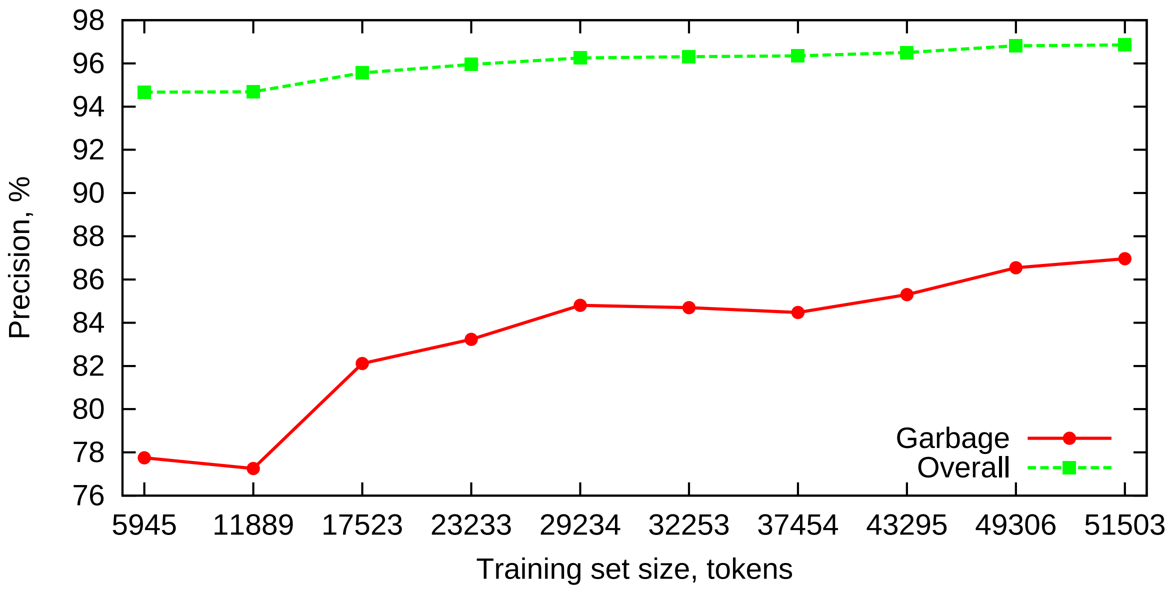


Figure 3.3: Garbage and overall detection precision for incremental training.

Note the stabilization of the overall precision curve of the POS tagger around 97%. While the overall precision stabilizes, we note that solid line showing precision per tag `G`, which indicates "garbage" to remove, is not stable yet. This precision might be improved further by increasing the size of our manually annotated training set.

### 3.4.2 Correctness and completeness, preliminary evaluation

We evaluated proposed set for KEA [96], and Machine Learning + Natural Language Processing method, recently proposed in [44, 48]. Evaluation shows that both methods, very different by their nature, have the overlap of true positive extracted keyphrases of about 55%. This indicates that different systems mine different keyphrases, so the dataset is "hard" or complete [31].

From other point of view the dataset is "naturally" correct, because it has editor assigned keyphrases of proven quality, and at least one keyphrase appears in each text at least once.

We perform preliminary evaluation with KEA, carrying out the experiments for keyphrases recognition using refined full texts and not refined ones. We see small but stable (*c.a.* 4%) improvement in keyphrases recognition using refined full texts. This is not major aim of the present work, to evaluate the effect of cleaning, moreover, since KEA does not use syntactic knowledge, it cannot improve too much with refined texts. But, since we have removed all "garbage" KEA has less chance to extract linguistically senseless information like piece of tag or formula.

## 3.5 Discussion

We have prepared and presented a large dataset for keyphrase extraction. The novelties of the dataset are:

- It is at least 10 times bigger than any of previously used datasets.

- It is a set of full texts of scientific papers, which is typical for the keyphrase extraction domain.
- It has author assigned and editor corrected keyphrases.
- It is verifiable and reproducable, because all presented information may be found through CiteseerX and ACM portal.
- It is public and available for use by researchers.
- It is refined for better NLP processing to get more syntactical and semantical knowledge.
- The dataset is hard because it presents different challenges for different state-of-the-art machine learning systems.

The proposed dataset may also be used for classification tasks, because all presented documents have classification labels which may be found on ACM portal. Another possible use of the dataset is the text parts detection.

# Chapter 4

# Automatic Keyphrases or Tags Extracting in ADL

In this chapter we study the use of Natural Language Processing (NLP) techniques to improve different machine learning approaches (Support Vector Machines (SVM), Local SVM, Random Forests), to tackle the problem of automatic keyphrases extraction from scientific papers. For the assessment we used a large high quality dataset: 2000 ACM papers from the Computer Science domain (see Chapter3). Evaluation shows promising results that outperform state-of-the-art Bayesian learning system KEA improving the average F-Measure from **22**% (KEA) to **30**% (Random Forest) on the same dataset without the use of controlled vocabularies. The assessment is performed by comparison with expert assigned keyphrases. Finally, we report a detailed analysis of the effect of the individual NLP features and data sets size on the overall quality of extracted keyphrases.

We organize the Chapter as follows: In Section 4.1 we present a brief review of the state-of-the-art in the domain and a discussion of relevant related work. Section 4.2 provides a detailed description of the dataset used in our experiments. Section 4.3 presents the details of the proposed extraction methodology (that we name hereafter ML+NLP), specific feature set, text processing tasks and result assessment methodology. In Section 4.4 we present the results of all four approaches, Bayesian Learning (KEA), Support Vector Machine, Fast Local Kernel Machine and Random Forrest training. In Section 4.5 we present a quantitative comparative analysis of the obtained quality measures as well as a qualitative analysis of the extracted keyphrases. Section 4.6 is devoted to the conclusions.

## 4.1 State-of-the-art

Exponential growth of information in web era made autonomous digital libraries very popular. Autonomy means automatic information harvesting, processing, classification and representation and it brings several challenges.

A specific challenge lies in the domain of scholarly papers [36], accumulated in autonomous digital libraries [20, 28, 53] like CiteSeerX[1][30], Google Scholar[2][17] or Rexa[3][58]. Library spider crawls the web for scientific papers. Having retrieving the paper, the crawler must convert it into a text format. Then it extracts relevant metadata (like title, authors, citations) and finally documents are properly analyzed (classified, identified, ranked, stored). Metadata helps to categorize or classify papers, simplifying and enhancing users' searches, but often metadata is not available explicitly.

Information extraction from scholarly papers contains two broad classes of tasks:

- recognition of structural information which is present inside the paper body (like authors, venues, title, abstract, text parts like sections,

[1]http://citeseer.ittc.ku.edu
[2]`http://scholar.google.com`
[3]http://rexa.info

tables figures, author assigned keyphrases (the ones that follow after word "Keywords:" from a new line in a header);

- extraction of information which is only implicitly present, such as generic keyphrases or tags, which are not explicitly assigned by the authors.

First task is well-investigated and accomplished with very high (up to 97%) precision by two groups, Giles [82] with help of Support Vector Machines and McCallum [59, 67] with help of Hidden Markov model and Conational Random Fields, on a benchmarking dataset taken from Rexa scientific crawler. We emphasize that extraction of implicit information is very different problem and most of methods applied to extraction of header part (explicit information) are not suited for it.

In this Chapter we focus on the second, more challenging task of extraction of implicit information. We analyze the effect of the use of Natural Language Processing (NLP) techniques and the use of specific NLP-based heuristics on the improvement of current Machine Learning (ML) approaches.

Machine learning methods are successfully used to support automatic information extraction tasks for short news, mails, web pages [86], and also for the problem of keyphrases extraction [93, 38, 37]. Keyphrase is a short phrase representing a concept from a document. They are useful for search, navigation and classification of text content.

Let us briefly outline text classification problem and show how keyphrases may help in it. The most common and primitive method of classification is bag-of-word plus vector space representation [1], where a document is converted into a vector of very large dimensionality (topically 5-10 thousands) where component equal to 1 if word presents in document and 0 if not. So distinct quantity of words in a set of documents we want to classify is the dimensionality of a vector space (so-called features space) we work with. Assuming that basis of the feature space is orthogonal, the scalar product of two vectors (documents) indicates how similar documents are. So categorization is simply a similarity between category vector and document vector. In this primitive approach we cannot deal with keywords or keyphrases. But if we will consider phrases as instead of simple tokens, and weight each phase more if it is a keyphrase we will get more accurate classification [74]. So as we see, keyphrases are indeed useful for text categorization task.

The most popular state-of-the-art system for keyphrases extraction is KEA [96]. It uses Naïve Bayes classificator and few heuristics. The best results reported by KEA team show about 18% of Precision [96] in the extraction of keyphrases from generic web pages. Usage of domain specific vocabularies may improve the result up to 28.3% for Recall and 26.1% for Precision [60].

Another approach is suggested by Tourney and uses GenEx algorithm [89]. GenEx is based on a combination of parameterized heuristic rules and genetic algorithms. The approach provides nearly the same precision and recall as KEA. In a more recent work [86], the author applies web-querying techniques to get additional information from the Web as background knowledge to improve the results. This method has a disadvantage: mining the Web for information and parsing the responses is a time and resource consuming operation. This is inconvenient for Digital Libraries with millions of documents. In this approach the author measures the results by the average number of correctly found phrases vs. total number of extracted phrases ratio.

Recent works by A. Hulth et al. took into account domain [38] and linguistic [37] knowledge to search relevant keyphrases. In particular, [37] used thesaurus trying to get domain knowledge. Recall reported in this work is very low, namely 4-6%. The approach proposed in [38] introduced a heuristic related to part-of-speech usage, and proposed training based on the three standard KEA features plus one linguistic feature. Authors reported relatively good results (F-Measure up to 33.9%). However, it is hard to compare their results with others due to the strong specificity of the used data set: short abstract with on average 120 tokens where around 10% of all words in the proposed set were keyphrases.

A recent interesting work with regard to the application of linguistic knowledge to the specific problem is reported in [23]. The authors used WordNet[24] and "lexical chains" structures based on synonyms and antonyms. Then they applied decision trees as a ML part on about 50 journal articles as the training set and 25 documents as the testing set. They reported high precision, up to 45%, but did not mention recall. This makes difficult any comparison with other techniques, and, as it will be investigated below, we think that such dataset is too small and biased for comparison.

Other ML technique, least square SVM [93] shows 21.0% Precision and 23.7% Recall in the analysis of web mined scientific papers. Also in this case the described experiments are limited to a very small testing dataset of 40 manually collected papers, which is again very small set.

Let us briefly consider some other methods all summarized in the recent paper [55], they are:

1. Word clustering: here the basic idea is to cluster words using unsupervised clustering method, for example [80]. Than, after the phrases are clustered, we induce an additional weight to a phrase's TF, making it proportional to summarized TF's of all phrases in a cluster.

2. Using the sentence salience score [69] as a feature. This score is a weight of sentence in which a phase is located. It based on the notion

of vector space, where we represent each sentence in a document or cluster of documents as a vector, and than compute the inner product between each pair of vectors. Having linear combination of scholar vector product, position of a sentence in a document and centroid score (the measure of centrality of a sentence in a single document or cluster of documents) we denote a notion of salience score as it defined in [69]. Salience score is also used for instance in [54].

3. Graph-based score is also may give an additional weighting, here we exploit the idea that important phrases are connected into a graph, such graphs may be constructed with help of iterative reinforcement algorithm [92]. Same statement is valid for sentences which are also bound in a graph. Reinforcement algorithm is the iterative method that recomputes the weight of a node (single word or sentence) in a graph until convergence. So it may be employed as an additional weighting.

Sticking all mentioned methods together we may assign each single phrase with a complex score, than, getting threshold we may take $n$ most "important" phrases and treat them as keyphrases. Evaluation of this method shows an improvement of F-Measure from 25% (TFxIDF baseline approach) to 29%. This improvement is quite similar (a bit less) with what we have obtained, but it is done on the smaller manually tagged by volunteers annotators set of meeting notes.

There is another side of machine learning: unsupervised learning or learning without a trainer. Such kind of problems does not require any training set (while, for sometime it may use some "seed" which are not necessary). Same as it is for supervised keyphrases extraction problem, here there is just one publicly available (just for research purposes, for commercial use there is another type of license) keywords extraction system (to best of our knowledge): Leximancer [34]. This system [78, 94] designed and developed for completely unsupervised keyword extraction. We say keyword instead keyphrases since Leximancer is able to extract merely one-token phrases. This limitation is caused by the nature of a method. For information extraction Leximancer team use so-called context based analysis, i.e. each word considered within context, the piece of text around a word. This methodology came from earlier works of David Yarowsky [99, 100] dedicated to the problem of word-sense disambiguation. We believe that context-based analysis with combination of proposed int the present Thesis methodology may improve keyphrases extraction.

This chapter extends and improves on a preliminary work describing our initial concepts and presenting initial results [44] obtained using standard SVM approaches, namely:

1. We extend the use of state-of-the-art NLP tools [62] for the extraction, definition and use of linguistic-based features such as part of speech and syntactic relations extracted by dependency parsers [64].

2. We apply the proposed NLP-based approach to a number of different ML methods namely traditional SVM, innovative Local SVM and Random Forests.

3. We define and publish a large high quality dataset of 2000 documents [47], available through internet, with experts assigned keyphrases in Computer Science field.

4. We analyze in details the effect of different NLP features and dataset size on the overall quality of extracted keyphrases.

5. We perform a comparative analysis of the computed quality measures and obtained keyphrases with the various ML techniques (enhanced with NLP) and the popular Bayesian learning system KEA.

## 4.2 Dataset Description, Characterization and Linguistic Processing

### 4.2.1 Dataset Description and Characterization

The dataset presented [47] contains a set of papers published by the ACM in the Computer Science domain in 2003–2005. The documents are included in the ACM portal[4] and their full texts were crawled by CiteSeerX digital library as PDFs, but we place them to internet as the text files. These text files names are unique ACM ids, so dataset may be easily verified online through ACM portal. In our pre-processing tasks, we separated different parts of papers, such as title and abstract, thus enabling extraction based on a part of an article text. Formulas, tables, figures and eventual LaTeX mark up were removed automatically. We share this dataset and welcome interested communities to use it as a benchmarks set for information extraction approaches.

For our investigations we separate existing keyphrases into two categories:

- author assigned: located inside each document in the header sections after the prefix "Keywords:";
- editor assigned: manually assigned by human experts in a particular domain.

Our experimental dataset consists of 2000 documents with keyphrases assigned by ACM editors. It is important to note that in our preparation of the above dataset, we have selected only papers that contain at least one expert assigned keyphrase in the full text of a document. So we are

[4]`http://portal.acm.org`; available also at `http://dit.unitn.it/~krapivin/`

not in the more challenging case of *completely* implicit extraction. In our dataset, each document has on average about 3 unique human assigned keyphrases (see Figure 4.1).

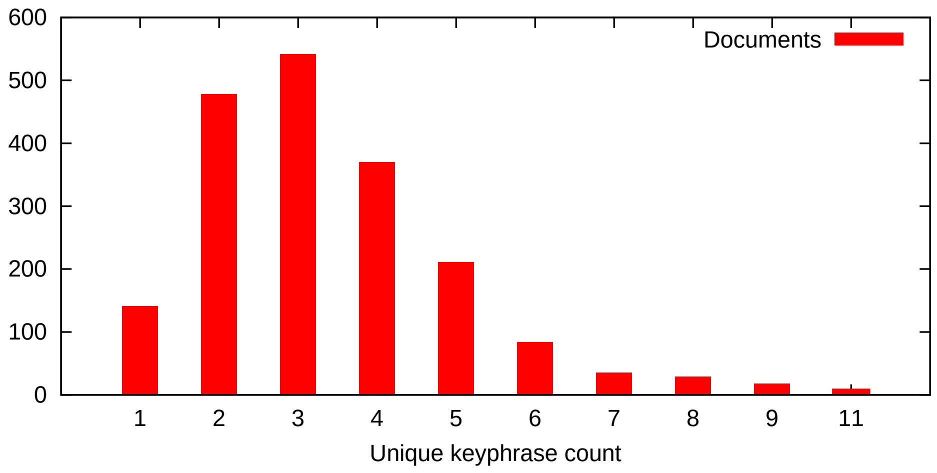


Figure 4.1: Distributions of unique assigned keyphrases per document.

## 4.2.2 Linguistic Analysis of Keyphrases

We performed a NLP analysis of the keyphrases to study their linguistic properties. This is the base for the proposal of heuristics and the definition of the features used in the machine learning step. This improves the quality of generated keyphrase candidates while simultaneously reducing their quantity.

We analyzed a sample of 100 random documents using OpenNLP tools. We applied tokenizer, Part of Speech (POS) tagger and chunker to explore differences between POS tags and chunk types for normal text documents and the corresponding keyphrases set. Fig. 4.2 shows POS tag distributions for the most common POS tags, such as nouns: NN, NNP, NNS; prepositions: IN, adjectives: JJ and verbs: VBN, VBP, VBG, VBD. Fig. 4.3 shows distributions for chunk types, such as noun phrases: B-NP, I-NP; prepositional phrases: B-PP; verbal phrases: B-VP, I-VP. To improve readability we have omitted values close to zero.

One can note from the figures that the distributions differ significantly between normal text and keyphrases sets. The major differences in POS tags distribution confirm that the majority of keyphrases consist of nouns, singular as well as plural, and adjectives. The difference in chunk types distribution also confirms and reinforces this hypothesis, adding to it that the overwhelming majority of keyphrases are noun phrases.

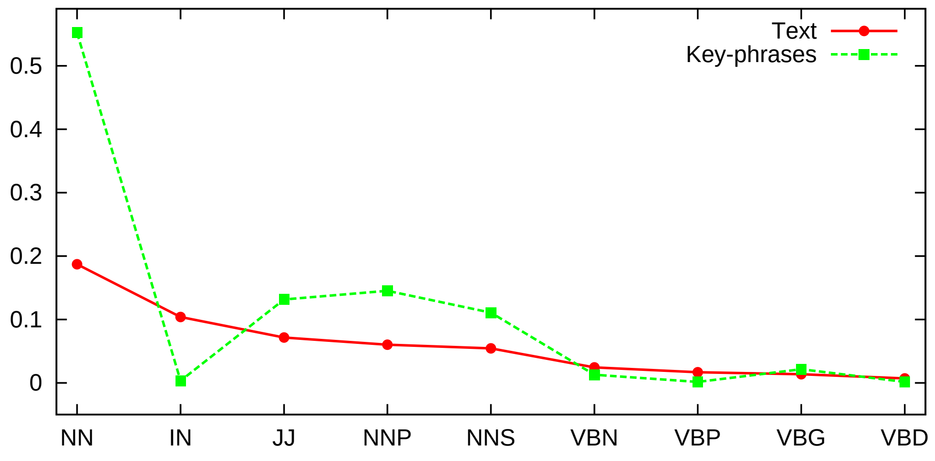


Figure 4.2: POS tags distributions for normal text and keyphrases.

We did another analysis using MaltParser[64] to explore differences between dependencies of keyphrases and the ones of normal text. Fig. 4.4 compares keyphrases and normal text dependency distributions. We use the results of this analysis to compose features set.

