P-TAG: Large Scale Automatic Generation of Personalized Annotation TAGs for the Web

Paul - Alexandru Chirita[1]*, Stefania Costache[1], Siegfried Handschuh[2], Wolfgang Nejdl[1]

[1]L3S Research Center / University of Hannover, Appelstr. 9a, 30167 Hannover, Germany
{chirita,costache,nejdl}@l3s.de

[2]National University of Ireland / DERI, IDA Business Park, Lower Dangan, Galway, Ireland
Siegfried.Handschuh@deri.org

ABSTRACT

The success of the Semantic Web depends on the availability of Web pages annotated with metadata. Free form metadata or tags, as used in social bookmarking and folksonomies, have become more and more popular and successful. Such tags are relevant keywords associated with or assigned to a piece of information (e.g., a Web page), describing the item and enabling keyword-based classification. In this paper we propose P-TAG, a method which automatically generates personalized tags for Web pages. Upon browsing a Web page, P-TAG produces keywords relevant both to its textual content, but also to the data residing on the surfer's Desktop, thus expressing a personalized viewpoint. Empirical evaluations with several algorithms pursuing this approach showed very promising results. We are therefore very confident that such a user oriented automatic tagging approach can provide large scale personalized metadata annotations as an important step towards realizing the Semantic Web.

Categories and Subject Descriptors

H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing; H.3.5 [**Information Storage and Retrieval**]: Online Information Services

General Terms

Algorithms, Experimentation, Design

Keywords

Web Annotations, Tagging, Personalization, User Desktop

1. INTRODUCTION

The World Wide Web has had a tremendous impact on society and business in recent years by making information instantly and ubiquitously available. The Semantic Web is seen as an extension of the WWW, a vision of a future Web of machine-understandable documents and data. One its main instruments are the *annotations*, which enrich content with metadata in order to ease its automatic processing. The traditional paradigm of Semantic Web annotation (i.e., annotating Web sites with the help of external tools) has been established for a number of years by now, for example in the form of applications such as OntoMat [20] or tools based on Annotea [23], and the process continues to develop and improve. However, this paradigm is based on manual or semi-automatic annotation, which is a laborious, time consuming task, requiring a lot of expert know-how, and thus only applicable to small-scale or Intranet collections. For the overall Web though, the growth of a Semantic Web overlay is restricted because of the lack of annotated Web pages. In the same time, the tagging paradigm, which has its roots in social bookmarking and folksonomies, is becoming more and more popular. A tag is a relevant keyword associated with or assigned to a piece of information (e.g., a Web page), describing the item and enabling keyword-based classification of the information it is applied to. The successful application of the tagging paradigm can be seen as evidence that a lowercase semantic Web[1] could be easier to grasp for the millions of Web users and hence easier to introduce, exploit and benefit from. One can then build upon this lowercase semantic web as a basis for the introduction of more semantics, thus advancing further towards the Web 2.0 ideas.

We argue that a successful and easy achievable approach is to automatically generate annotation tags for Web pages in a scalable fashion. We use tags in their general sense, i.e., as a mechanism to indicate what a particular document is about [4], rather than for example to organize one's tasks (e.g., "todo"). Yet automatically generated tags have the drawback of presenting only a generic view, which does not necessary reflect personal interests. For example, a person might categorize the home page of Anthony Jameson[2] with the tags "human computer interaction" and "mobile computing", because this reflects her research interests, while another would annotate it with the project names "Halo 2" and "MeMo", because she is more interested in research applications.

The crucial question is then how to automatically tag Web pages in a personalized way. In many environments, defining a user's viewpoint would rely on the definition of an interest profile. However, these profiles are laborious to create and need constant maintenance in order to reflect the changing interest of the user. Fortunately, we do have a rich source of user profiling information available: everything stored on her computer. This *personal Desktop* usually contains a very rich document corpus of personal information which can and should be exploited for user personalization! There is no need to maintain a dedicated interest profile, since the

*Part of this work was performed while the author was visiting Yahoo! Research, Barcelona, Spain.

WWW 2007, May 8–12, 2007, Banff, Alberta, Canada.
ACM 978-1-59593-654-7/07/0005.

[1]Lowercase semantic web refers to an evolutionary approach for the Semantic Web by adding simple meaning gradually into the documents and thus lowering the barriers for re-using information.
[2]http://www.dfki.de/~jameson

Desktop as such reflects all the trends and new interests of a user, while it also tracks her history.

Based on this observation, we propose a novel approach for a scalable automatic generation of annotation tags for Web pages, personalized on each user's Desktop. We achieve this by aligning keyword candidates for a given Web page with keywords representing the personal Desktop documents and thus the subject's / author's personal interest, utilizing appropriate algorithms. The resulting personalized annotations can be added on the fly to any Web page browsed by the user.

The structure of the paper is as follows: In Section 2 we discuss previous and related work. Section 3 describes the core algorithmic approaches for personalized annotation of Web pages by exploiting Desktop documents. In Section 4 we present the setup and empirical results of our evaluation. Finally, prior to concluding, we briefly discuss possible applications of our approach in Section 5.

2. PREVIOUS WORK

This paper presents a novel approach to generate personalized annotation tags for Web pages by exploiting document similarity and keyword extraction algorithms. Though some blueprints do exist, to our knowledge there has been no prior explicit formulation of this approach, nor a concrete application or empirical evaluation, as presented in this paper. Nevertheless, a substantial amount of related work already exists concerning the general goal of creating annotations for Web pages, as well as keyword extraction. The following sections will discuss some of the most important works in the research areas of annotation, text mining for keyword extraction, and keyword association.

