Towards a Keyword-Focused Web Crawler

Tomasz Kuśmierczyk[1], Marcin Sydow[2,1]

[1] Institute of Computer Science, Polish Academy of Sciences, Warsaw, Poland
[2] Polish-Japanese Institute of Information Technology, Warsaw, Poland,
t.kusmierczyk@phd.ipipan.waw.pl,msyd@poljap.edu.pl

Abstract. This paper concerns predicting the content of textual web documents based on features extracted from web pages that link to them. It may be applied in an intelligent, keyword-focused web crawler. The experiments made on publicly available real data obtained from Open Directory Project[1] with the use of several classification models are promising and indicate potential usefulness of the studied approach in automatically obtaining keyword-rich web document collections.

1 Introduction and Motivation

Web crawler is a network application that automatically fetches large collections of web documents via http protocol according to some clearly defined crawling strategy. It is an essential module in various applications including search engines, for example. For some applications it is important to limit the fetching process only to documents that satisfy some criteria concerning their content, for example presence of specific keywords, specific topic of document, etc. Such task is known as *focused crawling*.

A crawler generally works in iterations that, in short, are as follows. Picking a bulk of URL addresses to be fetched, out of an internal priority queue, fetching them via http protocol, parsing and recording the fetched documents, pushing the parsed links that lead to new web documents into the priority queue. In real systems, the process is much more complicated, but the idea is generally as presented.

One of the key technical problems in focused crawling is that the fact whether a document to be fetched is worth fetching (i.e. satisfies the specified criteria) may be verified only *after* it is fetched. In practice, the ratio of web documents satisfying the focused-crawling criteria to all documents that are available in standard, BFS-like crawling scheme may be arbitrarily low. Thus, to save crawler's resources such as network bandwidth, hard disk, CPU, etc. and to efficiently fetch large collection that is rich of documents that satisfy the crawling criteria it is necessary to *predict* the contents of documents to be fetched *without fetching them*.

This can be stated as a binary classification problem: given some specific criterion and the set of already-crawled documents that contain links to an unknown web document x, predict whether x satisfies the criterion without fetching it. More precisely, a supervised learning approach can be used, i.e. the model is learnt on a portion of linked web documents and it is subsequently applied to unknown portion of the web.

[1] http://www.dmoz.org/

In this paper, we study a specific problem of predicting the presence of pre-specified keyword phrase on a web page rather than its topicality that makes it subtly different from most of approaches previously studied in the literature.

Such a specified task has many important applications that usually involve preparing a corpus of documents rich in specific keywords to be further processed by other tools. It may be then used for various tasks ranging from information extraction to statistical analysis of keyword presence to be subsequently used for tuning keyword-based web ad campaigns, for example.

1.1 Related Work

The idea of biasing the crawled collections towards a pre-specified criteria has been intensively studied since early times of web mining. Below we list a selection of representative early works on the topic.

Focused crawling based on a classifier was proposed in [4] where a naive Bayes approach was applied to predict categories of web pages to be fetched.

The phenomenon of "topical locality", i.e. a topical correlation of web documents in a link neighbourhood in WWW was studied in [5].

The idea of taking into account, during web content prediction, the pages that are a few links away, with the concept of context graphs was studied in [6].

Most works concerning the topic use naive Bayesian classifier, though [8] studies many other models and observes that other models may perform better, e.g. SVM (Support Vector Machine). In this paper we apply SVM and CART-Trees, besides Bayesian classifier to evaluate our approach. At this level of work our goal is not to select the best possible classifier and configuration but to gain some knowledge and intuition about their properties in context of the task. Therefore, we decided to use three popular approaches that are also known to be successful in solving similar problems. In further research one can carry additional experiments leading to slight increase in quality.

The concept of intelligent crawler that learns during the crawling was introduced in [1]. The same work proposes to measure the quality of intelligent or focused crawling with *harvest rate* – the proportion of documents "relevant" to the crawl criteria to all harvested documents. The same measure is used in our paper.

As an example of a recent survey, [2] studies various algorithms for prioritising the pages to be crawled with a PageRank-based importance measure.