### 4.2.3 Text processing

Before any extraction task, text needs to be pre-processed to assure a reasonable quality of extraction [97]. Modern scientific papers are mostly

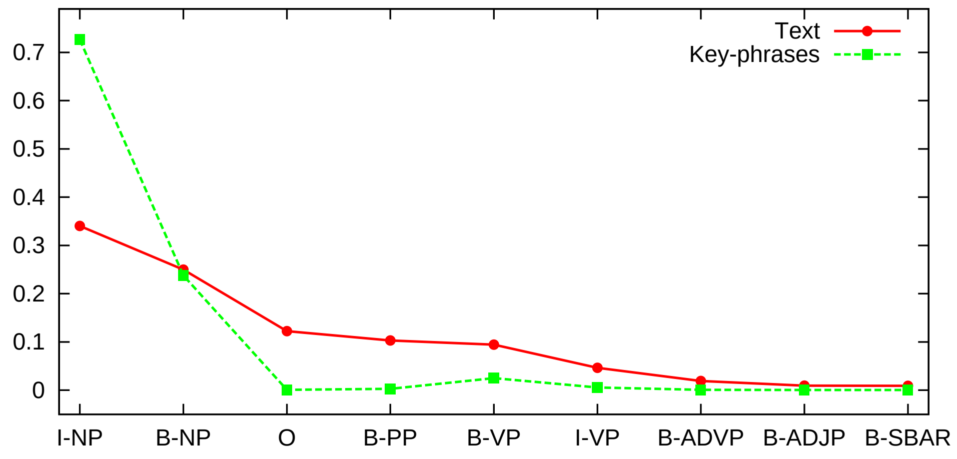


Figure 4.3: Chunk types distributions for normal text and keyphrases.

available in PDF format, thus first we need to convert them to plain text. Further preprocessing includes sentence boundary detection, tokenization, POS tagging, chunking, parsing, stemming and recognizing separate blocks inside the article, such as Title, Abstract, Section Headers, Reference Section, Body.

We used OpenNLP suite [62] to do standard steps of text processing. Namely, we apply sentence boundary detector, tokenizer, part of speech tagger and chunker consequently. Then we apply a heuristic, inspired by the previous linguistic analysis of keyphrases.

The heuristic consists of two steps. First we filter by chunk type, leaving only NP chunks for further processing. Then we filter the remaining chunks by POS. We leave only chunks with tokens belonging to the parts of speech from the top of the distribution in Fig. 4.2, such as NN, NNP, JJ, NNS, VBG and VBN. Table 4.1 shows an example sentence and extracted keyphrase candidates. This heuristic extracts for further analysis only linguistically meaningful keyphrase candidates.

We use S-removal stemmer, embedded into KEA[96], to avoid problems with the same words written in different forms. This is the stem-

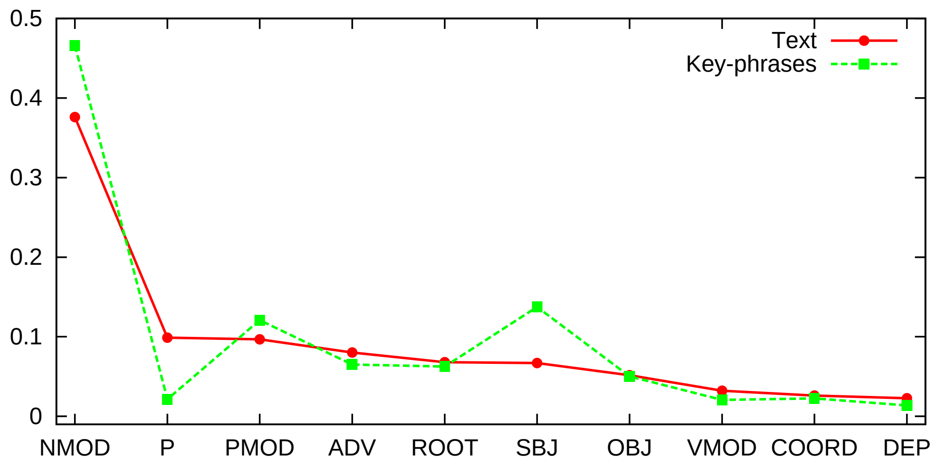


Figure 4.4: Dependencies distribution for normal text and keyphrases.

Table 4.1: Keyphrase candidates extracted by the heuristic.

| | |
|---|---|
| Sentence | Therefore, the seat reservation problem is an on-line problem, and a competitive analysis is appropriate. |
| Candidates | seat, seat reservation, seat reservation problem, reservation, reservation problem, problem, on-line problem, analysis, competitive analysis |

mer specially adopted by WEKA[97] group for the problem of keyphrases extraction, it is different from the other WEKA-embedded stemmers like snowball stemmer[85], Porter's[68] or Lovins' stemmer[56]. One of the core differences is that it does not remove "ing" form since such form changes the sense of a word significantly in rare cases. It called "s-removal" just because it distinguishes plural and single or in other words removes "s" in the end of a word. Table 4.2 shows the examples of original and stemmed forms.

In addition, we apply MaltParser to extract dependencies which we use as additional features for machine learning.

Table 4.2: Original and stemmed forms

| Original | Stemmed |
|---|---|
| Multiple selections | multipl selection |
| text editing | text editing |
| SPFD-based global rewiring | spfd base global rewiring |
| Glauber dynamics | glauber dynamic |
| networking | networking |
| network | network |

## 4.3 Enhancing Machine Learning with Natural Language Processing

### 4.3.1 Features selection

Proper feature space selection is a crucial step in information extraction. Many features may be used for accurate information extraction and their characteristics are strongly domain dependent.

The feature set we propose is detailed in Table 4.3. Features 1, 2 and 3 are common and widely used in most information extraction systems [86]. Less traditional features are: feature 4 quantity of tokens in a phrase, used in [82], feature 5 – the part of a text, successfully used in [93]. The features numbered in Table 4.3 from 6 to 20 are based on linguistic knowledge. We consider keyphrase containing a maximum of 3 tokens, with indices 1, 2 and 3 in the feature names referring to the first, second and third token of the candidate, respectively. Features 6 to 8 contain part of speech tags of the tokens of a keyphrase candidate.

The next set of features uses dependencies given by the MaltParser. Each dependency contains a head, a dependent and a labeled arc joining them. Dependencies help us to capture the relations between tokens, the position and the role of the keyphrase in the sentence. Specially, features

Table 4.3: The adopted Feature Set, $i \in [1..3]$

| # | Feature | # | Feature |
|---|---|---|---|
| 1 | term frequency | 6-8 | $i$-th token POS tag |
| 2 | inverse document frequency | 9-11 | $i$-th token head POS tag |
| 3 | position in text | 12,15,18 | $i$-th token dependency label |
| 4 | quantity of tokens | 13,16,19 | distance for $i$-th incoming arc |
| 5 | part of text | 14,17,20 | distance for $i$-th outgoing arc |

9-11 contain part of speech tag of a head of the token for each token of the candidate. Features 12-20 refer to the relations within the keyphrase and relations attaching a keyphrase to the sentence. They consist of three groups, one group for each token of the keyphrase candidate, and have similar meaning. Let us consider in detail the first group, features 12-14. Feature 12 refers to the label of the arc from the first token of the candidate to its head. It grasps the relation between the keyphrase and the sentence or between the tokens of keyphrase. Features 13 and 14 grasp the cohesion of keyphrase and its relative position in the sentence. Feature 13 refers to the distance between the first keyphrase token and its dependent if it exists. As a distance we take the difference between token indexes. Feature 14 refers to the distance between the first token and its head.

### 4.3.2 Machine Learning Methods used for comparison

**Random Forest.**

Nowadays ensemble learning is getting more popular, one may be divided into two main branches:

- bagging [8] and
- boosting [72].

The core idea of ensemble learning is in the construction of many decision trees (or other) classifiers and voting for the best result. Bagging and boosting main difference in a way of constructing decision trees. Random Forest is an extension of earlier “bagging” technique proposed by Leo Breiman in 2001 [9, 14]. The RF algorithm randomly takes piece of a training set to grow each tree, and does not need costly cross-validation procedure. Being scalable and relatively fast RF improves state-of-the-art in a different machine learning-based classification tasks.

### Support Vector Machines (SVMs)

[19] are classifiers with sound foundations in statistical learning theory [65] which are now considered the state-of-the-art classification method for a wide range of computational tasks. The reasons of their success are related to their ability to find the optimal solution, the possibility of highly non-linear mappings of the input space, the handling of noisy data with a soft-margin approach and their robustness to the curse of dimensionality. Differently from many text classification approaches based on the “bag of words” representation (where each text is encoded in a binary vector denoting which words are present) that causes a very high dimensionality of the data, we are working here with only 20 features. When the dimensionality is high a linear classifier is frequently the best choice, while with a reduced number of features a non-linear approach is needed. For this reason we adopt SVM with the Gaussian radial basis function (RBF) kernel in the form:

$$K(x, x') = \exp\left(-\frac{||x - x'||^2}{\sigma}\right) \tag{4.1}$$

where $\sigma$ is a non-negative constant, called kernel width, which regulates the level of locality of the kernel and needs to be tuned tuning model selection.

#### FaLK-SVM

[76] is a kernel method based on Local SVM [7] which is scalable for large datasets. There are theoretical and empirical arguments supporting the fact that local learning with kernel machines can be more accurate than SVM [77]. In FaLK-SVM[5] a set of local SVMs is trained on redundant neighborhoods in the training set selecting at testing time the most appropriate model for each query point. The global separation function is sub-divided in solutions of local optimization problems that can be handled very efficiently. This way, all points in the local neighborhoods can be considered without any computational limitation on the total number of SVs which is the major problem for the application of SVM on large and very large datasets.

#### KEA

[96] represents the state-of-the-art for keyphrase extraction tasks and it is based on the bag-of-words concept. The Bayes theorem is used to compute a probability of a phrase to be a keyphrase using frequencies gathered from a text. So in the end each phrase in the text has a probability to be a keyphrase. After that KEA takes top $q$ of phrases and calls them keyphrases. Naïve Bayes learning is widely used for other text-oriented tasks like spam filtering or text classification.

There are some more accurate machine learning used for classification, for instance so called "Bayesian learning", most promising techniques are RVM, or Relevance Vector Machine [84] or Gaussian Processes [95] that may potentially improve the accuracy of prediction, but they are too slow comparing even with SVM.

[5]FaLKM-lib [75] source is available at `http://disi.unitn.it/~segata/FaLKM-lib`

### 4.3.3 Training with unbalanced class cardinalities

In keywords and keyphrases extraction tasks it is natural that the number of key elements (and of candidate key elements) is dramatically lower than the number of non-key elements. In our dataset the keyphrases represent about the 1.8% of the total number of phrases taken into account, meaning that we have more than 55 times fewer positive examples than negative ones. Classification with unbalanced datasets is a challenging task which is very often handled assigning different misclassification costs to the classes.

In SVM classification this is possible associating different soft-margin regularization parameters ($C$) to the classes as discussed for the first time in [65]. Obviously the assignment of different regularization parameters to each class enlarges the set of parameters that needs to be tuned during model selection phase.

The same approach can be used for FaLK-SVM with the only difference being that the soft-margin regularization parameters ($C$) are set locally.

Random Forest also can handle unbalanced data using "weight" parameter in implementation [14].

### 4.3.4 Result assessment methodology

Usual IR performance measures are Precision, Recall and F-Measure. Defining with $A$ the set of true keyphrases that have not been recognized as keyphrases, with $B$ the correctly recognized keyphrases and with $C$ the set of phrases incorrectly recognized as keyphrases, the formal definition of precision ($P$), recall ($R$) and F-measure ($FM$) are:

$$P = 100\% \cdot \frac{B}{B+C} \tag{4.2}$$

$$R = 100\% \cdot \frac{B}{B+A} \tag{4.3}$$

$$FM = \frac{2P \cdot R}{P + R} \tag{4.4}$$

Here it is important to underline, that we consider each phrase as occurrence individually. This means that one phrase may be included into the paper text several times as a keyphrase, and several times as not keyphrase. It is difficult to judge whether a phrase is a keyphrase or not in such approach. The heuristic behind our judging about keyphrase is as follows: if the phrase has been recognized as a keyphrase at least once, we treat one as a keyphrase. We will call $P$, $R$ and $F$-measure based on it as *document-based* measures in the future which is also the methodology adopted by KEA [96]. Another approach is to take into account each phrase occurrence separately, obtaining *occurrence-based* precision, recall and F-Measure. This family of measures are not suitable for the final evaluation of a keyphrases extraction method since one phrase can be considered as a keyphrase several times. Even worse, it can be considered few times as a keyphrase, and few times as a non keyphrase. However, occurrence-based F-Measure is effective as measure to be maximized during SVM model selection because it follows the formal representation of numerical SVM data.

## 4.4 Experimental evaluation

In this section we give the details of the experiments carried out for the analysis of the discussed keyphrase approaches. To assess the results we used standard IR performance measures: Precision, Recall and F-Measure [65, 82].

### 4.4.1 Dataset splitting

We divided the whole dataset of 2000 documents into 3 sets: training set (TR), validation set (VS) and testing set (TS) respectively with 1400, 200 and 400 documents each. To investigate the optimal dataset size we further divided the training set into 7 subsets of 200 documents each. All sets are selected randomly assuring however the balancing with respect to the year of publication, namely assuring that all the sets have the proportional quantity of papers published in a given year.

### 4.4.2 Experiment 1. Comparison of ML Methods Enhanced by NLP

#### Random Forest

: four parameters to tune are the number of trees in the ensemble $I$, the splitting parameter $K$, the balancing parameter $w$ and the depth of a tree $d$. Experimentally we discovered the following tricks to reduce training tries:

- take 3 different $K$ parameters: default, half and double of default;
- stop the algorithm as soon as an increase in the number of trees does not improve significantly the solution;
- the depth of tree usually should not overcome the quantity of selected features, and should not be much smaller than them.

We found experimentally the best tuning ranges. We used the fast open source implementation[6] compatible with WEKA [97].

[6]http://code.google.com/p/fast-random-forest/

Table 4.4: The results for SVM, FaLK-SVM, RF and KEA. Best values in bold

| | Precision | Recall | F-Measure |
|---|---|---|---|
| FaLK-SVM | 24.59% | 35.88% | 29.18% |
| SVM | 22.78% | **38.28%** | 28.64% |
| Random Forest | **26.40%** | 34.15% | **29.78%** |
| KEA (best $q$) | 18.61% | 26.96% | 22.02% |

**SVM**

: the hyper-parameters we tune are the regularization parameters of the positive and negative classes ($C^+$ and $C^-$ respectively) and the width $\sigma$ of the RBF kernel. These parameters are selected using 10 fold cross-validation with a three-dimensional grid search in the space of the parameters. The model selection is performed maximizing in this parameter space the occurrence-bases F-Measure. For SVM training and prediction we use LibSVM [13].

**FaLK-SVM**

: in addition to the SVM parameters ($C^-$, $C^+$ and $\sigma$) we have to set the neighborhood size $k$ used for the local learning approach. Model selection is thus performed as described for SVM but using a four-dimensional grid-search.

**KEA**

: KEA has one tuning parameter $q$ which is threshold, experimentally we have found that $q = 5$ produces the best F-Measure (see Table 4.4.2).

Table 4.4 summarizes the results. We see that the best result in F-Measure is achieved with the Random Forest using all 20 proposed NLP features. FaLKM and SVM follow very closely while KEA is much lower. The difference between three best methods is not very big (RF outper-

Table 4.5: KEA results for different threshold $q$ values. The best precision, recall and F-Measure among all $q$ values are in bold.

| $q$ | P, % | R, % | F-Measure, % |
|---|---|---|---|
| 6 | 17.31 | **30.10** | 21.98 |
| 5 (default) | 18.61 | 26.96 | **22.02** |
| 3 | 21.47 | 18.41 | 19.98 |
| 2 | **24.87** | 14.41 | 18.25 |

forms SVM by about 4%), therefore it is important to understand what are the most important factors: particular features or peculiarities of the dataset. These has led us to investigate both dimensions in the next set of experiments.

### 4.4.3 Experiment 2. Training set size analysis

An increase in training set size may bring an improvement of prediction quality. However, training on a large amount of data is computationally expensive. Thus it is relevent to estimate which dataset size is enough to obtain the best prediction performance. To study this, we carried out experiments at increasing training set sizes as summarized in Figure 4.5. One can see that i) F-Measure improves as the training set size increases; ii) the improvement levels off after ca. 400 documents. We can conclude that for the task of keyphrases extraction it is important to have rather large training sets, but training sets with more than 400 documents are very likely to experience computational difficulties without a relevant increase in prediction ability.

### 4.4.4 Experiment 3. NLP features analysis

In this experiments we analyze the individual effect of the features on the prediction ability. We performed experiments omitting features one by

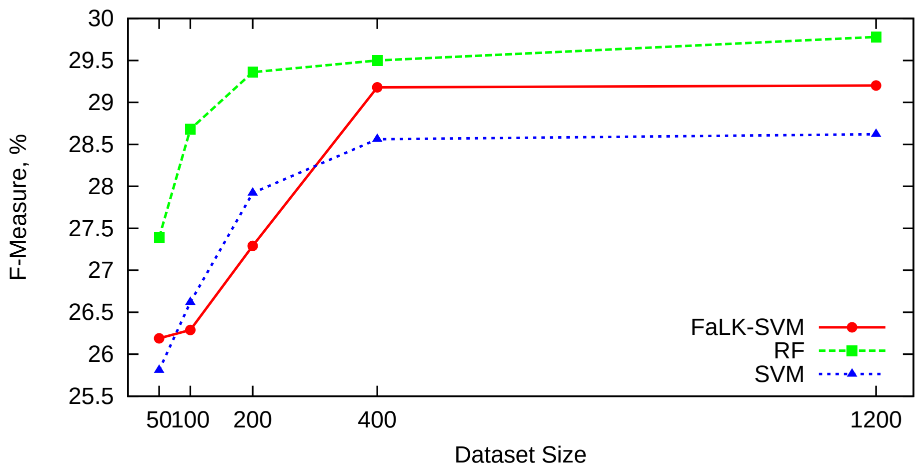


Figure 4.5: F-Measure behavior with dataset size growth.

one and monitoring the effect on the overall quality. Assuming that our features are logically grouped we decided to exclude the following groups of features sequentially:

- Arcs (11 features left)
- Head POS Tags (8 features left)
- POS tags (5 features left)
- TFxIDF and relative position (3 features left).