2.1 Generating Annotations for the Web

Brooks and Montanez [4] analyzed the effectiveness of tags for classifying blog entries and found that manual tags are less effective content descriptors than automated ones. We see this as a support for our work, especially since our evaluation from Section 4 proves that the tags we create do result in high precision for content description. They further showed that clustering algorithms can be used to construct a topical hierarchy amongst tags. We believe this finding could be a useful extension to our approach.

Cimiano et. al. [10] proposed PANKOW (Pattern-based Annotation through Knowledge on the Web), a method which employs an unsupervised, pattern-oriented approach to categorize an instance with respect to a given ontology. Similar to P-TAG, the system is rather simple, effortless and intuitive to use for Web page annotation. However, it requires an input ontology and outputs instances of the ontological concepts, whereas we simply annotate Web pages with user specific tags. Also, PANKOW exploits the Web by means of a generic statistical analysis, and thus their annotations reflect more common knowledge without considering context or personal preferences, whereas we create personalized tags. Finally, the major drawback of PANKOW is that it does not scale, since it produces an extremely large number of queries against one target Web search engine in order to collect its necessary data.

The work in [11] presents an enhanced version of PANKOW, namely C-PANKOW. The application downloads document abstracts and processes them off-line, thus overcoming several shortcomings of PANKOW. It also introduces the notion of context, based on the similarity between the document to be annotated and each of the downloaded abstracts. However, it is reported that annotating one page can take up to 20 minutes with C-PANKOW. Our system annotates Web pages on the fly in seconds. Note that the tasks are not entirely comparable though, since our system does not produce ontology-based annotations, but personalized annotation tags. Further, our notion of context is much stronger, since we consider documents from the personal Desktop, which leads to highly personalized annotations. Finally, C-PANKOW uses the proper nouns of each Web page for annotation candidates, and thus annotation is always directly rooted on the text of the Web page. On the other hand, the algorithms we propose in this paper generate keywords that not necessarily appear literally on the Web page, but are in its context, while also reflecting the personal interests of the user.

Dill et. al. [14] present a platform for large-scale text analytics and automatic semantic tagging. The system spots known terms in a Web page and relates it to existing instances of a given ontology. The strength of the system is in the taxonomy based disambiguation algorithm. In contrast, our system does not rely on such a handcrafted lexicon and extracts new keywords in a fully automatic fashion, while also supporting personalized annotations.

2.2 Text Mining for Keywords Extraction

Text data mining is one of the main technologies for discovering new facts and trends about the currently existing large text collections [21]. There exist quite a diverse number of approaches for extracting keywords from textual documents. In this section we review some of those techniques originating from the Semantic Web, Information Retrieval and Natural Language Processing environments, as they are closest to the algorithms described in this paper. In Information Retrieval, most of these techniques were used for Relevance Feedback [29], a process in which the user query submitted to a search engine is expanded with additional keywords extracted from a set of relevant documents [32]. Some comprehensive comparisons and literature reviews of this area can be found in [17, 30]. Efthimiadis [17] for example proposed several simple methods to extract keywords based on term frequency, document frequency, etc. We used some of these as inspiration for our Desktop specific annotation process. Chang and Hsu [7] first applied a clustering algorithm over the input collection of documents, and then attempted to extract keywords as cluster digests. We moved this one step further, by investigating the possibilities to acquire such keywords using Latent Semantic Analysis [13], which results in more qualitative clusterings over textual collections. However, this turned out to require too many computational resources for the already large Desktop data sets.

The more precise the extracted information is, the closer we move to applying NLP algorithms. Lam and Jones [26] for example used summarization techniques to extract informative sentences from documents. Within the Semantic Web / Information Extraction area, we distinguish the advances achieved within the GATE system [12, 27], which allows not only for NLP based entity recognition, but also for identifying relations between such entities. Its functionalities are exploited by quite several semantic annotation systems, either generally focused on extracting semantics from Web pages (as for example in KIM [25]), or more guided by a specific purpose underlying ontology (as in Artequakt [1]).

2.3 Text Mining for Keywords Association

While not directly related to the actual generation of semantic entities, keyword association is useful for enriching already discovered annotations, for example with additional terms that describe them in more detail. Two generic techniques have been found useful for this purpose. First, such terms could be identified utilizing co-occurrence statistics over the entire document collection to annotate [24]. In fact, as this approach has been shown to yield good results, many subsequent metrics have been developed to best assess "term relationship" levels, either by narrowing the analysis for

only short windows of text [19], or broadening it towards topical clusters [31], etc. We have also investigated three of these techniques in order to identify new keywords related to a set of terms that have been already extracted from the Web page which requires annotation. Second, more limited, yet also much more precise term relationships can be obtained from manually created ontologies [6], or thesauri, such as WordNet [28].

3. AUTOMATIC PERSONALIZED WEB ANNOTATIONS

We distinguish three broad approaches to generate personalized Web page annotations, based on the way user profiles (i.e., Desktops in our case) are exploited to achieve personalization: (1) a document oriented approach, (2) a keyword oriented approach, and (3) a hybrid approach.