In contrast to the cited works, and many others, our work focuses on a specific task of crawling web pages that are rich in pre-specified *keywords* rather than of a specific topic. In addition, while we adapt a combination of the machine-learning approaches studied in other works before, including context-graphs, for example, the techniques presented in this paper are very simple, efficient and topic-independent.

2 Problem Statement

In the work described in this paper we focus on a specific issue of short crawls based on usage of a small list of keywords with well chosen seed pages. This keywords might be user's queries or names of entities therefore their length is limited to just several

words. By short crawls we understand crawls going no farther than ten or twenty jumps from layer of seed pages (in experiments we used 25 layers). By well chosen seed we understand the set of pages' URLs with high initial Harvest Ratio:

$$HR = \frac{|valuable\ pages|}{|fetched\ pages|}$$

where we define $valuable\ pages$ ={pages from set $fetched\ pages$ that contain each of keywords at least once} and by $fetched\ pages$ we understand all pages downloaded by crawler in specific set of crawl layers (layer = set of pages fetched by crawler in single work cycle).

3 General ideas and design

In our approach to focused crawling we utilized two main concepts in this area. Our design of a classifier can be understood as a combination of simplified content and link (graph context) analysis approaches. In contrast to the first type of crawlers we decided to use simple keyword-based features that can be computed in a very fast way. Also context is analysed in a simplified way: possible partial overlapping of different link paths is not investigated.

General design of a classification scheme is shown on Figure 1. The scheme presents a process of deciding whether to fetch or not considered page. The process is composed of several steps that produces different outcomes. Outcomes are denoted with consecutive letters of alphabet. The result of last step is a final decision that can be then applied by fetching module. This decision can be interpreted as mentioned in introduction "prediction of content" e.g. system predicts whether considered page contains keywords or not.

The first step of a decision making process is to extract features (denoted with letter A on the Figure 1) of all link paths that lead to the considered URL. It is obvious that only paths included in known part of the web-graph are considered. To avoid loops and filter out irrelevant paths we consider only these links that lead from lower to higher layer. In our implementation we limited paths' length to tree hops.

For every link in the path simple features are extracted:

- whether it points out at valuable (in the sense of the criteria) page or not (of course this feature is not known for the last link in path)
- how many hops backward is needed to get to valuable page
- what is minimum/average distance (measured in number of words) between link position on the page and criteria keywords
- what is the number of keywords occurrences in the link source page
- what would be the fraction of valuable pages considering layers up to the one where the source page is placed if we have not applied cutting-off

As one can see at this point we exploit textual content of previously fetched pages. Nevertheless the content is used in a very simplified and therefore low-cost way e.g. single occurrences and relative positions of keywords are considered. Such an approach