Figure 4.6 summarizes the results. We see that in case of Random Forest using only first three features decreases F-Measure essentially to KEA results. This is very interesting, because bayesian learning of KEA is only possible considering just 3 features (an increase of features quantity will break KEA). In our comparison of four methods we have two "statistical learning" methods (SVM and FaLK-SVM), and two "probabilistic" methods which give close results having the same three basic features that regard simple count of tokens. Figure 4.6 shows that the various methods capture different features in different ways: arcs are important for Random Forest

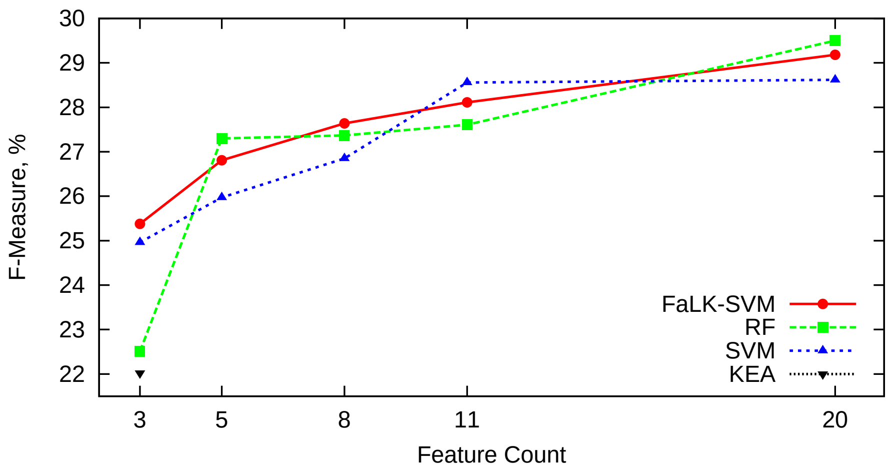


Figure 4.6: F-Measure behavior with feature count growth.

and POS tags a most important for SVM. Moreover, while there is a tendency for the overall quality to level off, the experiments do not indicate clearly to have reached a “plateau” behavior (other relevant features may be found).

## 4.5 Comparative analysis of extracted keyphrases

Aside from the detailed quantitative analysis of the standard quality measures used to compare the different approaches performance, some important insights on the specific characteristics of each approach, can be gained by a direct analysis of the extracted keyphrases. To this end we have focused our attention on the best results obtained respectively by KEA and by our proposed approach (RF+NLP).

In Table 4.6 we have collected statistics from our experiments related to correctly and incorrectly recognized keyphrases. Figure 4.7 and Figure 4.8 show the distribution of the numbers of correctly and incorrectly extracted keyphrases per document by all four approaches, respectively.

Table 4.6: Comparison between Found/Not found keyphrases counts for the best results.

| Keyphrases type | Keyphrases count |
|---|---|
| KEA Correct | 360 |
| RF Correct | 463 |
| KEA Incorrect | 1575 |
| RF Incorrect | 1280 |
| ACM Total | 1332 |
| Correct keyphrases overlapped for KEA *and* RF | 264 |
| Correct keyphrases uniquely for KEA *or* RF | 559 |

From these data, apart from the generic improvements in the recognition performance of the proposed SVM+NLP approach already presented in Table 4.4 and discussed in the previous section, we can identify some other interesting characteristics and differences in the two approaches, namely:

- There is an overlap between the extracted true keyphrases in the two approaches: approximately 57% of RF+NLP correct keyphrases are also extracted by the KEA system. However, there exists a significant number of distinct keyphrases extracted only by KEA and only by RF+NLP. This can also be seen in last 2 rows of the Table 4.6. Let us call varieties of correctly extracted by KEA keyphrases as (KEA-c) and by RF+NLP approach as (RF+NLP-c). The intersection of (KEA-c) and (RF+NLP-c) (see next-to-last row) contains 264 keyphrases, while the union of (KEA-c) and (RF+NLP-c) (the last row) is nearly two times bigger. That allows us hypothesize that a combination of the two approaches may improve keyphrases extraction performance.

- As already noticed in Table 4.4 RF+NLP recognizes less incorrect keyphrases than KEA or other methods. In addition we can notice from Figure 4.8 that ML+NLP approaches have a lot more documents with few or zero incorrectly recognized keyphrases, while KEA often makes 4 or 5 errors;

- Both approaches provide a similar "coverage" of correctly extracted keyphrases per document. Here "coverage" means the property of a given extraction approach to identify "at least" one correct keyphrase per document. In fact the total number of documents with correctly extracted keyphrases increases from 66% in KEA to 73% in RF+NLP.

From these observations we can conclude that KEA looses to ML+NLP approaches in precision and in recall both by 44% and 30% respectively. From the recall viewpoint KEA is more competitive with ML+NLP due to the higher number of extracted keyphrases it proposes, thus causing a much lower precision. Thus, even if there is a rather relevant number of correct keyphrases extracted only by KEA than can potentially enhance the ML+NLP recall, combining the two approaches might not be a good idea, because incorrectly recognized keyphrases must be merged too and thus the final precision (and the F-Measure) sensibly decreases.

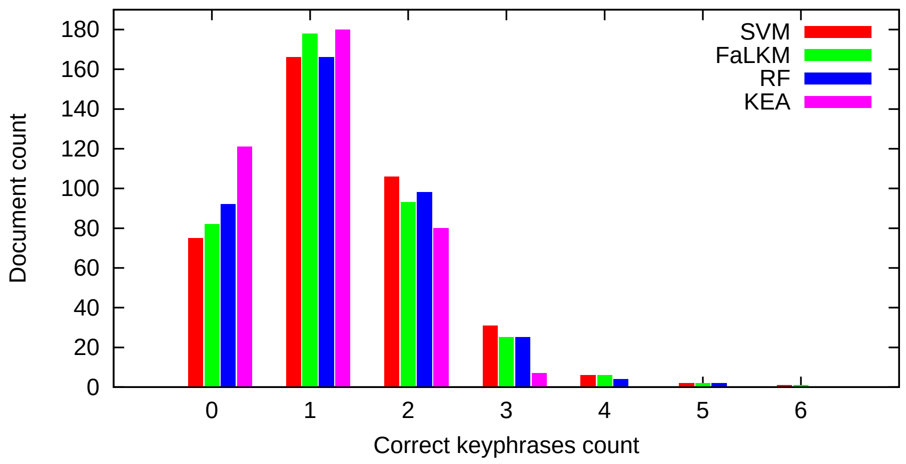


Figure 4.7: Comparison of KEA and ML+NLP distributions of correctly extracted keyphrases per document.

Table 4.7: Examples of *top* results for RF+NLP approach

| # | **ACM Keyphrases stems** | **RF+NLP Keyphrases stems** |
|---|---|---|
| 1 | data clustering, constructive induction, bayesian network, em algorithm | *data clustering, constructive induction, bayesian network, em algorithm*, bayesian multinet |
| 2 | creg, register allocation, graph coloring, register | *creg, register allocation, graph coloring* |
| 3 | software prefetching, software pipelining, vliw machine, locality analysi | *software prefetching, software pipelining, vliw machine,* modulo scheduling |
| 4 | two-variable fragment, controlled language, natural language, logic | *two-variable fragment, controlled language, natural language* |
| 5 | order-sorted logic, knowledge representation, terminological knowledge, resolution system | *order-sorted logic, knowledge representation, terminological knowledge*, label-based formula, hierarchical representation |

Table 4.8: Examples of *bad* results for RF+NLP approach

| # | **ACM Keyphrases stems** | **RF+NLP Keyphrases stems** |
|---|---|---|
| 1 | distributed environment, persistent object, distributed system, modularity, object-oriented approach, distributed programming system, replication, migration | *distributed environment, persistent object, distributed system*, database system |
| 2 | flexible transaction, concurrency control, transaction management, serializability | *flexible transaction, concurrency control*, heterogeneou distributed database, distributed database, distributed database environment, database environment, database system, multidatabase system, multidatabase |
| 3 | knowledge-based query processing | knowledge-based approach |
| 4 | security, partial equivalence relation, semantic, powerdomain, noninterference | secure information, secure information flow, information flow, sequential program, security properti |
| 5 | reflection, program transformation | generic reification technique, reflective language |

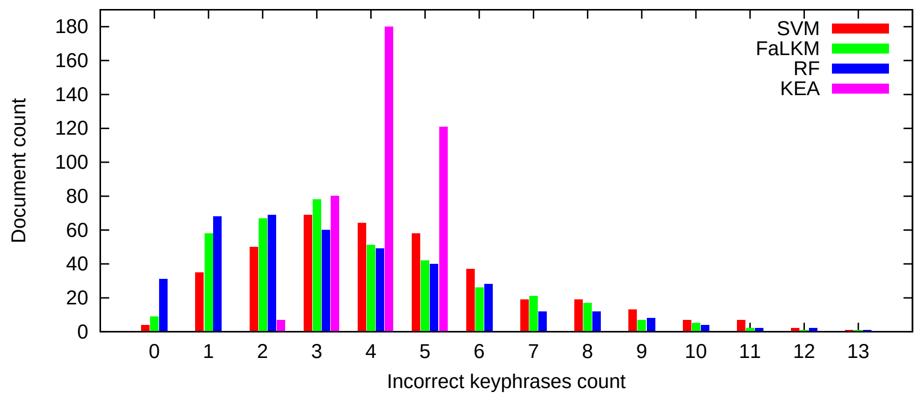


Figure 4.8: Comparison of KEA and ML+NLP distributions of incorrectly extracted keyphrases per document.

As a last analysis, we have explored qualitatively the keyphrases extracted by RF+NLP method. First, we have looked at the best results: as examples of the type of correct matches between original ACM keyphrases stems and correctly extracted keyphrases stems, we collected in Table 4.7 the top 5 results[7]: 2 documents have 100% match, and 3 documents with almost all keyphrases recognized. We see that unrecognized keyphrases are synonyms or related to the recognized ones, for instance instead of "bayesian network" we have "bayesian multinet", or "hierarchical representation" instead of "knowledge representation".

More interesting is to look at the *bad* results of our approach, that is at the results at the bottom of the distribution where no correct keyphrases (the ones included in the ACM list) has been extracted. Table 4.8 provides 5 examples of such results. In this case we cannot say "the bottom 5" because we have a tail of equally *bad* results, so we have randomly selected 5 examples among all documents with no correctly extracted keyphrases. For completeness we report that we had 92 such documents out of 400 total. A preliminary qualitative analysis of the data in the table confirms the lack

[7]Keyphrases in the table are stemmed with S-Removal stemmer

of correct keyphrases. However, the extracted keyphrases seem related to the human assigned keyphrases that we use for assessing the quality of our approach. Specifically, it seems that the human assigned keyphrases may be more general and tend to have a higher level of abstraction while the keyphrases extracted from the actual text via RF+NLP tend to be more specific. For instance: "security $\rightarrow$ secure information", "reflection $\rightarrow$ reflective language".

This behavior, if confirmed by further and more comprehensive data, would enhance the trust in the use of automatic keyphrases extraction. Automatically extracted keyphrases cannot really be used *instead* of assigned ones, but they potentially could enhance them. We know that it is not possible to draw out general trends from such a limited number of instances. However, exploration and analysis of the intrinsic qualities of automatically extracted keyphrases seems an interesting direction for future work.

## 4.6 Discussion

In this Chapter we present the application of NLP-based knowledge to a number of different ML methods: traditional SVM, a local variant of SVM and Random Forests for automatic keyphrases extraction from scientific documents. The proposed NLP-based approach shows promising results. We have performed a detailed evaluation of the performance of all ML methods by comparing the extracted keyphrases with human assigned keyphrases on a subset of 2000 ACM papers in the Computer Science domain.

Evaluation shows that adding the syntactic knowledge to the problem of keyphrases extraction improves the quality of extraction. Talking about machine learning part we can conclude that the best tradeoff between extraction quality and computation speed is Random Forrest. Baseline method KEA is extremely fast and simple and despite it uses just two features i) TFxIDF and ii) the relational position of the first occurrence of a phrase in a document, KEA shows relatively good performance, namely 22% of F-Measure. For comparison, the best result of Random Forrest is F-Measure 30%. We see 36% improvement comparing with KEA, which seems to be interesting impact without a usage of controlled vocabularies.

The proposed hybrid ML+NLP approach may also be valid with different data like news, emails, abstracts and web pages. One limitation of the present work (and of all the works based on the instance learning) is in the assumption of the presence of the searched keyphrases inside the documents (assumption that has been used in the construction of our dataset). Indeed, our learning method cannot find (without additional supporting knowledge) a specific keyphrase in a document when the document does not contain at least one instance of the keyphrase. To tackle such a challenging keyphrase assignment task one needs to take into account documents or keyphrases similarities. For example, one may forecast that documents with similar topics may have similar keyphrases. Alternatively, we have to move from syntactic to semantic relations between words in order to access (implicitly) related keyphrases. Possible future work may also focus on deeper analysis of a quality of extracted keyphrases, because “incorrect” ones are not necessarily the “bad” ones. There is a chance they may be better than author or editor assigned keyphrases.

# Chapter 5

# Ranking of Items in ADL

In this Chapter we exploit the idea of ranking items in the graph-based structures. To perform ranking we propose few modifications of Google PageRank. If Autonomous Digital Library documents refer to each other, like web pages do, they are connected into a graph. This is true for scientific publications, where a paper usually has references. We apply PageRank and PageRank-based ranking schemas to the set of scientific papers which conform the graph of 266,000 nodes. We describe here also innovative visualization techniques to plot the difference between various ranks, and, in the end, we adopt graph-based ranking techniques to evaluate an impact of an individual researcher.

## 5.1 State-of-the-art

### 5.1.1 Scientometrics: different ranks

After the Second World War, with the increase in funding of Science and Technology (S&T) initiatives (especially by public institutions), the need for supervising and measuring the productivity of research projects, institutions, and researcher themselves became apparent [26, 27]. Scientometrics

[90] was then born as a science for measuring and analysing quantitatively science itself [21]. Nowadays, the quantitative study of S&T is a rapidly developing field, also thanks to a greater availability of information about publications in a manner that is easy to process (query, analyze). The easiest measure to show any individual scientist's output is the total number of publications. However, this index does not express the quality or impact of the work, as the high number of conferences and journals make it easy to publish even low quality papers. To take quality and impact into account, the citations that a paper receives emerged, in various forms, as a leading indicator. The citation concept for academic journals was proposed in the fifties by Eugene Garfield, but received the deserved attention in 1963 with the birth of the Science Citation Index (SCI) [26]. SCI was published by the Institute for Scientific Information (ISI) founded by Garfield himself in 1960 and currently known as Thomson Scientific that provides the Web of Science on-line commercial database. The most studied and commonly used indexes (related to SCI) are, among others [61]:

1. P-index: or just number of articles of author.
2. CC-index: number of citations excluding self-citations.
3. CPP: or average number of citations per article.
4. Top 10% index: the number of papers of a person that are in the top 10% most frequently cited papers in the domain during the past 4 years.
5. Self-citation percentage.
6. Career length in years.
7. Productivity: quantity of papers per time-unit.

Although most of the indexes are related mainly to authors, they can also be applied to measuring communities, institutions or journal, using various forms of aggregation. In the last decade new indexes have been proposed. These indexes are rapidly gaining popularity over the more traditional citation metrics described above:

1. H-index, proposed by Hirsch in [35]. The H-index for an author is the maximum number h such that the author has at least h articles with h citations each. This index is widely used (including in our University), and comes in different flavors (e.g., normalized based on average number of authors of papers, on the average citations in a community, etc).

2. The G-index for an author is the maximum number g such that the most cited g papers of an author collectively received g2 citations. The g index takes into account papers with very high citations, which is something that is smoothed out by the h-index.

In addition, we mention below some algorithm for ranking Web pages. They are relevant as many of them have been very successful for ranking web content, and papers share some similarities with Web sites, as they can be seen as a sort of hypertext structure is papers are seen as web pages and citations are seen as links.

1. Hypertext-Induced Topic Selection (HITS) [43]: based on graph linkage investigation, it operates with two notions: “authority” and “hub”, where authority represents relevance of the page (graph node) to query and hub estimates the value of the node’s links to other pages.

2. PageRank (described in more detailed in the following): a well-known and successful ranking algorithm for Web pages [10], based on net random walking probabilistic model. When modified for ranking scientific papers, it has been shown to give interesting results [15].

3. Hilltop [4]. This algorithm is based on the detection of “expert pages”, i.e., pages that have many outgoing links (citations) and are relevant to a topic. Pages that are linked by expert ones have better rank.

In our work we adopt a variation of PageRank as one of the main indexes used for the analysis of differences among indexes. The intuition behind PageRank is that a web page is important if several other important web pages point to it. Correspondingly, PageRank is based on a mutual reinforcement between pages: the importance of a certain page influences and is being influenced by the importance of some other pages. From a computational point of view, PageRank is a statistical algorithm: it uses a relatively simple model of “Random Surfer” [10] to determine the probability to visit a particular web page. Since random browsing through a graph is a stochastic Markov process, the model is fully described by Markov chain stochastic matrix. The most intriguing question about PageRank is how to compute one for a dataset as huge as the web. The inventors of PageRank, Brin and Page, proposed a quite effective polynomial convergence method [10] (see please more in Sec. 5.2.3), similar to the Jacobi methods. Since then, a significant amount of research has been done in the exploration of the meaning of PageRank and proposals for different computation procedures [6, 57, 15]. When the attention is shifted from web pages to scientific citations, the properties of the citation graph - mainly its sparseness - has been used to simplify the computational problem [81]. In our work, we have based our computations on a variation of Page Rank (called Paper Rank) for ranking scholarly documents explained in detail in Section 4. From a computational perspective, the difference is that the algorithm we propose exploits the fact that in citations, unlike in web links, “cycles” or cross-citations (when paper $A$ cites paper $B$ and visa versa) are very rare, and should be considered as “unfair” citing [21]. Paper must be published with all it’s references *before* being cited. In terms of comparison among scientific metrics for determining the difference in the ranking results they generate (and methods for evaluating such differences), there is no prior art to the best of our knowledge.

### 5.1.2 PageRank evolution and computation

Ten years ago Google[18] corporation with great success applied PageRank algorithm[10] to the problem of web-pages ranking. PR algorithm is purely statistical, and there is no need to analyze the content of each page lexically. It uses "Random Surfer" model, in which the process of browsing through the web pages links is modeled by the stochastic Markov process, fully described by Markov chain matrix. Recently Page Rank has been studied from several points of view including computational feasibility, modifications and adaptations to the different types of graphs and network models, probabilistic model, mathematical background[22]. Its popularity for ranking web-pages makes it popular in other domains, like ranking of scholarly publications. The most intriguing question about PR is how to compute it for the whole web? Whole internet contains terabytes of information, and being represented as a graph it exceeds modern computers memory. It is a creative engineering task to design fast access storage to compute PR. Let us briefly outline major methods for PR computation.

- The simplest one is the cyclic PR computation for all nodes in the graph one by one, using recursive formula (1) until convergence [9]. This method takes unit vector as initial rank approximation.
- PR authors, Brin and Page proposed polynomial convergence method[10], similar to Jacobi methods.
- This method was improved by Kamvar *et al.*, 1999 [41] using "block-based strategy", similar to implementations in relational database products.

- In 2004 Langville [52] invented the procedure with reduction of the iterations number with lucky initial approximation.
- In 2005 Haveliwala *et al.* [57], proposed quadratic extrapolation method to accelerate PR convergence and evaluated their methodology under roughly 81 millions of pages.

Most of mentioned above works are related to the WEB links ranking problem which usually deals with much larger graphs than scientific citing problem. So, the computation problem has been studied well enough and looks feasible.

## 5.2 Proposed Approach

In this section we consider dataset we used, different ranks applicable for scientific citing as a measure of research impact of a single paper or author.