3.1 Document Oriented Extraction

The general idea of the document oriented approach is as follows. For a given Web page, the system retrieves similar documents from the personal Desktop. This set of related personal documents reflects user's viewpoint regarding the content of the Web page. Hence, the set will be used to extract the relevant annotation tags for the input Web page. The generic technique is also depicted in the algorithm below.

Algorithm 3.1.1. Document oriented Web annotations generation.

1: Given a browsed Web page p,
2: **Find** similar Desktop documents $DS_i, i \in \{1, \ldots, nd\}$
3: **For** each document DS_i:
4: **Extract** its relevant keywords.
5: **Select** Top-K keywords using their confidence scores.

We will now detail the specific algorithms for accomplishing steps 2 and 4 in the following two subsections.

3.1.1 Finding Similar Documents

We consider two approaches for this task, namely Cosine Similarity (i.e., nearest neighbor based search), and Latent Semantic Analysis (i.e., clustering based search). In both cases, we apply the algorithms only to those terms with a Desktop document frequency above 10 and below 20% from all Desktop documents of the current user. This is necessary in order to avoid noisy results (i.e., documents with many words in common with an input page p, yet most of these being either very general terms, or terms not describing the interests of the surfer). Moreover, such optimizations significantly improve computation time.

Cosine Similarity. Nearest neighbor search, based on TFxIDF (Term Frequency multiplied by Inverse Document Frequency), outputs a similarity score between two documents utilizing the following simple formula:

$$Sim(D_i, D_j) = v(D_i) \cdot v(D_j) = \sum_{t \in D_i \cup D_j} w_{i,t} \cdot w_{j,t} \qquad (1)$$

where $v(D_i), v(D_j)$ are the term vectors describing documents D_i, D_j using TFxIDF, as within the Vector Space Model [3], and $w_{k,t}$ represents the actual TFxIDF weight associated to term t within document D_k.

Latent Semantic Analysis. Similar to clustering, LSA can compute document to document similarity scores. However, we decided to abandon experimenting with this algorithm, as it turned out to be too computationally expensive for regular Desktops consisting of several dozens of thousands of indexable items. Nevertheless, since the quality of its results might be high, we are currently investigating several techniques to optimize it or to approximate its results in order to enable evaluating it in a further work.

3.1.2 Extracting Keywords from Documents

Introduction. Keyword extraction algorithms usually take a text document as input and then return a list of keywords, which have been identified by exploiting various text mining approaches. To ease further processing, each keyword has associated a value representing the confidence with which it is thought to be relevant for the input document. Once such keywords have been identified for the input set of relevant Desktop documents (i.e., as determined using the previously described techniques), we propose to generate annotations for the original input Web page by sorting all these generated terms utilizing their confidence levels, and then taking the Top-K of them.

We investigated three broad approaches to keyword extraction with increasing levels of granularities, denoted as "Term and Document Frequency" (keyword oriented), "Lexical Compounds" (expression oriented) and "Sentence Selection" (summary oriented).

Term and Document Frequency. As the simplest possible measures, TF and DF have the advantage of being very fast to compute. Moreover, previous experiments showed them to yield very good keyword identification results [5, 17]. We thus associate a score with each term, based on the two statistics (independently). The TF based one is obtained by multiplying the actual frequency of a term with a position score descending as the term first appears more towards the end of the document. This is necessary especially for longer documents, because more informative terms tend to appear towards their beginning [5]. The complete TF based keyword extraction formula is as follows:

$$TermScore = \left[\frac{1}{2} + \frac{1}{2} \cdot \frac{nrWords - pos}{nrWords}\right] \cdot TF \qquad (2)$$

where $nrWords$ is the total number of terms in the document, pos is the position of the first appearance of the term, and TF is the frequency of each term in the considered Desktop document.

The identification of suitable expansion terms is even simpler when using DF: The terms from the set of Top-K relevant Desktop documents are ordered according to their DF scores. Any ties are resolved using the above mentioned TF scores [17].

Note that a hybrid TFxIDF approach is not necessarily efficient, since one Desktop term might have a high DF on the Desktop, yet it may occur rarely on the Web. For example, the term "RDF" could occur frequently accross the Desktop of a Semantic Web scientist, thus achieving a low score with TFxIDF. However, as it is encountered more rarely on the Web, it would make a better annotation than some generic keyword.

Lexical Compounds. Anick and Tipirneni [2] defined the *lexical dispersion hypothesis*, according to which an expression's lexical dispersion (i.e., the number of different compounds it appears in within a document or group of documents) can be used to automatically identify key concepts over the input document set. Although there exist quite several possible compound expressions, it has been shown that simple approaches based on noun analysis produce results almost as good as highly complex part-of-speech pattern identification algorithms. We thus inspect the selected Desktop documents (i.e., at step 1 of the generic algorithm) for all their lexical compounds of the following form:

$$\{\ adjective?\ noun+\ \}$$

We note that all such compounds could be easily generated off-line, at Desktop indexing time, for all the documents in the local repository. Moreover, once identified, they could be further sorted depending on their dispersion within each document in order to facilitate fast retrieval of the most frequent compounds at run-time.