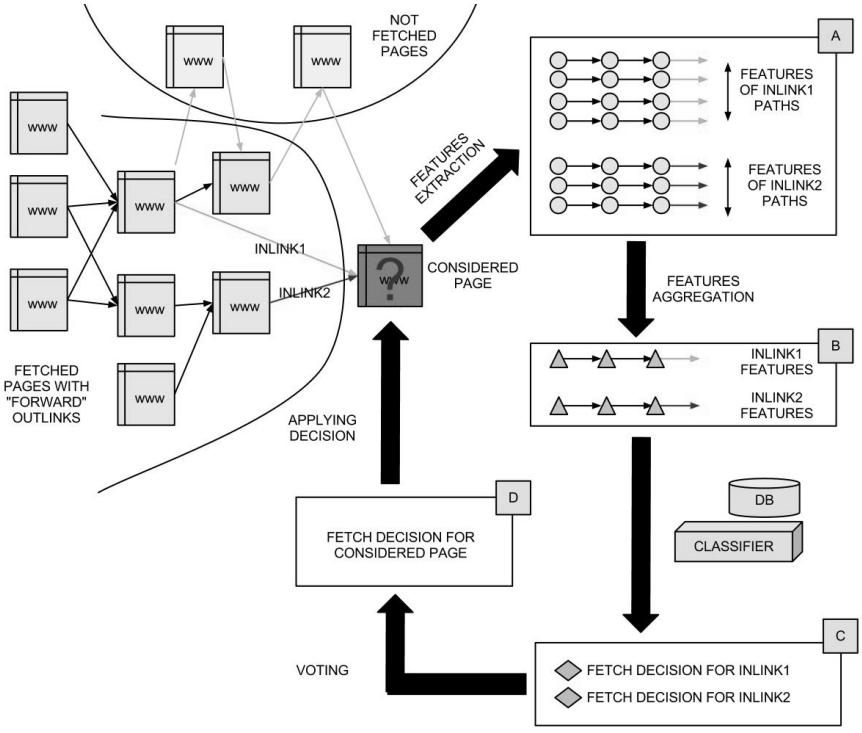

Fig. 1. General schema of features extraction and classification process

uses only small part of available information but as further experiments showed final result, after incorporating these features into graph context structures, is very promising.

During the evaluation process we tested three different subsets of features. First, denoted as Rich, consist of all possible features. In the second, denoted as Poor, we include only 5 features: number of keywords for all pages in considered path (3 features), whether previous page was valuable or not (1 feature) and what is the minimum distance between link and keywords in the current page (1 feature). Third set of features, denoted as Medial, includes a number of features in between Rich and Poor sets.

In the second step, lists of features for paths are grouped according to the last link in the path and then aggregated. For example, having two in-links for the considered URL we obtain two vectors of features (one per each URL). These vectors (denoted with letter B on the Figure 1) are used in classification process: either for training or for test/final classifying.

Classifiers need to define positive and negative class for them. The simplest approach (denoted Simple) is to select links that directly point to a valuable page. Fetching or not of some page may influence reachabilitiy of other pages. Therefore we also considered strategies based on harvest ratio estimation: link is added to positive class if fraction of valuable pages reachable (in farther layers) from link destination page fulfils condition:

$$C \cdot fraction > harvestRatioEstimation$$

General idea is to teach classifier whether link leads to some valuable (e.g. richer in valuable pages than the average) sub-graph or not. Right side of the inequality has the meaning of what is current believe on what is average harvest ratio. It is calculated basing on pages fetched up to the current moment. Left side consist of real (known during learning, but not known after deployment) fraction of future-reachable, valuable pages and constant C that controls what level of harvest ratio is satisfying. It is important to remember that some of the pages in the reachable sub-graph are also reachable from other links that also will be considered, therefore C has not obvious meaning. It can be interpreted as a measure of how much we want to risk fetching unsuitable pages. For $C = 1.0$ we denote this fetch criteria as Harvest0, for $C = 0.3$ as Harvest1 and for $C = 0.001$ as Harvest2.

The last step in URL classification process is to decide whether page should be fetched or not basing on decisions for all in-links (denoted with letter C on the Figure 1). This decision (denoted with letter D on the Figure 1) is made by voting. If there is $\geq F$ votes then page is fetched; otherwise not. Empirically we chose $F = 0.95$ what in practise means that all votes must say 'yes' for fetching.

At current level of advance of our project we set up experimental environment using *python* scripts. The scripts process already downloaded results of crawls and simulate behaviour of focused crawler on off-line data. The data was gathered with Apache Nutch 1.6 crawler and logical consequence of current works would be to reimplement system using Java language as a plug-in to this crawler.

4 Experiments

4.1 Data Characterisation

In experiments we used results of crawls gathered by Apache Nutch 1.6. For seed URLs we used three publicly available Open Directory Project directories:

1. business/e-commerce (655 URLs)
2. recreation/theme parks (485 URLs)
3. computers/mobile computing (510 URLs)

For each of these sets we crawled the web with depth parameter set to 25 layers and maximum breadth set to 1000. It led to fetching more than 20 thousand of pages with hundreds of thousands of links. We analysed the resulting crawls with different sets of keywords. On Figure 2 we show dependence of harvest ratio in different layers for different keywords in considered crawls. Brief review of this figure leads to the conclusion that typical assumption that the ratio of the pages satisfying the crawling criteria decreases with the distance from the seed set, is not necessarily true when taken verbatim. However, after smoothing (not shown) the charts, although quite flat, they generally indicate weak signals of such phenomenon.