### 5.2.1 Data set description and data preprocessing

The starting point for our analysis is a dataset of 266788 papers published in ACM conferences or journals, and authored by 244782 different authors. The dataset was available as XML documents that for each paper describes information such as authors, title, year of publication, journal, classification and keywords (for some of the papers), journal volume and pages, and citations. A sample of the dataset format is available at the companion web page mentioned earlier. The set is biased in terms of citation information. For any given paper in the set, we have all its references (outgoing citations), but we only have citations to it (incoming citations) from other papers in the dataset, and hence from ACM papers. To remove the bias (to the possible extent), we disregard references to non-ACM papers. In other words, we assume that the world, for our citation analysis, only consists of ACM papers. Although we have no measurable evidence, given that we are comparing citation-based metrics we believe that the restriction to an "ACM world" does not change the qualitative results of the analysis. Including references to non-ACM papers would instead unfairly lower the measure for Paper Rank since, as we will show, Paper Rank is based on both incoming and outgoing citations. This being said, we also observe that the quality of the chosen dataset is very high. The majority of papers have been processed manually during the publishing process and all author's names have been disambiguated by humans. This is crucial since systems like Google Scholar or Citeseer contain errors in the disambiguation of authors names and citations. In fact, both Goodle Scholar or other autonomous digital libraries like Citeseer or Rexa use machine learning-based unsupervised techniques to disambiguate the information and are prone to introduce mistakes. A preliminary study of these errors in Google Scholar is presented in [73]. Besides disambiguation errors, crawled information may include spurious types of documents like deliverables, reports, white papers, etc. Indeed, Scholar includes in its statistics the citations coming from project deliverables or even curricula vitae, which are not commonly considered to be academically meaningful citations. Thus, although incomplete, the ACM dataset has a high level of quality in particular in respect to authors and citations. The full citation graph of the ACM dataset has 951961 citations, with an average of 3.6 outgoing citations per paper (references to other ACM papers). Figure 5.1 shows instead how many papers have a given (incoming) citation count (hereafter called CC). As expected, there is a very large number of papers with low, near-zero citations and a few papers with a high number of citations.

The years of publication of the papers in the dataset vary from 1950 to 2005 with most emphasis on the recent two decades due to the increase in

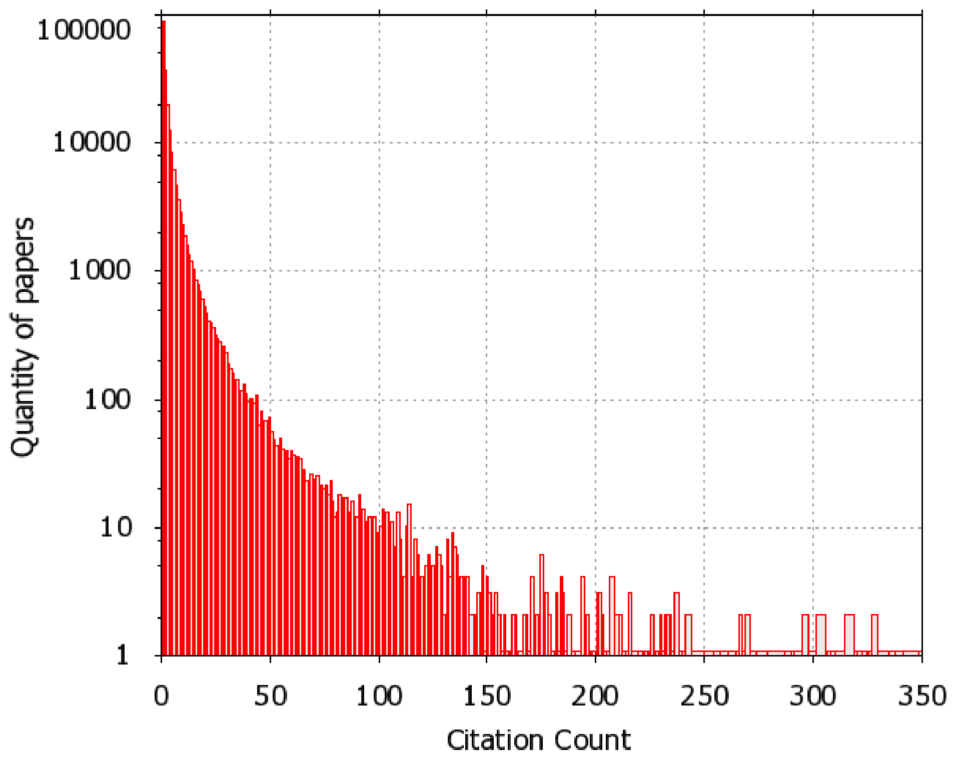


Figure 5.1: Distribution of papers by Citation Count.

the number of publications.

### 5.2.2 Page Rank outline

The original Page Rank algorithm [10] ranks the nodes of a directed graph with $N$ vertices. Considering edges between nodes we may come to the following formalization: let us numerate all nodes in graph from 1 to $N$, so that each node has it's own unique sequence index. Each node has to be ranked by PageRank algorithm so that node number $i$ has PageRank value $P_i$. Mapping this consideration to the problem of scientific citing we may conclude that a paper is a node, and citation is an edge of a graph. Rank of the node represents it's weight assuming that the more node is "referred" (cited in case of papers), the better should be the rank. The trick of a PageRank is that it considers not only the *quantity* of citations, but also the PageRank (*quality*) of all citing papers. The rank of a node is determined by the following recursive formula, where $S(j)$ is the quantity of outgoing links from a node number $j$. are just sequence numbers and $D$ is the set of nodes such that there is a path in the graph from them to node $i$.

$$P_i = \sum_{i \neq j}^{j \in D} \frac{P_j}{S(j)} \tag{5.1}$$

The formula can be seen in matrix form and the computation can be rewritten as an eigenvector problem:

$$\vec{r} = A\vec{r} \tag{5.2}$$

where $A$ is the transition matrix, or stochastic Markov matrix.

This consideration exposes several potential problems in rank computation as discussed in [6, 51]. One of them is the presence of the nodes which link to other nodes but are not linked by other nodes, called dangling nodes. In this case, equation 5.2 may have no unique solution, or it may have no solution at all (it will lead to zero-rows occurrence in the transition matrix and uncertainty of the rank of the dangling nodes). Such problem may be resolved with the introduction of a damp-factor $d$. The dump (or decay) factor is a positive double number $0 < d < 1$:

$$P_i = (1-d) \sum_{i \neq j}^{j \in D} \frac{P_j}{S(j)} + \frac{d}{N} \tag{5.3}$$

The damp factor was proposed by the PageRank inventors, Page and Brin. In their publication [10], Page and Brin give a very simple intuitive justification for the PageRank algorithm: they introduce the notion of 'random surfer'. Since in the specific case of web pages graph, the equivalent stochastic Markov matrix can be described as browsing through the links, we may imagine a 'surfer' who makes random paths through the links.

When the surfer has a choice of where to go, it chooses randomly the next page to visit among the possible linked pages The damp factor models the fact that surfers at some point get bored of following links and stop (or begin another surf session). The damp factor therefore also reduces the probability of surfers ending up in dangling nodes, especially if the graph is densely connected and dangling nodes are few. The damp factor helps to achieve two goals at once: 1) faster convergence using iterative computational methods, 2) ability to solve the equation, since all the nodes must have al least $d/N$ Page Rank even if they are not cited at all.

### 5.2.3 PageRank computation, simple cases

The most intriguing question about PR is how to compute it for the whole web? Whole internet contains terabytes of information, and being represented as a graph it exceeds modern computers memory. It is a creative engineering task to design fast access datastructures and algorithm to compute PR. But since Google works it is obvious that such problem is solved. Let us briefly outline major methods for PR computation

- The simplest one is the cyclic PR computation for all nodes in the graph one by one, using recursive formula 5.1 until convergence [15] (the simple implementation is available through [83] for free). This method takes unit vector (or vector with just one not zero component equal to constant) as initial rank approximation.

- PR authors, Brin and Page proposed polynomial convergence method [10], similar to Jacobi methods (will be further examined in details in this section).

- Method 2 above was improved by Kumvar *et al.*, 2003 [41] using quadratic extrapolations and was applied to 83 millions of web pages.

- In 2004 Langille [52] invented produced a specialized iterative aggregation algorithm for updating any Markov chain with any type of update, link or state. They reached speed up 5-10 times comparing with the previous results.

So the problem of computation has been found and it is easily scalable to the WWW dimension. Since we have much smaller dataset than [41] had, we can apply the simplest methods to get more or less good result.

Let us consider one of the simplest methods of computation of PageRank, or “power convergence method” in more details. In particular we may set initial approximation as a vector with all components equal to 1. Than to compute PageRank we should follow the recursive formula:

$$I_{k+1} = A \cdot I_k \tag{5.4}$$

, where $A$ is a Markov matrix and $I$ is an eigenvector. Lawrence and Page [10] reported that $k = 50 \div 100$ iterations are enough to converge to eigenvector with appropriate accuracy and it took up to 6 days for all available WEB pages graph those time (more than 10 years ago). Formula 5.4 is applicable to millions of nodes in a graph since it’s simplicity and matrix sparsity. Reader may find concrete samples in [11] or in the web [3].

### 5.2.4 Paper Rank

PageRank has been very successful in ranking web pages, essentially considering the reputation of the web page referring to a given page, and the outgoing link density (pages P linked by pages L where L has few outgoing links are considered more important than pages P cited by pages L where L has many outgoing links). Paper Rank (PR) applies page rank to papers by considering papers as web pages and citations as links, and hence trying to consider not only citations when ranking papers, but also taking into account the rank of the citing paper and the density of outgoing citations from the citing paper. From a computation perspective, PR is different from Page Rank in that loops are very rare, almost inexistent. Situations with loop where a paper A cites a paper B and B cites A are possible when authors exchange their working versions and cite papers not yet published but accepted for publication. In our dataset, we have removed these few loops (around 200 loops in our set). This means that the damp factor is no longer needed to calculate PR. Because of the above analysis, we can compute PR directly according to the formula 5.1. Furthermore, considering that a citation graph has $N >> 1$ nodes (papers), each paper may potentially have from 1 to $N - 1$ inbound links and the same quantity of outgoing ones. However, in practice citation graphs are extremely sparse, (articles in our ACM dataset normally have from 5 to 20 references[1]) and this impact the speed of the computation of PR. However, also in this case the matrix form of the problem (i.e. formula 5.2 may have no solution, now because of initial nodes (nodes who are cited but do not cite). To avoid this problem we slightly transform initial problem assigning a rank value equal to 1 to all initial nodes, and resetting it to zero at the end of the computation (as we want to emphasize that papers who are never cited have a null paper rank). Now the problem became solvable and the Markov matrix may be easily brought to the diagonal form. We used fast and scalable recursive algorithm for calculating Paper Rank, which corresponds to the slightly different equation:

$$\vec{r} = A\vec{r} + \vec{r_0} \tag{5.5}$$

, where $r_0$ is just a constant vector. The algorithm takes all initial nodes and propagates their PaperRank according to the formula 5.5. The power

[1]for computer science domain in average

of PageRank or PaperRank is in the fact that *every* paper may change the value of all nodes in a graph. PageRank is a unique and stable solution for a graph (www pages set, or scientific papers set) as *whole*. In other words every paper counts. It is very hard to judge which index is "better", but world-acknowledged company Google [18] may be treated as a live proof that the PageRank is a "good" measure.

### 5.2.5 PR-Hirsch

One of the most widely used indexes related to author is the H-index proposed by Jorge Hirsch in 2004 [35] and presented earlier. The H-index tries to value consistency in reputation: it is not important to have many papers, or many citations, but many papers with many citations. We propose to apply a similar concept to measure authors based on PR. However, we cannot just say that PRH is the maximum number q such that an author has q papers with rank q or greater. This is because while for H-index it may be reasonable to compare number of papers with number of citations the papers have, for PRH this may not make sense as PR is for ranking, not to assign a meaningful absolute number to a paper. The fact that a paper has a CC of 45 is telling us something we can easily understand (and correspondingly we can understand the H-index), while the fact that a paper has a PR of 6.34 or 0.55 has little "physical meaning". In order to define a PR-based Hirsch index, we therefore rescale PR so that it gets to a value that can be meaningfully compared with the number of papers. Let's consider in some detail our set: we have a graph with $N$ nodes (vertices) and $n$ citations (edges). Each $i$-th node has PR equal to $P_i$, that expresses the probability for a random surfer to visit a node, as in the Page Rank algorithm. So let's assume that we run exactly n surfers (equal to quantity of citations), and calculate the most probable quantity of surfers who visited node $i$. If the probability to visit the node $i$ for one surfer is $p_i$, expectation value $Q_i$ for $n$ surfers to visit the node $i$ will be $p_i \cdot n$, which is most probable quantity of surfers, who visited node $i$. We multiply probabilities since all surfers are independent. To be precise we should first normalize PR for each node according to full probability condition: $\sum_i p_i = 1$. If the total sum of all PRs equals to $M$, the expected value for $n$ surfers is as follows:

$$Q_i = P_i \frac{n}{M} \tag{5.6}$$

Where $P_i$ is a Paper Rank of the paper $i$, $n/M$ is the constant $\approx 5.9169$ for our citation graph. So in other words we rescale PR to make it comparable with the quantity of citations. Indeed, $Q_i$ is the most probable quantity of surfers who visited a specific paper $i$, whereas to compute Hirsch index we use quantity of citations for the paper $i$. It is interesting to compare the ranges of $Q$ and citation count (see 5.1). Following the definition of $H$-index and the previous discussion, we define PR-Hirsch as the maximum integer number $h$ such that an author has at least $h$ papers with $Q$ value (i.e. rescaled PR following equation 5.6) equal or greater than $h$.

| Average Q | Maximum Q | Average CC | Maximum CC |
|---|---|---|---|
| 3.57 | 1326.77 | 3.57 | 1736 |

Table 5.1: Comparison of citation count and random surfers count mathematical expectation values for all papers in graph.

## 5.3 Exploring Paper Metrics

This section explores the extent of the differences between paper metrics PR and CC when ranking papers, and their causes. As part of the analysis we introduce concepts and indexes that go beyond the PR vs CC analysis, and that are generally applicable to understanding the effects and implications of using a certain index rather than another for assessing papers' value.

### 5.3.1 Plotting the difference

The obvious approach to exploring the effect of using PR vs CC in evaluating papers would consist in plotting these values for the different papers. Then, the density of points that have a high CC and low PR (or vice versa) would provide an indication of how often these measures can give different quality indication for a paper. This leads however to charts difficult to read in many ways: first, points overlap (many papers have the same CC, or the same PR, or both). Second, it is hard to get a qualitative indication of what is "high" and "low" CC or PR. Hence, we took the approach of dividing the CC and PR axis in bands. Banding is also non-trivial. Ideally we would have split the axes into 10 (or 100) bands, e.g., putting in the first band the top 10% (top 1%) of the papers based on the metric, to give qualitative indications so that the presence of many papers in the corners of the chart would denote a high divergence. However the overlap problem would remain, and it would distort the charts in a significant way since the measures are discrete. For example the number of papers with 0 citations is well above 10%. If we neglect this issue and still divide in bands of equal size (number of papers), papers with the same measure would end up in different bands. This gives a very strong biasing in the chart (examples are provided in the companion page). Finally, the approach we took (Figure 5.2) is to divide the X-axis in bands where each band corresponds to a different citation count measure. With this separation we built 290 different bands, since there are 290 different values for CC (even if there are papers with much higher CC, there are only 290 different CC values in the set). For the Y-axis we leverage mirrored banding, i.e., the Y-axis is divided into as many bands as the X-axis, also in growing values of PR. Each Y band contains the same number of papers as X (in other words, the vertical rectangle corresponding to band i in the X axis contains the same number of papers qi as the horizontal rectangle corresponding to band i of the Y-axis). We call a point in this chart as a square, and each square can contain zero, one, or many papers. The reasoning behind the use of mirrored banding is that this chart emphasizes divergence as distance from the diagonal (at an extreme, plotting a metric against itself with mirrored banding would only put papers in the diagonal). Since the overlap in PR values is minimal (there are thousands of different values of PR and very few papers with the same PR values, most of which having very low CC and very low PR, and hence uninteresting), it does not affect in any qualitatively meaningful way the banding of the Y-axis.

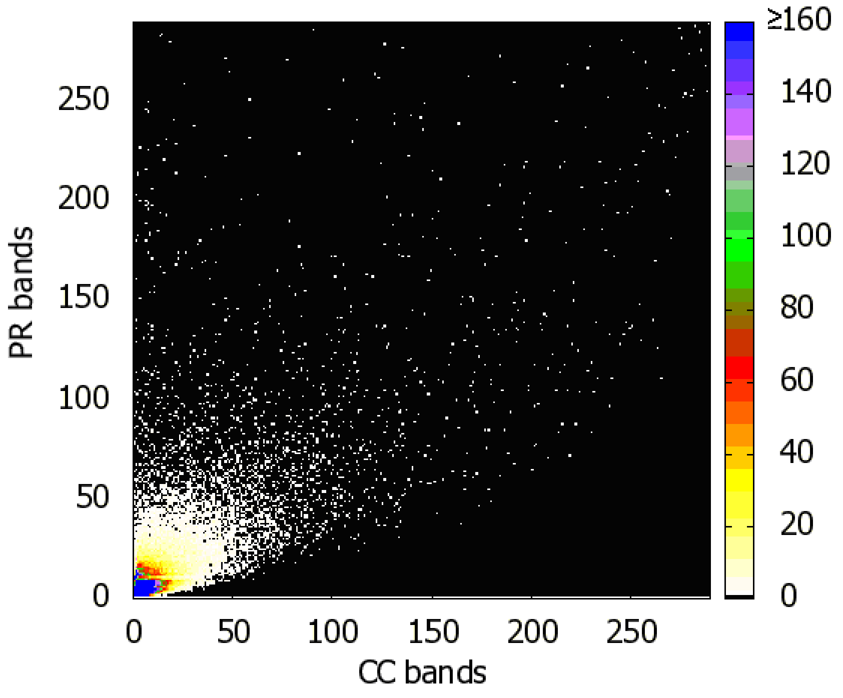


Figure 5.2: CC vs PR. X axis plots CC bands, Y axis plots PR mirror-banded by CC. The color corresponds to the number of papers within a band. (For actual values of PR and CC for each band see Table 5.2).

Table 5.2 gives an indication of the actual citation and PR values for the different bands.

| Number of band both for CC and PR | CC | PR |
|---|---|---|
| 50 | 50 | 6.23 |
| 100 | 100 | 14.74 |
| 150 | 151 | 26.57 |
| 200 | 213 | 38.82 |
| 250 | 326 | 58.86 |
| 280 | 632 | 113.09 |
| 290 | 1736 | 224.12 |

Table 5.2: Mapping of band number to the actual value of CC or average actual value for PR.