Sentence Selection. This technique builds upon sentence oriented document summarization: Having as input the set of Desktop documents highly similar to the input Web page, a summary containing their most important sentences is generated as output. Thus, sentence selection is the most comprehensive of these approaches, as it produces the most detailed annotations (i.e., sentences). Its downside is that, unlike the first two algorithms, its output cannot be stored efficiently, and thus it cannot be precomputed off-line. We generate sentence based summaries by ranking the document sentences according to their salience score, as follows [26]:

$$SentenceScore = \frac{SW^2}{TW} + PS \ \ [+\frac{TQ^2}{NQ}]$$

The first term is the ratio between the square amount of significant words within the sentence and the total number of words therein. A word is significant in a document if its frequency is above a threshold as follows:

$$TF > ms = \begin{cases} 7 - 0.1 * [25 - NS] & , if\ NS < 25 \\ 7 & , if\ NS \in [25, 40] \\ 7 + 0.1 * [NS - 40] & , if\ NS > 40 \end{cases}$$

with NS being the total number of sentences in the document (see [26] for more details). The second term is a position score. We set it to $1/NS$ for the first ten sentences, and to 0 otherwise. This way, short documents such as emails are not affected, which is correct, since they usually do not contain a summary in the very beginning. However, it is known that longer documents usually do include overall descriptive sentences in the beginning [16], and these sentences are thus more likely to be relevant. The final term is an optional parameter which is not used for this method, but only for the hybrid extraction approaches (see Section 3.3). It biases the summary towards some given set of terms, which in our case will correspond to the terms extracted from the input Web page. More specifically, it represents the ratio between the square number of set terms present in the sentence and the total number of terms from the set. It is based on the belief that the more terms in the set are contained in a sentence, the more likely will that sentence convey information highly related to the set, and consequently to the Web page to annotate.

3.2 Keyword Oriented Extraction

In the keyword oriented approach we start from extracting some keywords from the Web page to annotate. This set of keywords reflects a generic viewpoint on the document. In order to personalize this viewpoint, we then align these keywords with related terms from the personal Desktop, as in the algorithm presented below.

Algorithm 3.2.1. Keyword oriented Web annotations generation.

1: Given a browsed Web page p,
2: **Extract** relevant keywords from p
3: **For** each relevant keyword
4: **Find** related keywords on the local Desktop.
5: **Select** Top-K keywords using their confidence scores.

Step 1 can be accomplished using the algorithms from Section 3.1.2. The next two subsections will introduce the techniques we propose for the keyword alignment process, i.e., the finding of similar keywords on the Desktop corpus, namely (1) term co-occurrence statistics and (2) thesaurus based extraction.

Term Co-occurrence Statistics. For each term, one could easily compute off-line those terms co-occurring with it most frequently in a collection, such as user's Desktop, and then exploit this information at run-time in order to infer keywords highly correlated with the content of a given Web page. Our generic Desktop level co-occurrence based keyword similarity search algorithm is as follows:

Algorithm 3.2.2. Co-occurrence based keyword similarity search.

Off-line computation:
1: **Filter** potential keywords k with $DF \in [10, \ldots, 20\% \cdot N]$
2: **For** each keyword k_i
3: **For** each keyword k_j
4: Compute SC_{k_i,k_j}, the similarity coefficient of (k_i, k_j)

On-line computation:
1: **Let** S be the set of terms potentially similar to E,
the set of keywords extracted from the input Web page.
2: **For** each keyword k of E:
3: $S \leftarrow S \cup TSC(k)$, where $TSC(k)$ contains the Top-K terms most similar to k
4: **For** each term t of S:
5a: **Let** $Score(t) \leftarrow \prod_{k \in E}(0.01 + SC_{t,k})$
5b: **Let** $Score(t) \leftarrow \#Hits(E|t)$
4: **Select** Top-K terms of S with the highest scores.

The off-line computation needs an initial trimming phase (step 1) for optimization purposes. Also, once the co-occurrence levels are in place and the terms most correlated with the keywords extracted from the input Web page have been identified, one more operation is necessary, namely calculating the correlation of each proposed term with the entire set of extracted keywords (i.e., with E), rather than with each keyword individually. Two approaches are possible: (1) using a product of the correlation between the term and all keywords in E (step 5a), or (2) simply counting the number of documents in which the proposed term co-occurs with the entire set of extracted keywords (step 5b). Small scale tuning experiments performed before the actual empirical analysis indicated the latter approach to yield a slightly better outcome. Finally, we considered the following Similarity Coefficients [24]:

- *Cosine Similarity*, defined as:

$$CS = \frac{DF_{x,y}}{\sqrt{DF_x \cdot DF_y}} \quad (3)$$

- *Mutual Information*, defined as:

$$MI = log\frac{N \cdot DF_{x,y}}{DF_x \cdot DF_y} \quad (4)$$

- *Likelihood Ratio*, defined below.

DF_x is the Document Frequency of term x, and $DF_{x,y}$ is the number of documents containing both x and y.

Dunning's Likelihood Ratio λ [15] is a co-occurrence based metric similar to χ^2. It starts from attempting to reject the null hypothesis, according to which two terms A and B would appear in

text independently from each other. This means that $P(A\ B) = P(A\neg B) = P(A)$, where $P(A\neg B)$ is the probability that term A is *not* followed by term B. Consequently, the test for independence of A and B can be performed by looking if the distribution of A given that B is present is the same as the distribution of A given that B is not present. Of course, in reality we know these terms are not independent in text, and we only use the statistical metrics to highlight terms which are frequently appearing together. We thus compare the two binomial processes by using likelihood ratios of their associated hypotheses, as follows:

$$\lambda = \frac{max_{\omega\in\Omega_0}\ H(\omega;k)}{max_{\omega\in\Omega}\ H(\omega;k)} \quad (5)$$

where ω is a point in the parameter space Ω, Ω_0 is the particular hypothesis being tested, and k is a point in the space of observations K. More details can be found in [15].