4.2 Links classifier parameters selection

Classification process that is shown on Figure 1 depends on many parameters. To choose them we performed several experiments using first crawl results (for business/e-commerce seed). We split links basing on layers into two sets: training set out of layers 3-10 and test set out of layers 11-25. For various keyword sets (shown on Figure 2) we tested different classifiers (Gaussian Naive Bayes [7], CART-Tree [3], Linear-SVM [9]), sets of features (Poor, Medial, Rich) and fetch criteria (Simple, Harvest0/1/2).

To select the best classifier we compared plots of F_1 measures for test set. We fixed other parameters and measured quality for different keywords with different overall harvest ratio (harvest ratio calculated for whole crawl results). All of the plots look similarly to each other. Three sample plots are shown on Figure 3. In general Gaussian Naive Bayes classifier performed the best. The worst results were obtained for CART-Trees. What can also be observed is that results of link classification increase with number of valuable pages. It can be an effect of better representation of positive class in training set.

To select the best subset of features and fetch criteria we performed similar procedures. Table 1 shows averaged values of F_1 for different features subsets and Table 2 for different fetch criteria. Final conclusion is that the best results are obtained for the features denoted as Poor with either Simple or Harvest0 fetch criteria.

4.3 Pages classification

To evaluate classification system's ability to select properly pages to be fetched we performed further simulation. We used layers 3-10 as a training set and 11-25 as a test set. At first we calculated harvest ratio without cutting-off any branches (Original

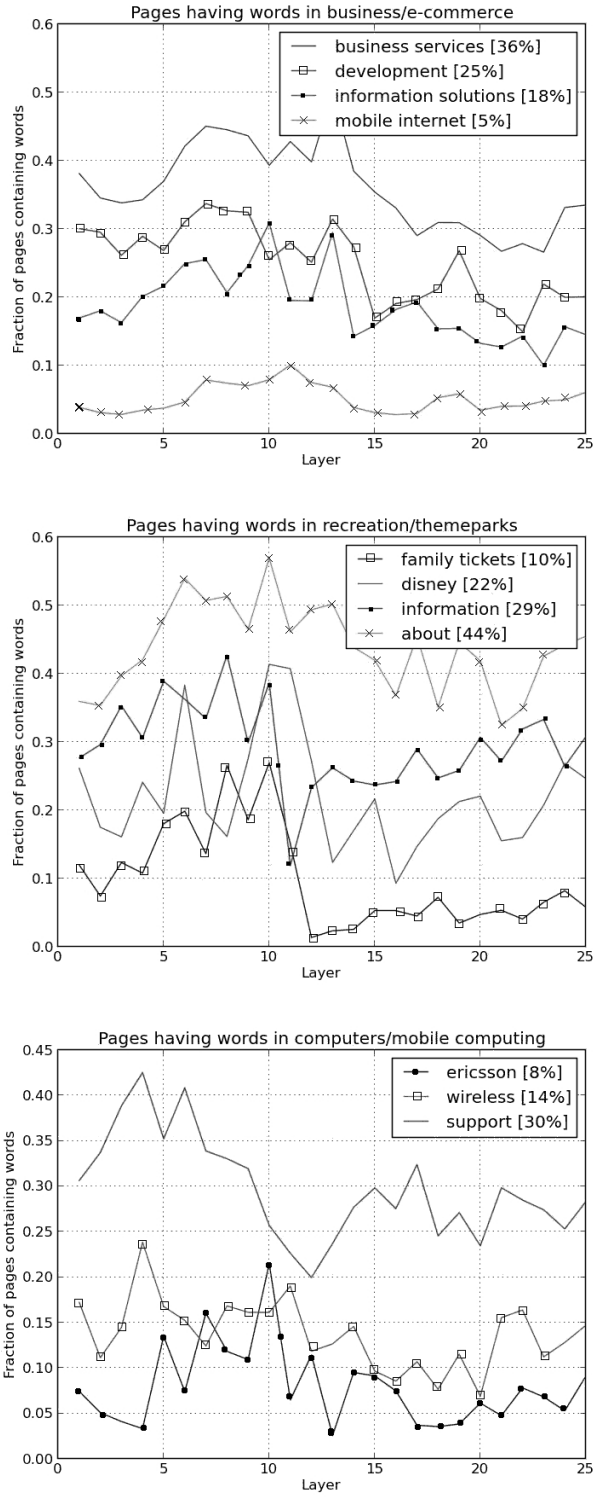

Fig. 2. Harvest ratio in layers for different crawls and keywords (in brackets means are given).