The chart in Figure 5.2 shows a very significant number of papers with a low CC but a very high PR. These are the white dots (a white color corresponds to one paper). Notice that while for some papers the divergence is extreme (top left) and immediately noticeable, there is a broad range of papers for which the difference is still very significant from a practical perspective. Indeed, the very dense area (bands 1-50) includes many excellent papers (CC numbers of around 40 are high, and even more considering that we only have citations from ACM papers). Even in that area, there are many papers for which the band numbers differ significantly if they are ranked by CC or PR.

To give a quantitative indication of the difference, Table 5.3 below shows how far apart are the papers from the diagonal. The farther away the papers, the more the impact of choosing an index over another for the evaluation of that paper.

The mean value for the distance from the main diagonal is 3.0 bands, while the standard deviation is 3.4. This deviation from the average is rather significant, i.e. in average the papers are dispersed through 3 bands around main diagonal. In the subsequent discussion, we will qualitatively

| Distance in bands from the diagonal | % of papers with this distance |
|---|---|
| 0 | 36.83 |
| 1 | 24.30 |
| 2 | 13.02 |
| 3 | 5.76 |
| 4 | 5.43 |
| 5 | 2.50 |
| 6 | 1.70 |
| 7 | 1.34 |
| 8 | 1.86 |
| 9 | 1.57 |
| 10 | 0.79 |
| ≥11 | 4.89 |

Table 5.3: Deviation of papers around main diagonal.

refer to papers with high PR and high CC as popular gems, to paper with high PR and low CC as hidden gems, to papers with low PR and high CC as popular papers, and to papers with low CC and PR as dormant papers (which is an optimistic term, on the assumption that they are going to be noticed sometime in the future).

### 5.3.2 Divergence

The plots and table 5.3 above are an attempt to see the difference among metrics, but it is hard from them to understand what this practically means. We next try to quantitatively assess the difference in terms of concrete effects of using a metric over another for what metrics are effectively used, that is, ranking and selection. Assume we are searching the Web for papers on a certain topic or containing certain words in the title or text. We need a way to sort results, and typically people would look at the top result, or at the top 10 or 20 results, disregarding the rest. Hence, the key metric to understand divergence of the two indexes is how often, on average, the top t results would contain different papers, with significant values for $t = 1, 10, 20$. In the literature, the typical metric for measuring a difference between two rankings is the Kendall $\tau$ distance [42], measured as the number of steps needed to sort bi-ranked items so that any pair A and B in the two rankings will satisfy to the condition

$$sign(R_1(A) - R_1(B)) = sign(R_2(A) - R_2(B)) \tag{5.7}$$

where $R_1$ and $R_2$ are two different rankings. However, this measure does not give us an indication of the practical impact of using different rankings, both for searching papers and, as we will see later, for authors. What we really want to understand is to see the distance between two rankings based on the actual paper search patterns. Assume we are searching the Web for papers on a certain topic or containing certain words in the title or text. We need a way to sort results, and typically people will look at the top result, or at the top 10 or 20 results, disregarding the rest. Hence, the key metric to understand divergence of the two indexes is how often, on average, the top t results would contain different papers, with significant values for $t = 1, 10, 20$. For example, the fact that the papers ranked 16 and 17 are swapped in two different rankings is considered by the Kendall distance, but is in fact irrelevant from our perspective. To capture this aspect, we propose a metric called divergence, which quantitatively measures the impact of using one scientometric index versus the other. Consider two metrics M1 and M2 and a set of elements (e.g., of papers) S. From this set S, we take a subset n of elements, randomly selected. For example, we take the papers related to a certain topic. These n papers are ranked, in two different rankings, according to two metrics M1 and M2, and we consider the top t elements. We call divergence of the two metrics, $Div_{M1,M2}(t, n, S)$, the average number of elements that differ between the two sets (or, t minus the number of elements that are equal). For example, if S is our set of ACM papers, and n are 1000 randomly selected papers (say, the papers related to a certain topic or satisfying certain search criteria), $Div_{CC,PR}(20, 1000, S)$ measures the average number of different papers that we would get in the typical 20-item long search results page. We measured the divergence experimentally for CC and PR, obtaining the results in the table 5.3.2 below. As a particular case, $Div_{M1,M2}(1, n, S)$ measures how often does the top paper differs with the two indexes.

| $t$ | $Div_{PR,CC}(t, 1000, S)$, in % | $Div_{PR,CC}(t, 1000, S)$ |
|---|---|---|
| 1 | 62.40 | 0.62 |
| 10 | 49.94 | 4.99 |
| 20 | 46.42 | 9.28 |
| 40 | 43.29 | 17.31 |
| 60 | 42.51 | 25.5 |
| 80 | 41.75 | 33.39 |
| 100 | 40.52 | 40.52 |

Table 5.4: Experimentally measured divergence for the set of ACM papers.

The table 5.3.2 is quite indicative of the difference, and much more explicit than the plots or other evaluation measures described above. In particular, the table shows that more than almost 2/3 of the times, the top ranked paper differs with the two metrics. Furthermore, and perhaps even more significantly, for the traditional 20-element search result page, nearly half of the paper would be different based on the metric used. This means that the choice of metric is very significant for any practical purposes, and that a complete search approach should use both metrics (provided that they are both considered meaningful ways to measure a paper). In general we believe that divergence is a very effective way to assess the difference of indexes, besides the specifics of CC and PR. We will also see the same index on authors, and the impact that index selection can therefore have on people's careers. Details on the experiments for producing these results and the number of measures executed are reported in the companion web page.

### 5.3.3 Understanding the difference

We now try to understand why the two metrics differ. To this end, we separate the two factors that contribute to PR, see equation 5.1: the PR measure of the citing papers and the number of outgoing links of the citing papers (or numerator and denominator). To understand the impact of the weight, we consider for each paper $P$ the weight of the papers citing it (we call this the potential weight, as it is the PR that the paper would have if all the citing papers $P$ only cited $P$). We then plot (Figure 5.3) the average potential weight for the papers in a given square (intersection of a CC and a PR band) in the banded chart. The estimation of the impact of outgoing links will be done in the following way.

What we want to see when examining the effect of outgoing links from citing paper, is the "weight dispersion", that is, how much weight of the incoming papers (i.e., how much potential weight) is dispersed through other papers as opposed to being transmitted to P. This is really the measure of the "damage" that outgoing links do to a Paper Rank. We compute the dispersed weight index for a paper P (DW(P)) as the sum of the PR of the citing papers C(P) (that is, the potential weight of P) divided by the PR of P (the actual weight). Figure 5.4 plots the average dispersed weight for each square, as usual by CC and PR. The dark area in the bottom right corner is because there are no papers there.

These two charts very clearly tell us that outgoing links are the dominant effect for the divergence between CC and PR. Papers having a high CC and low PR have a very high weight dispersion, while papers with high PR and low CC are very focused and able to capture nearly all potential weight. The potential weight chart (Figure 5.3) also tends to give higher numbers

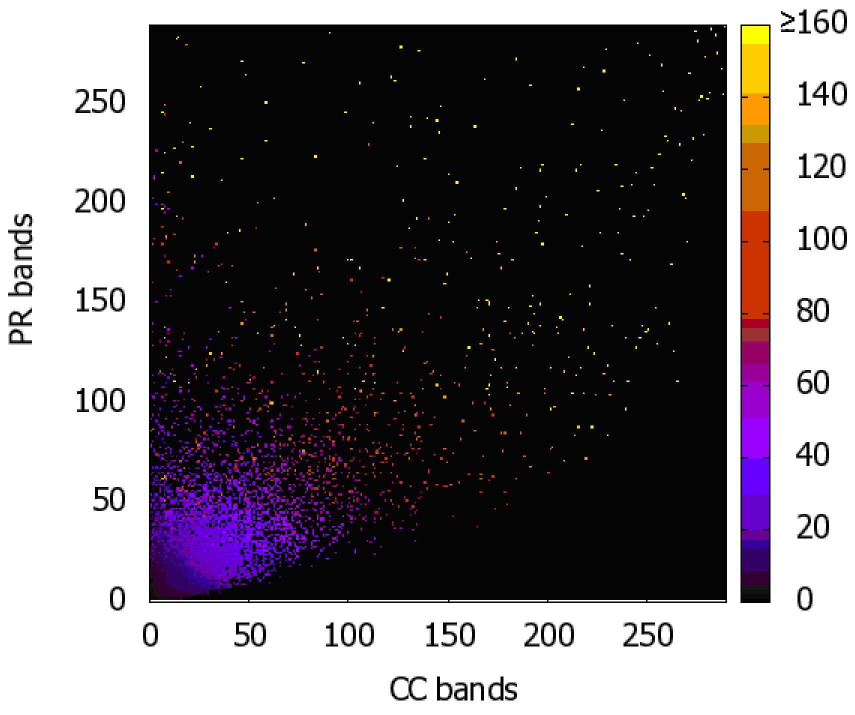


Figure 5.3: Average potential weight for all papers in a square The color in the Z-axis denotes the weight X axis plots CC bands, Y axis plots PR mirror-banded by CC.

for higher PR papers but the distribution is much more uniform in the sense that there are papers in the diagonal or even below the diagonal and going from the top left to the bottom right the values do changes but not in a significant way (especially when compared to the weight dispersion chart). To see the difference concretely on a couple of example, we take a "hidden gem" and a "popular paper", see Figure 5.5.

The specific gem is the paper Computer system for inference execution and data retrieval, by R. E. Levien and M. E. Maron, 1967. This paper has 14 citations in our ACM-only dataset (Google Scholar shows 24 citations for the same paper). The PR of this "hidden gem" is 116.1, which is a very high result: only 9 papers have a greater rank. Let's go deep inside the graph to see how this could happen. Figure 5.6 shows all the incoming citations for this paper up to two levels in the citation graph.

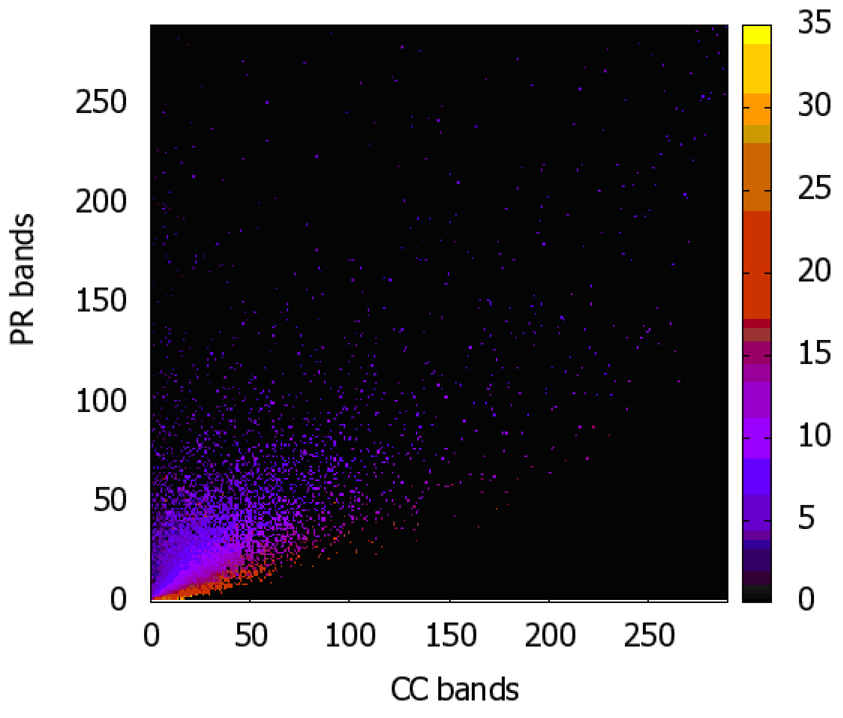


Figure 5.4: Average dispersed weight for all papers in a square The color in the Z-axis denotes the weight X axis plots CC bands, Y axis plots PR mirror-banded by CC.

The paper in the center is our "gem", and this is because it is cited by an heavyweight paper that also has little dispersion: it cites only two papers. We observe that this also means that in some cases a pure PR may not be robust, meaning, the fact that our gem is cited by a heavyweight paper may be considered a matter of "luck" or a matter of great merit, as a highly respected "giant" is citing it. Again, discussing quality of indexes and which is "better" or "worse" is outside our analysis scope, as is the suggestion for the many variations of PR that could make it robust.

We now consider a paper in the bottom of the CC vs PR plot, a paper with high number of citations but relatively low PR. The corresponding citation graph is shown in Figure 5.7. This paper has 55 citations in our ACM dataset (158 citations in Google Scholar) and a relatively poor PR of 1.07. This result is not particularly bad, but it is much worse than

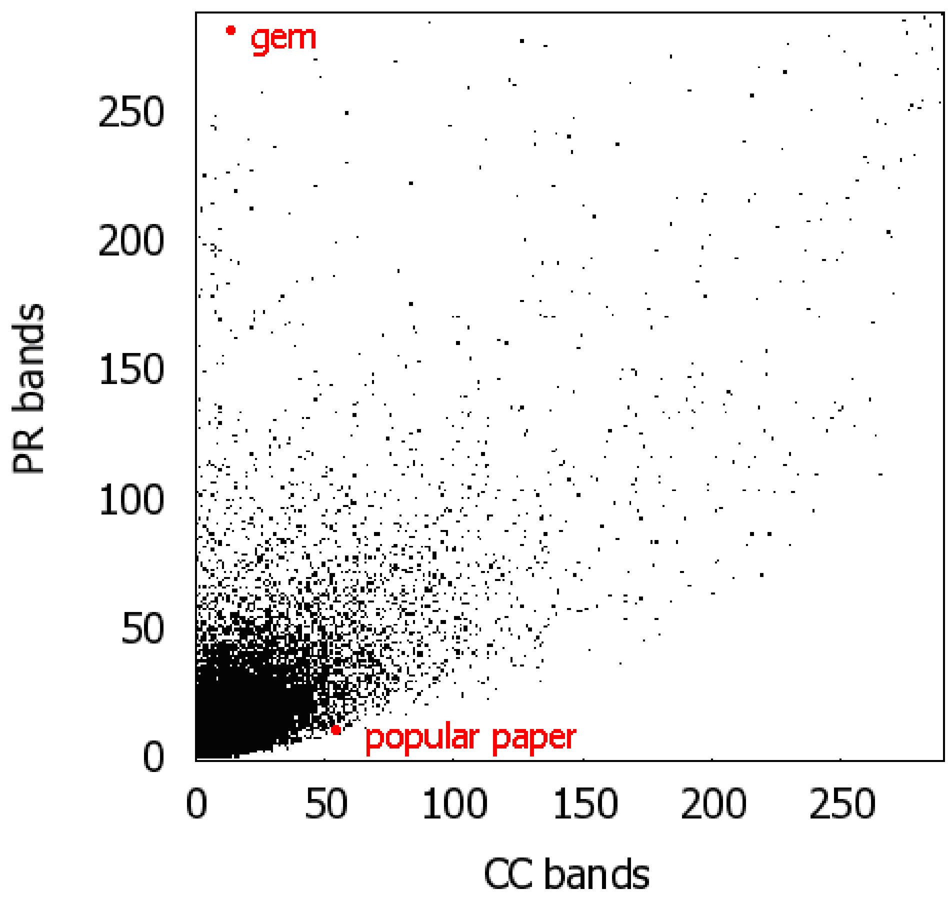


Figure 5.5: “Gem” and “popular paper” (or “stone”) relative positions.

other papers with similar number of citations. There are 17143 papers in the dataset that have grater Paper Rank and just 1394 papers with better citation count. Comparing with papers in the same CC and PR band, this paper has a weight dispersion factor that is over twice that of papers in the same CC band and three times the one of papers in the same PR band, which explain why the increased popularity with respect to papers in the same PR band did not correspond to a higher PR. As a final comment, we observe that very interestingly there are papers with very low CC and very high PR, but much less papers - almost none - with very high CC and very low PR. If we follow the dispersion plot this is natural, as it would assume that the dispersed weight should be unrealistically high (many papers with hundreds of citations) which does not happen in practice, while it is possible to have "heavyweight" papers with very few citations that make the presence of paper gems (papers in the top left part) possible. However, we believe that the absence of papers in the bottom right part and, more in general, the skew of the plot in Figure 5.2 towards the upper left is indicative of a "popularity bias". In the ideal case, an author A would read all work related to a certain paper P and then decide which papers to reference. In this case, citations are a very meaningful measure (especially if they are positive citations, as in the motto "standing on the shoulders of giants"). However this is impossible in practice, as nobody can read such a vast amount of papers. What happens instead is that author A can only select among the papers she "stumbles upon", either because they are cited by other papers or because they are returned first in search results (again often a result of high citation count) or because they are published in important venues. In any event, it is reasonable to assume that authors tend to stumble upon papers that are cited more often, and therefore these papers have a higher chance of being cited than the "hidden gems", even if maybe they do not necessarily have the same quality. We believe that it is for this reason that over time, once a paper increases with citation count, it necessarily increases with the weight, while gems may remain "hidden" over time. A detailed study of this aspect (and of the proper techniques for studying it) is part of our future work.

## 5.4 Focusing, seeking for "golden middle". De Solla Price principle.

The computational procedures to acquire PageRank or PaperRank have been described in Subsection 5.2.3, let us now try to look over this problem from another end. How PageRank or PaperRank may be changed and why. From the variety of possible PageRank modifications I would like to count the following:

- PR Computation with or without dump factor (see formula 5.1, 5.3 above).
- Personalized Page Rank with some initial personalization vector is more common for web-search engines. Here all pages have their own personal weights before PR calculation.
- Focusing of PR, or redistribution of links to link probabilities in the stochastic Markov matrix. This means that core PR model of Random Surfer is no longer Random, it becomes focused. This model was successfully applied to the web pages ranking problem by Tony Abou-Assaleh *et al.* [2] and by Fuyong Yuan *et al.* [98] in 2007. Most recent application of Focused Random Surfer model is applied by Prof. Lee Giles for ranking items in Autonomous Digital Library CiteSeerX [82]. This is the closest research to the present one to the best of our knowledge.
- Double (or more) focusing of PR takes into account more deep properties of citation graph entities during stochastic Markov matrix composition. For example, it may first focus on site name and then on site content.

### 5.4.1 Focused Surfer

The Random Surfer model is the basis of PageRank algorithm. PageRank of the certain node is proportional to the probability to reach this node by randomly riding the graph. At each step rider randomly chooses the link to follow. Focused Surfer decides which path is more preferable for him.

Formula 5.1 may be rewritten to better expresses this mathematically,

$$P_i = (1-d) \sum_{i \neq j}^{j \in D} P_j \cdot s(j|i) + \frac{d}{N} \tag{5.8}$$

where $s(j|i)$ is the probability to follow the reference $i$ being at the place $j$. $s$ is a function that may be arbitrary. We propose to use the simplest variant of it, which we show in formula

$$s(j|i) = \frac{C(i)}{\sum_{k \in D} C(k)} \tag{5.9}$$

where C(m) is paper m citations count, and D is the set of all references in paper C(j). This means that more cited nodes have advantage and they are more visible and attractive for further citation. More complex analogue of such focusing proposal author has found after publishing of this contribution [49] in the paper of Prof. Lee Giles [82].

### 5.4.2 Evaluation, comparison with usual PR

Evaluation for the problem of Focused Paper Rank is performed for the dataset presented in the Section 5.2.1. We use the same mirrored-plotting methodology described in Subsection 5.3.1. The result is shown in Figure 5.8. In the top (sub-figure a)) we see the original Paper Rank, in the bottom b) the focused one.