Thesaurus Based Extraction. Large scale thesauri encapsulate global knowledge about term relationships. Thus, after having extracted the set of keywords describing the input Web page (i.e., as with Step 1 of Algorithm 3.2.1), we generate an additional set containing all their related terms, as identified using an external thesauri. Then, to achieve personalization, we trim this latter set by keeping only those terms that are frequently co-occurring over user's Desktop with the initially identified keywords. The algorithm is as follows:

Algorithm 3.2.3. Thesaurus based keyword similarity search.

1: **For** each keyword k of a set P, describing Web page p:
2: **Select** the following sets of related terms using WordNet:
2a: Syn: All Synonyms
2b: Sub: All sub-concepts residing one level below k
2c: Super: All super-concepts residing one level above k
3: **For** each set S_i of the above mentioned sets:
4: **For** each term t of S_i:
5: **Search** the Desktop with $(P|t)$, i.e., the original set, as expanded with t
6: **Let** H be the number of hits of the above search (i.e., the co-occurrence level of t with P)
7: **Return** Top-K terms as ordered by their H values.

We observe three types of term relationships (steps 2a-2c): (1) synonyms, (2) sub-concepts, namely hyponyms (i.e., sub-classes) and meronyms (i.e., sub-parts), and (3) super-concepts, namely hypernyms (i.e., super-classes) and holonyms (i.e., super-parts). As they represent quite different types of association, we investigated them separately. Further, we limited the output expansion set (step 7) to contain only terms appearing at least T times on the Desktop, in order to avoid any noisy annotations, with $T = \min(\frac{N}{\text{DocsPerTopic}}, \text{MinDocs})$. We set $DocsPerTopic = 2,500$, and $MinDocs = 5$, the latter one coping with small input Desktops.

3.3 Hybrid Extraction

The hybrid approach mixes the document oriented approach with the keyword oriented one. The system first extracts some relevant keywords from an input Web page. This set of keywords is the same as in Step 1 of Algorithm 3.2.1 and reflects a generic viewpoint on the document. We personalize this viewpoint by retrieving the Desktop documents most relevant to it (i.e., using Lucene Desktop search) and then by extracting relevant keywords from them. The generic algorithm is as follows:

Algorithm 3.3.1. Hybrid generation of Web annotations.

1: Given a browsed Web page p,
2: **Extract** the relevant keywords P_i from p
3: **For** each keyword P_i:
4: **Find** the most relevant Desktop documents $D_{i,j}$.
5: **For** each document $D_{i,j}$:
6: **Extract** its relevant keywords.
7: **Select** Top-K keywords using their confidence levels.

Steps 2 and 6 utilize the keyword extraction techniques described in Section 3.1.2. Step 4 represents a regular Desktop search, in which the results are ordered by their relevance to the query, i.e., to each keyword P_i.

4. EXPERIMENTS

4.1 Experimental Setup

System Setup. We have asked 16 users (PhD and Post-Doc students in various areas of computer science and education) to support us with the experiments. Our Lucene[3] based system was running on each personal Desktop. Lucene suited our interests best, given its rapid search algorithms, its flexibility and adaptivity, and last but not least its cross-platform portability. Thus, each user's corpus of personal documents has been indexed for later faster retrieval. More specifically, we indexed all Emails, Web Cache documents, and all Files within user selected paths. The Web pages to be annotated have been selected randomly from each user's cache. To ensure that the user did not artificially collect a lot of data on the topic of the selected pages *after* having browsed them, we only picked pages browsed within the last week. For the annotation, we chose two input pages per each category: small (below 4 KB), medium (between 4 KB and 32 KB), and large (more than 32 KB) [18]. In total, 96 pages were used as input over the entire experiment, and over 2,000 resulted annotations were graded.

Algorithms. We applied our three approaches for keyword extraction and annotation to the input pages, i.e., Document oriented, Keyword oriented and Hybrid. In all cases, in order to optimize the run-time computation speed, we chose to limit the number of output keywords extracted per Desktop document to the maximum number of annotations desired (i.e., four).

We applied the *Document oriented approach* with the following algorithms:

1. **TF, DF**: Term and Document Frequency;
2. **LC**: Lexical Compounds;
3. **SS**: Sentence Selection.

Second, we investigated the automatic annotation by applying the *Keyword oriented approach*. For step one of our generic approach (see Algorithm 3.2.1) we used the keyword extraction algorithms TF, DF, LC and SS. For step two we investigated the following algorithms for finding similar keywords:

1. **TC[CS], TC[MI], TC[LR]**: Term Co-occurrence Statistics using respectively Cosine Similarity, Mutual Information, and Likelihood Ratio as similarity coefficients;
2. **WN[SYN], WN[SUB], WN[SUP]**: WordNet based analysis with synonyms, sub-concepts, and super-concepts.

All in all, this gives us 24 possible combinations, i.e., TF+TC[CS], TF+TC[MI], . . . , SS+WN[Sub], and SS+WN[Sup].