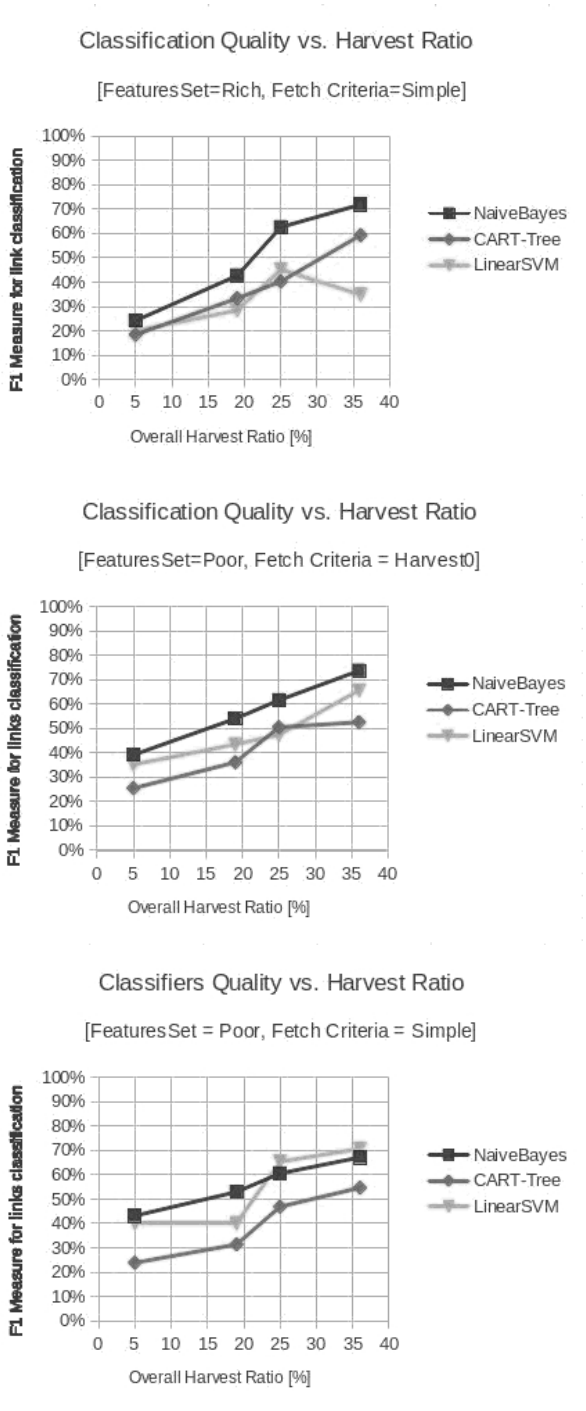

Fig. 3. Link classification quality (F_1) for different classifiers.

Table 1. Average (over different keyword sets) F_1 of links classification for different sets of features [Fetch Criteria = Simple].

Set of Features	NaiveBayes	CART-Tree	LinearSVM
Poor	61%	41%	59%
Medial	52%	40%	44%
Rich	52%	39%	35%

Table 2. Average (over different keyword sets) F_1 of links classification for different Fetch Criteria [Features = Poor].

Fetch Criteria	NaiveBayes	CART-Tree	LinearSVM
Simple	61%	41%	59%
Harvest0	61%	42%	49%
Harvest1	59%	43%	44%
Harvest2	57%	42%	49%

Harvest Ratio). Then we calculated harvest ratio using classification system to cut-off some URLs (and eventually branches). It is important to mention that this behaviour would be quite different in real, on-line focused crawler whereas skipped (cut-off) pages would be replaced with another ones. Anyway, assuming that in this new set of pages we can also successfully perform classification, final ratio would be even better.