Figure 5.8 b) (in the bottom) illustrates the Focused Surfer model and FPR algorithm instead of PR. Focused Surfer model gives better chances to more cited papers, at the same time stealing the part of the weight from their poorly cited neighbors. This idea leads us to the conclusion that in general total FPR rank remains the same as PR, it just gets re-distributed. This idea is supported by computation of average FPR and PR which are nearly the same: $< FPR >$=0.603 and $< PR >$=0.602. Now let us observe effects present in Figure 5.8 b). The points are located closer to the main diagonal (comparing with plot a)) and there is significantly less papers with big CC and small PR (reducing of the effect of outbound links). On the other hand we see that "gems"-effect is still noticeable. This means that FPR tends to be a "middle" between PR and CC.

### 5.4.3 Is FPR "better" than PR?

Focused Page Rank major strong points are:

1. It is the tradeoff between Page Rank and Citation Count. So it may serve as an agreement between the followers of pure citation count and Page Rank followers.

2. Proposed solution less suffers from the effect of outbound links.

3. It reflects one of the fundamental principles of Scientometrics, first time formulated by de Solla Price in 1976: *"Success seems to breed success. A paper which has been cited many times is more likely to be cited again than one which has been little cited. An author of many papers is more likely to publish again than one who has been less prolific. A journal which has been frequently consulted for some purpose is more likely to be turned to again than one of previously infrequent use"*.

4. It captures the power of Page Rank, where not only the quantity of citations, but also the quality of ones counts.

## 5.5 Exploring Author Metrics

### 5.5.1 Plotting the difference

We now perform a similar analysis on authors rather than papers. For this, we initially consider PRH and Hirsch as main metrics, and then extend to other metrics. The plot to visualize the differences (Figure 5.9) is similar in spirit to the one for CC vs PR. The X-axis has Hirsch values, while the Y-axis has PRH values. A first observation is that applying “Hirsching” to CC and PR to get H-index and PRH smoothes the differences, so we do not have points that are closer to the top left and bottom right corners. This could only happen, for example, if one author had many papers that are hidden gems.

Since the authors with low Hirsch and PRH are dominant, a log scale was used plotting Figure 5.9. This increased similarity is also shown in Table 5.5.1, where many papers are on the diagonal (this is also due to the fact that we have a much smaller number of squares in this chart). The mean distance from the diagonal is 0.25 bands, while the standard deviation is 0.42 bands. Interestingly, as we will see, though at first look the differences seem less significant, the impact of using one rather than the other index is major.

| Distance in bands from the main diagonal | Percent of authors with this distance |
|---|---|
| 0 | 83.07% |
| 1 | 12.23% |
| 2 | 2.90% |
| 3 | 0.99% |
| 4 | 0.40% |
| 5 | 0.19% |
| 6 | 0.09% |
| 7 | 0.05% |
| 8 | 0.03% |
| 9 | 0.02% |
| 10 | 0.01% |
| ≥11 | 0.01% |

Table 5.5: Deviation of authors around main diagonal.

### 5.5.2 Divergence

The same measure of divergence described for papers can be computed for authors (since divergence is a universal measure and may be applied to different ranking schemas). The only difference is that now the set S is a set of authors, and that the indexes are H-index and PRH instead of CC and PR. We also compute it for n=100, as the experiment we believe it is meaningful here is to consider replies to a typical job posting for academia or a research lab, generating, we assume, around 100 applications.

| $t$ | $Div_{PRH,H}(t)$ divergence for PR-Hirsch and Hirsch |
|---|---|
| 1 | 59.3% |
| 5 | 50.04% |
| 10 | 46.13% |
| 20 | 43.47% |

Table 5.6: Divergence between $PRH$ and $H$, $n = 100$.

Although nobody would only make a decision based on indexes, they are used more and more to filter applications and to make a decision in case of close calls or disagreements in the interview committees. The Table 5.5.2 tells us that almost two third of the times, the top candidate would differ. Furthermore, if we were to filter candidates (e.g., restrict to the top 20), nearly half of the candidates passing the cutoff would be different based on the index used. This fact emphasizes once again that index selection, even in the case of both indexes based on citations, is key to determining the result obtained, be them searching for papers or hiring/promotion of employees. Notice also that we have been only looking at differences in the elements in the result set. Even more are the cases where the ranking of elements differ, even when the t elements are the same. Another interesting aspect is that the divergence is so high even if the plot and Table 5.5.1 show values around the diagonal. This is because most of the authors have a very low H and PRH (these accounts for most of the reasons why authors are on average on the diagonal). However, and this can also be seen in the plot, when we go to higher value of H and PRH, numbers are lower and the distribution is more uniform, in the sense that there are authors also relatively far away from the diagonal (see the softer colors and the distributions also far from the diagonal towards the top-right quadrant of Figure 5.9). Incidentally, we believe that this confirms the quality of divergence as a metric in terms of concretely emphasizing the fact that the choice of index, even among citation-based ones, has a decisive effect on the result. We omit here the section on "understanding the difference" as here it is obvious and descends from the difference between CC and PR, described earlier and used as the basis for PRH and Hirsch respectively.

### 5.5.3 Divergence between other indexes

The discussion above has focused on PRH vs H. We now extend the same analysis to other indexes. The table 5.5.3 below shows a comparison for PRH, H, G index, and the total citation count for an author (the sum of all citations for the paper by an author, denoted as TCC in the table).

| $t$ | $PRH$ vs $G$ | $PRH$ vs $TCC$ | $H$ vs $TCC$ | $H$ vs $G$ | $G$ vs $TCC$ |
|---|---|---|---|---|---|
| 1 | 56.3 | 56.4 | 38.2 | 34.6 | 29.9 |
| 5 | 45.66 | 46.38 | 29.48 | 25.58 | 23.84 |
| 10 | 43.05 | 43.03 | 27.9 | 22.94 | 22.95 |
| 20 | 41.3 | 41.66 | 27.63 | 21.70 | 22.62 |

Table 5.7: Divergence for the different indexes in %, $n = 100$ (for simplicity the $Div()$ notation is omitted).

The first lesson we learn from the table is that no two indexes are strongly correlated. The higher correlation is between G and the total citation count, and we still get the top choice different in one out of four cases. The other interesting aspect is that PRH and H are the pair with the highest divergence, which makes them the two ideal indexes to be used (in case one decides to adopt only two indexes).

## 5.6 Discussion

We have explored the problem of ranking of scholarly papers in citation networks. We argue in favor of usage of invented modification of PageRank – PaperRank as the proper measure of an impact of scientific paper. The major argument here is that the PaperRank takes into account not just the quantity of citations but also the quality of ones. Another achievement of the Thesis is the adaptation of Focused Paper Rank for scientific citing. Of course, it cannot displace traditional measure: Citation Count, but can (and we believe should) be used in parallel.

The same thing is about PR-based Hirsch index, it is a try to grasp not just the broadness of an author in terms of quantity of citations per paper, but also an attempt to estimate the real impact of a certain researcher.

The other interesting impact of the present Thesis is in understanding and visualizing the difference between various indexes. Here we mention the innovative methodology of plotting the difference and computation of divergence among different indexes.

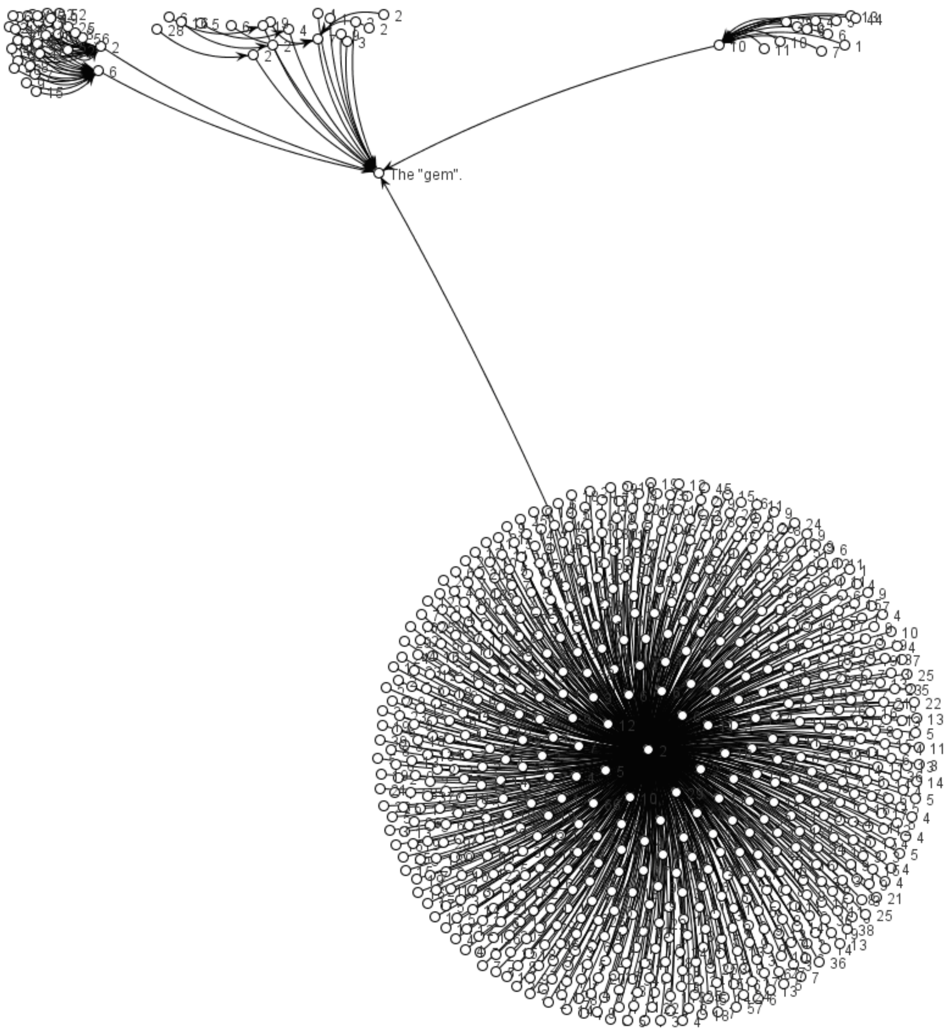


Figure 5.6: One of the "hidden gem" in the dataset, paper of E. Levien and M. E. Maron (in the center). Arrows refer to incoming citations. The digits near the papers refer to the quantity of outgoing links.

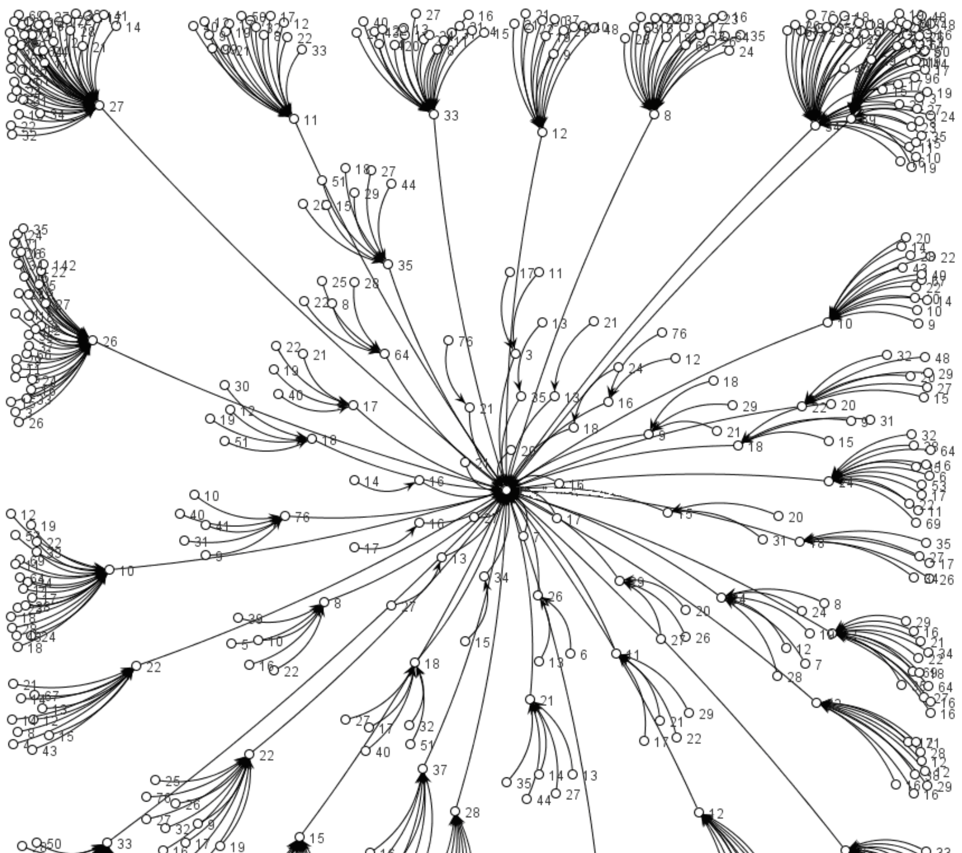
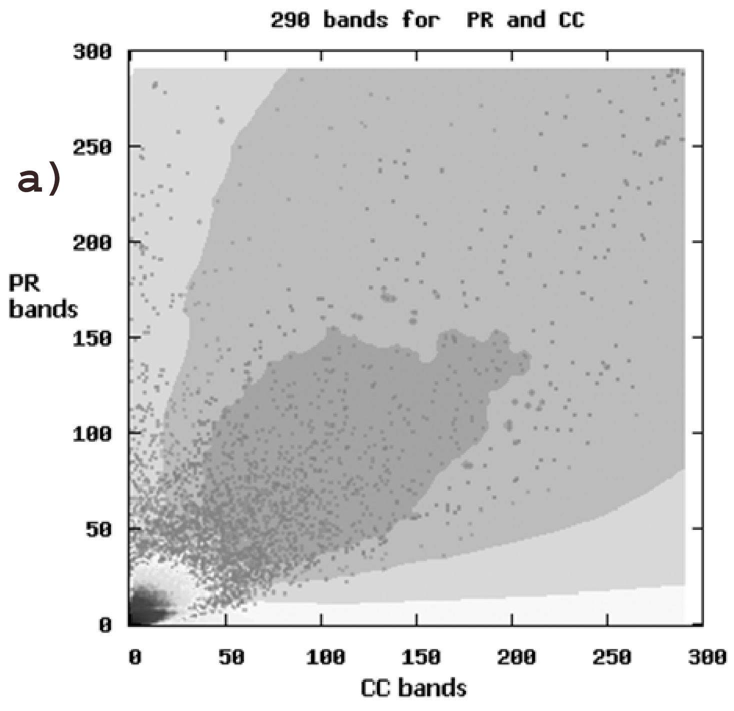


Figure 5.7: "Popular paper" (in the center): relatively highly cited but not very well-ranked.



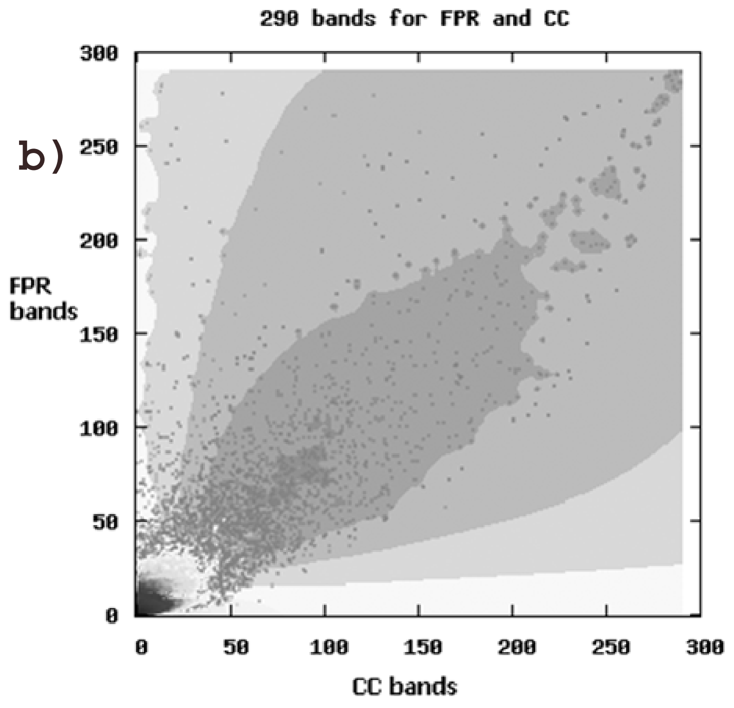


Figure 5.8: Diversity of PR a), FPR b) and Citation Count CC. White and black points in the bottom-left corner does not mean absence of papers. This is a gray-scale of colored map, where the major quantity of papers has small number of CC, and since lie exactly in the bottom-left corner and it is nearly the same for the both plots.

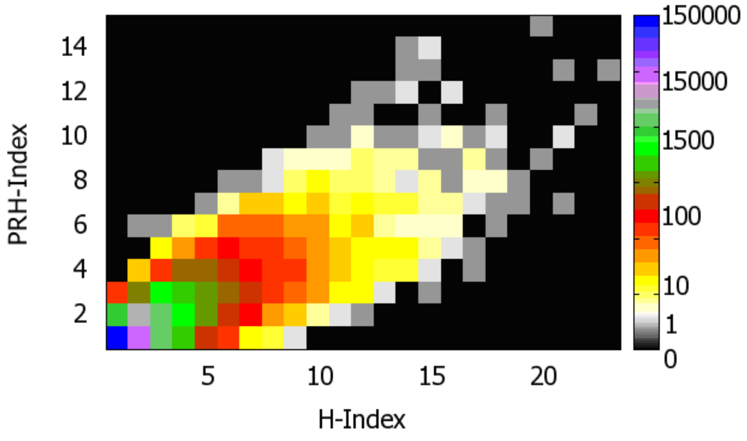


Figure 5.9: The gradient of Hirch and PRHirch in log scale. Author's density is plotted with colors: authors' number goes from 1 to 149170 of authors per square. PR-Hirch has been rounded.