Finally, we conducted the study with the *Hybrid approach*. We used the four keyword extraction algorithms for both the input Web

[3]http://lucene.apache.org

Algorithm	P@1	P@2	P@3	P@4
TF	0.78	0.86	0.85	0.81
DF	0.83	0.76	0.76	0.74
LC	0.78	0.76	0.81	0.82
SS	0.67	0.75	0.75	0.73

Table 1: P@1-4 for document oriented extraction for small Web pages.

page and the identified Desktop documents. These gave 16 combinations, i.e., TF+TF (applying TF for the Web page and TF for the Desktop documents), TF+DF, . . . , SS+LC and SS+SS.

Rating the Personal Annotations. For each input page, the annotations produced by all algorithms were shuffled and blinded such that the user was not aware of either the algorithm which produced them, or of their ranking within the list of generated results. Each proposed keyword was rated 0 (not relevant) or 1 (relevant). It did not matter *how much* relevant each proposed annotation was, as it would have been a valid result anyway (i.e., if it has at least a reasonable degree of relevance to the input Web page, then it can be returned as annotation). It can be easily shown that those annotations closer to the actual content of the input Web document are generated more often.

We measured the quality of the produced annotations using Precision, a standard Information Retrieval evaluation measure. As our algorithms also generated a confidence score for each proposed annotation, we were able to order these suggestions and calculate the precision at different levels, namely P@1, P@2, P@3 and P@4. The precision at level K (P@K) is the precision score when only considering the Top-K output[4]. It represents the amount of relevant annotations proposed within the Top-K suggestions divided by K, i.e., the total number of annotations considered. Four different levels of K were analyzed, as it was not clear which is the best amount of annotations to generate using our approaches.

First, the P@K scores were computed for each user, algorithm, and Web page in particular. We averaged these values over the Web pages of the same type, obtaining user's opinion on the tuple <algorithm, type of Web page>. We further averaged over all subjects, and the resulting values are listed in the tables to come.

When doing the experiments, the users were specifically asked to rate the generated annotations taking into account both the target Web page and their personal interests. The idea was to generate for each subject not only generic keywords extracted from the current Web page, but others that connect it to her Desktop, and therefore her interests. Hence, the experiment is designed to evaluate our two main research questions: First, how well can automatic and personal annotation of Web pages be conducted with our approach? Second, which algorithm is producing the best results?

4.2 Results

We will split our discussion of the experimental results in two parts, first analyzing the output of each approach in particular, and then comparing them to find the best method for generating personalized annotations. We compare our algorithms according to their P@3 scores (i.e., the average value over the ratings given to the Top-3 annotations, as ordered using their confidence levels), as most of them did not always produce more than 3 annotations above the strict minimal confidence thresholds we set.

[4]In order to produce only highly qualitative results, we set some minimal thresholds for the confidence scores of each algorithm. Whenever they were not reached, the algorithm simply generated K' < K annotations for that input page.

Algorithm	P@1	P@2	P@3	P@4
TF	0.76	0.76	0.73	0.71
DF	0.59	0.51	0.55	0.53
LC	0.76	0.73	0.70	0.71
SS	0.76	0.67	0.68	0.67

Table 2: P@1-4 for document oriented extraction for medium Web pages.

Algorithm	P@1	P@2	P@3	P@4
TF	0.76	0.80	0.80	0.80
DF	0.63	0.63	0.60	0.54
LC	0.83	0.74	0.72	0.75
SS	0.82	0.74	0.71	0.69

Table 3: P@1-4 for document oriented extraction for large Web pages.

Document Oriented Extraction. The investigation of our document based approaches provided us with an interesting insight: The Term Frequency metric performs best overall, thus generating better annotations than more informed techniques such as Lexical Compounds or Sentence Selection. After looking at each result in particular, we found that TF produces constantly good or very good output, whereas LC for example was much more unstable, yielding for some Web pages excellent annotations, but for some others rather poor ones. Nevertheless, both LC and SS had very good ratings, even the best ones for those subjects with dense Desktops. Thus, for future work one might consider designing an adaptive personalized annotation framework, which automatically selects the best algorithm as a function of different parameters, such as the amount of resources indexed within the personal collection. DF was only mediocre (except for the case of small input Web pages), which is explainable, as its extractions are based on generic entire Desktop statistics, thus diverging too much from the context of the annotated Web page.

Averaging over all types of input files, the best precisions were 0.79 when considering only the first output annotation, 0.81 when the top two results were analyzed, 0.79 for the top three, and 0.77 for all four. As LC yielded the first value, and TF the latter three ones, we conclude that LC is the method of choice if only one annotation is desired. However, its quality degrades fast, and TF should be used when looking for more results. All document oriented results are summarized in Tables 1, 2, and 3, for the small, medium, and large pages respectively.

Keyword Oriented Extraction. We will split the analysis of the keyword based algorithms based on their two steps. For extracting keywords from the single Web page used as input, all methods perform rather similarly for small pages, with TF performing slightly better. Again, we notice that the small input is the easiest one for our algorithms. More interesting, Document Frequency yields the best basis for keyword similarity search when used with medium and large input pages. It thus selects best those terms from the input Web page which are most representative to the user. This is correct, as it is the only algorithm that relates the extraction also to the Desktop content by using DF values from user's personal collection.

In the case of keyword similarity search, the results are very clear. WordNet based techniques seem to perform rather poorly, indicating external thesauri are not a good choice for annotations, as they cover a too general content. Note that one might obtain a better outcome when targeting this scenario to application specific thesauri and input Web pages. On the other hand, all co-occurrence based algorithms produce very good results, with TC[CS] and TC[MI] being slightly better.