Figure 4 presents plots of harvest ratio before and after cutting-off for different fetch criteria. Change of these criteria should strongly influence behaviour of the whole system. Intuitively, when changing this parameter we change what classifier is learned to achieve: either to predict that a single page satisfies the criteria or it contains links that can lead to such. Short analysis of this figure shows that the best results are obtained for strategy Simple and Harvest0.

Figure 5 presents changes of harvest ratio in test set in crawl results for business/e-commerce seed. For all of considered keyword sets quality increased visibly. Ratio improved in the best case of 50%.

4.4 Final evaluation

To confirm that our results apply to different crawl results and keyword sets we repeated simulation for the rest of configurations from Figure 2. In each case we split pages into training and test sets similarly as before. Then, we performed classification and measured harvest ratio change. Charts are shown on Figure 6. These results indicate that the approach is quite successful as the harvest ratio clearly increases. The improvement factor varies from about 1.5 (e.g. business/e-commerce:“business services”) to about 10 times (e.g. computers/mobile computations:“ericsson”).

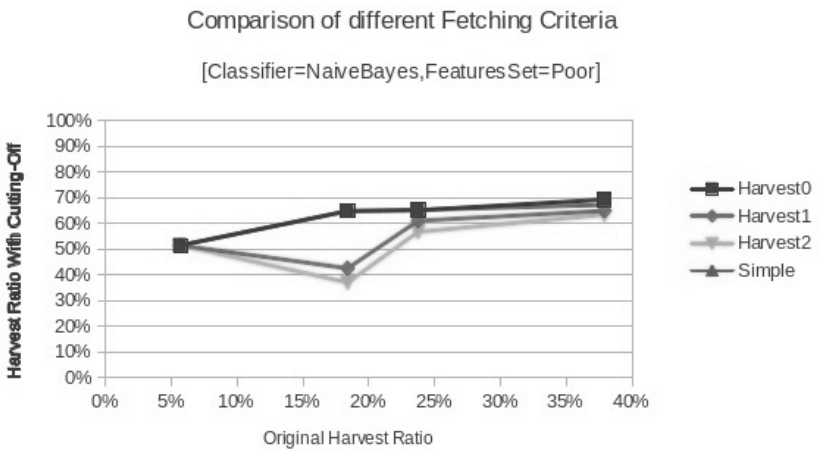

Fig. 4. Harvest ratio for different fetch criteria.

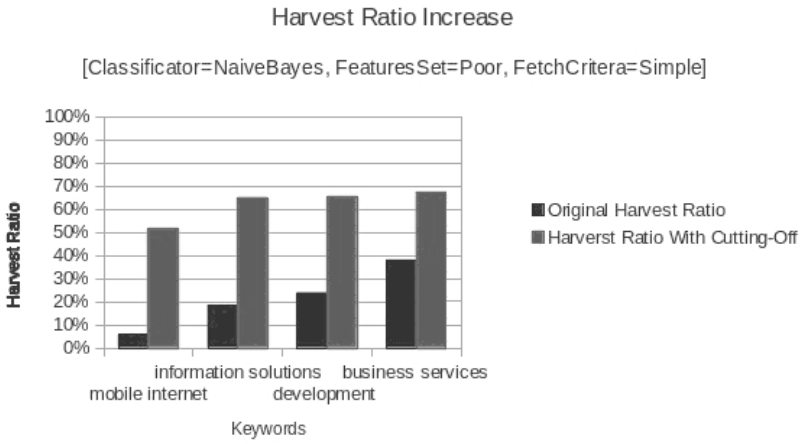

Fig. 5. Harvest ratio increase in crawl for business/e-commerce seed.