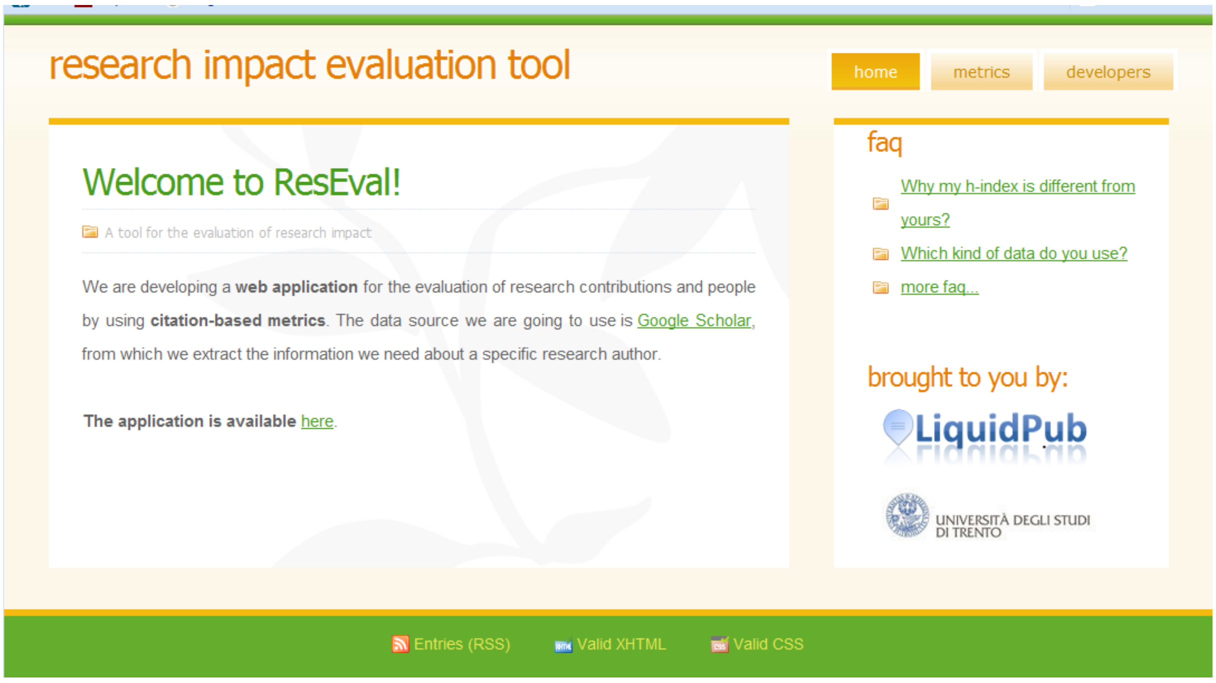
# Chapter 6

# Improving Search, Navigation and Quality in ADL: MockUp

This Chapter is dedicated to the possible application of the mining methods and ranking schemas described above.

## 6.1 PageRank in ResEval tool

In the web era we see rapid evolution of internet, for instance a lot of web-based communities, blogs and sites appear. Connecting people into a huge graph named internet is a web 2.0, this is interactive web. With help of gathering users' feedbacks, web community may pick-up the most interesting opinions, comments, short articles and news and advertise them with tremendous speed around the world. This idea of discussing primary visions, short opinions and make process of maturing of ideas very quick is the message of LIQUIDPUB [12] project. This project is started in 2008 and has been running until now. I contributed to it with the tool for gathering information from Google Scholar [17] and computing $H$, $G$ and some other indexes. The tool may compute the indexes excluding self-citations and citations from co-authors. The tool looks like in Figure 6.1:



Welcome to ResEval!

A tool for the evaluation of research impact

We are developing a **web application** for the evaluation of research contributions and people by using **citation-based metrics**. The data source we are going to use is Google Scholar, from which we extract the information we need about a specific research author.

**The application is available** here.

Why my h-index is different from yours?

Which kind of data do you use?

more faq...

Figure 6.1: ResEval home.

The details a reader may find by URL `http://project.liquidpub.org/reseval/`. There are some explanations about how it is done and why ResEval tool is more informative than Google Scholar [17]. The detailed scientific part is presented in a set of our publications [50, 49, 46]. The UI of the tool and the functionality is shown in Figure 6.2.

In ResEval a user asks for author he wants to evaluate and restricts the area of search with: publication years, broad branch of science like: "Physics" or "Computer Science" etc. The Figure 6.2 illustrates the tool search result by query for Professor F. Giunchiglia [1], who is the Full Professor in the University of Trento, Italy. Prof. Giunchiglia is the founder of PhD School in Trento [2] and the top researcher in semantics, including matching, semantic search, ontologies creation etc. In the Figure 6.2 we

[1]`http://www.dit.unitn.it/~fausto/`
[2]http://ict.unitn.it

see five indexes, namely *G*-index, *H*-index, average quantity of citations by paper, total quantity of papers and total quantity of citations. It is impossible to compute RageRank or PaperRank at this stage, since we have no whole citation graph used in Google Scholar, but in case we would have one, it would be feasible to have an ability to sort search result not just by citation count (now it is implemented so) but also by PaperRank. The more queries users do, the more information we collect in a database, so we hope soon we will be able to reconstruct a piece of Google graph which will give us a chance to compute the PageRank.

This is the real use-case of usage of various indexes for assessing scientific progress of authors and individual papers.

## 6.2 Tagged search tool

Another potential application of the results presented in this Thesis is tagging. Tagging is the technique which came from web 2.0, and it means marking the content in the web by author of a content or by reader. This is kind of user-created manually classification of information. It is widely used in various web-communities like youtube [3], twitter [4], linked-in [5], live-journal [6] or facebook [7]. Each tag is a short representation of a topic user write about. Users may see tags of other people, make their own tags, and finally search by tag. Same thing may be implemented in scientific digital library. Here intuitively we see that a keyphrase or keyword carries the same sense as a tag. Thus both types of keyphrases, assigned by users/editors/reviewers and extracted automatically may enhance usability of large amounts of information, being self-descriptive, they may make

[3]http://www.youtube.com/
[4]http://twitter.com/
[5]http://www.linkedin.com
[6]http://livejournal.com
[7]http://www.facebook.com/

easier not the search only, but also be useful for:

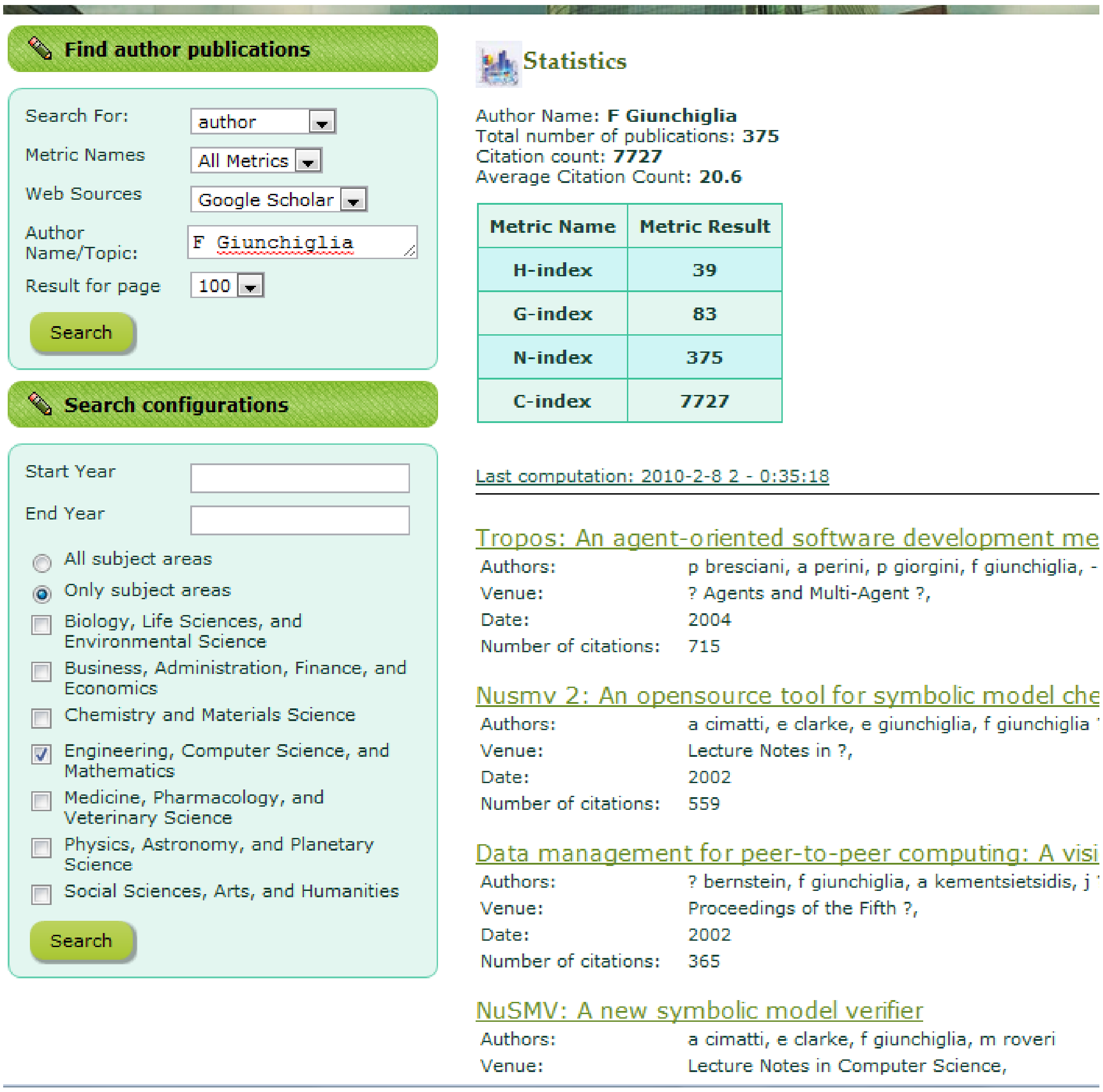
- State-of-the-art bounds detection.
- Seeking for experts/best papers in a small sub-domain.
- Define classifications and categories more precisely.

Here it is important to emphasize, that good ranking may serve for the same reasons, indeed, for detection of best experts in domain we may exploit Hirsch or G-indexes. PageRank-based indexes in combination with Citation Count or separately may be adopted for seeking for best papers. Thus we claim in favor of usage of combination of both approaches: proper ranks with categorized or tagged content. The draft of a tool for it is presented in the Figure 6.3, where all new parts are marked in red.

They are:

- new option for narrowing search for an author/authors: keyphrases input box;
- kind of a "cloud of tags" where all important keyphrases which describe areas of expertise of an author are presented. In case of too many of them we may choose the most crucial using most frequent and linguistically commonsensical ones, for instance noun phrases. Most important tags of keyphrases may be highlighted or printed in bigger font;
- Co-occurrence of particular keyphrases may establish connections between documents and so we can find "related papers" (marked in red in the bottom), which may improve the navigation and search for bounds of a state-of-the-art;
- More indexes to assess an author;

We see that such kind of a tools may be exploited as an effective instrument for finding more interesting and relevant information about top people in domain, seeking for bounds of state-of-the-art, assessing candidates for promotion or candidates for hiring in academia. Evaluating scientific progress with the wide spectrum of ranks is more expressive, and keyphrases may effectuate navigation between domains, authors or related articles easier. The problem of search is getting more sharp since nowadays just one particular problem, for instance PageRank studied in Chapter 5, has thousands papers devoted to. It is getting simply impossible to read at least 30% of them. But with help of appropriate tools this problem become solvable, most "noisy" (duplicated, incremental, erroneous) papers may be cut with help of proper ranking.



**Find author publications**

| | |
|---|---|
| Search For: | author |
| Metric Names | All Metrics |
| Web Sources | Google Scholar |
| Author Name/Topic: | F Giunchiglia |
| Result for page | 100 |

**Search configurations**

| | |
|---|---|
| Start Year | |
| End Year | |

- [ ] All subject areas
- [✓] Only subject areas
- [ ] Biology, Life Sciences, and Environmental Science
- [ ] Business, Administration, Finance, and Economics
- [ ] Chemistry and Materials Science
- [✓] Engineering, Computer Science, and Mathematics
- [ ] Medicine, Pharmacology, and Veterinary Science
- [ ] Physics, Astronomy, and Planetary Science
- [ ] Social Sciences, Arts, and Humanities

**Statistics**

Author Name: **F Giunchiglia**
Total number of publications: **375**
Citation count: **7727**
Average Citation Count: **20.6**

| Metric Name | Metric Result |
|---|---|
| H-index | 39 |
| G-index | 83 |
| N-index | 375 |
| C-index | 7727 |

Last computation: 2010-2-8 2 - 0:35:18

Tropos: An agent-oriented software development me

| | |
|---|---|
| Authors: | p bresciani, a perini, p giorgini, f giunchiglia, - |
| Venue: | ? Agents and Multi-Agent ?, |
| Date: | 2004 |
| Number of citations: | 715 |

Nusmv 2: An opensource tool for symbolic model che

| | |
|---|---|
| Authors: | a cimatti, e clarke, e giunchiglia, f giunchiglia |
| Venue: | Lecture Notes in ?, |
| Date: | 2002 |
| Number of citations: | 559 |

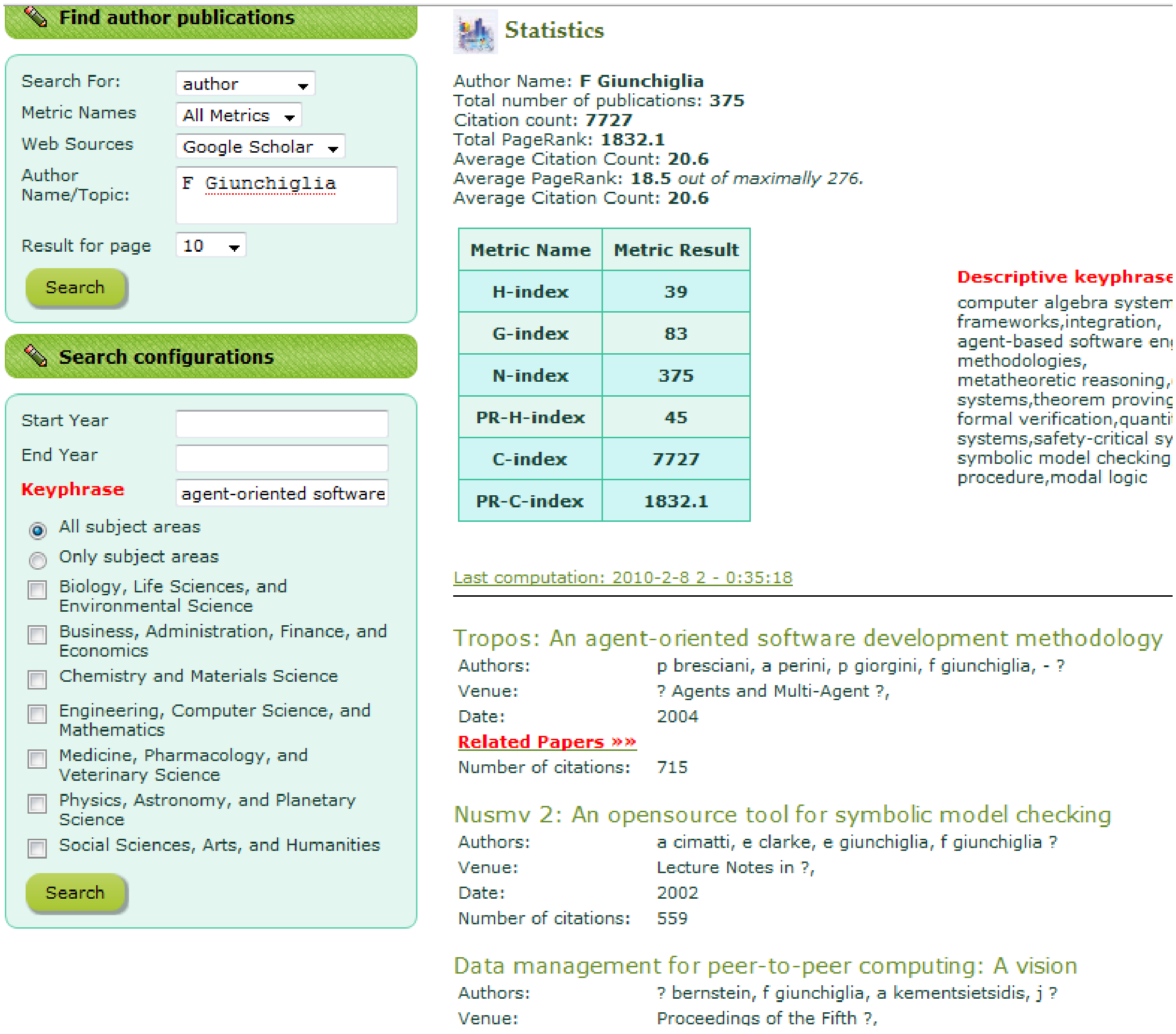
Data management for peer-to-peer computing: A visi

| | |
|---|---|
| Authors: | ? bernstein, f giunchiglia, a kementsietsidis, j |
| Venue: | Proceedings of the Fifth ?, |
| Date: | 2002 |
| Number of citations: | 365 |

NuSMV: A new symbolic model verifier

| | |
|---|---|
| Authors: | a cimatti, e clarke, f giunchiglia, m roveri |
| Venue: | Lecture Notes in Computer Science, |

Figure 6.2: ResEval tool: with several indexes computed for an author.



| | |
|---|---|
| Search For: | author |
| Metric Names | All Metrics |
| Web Sources | Google Scholar |
| Author Name/Topic: | F Giunchiglia |
| Result for page | 10 |

| | |
|---|---|
| Start Year | |
| End Year | |
| Keyphrase | agent-oriented software |

[✓ All subject areas]
Only subject areas
Biology, Life Sciences, and Environmental Science
Business, Administration, Finance, and Economics
Chemistry and Materials Science
Engineering, Computer Science, and Mathematics
Medicine, Pharmacology, and Veterinary Science
Physics, Astronomy, and Planetary Science
Social Sciences, Arts, and Humanities

Author Name: **F Giunchiglia**
Total number of publications: **375**
Citation count: ***7727***
Total PageRank: **1832.1**
Average Citation Count: **20.6**
Average PageRank: **18.5** *out of maximally 276.*
Average Citation Count: **20.6**

| Metric Name | Metric Result |
|---|---|
| H-index | 39 |
| G-index | 83 |
| N-index | 375 |
| PR-H-index | 45 |
| C-index | 7727 |
| PR-C-index | 1832.1 |

**Descriptive keyphrase**
computer algebra system
frameworks,integration,
agent-based software en
methodologies,
metatheoretic reasoning,
systems,theorem proving
formal verification,quanti
systems,safety-critical sy
symbolic model checking
procedure,modal logic

Last computation: 2010-2-8 2 - 0:35:18

Tropos: An agent-oriented software development methodology

| | |
|---|---|
| Authors: | p bresciani, a perini, p giorgini, f giunchiglia, - ? |
| Venue: | ? Agents and Multi-Agent ?, |
| Date: | 2004 |
| **Related Papers »»** | |
| Number of citations: | 715 |

Nusmv 2: An opensource tool for symbolic model checking

| | |
|---|---|
| Authors: | a cimatti, e clarke, e giunchiglia, f giunchiglia ? |
| Venue: | Lecture Notes in ?, |
| Date: | 2002 |
| Number of citations: | 559 |

Data management for peer-to-peer computing: A vision

| | |
|---|---|
| Authors: | ? bernstein, f giunchiglia, a kementsietsidis, j ? |
| Venue: | Proceedings of the Fifth ?, |

Figure 6.3: Enhanced ResEval tool: all new parts are marked as red.

# Chapter 7

# Conclusion

## 7.1 Thesis objectives and achievements

In the present Thesis we address two problems of modern autonomous digital libraries. The first one is the ranking of a scholarly papers.