Algorithm	P@1	P@2	P@3	P@4
TF+TC[CS]	**1.00**	**0.96**	**0.89**	**-**
TF+TC[MI]	0.83	0.83	0.89	-
TF+TC[LR]	0.75	0.88	0.89	-
TF+WN[Syn]	0.50	0.50	0.67	-
TF+WN[Sub]	0.41	0.41	0.37	0.18
TF+WN[Sup]	0.38	0.40	0.35	-
DF+TC[CS]	0.92	0.88	0.83	-
DF+TC[MI]	**0.92**	**0.88**	**0.86**	**-**
DF+TC[LR]	0.67	0.67	0.67	-
DF+WN[Syn]	-	-	-	-
DF+WN[Sub]	0.39	0.42	0.46	0.08
DF+WN[Sup]	0.67	0.53	0.48	0.20
LC+TC[CS]	**0.92**	**0.92**	**0.81**	**-**
LC+TC[MI]	0.92	0.88	0.81	-
LC+TC[LR]	0.65	0.70	0.74	0.65
LC+WN[Syn]	0.55	0.38	0.35	-
LC+WN[Sub]	0.26	0.34	0.36	0.24
LC+WN[Sup]	0.66	0.48	0.49	-
SS+TC[CS]	1.00	1.00	0.87	-
SS+TC[MI]	**0.83**	**0.83**	**0.89**	**-**
SS+TC[LR]	0.66	0.77	0.76	-
SS+WN[Syn]	0.43	0.43	0.43	-
SS+WN[Sub]	0.54	0.27	0.33	0.17
SS+WN[Sup]	0.53	0.47	0.37	-

Table 4: P@1-4 for kewyord oriented extraction for small Web pages.

Algorithm	P@1	P@2	P@3	P@4
TF+TC[CS]	0.64	0.64	0.64	-
TF+TC[MI]	**0.68**	**0.66**	**0.64**	**-**
TF+TC[LR]	0.70	0.66	0.63	-
TF+WN[Syn]	0.33	0.30	0.27	-
TF+WN[Sub]	0.41	0.39	0.36	0.08
TF+WN[Sup]	0.40	0.33	0.26	0.09
DF+TC[CS]	0.71	0.79	0.69	-
DF+TC[MI]	**0.71**	**0.75**	**0.75**	**-**
DF+TC[LR]	0.54	0.51	0.48	-
DF+WN[Syn]	0.15	-	-	-
DF+WN[Sub]	0.27	0.29	0.32	0.14
DF+WN[Sup]	0.38	0.30	0.27	0.10
LC+TC[CS]	0.60	0.63	0.57	-
LC+TC[MI]	**0.61**	**0.60**	**0.60**	**-**
LC+TC[LR]	0.59	0.61	0.58	0.35
LC+WN[Syn]	0.40	0.22	0.07	-
LC+WN[Sub]	0.31	0.41	0.36	0.09
LC+WN[Sup]	0.53	0.53	0.48	-
SS+TC[CS]	0.58	0.56	0.59	-
SS+TC[MI]	**0.62**	**0.60**	**0.60**	**-**
SS+TC[LR]	0.54	0.61	0.58	-
SS+WN[Syn]	0.23	0.13	0.09	-
SS+WN[Sub]	0.38	0.39	0.34	0.09
SS+WN[Sup]	0.33	0.30	0.18	-

Table 5: P@1-4 for kewyord oriented extraction for medium Web pages.

The best average ratings overall were 0.81 at the first annotation (DF+TC[CS] and DF+TC[MI]), 0.82 at the second one (DF+TC[CS]) and finally 0.80, when considering only the top third output (DF+TC[MI]). No method managed to generally produce more than three highly qualitative annotations. All keyword oriented results are summarized in Tables 4, 5, and 6, for the small, medium, and large pages respectively. In order to ease the inspection of results, for each keyword extraction algorithm (i.e., first step), the best method with respect to P@3 is in bold.

Hybrid Extraction. Our hybrid algorithms are composed of two phases, first a single page keyword extraction, and then another one, at the Desktop level and over those documents matching the best formerly extracted keywords (i.e., as obtained by searching the local Desktop with Lucene). All methods performed rather similar, with small differences between each other. An inspection of the actual data showed that again DF performed particularly well at the first extraction step. However, this single word output was not always discriminative enough for a regular Desktop search, i.e., for finding Desktop documents highly similar to the input Web page. In fact, this is true for all keyword extraction techniques, and this is why the keyword oriented methods managed to surpass the hybrid ones in the quality of their output. For the second step, TF performed visibly better with medium and large pages, and all methods were close to each other for small input data. This is in accordance with the document oriented approaches, which use a similar technique.

SS+TF is the best overall algorithm, with a precision of 0.80 at only the first result, and of 0.74 at the top two ones. Then, TF+TF performs best, yielding a score of 0.73 at the top three annotations, and 0.74 when all results are included in the analysis. All results are also depicted in Tables 7, 8, and 9, for the small, medium, and large pages respectively. For each keyword extraction algorithm (i.e., first step), the best method is in bold.

Comparison. We now turn our attention to finding global conclusions over all our proposed algorithms. In order to better pursue this analysis, we depict the three best performing algorithms of each category (i.e., document oriented, keyword oriented, and hybrid) in Figure 1.