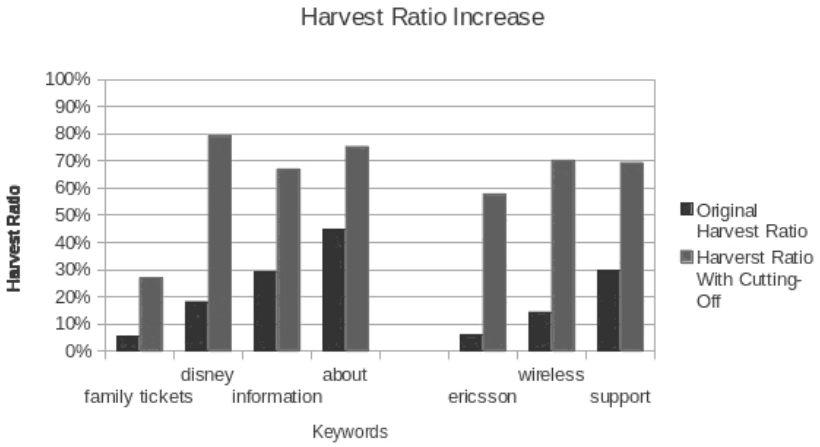

Fig. 6. Harvest ratio increase in crawls for computers/mobile-computations and recreation/theme-parks seed sets.

5 Conclusions and Future Work

We proposed a machine-learning approach to efficiently predict the presence of pre-specified keywords on unknown textual web pages, based on features extracted from web pages in a close link neighbourhood. The features are based both on link structure and textual content. The studied issue has applications in designing an efficient crawler that effectively collects web documents that are rich in pre-specified keywords.

In the reported experimental evaluation we tested numerous combinations of several parameter settings, including various feature sets, classification algorithms and keyword sets. Preliminary experimental results, that are done in a repeatable manner on a publicly available set of web documents from the dmoz.org web site are promising and indicate that the applied approach seems to be successful in obtaining keyword-rich web document collections, despite the simplicity of the applied model.

A continuation of this work would involve more systematic and extensive experimentation, including statistical significance analysis, larger data sets and more sophisticated prediction models, including multi-phrase criterion, for example.

An important improvement of the presented approach would involve incorporating the prediction module into an intelligent crawler that incrementally learns on-line, during the crawling process instead of the off-line learning model presented in this paper. This would also make it possible to apply even more practical evaluation measures that take into account also the consumption of important resources such as crawling time, used bandwidth, etc.

Acknowledgements

The first author was supported by research fellowship within "Information technologies: research and their interdisciplinary applications" agreement between IPS PAS and Polish Ministry of Science and Higher Education POKL.04.01.01-00-051/10-00, the second author was supported by PJIIT grant ST/SI/02/2011.

References

1. Charu C. Aggarwal, Fatima Al-Garawi, and Philip S. Yu. Intelligent crawling on the world wide web with arbitrary predicates. In *Proceedings of the 10th international conference on World Wide Web*, WWW '01, pages 96–105, New York, NY, USA, 2001. ACM.
2. Md.Hijbul Alam, JongWoo Ha, and SangKeun Lee. Novel approaches to crawling important pages early. *Knowledge and Information Systems*, 33:707–734, 2012.
3. L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees*. Wadsworth and Brooks, Monterey, CA, 1984.
4. Soumen Chakrabarti, Martin van den Berg, and Byron Dom. Focused crawling: a new approach to topic-specific web resource discovery. *Computer Networks*, 31(11–16):1623 – 1640, 1999.
5. Brian D. Davison. Topical locality in the web. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '00, pages 272–279, New York, NY, USA, 2000. ACM.
6. Michelangelo Diligenti, Frans Coetzee, Steve Lawrence, C. Lee Giles, and Marco Gori. Focused crawling using context graphs. In *Proceedings of the 26th International Conference on Very Large Data Bases*, VLDB '00, pages 527–534, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.
7. George H. John and Pat Langley. Estimating continuous distributions in bayesian classifiers. In *Proceedings of the Eleventh conference on Uncertainty in artificial intelligence*, UAI'95, pages 338–345, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc.
8. Gautam Pant and Padmini Srinivasan. Learning to crawl: Comparing classification schemes. *ACM Trans. Inf. Syst.*, 23(4):430–462, October 2005.
9. Ingo Steinwart and Andreas Christmann. *Support Vector Machines*. Springer Publishing Company, Incorporated, 1st edition, 2008.