The Thesis has explored and tried to understand and explain the differences among citation-based indexes. In particular, we have focused on a variation of Page Rank algorithm specifically design for ranking papers - that we have named Paper Rank - and compared it to the standard citation count index. Moreover, we have analyzed related indexes for authors, in particular the Paper Rank Hircsh-index and the commonly-used H-index. We have explored in details the impact they can have in ranking and selecting both papers and authors. The following are the main findings of this Thesis:

i) PR and CC are quite different metrics for ranking papers. A typical search would return half of the times different results.

ii) The main factor contributing to the difference is weight dispersion, that is, how much weight of incoming papers is dispersed through other papers as opposed to being transmitted to a particular paper.

iii) For authors, the difference between PRH and H is again very significant, and index selection is likely to have a strong impact on how people are ranked based on the different indexes. Two thirds of the times the top candidate is different, in an average application/selection process as estimated by the divergence.

iv) An analogous exploration of divergence between several citation-based indexes reveal that all of them are different in ranking papers, with g-index and total citation count being the most similar.

In addition to the findings, we believe that:

i) Divergence can be a very useful and generally applicable metric, not only for comparing citation-based indexes, but also for comparing any two ranking algorithms based on practical impact (results).

ii) There are a significant number of ”hidden gems” while there are very few ”popular papers” (non gem). The working hypothesis for this fact (to be verified) is that this is due to citation bias driven by a ”popularity bias” embedded in the author’s citation practices, i.e. authors tend to stumble upon papers that are cited more often, and therefore these papers have a higher chance of being cited.

The second problem that explored in the Thesis is the problem of extraction of keyphrases from scientific papers. Here we used smaller dataset, specially prepared and refined. The evaluation on prepared dataset shows that:

i) The best results of NLP-based ML methods always outperform KEA in all quality parameters: Precision, Recall and overall F-measure; in particular, it improves the average F-Measure from 22% (KEA) to 30% (Random Forrest) without the use of controlled vocabularies;

ii) A combination of KEA and RF may produce interesting result because both capture different keyphrases.

iii) Feature removal leads to stable decrease of F-Measure for all considered machine learning methods.

iv) Training set size increase improves F-Measure and reaches a "plateau" with training set size around 400 documents.

v) Random Forests is a good tradeoff between quality of keyphrases extraction and computational speed.

vi) NLP helps to avoid "strange" keyphrases candidates often extracted by a purely statistical systems; For instance KEA in some cases recognizes two keyphrases: "ad" and "hoc" instead of capturing "ad hoc" as the whole phrase.

vii) NLP is computationally expensive, but it provides more accurate keyphrases and proposed NLP heuristics cut search space by 50%.

We think that both problems: information extraction and ranking of more coarse-grained pieces of information (*c.a.* documents) are very related for good quality digital libraries construction. To support this opinion we have constructed two research tools: ResEval, the tool for observing major indexes for a particular author using web crawling; and the second one is the prototype of scientific ADL's search system with both paradigms implemented: search by keyphrase and ranked search with PaperRank usage.

## 7.2 Future work

### 7.2.1 Data mining part

Apart from withdrawing syntactic knowledge to improve keyphrases extraction there is another powerful method of statistical texts analysis: each phrase should be taken into account in context. This means the piece of text around a phrase is very important for understanding how valuable is the phrase. This leads us to the area of *unsupervised* statistical learning. We do consider this method as a future of machine learning-based information mining.

### 7.2.2 Ranking

Ranking papers and researchers is a very delicate problem. We know for sure that different co-authors make different impact in the same paper, some people usually cite other works just because it is traditional way to have certain quantity of citations, so some works may be cited by chance. Separating "noisy" papers, papers cited "by chance" from a really breakthrough papers seems to be very challenging and interesting research field. Having certain experience in the domain we do believe that it is statistically possible to draw some interesting conclusions about paper impact just investigating graph structure. We think that the same importance papers have similar graph structure "around". So it is going to be similar to observing a "trace" of a paper in a net of citations.

# Bibliography

[1] *Feature Engineering for Text Classification*, 1999.

[2] T. Abou-Assaleh, T. Das, G. Weizheng, M. Yingbo, P. O'Brien, and Z. Zhen. A link-based ranking scheme for focused search. In *In Proc. WWW2007*, pages 668–677, New York, USA, 2007. ACM Press.

[3] David Austin. Page rank simple samples, computaion, explanation. http://www.ams.org/featurecolumn/archive/pagerank.html.

[4] K. Bharat and G. A. Mihaila. When experts agree: Using non-affiliated experts to rank popular topics. In *Tenth International World Wide Web Conference*, 2001.

[5] Jinbo Bi and Vladimir Vapnik. Learning with rigorous support vector machines. In *COLT*, pages 243–257, 2003.

[6] M. Bianchini, M. Gori, and F. Scarselli. Inside pagerank. *ACM Transactions on Internet Technology*, 5(1), 2005.

[7] Enrico Blanzieri and Farid Melgani. Nearest neighbor classification of remote sensing images with the maximal margin principle. *IEEE Transactions on Geoscience and Remote Sensing*, 46(6):1804–1811, 2008.

[8] Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

[9] Leo Breiman. Random forests. *Machine Learning*, pages 5–32, 2001.

[10] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, 30(1-7):107–117, 1998.

[11] Kurt Bryan and Tanya Leise. The 25,000,000,000 eigenvectors the linear algebra behind google. *SIAM Review*, 48(3):569–582, 2006.

[12] Fabio Casati, Fausto Giunchiglia, and Maurizio Marchese. Publish and perish: why the current publication and review model is killing research and wasting your money. *Ubiquity*, pages 1–1, January 2007.

[13] C.C. Chang and C.J. Lin. *LIBSVM: a library for support vector machines*, 2001.

[14] Chao Chen, Andy Liaw, and Leo Breiman. Using random forest to learn imbalanced data. Technical report, Department of Statistics, UC Berkeley, 2004.

[15] P. Chen, H. Xie, S. Maslov, and S. Redner. Finding scientific gems with google. *Journal of Informetrics*, 2007.

[16] TREC conference. Trec collection. `http://trec.nist.gov/data.html`.

[17] Google Corporation. Google scholar autonomous digital library. http://scholar.google.com/.

[18] Google Corporation. Google search engine. http://www.google.com/.

[19] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[20] Isaac G. Councill, C. Lee Giles, Ernesto Di Iorio, Marco Gori, Marco Maggini, and Augusto Pucci. Towards next generation citeseer: A flexible architecture for digital library deployment. In *Research and Advanced Technology for Digital Libraries*, pages 111–122, 2006.

[21] D. deSolla Price. *Little Science - Big Science.* Columbia Univ. Press, New York, 1963.

[22] M. Diligenti, M. Gori, and M. Maggini. Web page scoring systems for horizontal and vertical search. In *World Wide Web Conference (WWW'2002)*, Honolulu, Hawaii, USA, May 2002.

[23] C. Ercan and I. Cicekli. Using lexical chains for keyword extraction. *Information Processing and Management*, 43:1705–1714, 2007.

[24] Christiane Fellbaum. *Wordnet: An Electronic Lexical Database.* Bradford Books, 1998.

[25] Association for Computing Machinery. Acm digital library. http://portal.acm.org/portal.cfm.

[26] E. Garfield. *Citation Indexing.* ISI Press, 1979.

[27] E. Garfield. The agony and the ecstasy - the history and meaning of the journal impact factor. In *International Congress on Peer Review And Biomedical Publication*, pages 605–612, Chicago, USA, 2005.

[28] C. Lee Giles. Citeseer: Past, present, and future. In *Atlantic Web Intelligence Conference (AWIC)*, 2004.

[29] Lee Giles. Citeseer autonomous digital library. http://citeseer.ist.psu.edu/.

[30] Lee Giles. Citeseerx autonomous digital library. http://citeseerx.ist.psu.edu/.

[31] Fausto Giunchiglia, Mikalai Yatskevich, Paolo Avesani, and Pavel Shvaiko. A large dataset for the evaluation of ontology matching. *Knowledge Eng. Review*, 24(2):137–157, 2009.

[32] Apache group. Lucene, java-written text search engine library. http://lucene.apache.org/.

[33] Apache Group. Nutch crawler. http://lucene.apache.org/nutch/.

[34] Leximancer group. Leximancer. `http://leximancer.com/`.

[35] J[orge]. E. Hirsch. An index to quantify an individual's scientific research output. In *Proceedings of the National Academy of Sciences*, pages 16569–16572, November 2005.

[36] H. Hui, C. L. Giles, E. Manavoglu, H. Zha, Z. Zhang, , and E. A. Fox. Automatic document metadata extraction using support vector machines. In *in Proceedings of the 3rd ACM/IEEE-CS JCDL)*, pages 37–48, 2003.

[37] A. Hulth. Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the 2003 conference on Empirical Methods in Natural Language Processing)*, volume 10, pages 216–223, 2003.

[38] A. Hulth. Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the 2003 conference on Empirical Methods in Natural Language Processing)*, volume 10, pages 216–223, 2003.

[39] IEEE. Ieee digital library. http://www.ieee.org/.

[40] Alexander Ivanyukovich and Maurizio Marchese. Unsupervised free-text processing and structuring in digital archives. In *in Ist Inter-*

*national Conference on Multidisciplinay Information Sciences and Technologies*, 2006.

[41] Sepandar D. Kamvar, Taher H. Haveliwala, Christopher D. Manning, and Gene H. Golub. Extrapolation methods for accelerating pagerank computations. In *WWW*, Budapest, Hungary, 2003.

[42] M. Kendall and J. D. Gibbons. *Rank Correlation Methods.* Edward Arnold, London, 1990.

[43] J. Kleinberg. Authoritative sources in a hyperlinked environment. In *In Proc. Ninth Ann. ACM-SIAM Symp. Discrete Algorithms*, pages 668–677, New York, USA, 1998. ACM Press.

[44] M. Krapivin, M. Marchese, A. Yadrantsau, and Yanchun Liang. Automated key-phrases extraction from scientific papers using domain and linguistic knowledge. In *Digital Information Management. ICDIM 2008)*, pages 105–112, London, GB, Nov 2008.

[45] Mikalai Krapivin. Mikalai krapivin's homepage. `http://dit.unitn.it/~krapivin/`.

[46] Mikalai Krapivin. Focused page rank application for scientific papers ranking in digital libraries. In *VLDL workshop, ECDL conference*, Aarhus, Denmark, 2008.

[47] Mikalai Krapivin, Aliaksandr Autayeu, and Maurizio Marchese. Large dataset for keyphrases extraction. Technical Report DISI-09-055, DISI, Trento, Italy, May 2008. http://eprints.biblio.unitn.it/archive/00001671/01/disi09055-krapivin-autayeu-marchese.pdf.

[48] Mikalai Krapivin, Aliaksandr Autayeu, Maurizio Marchese, Enrico Blanzieri, and Nicola Segata. Improving machine learning approaches

for keyphrases extraction from scientific documents with natural language knowledge. Technical Report DISI-10-003, Trento, DISI, January 2010.

[49] Mikalai Krapivin and Maurizio Marchese. Focused page rank in scientific papers ranking. In *ICADL*, pages 144–153, 2008.

[50] Mikalai Krapivin, Maurizio Marchese, and Fabio Casati. Exploring and understanding scientific metrics in citation networks. In *Complex (2)*, pages 1550–1563, 2009.

[51] A. N. Langville and C. D. Meyer. Deeper inside pagerank. *Internet Mathematics*, 1(3):335–380, 2004.

[52] Amy N. Langville and Carl D. Meyer. Deeper inside pagerank. *Internet Mathematics*, 1:2004, 2004.

[53] S. Lawrence, C. L. Giles, and K. Bollacker. Digital libraries and autonomous citation indexing. *IEEE Computer*, 32(5):67–71, 1999.

[54] Fei Liu, Feifan Liu, and Yang Liu. Automatic keyword extraction for the meeting corpus using supervised approach and bigram expansion. In *Spoken Language Technology Workshop, 2008. SLT 2008. IEEE*, pages 181–184, Dec. 2008.

[55] Feifan Liu, Deana Pennell, Fei Liu, and Yang Liu. Unsupervised approaches for automatic keyword extraction using meeting transcripts. In *The 2009 Annual Conference of the North American Chapter of the ACL*, pages 620–628, June 2009.

[56] J. B. Lovins. Development of a stemming algorithm. *Mechanical Translation and Computational Linguistics*, 11(5):22–31, 1968.

[57] Del Corso G. M., Gull A., and Romani F. The anatomy of a large-scale hypertextual web search engine. *Internet Mathematics*, 2(3):251–273, August 2005.

[58] A. McCallum. Rexa scholar autonomous digital library. http://rexa.info/.

[59] Andrew McCallum, Dayne Freitag, and Fernando C. N. Pereira. Maximum entropy markov models for information extraction and segmentation. In *ICML*, pages 591–598, 2000.

[60] O. Medelyan and I.H. Witten. Thesaurus based automatic keyphrase indexing. In *JCDL*, 2006.

[61] H. Moed. *Citation Analysis in Research Evaluation.* Springer, 2005.

[62] Thomas Morton. *Using Semantic Relations to Improve Information Retrieval.* PhD thesis, University of Pennsylvania, 2005.

[63] Thuy Dung Nguyen and Min yen Kan. Keyphrase extraction in scientific publications. In *In Proc. of International Conference on Asian Digital Libraries ICADL T07*, pages 317–326. Springer, 2007.

[64] Joakim Nivre, Johan Hall, and Jens Nilsson. Maltparser: A data-driven parser-generator for dependency parsing. In *In Proc. of LREC-2006*, pages 2216–2219, 2006.

[65] E. Osuna, R. Freund, and F. Girosi. Support vector machines: Training and applications. In *Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR T97)*, page 130, Washington, DC, USA, 1997. IEEE Computer Society.

[66] F. Peng and A. McCallum. Accurate information extraction from research papers using conditional random fields. In *Proceedings of Human Language Technology Conference*, 2004.

[67] Fuchun Peng and Andrew McCallum. Information extraction from research papers using conditional random fields. *Inf. Process. Manage.*, 42(4):963–979, 2006.

[68] M. F. Porter. An algorithm for suffix stripping. In *Program*, pages 130–137, 1980.

[69] D. Radev, S. Blair-Goldensohn, and Z. Zhang. Experiments in single and multi-document summarization using mead. In *In Proceedings of The First Document Understanding Conference*, 2001.

[70] A. Ratnaparkhi. A maximum entropy part of speech tagger. In *Proceeding of the Conference on Empirical Methods in Natural Language Processing*. University of Pennsylvania, 1996.

[71] A. Ratnaparkhi. A simple introduction to maximum entropy models for natural language processing. Technical Report 97-08, Institute for Research in Cognitive Science, University of Pennsylvania, aug 1997.

[72] Robert E. Schapire, Yoav Freund, Peter Bartlett, and Wee Sun Lee. Boosting the margin: A new explanation for the effectiveness of voting methods. *The Annals of Statistics*, 26(5):1651–1686, 1998.

[73] R[oberto] Scrinzi. Sviluppo di un tool per l'esplorazione semi-automatica all'interno del servizio di google scholar. Master's thesis, University of Trento, Italy, 2008.

[74] Fabrizio Sebastiani. Machine learning in automated text categorization. *ACM Comput. Surv.*, 34(1):1–47, 2002.

[75] Nicola Segata. FaLKM-lib v1.0: a Library for Fast Local Kernel Machines, 2009.

[76] Nicola Segata and Enrico Blanzieri. Fast Local Support Vector Machines for Large Datasets. In Petra Perner, editor, *Machine Learning and Data Mining in Pattern Recognition: 6th International Conference (MLDM 09)*, number 5632 in Lecture Notes in Artificial Intelligence, pages 295–310, Leipzig, Germany, 2009. Springer.

[77] Nicola Segata and Enrico Blanzieri. Fast and Scalable Local Kernel Machines. Technical report, University of Trento, Trento, Italy, 2009. Submitted to a journal.

[78] Andrew E. Smith. Automatic extraction of semantic networks from text using leximancer. In *HLT-NAACL*, 2003.

[79] Springer. Springer digital library. http://www.springer.com.

[80] A. Stolke. Srilm an extensible language modeling toolkit. In *In Proceedings of ICSLP*, page 901904, 2002.

[81] Y. Sun and C. L. Giles. Popularity weighted ranking for academic digital libraries. In *29th European Conference on IR Research (ECIR'2007)*, pages 605–612, Rome, Italy, 2007.

[82] Yang Sun, Huajing Li, Isaac G. Councill, Jian Huang, Wang-Chien Lee, and C. Lee Giles. Personalized ranking for digital libraries based on log analysis. In *10th ACM International Workshop on Web Information and Data Management (WIDM2008)*, pages 133–140, New York, USA, 2008. ACM Press.

[83] JUNG team. Java universal network/graph framework. http://jung.sourceforge.net/.

[84] Michael E. Tipping. Sparse bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, 1:211–244, 2001.

[85] Stephen Tomlinson. Compared the snowball stemmers with the hummingbird lexical stemming (lemmatization) system. In *CLEF*, 2003.

[86] P. Tourney. Coherent keyphrase extraction via web mining. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence (IJCAI-03)*, pages 434–439, 2003.

[87] Kristina Toutanova and Christopher D. Manning. Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP/VLC-2000)*, pages 63–70, 2000.

[88] DBLP Univesität Trier. Dblp digital library. `http://www.informatik.uni-trier.de/~ley/db/`.

[89] P. Turney. Learning to Extract Keyphrases from Text. Technical report, NRC/ERB-1057, feb 1999.

[90] A. F. J. Van Raan. Scientometrics: State-of-the-art. *SCIENTOMETRICS*, 38(1):205–218, 1997.

[91] Vladimir N. Vapnik. *The nature of statistical learning theory.* Springer-Verlag New York, Inc., New York, NY, USA, 1995.

[92] X. Wan, J. Yang, and J. Xiao. Towards an iterative reinforcement approach for simultaneous document summarization and keyword extraction. In *In Proceedings of ACL*, page 552559, 2007.

[93] J. Wang and H. Peng. Keyphrases extraction from web document by the least squares support vector machine. In *IEE/WIC/ACM International Conference on Web Intelligence)*, 2005.

[94] Marcus Watson, Andrew Smith, and Scott Watter. Leximancer concept mapping of patient case studies. In *KES (3)*, pages 1232–1238, 2005.

[95] Christopher K. I. Williams and David Barber. Bayesian classification with gaussian processes. *IEEE Trans. Pattern Anal. Mach. Intell.*, 20(12):1342–1351, 1998.

[96] I. H. Witten, G. W. Paynte, E. Frank, C. Gutwin, and C. G. Nevill-Manning. Kea: Practical automatic keyphrase extraction. In *Proceedings of DL '99*, pages 254–256, 1999. http://www.nzdl.org/Kea/.

[97] Ian .H. Witten and Eibe Frank. *Data Mining: Practical machine learning tools and techniques.* Morgan Kaufmann, San Francisco, 2005.

[98] Fuyong Y., Chunxia Y., and Jian L. Improvement of pagerank for focused crawler. In *Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing*, pages 797–802, 2007.

[99] David Yarowsky. Word-sense disambiguation using statistical models of roget's categories trained on large corpora. In *COLING*, pages 454–460, 1992.

[100] David Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *ACL*, pages 189–196, 1995.