Algorithm	P@1	P@2	P@3	P@4
TF+TC[CS]	0.62	0.66	0.62	-
TF+TC[MI]	0.66	0.67	0.61	-
TF+TC[LR]	**0.65**	**0.68**	**0.64**	**-**
TF+WN[Syn]	0.25	0.08	0.05	-
TF+WN[Sub]	0.44	0.36	0.32	0.15
TF+WN[Sup]	0.44	0.46	0.33	-
DF+TC[CS]	**0.80**	**0.80**	**0.79**	**-**
DF+TC[MI]	0.80	0.77	0.79	-
DF+TC[LR]	0.85	0.71	0.70	-
DF+WN[Syn]	-	-	-	-
DF+WN[Sub]	0.44	0.38	0.37	0.02
DF+WN[Sup]	0.46	0.30	0.28	0.08
LC+TC[CS]	**0.57**	**0.61**	**0.58**	**-**
LC+TC[MI]	0.61	0.59	0.56	-
LC+TC[LR]	0.63	0.64	0.52	0.35
LC+WN[Syn]	0.31	0.19	0.16	-
LC+WN[Sub]	0.33	0.37	0.32	0.10
LC+WN[Sup]	0.53	0.43	0.39	0.06
SS+TC[CS]	**0.64**	**0.60**	**0.62**	**-**
SS+TC[MI]	0.60	0.60	0.60	-
SS+TC[LR]	0.64	0.63	0.61	-
SS+WN[Syn]	0.27	0.14	0.09	-
SS+WN[Sub]	0.51	0.40	0.38	0.14
SS+WN[Sup]	0.34	0.38	0.25	-

Table 6: P@1-4 for kewyord oriented extraction for large Web pages.

For small Web pages (the leftmost bar), the keyword oriented approaches are by far the best ones, being placed on positions one, two and four. They are followed by the document oriented ones, and finally by the hybrid methods, all global differences being very clear. In the case of medium sized pages, the proposed algorithms are harder to separate, performing similarly when analyzed over their best three representatives. Interesting here is the strong drop of TF+TC[LR], indicating that term frequency is good at single document keyword extraction only with small Web pages. Finally, we observe that the document oriented approaches are the best with large pages, which is reasonable, as they have the most amount of information available when searching the local Desktop for documents similar to the input Web page. They are followed by the keyword oriented techniques, and then by the hybrid ones.

Algorithm	P@1	P@2	P@3	P@4
TF+TF	**0.86**	**0.76**	**0.79**	**0.81**
TF+DF	0.92	0.71	0.69	0.69
TF+LC	0.75	0.79	0.78	0.73
TF+SS	0.92	0.79	0.75	0.76
DF+TF	0.83	0.88	0.83	0.77
DF+DF	0.92	0.71	0.69	0.67
DF+LC	0.67	0.67	0.64	0.63
DF+SS	**0.92**	**0.88**	**0.83**	**0.79**
LC+TF	0.86	0.74	0.74	0.76
LC+DF	0.92	0.71	0.69	0.63
LC+LC	**0.75**	**0.75**	**0.78**	**0.75**
LC+SS	0.86	0.72	0.73	0.74
SS+TF	0.89	0.78	0.77	0.79
SS+DF	1.00	0.75	0.72	0.65
SS+LC	**0.92**	**0.92**	**0.89**	**0.83**
SS+SS	1.00	0.79	0.78	0.72

Table 7: P@1-4 for hybrid extraction for small Web pages.

Algorithm	P@1	P@2	P@3	P@4
TF+TF	**0.71**	**0.76**	**0.74**	**0.74**
TF+DF	0.62	0.58	0.58	0.56
TF+LC	0.63	0.65	0.60	0.61
TF+SS	0.69	0.68	0.66	0.59
DF+TF	**0.70**	**0.66**	**0.59**	**0.59**
DF+DF	0.59	0.60	0.56	0.56
DF+LC	0.45	0.42	0.43	0.44
DF+SS	0.62	0.59	0.57	0.53
LC+TF	**0.73**	**0.78**	**0.74**	**0.71**
LC+DF	0.59	0.58	0.57	0.56
LC+LC	0.67	0.65	0.61	0.61
LC+SS	0.77	0.70	0.69	0.64
SS+TF	**0.80**	**0.76**	**0.70**	**0.68**
SS+DF	0.59	0.58	0.57	0.55
SS+LC	0.56	0.59	0.57	0.54
SS+SS	0.71	0.66	0.64	0.62

Table 8: P@1-4 for hybrid extraction for medium Web pages.

Figure 1 shows the quality of the output as a function of the input page size. As small Web pages contain few terms, they yield a clearer output, either when searching for related documents (as with document oriented techniques), or when extracting their keywords (as in the keyword oriented approaches), etc. As the content size increases, more noise appears, and processing becomes more difficult. Yet when Web pages have reached a reasonably large size, a number of informative terms tend to stand out, thus easing their processing to some extent.

We also averaged the results of all algorithms over all evaluated pages (in the rightmost column). We observed an obvious "ranking" across our generic approaches: (1) Keyword based algorithms (especially DF+TC[MI]), (2) Document based techniques, and (3) Hybrid ones. Though one would probably expect the latter ones to be the best, they suffered from the fact that the extracted keywords were insufficient to enable the retrieval of highly similar Desktop documents. On the contrary, the keyword based algorithms offer the optimal balance between the content of the input Web page and the personal files, producing a good selection of keywords which are both contained, as well as missing from the input page.

Finally, in Table 10 we give several examples for the output produced by the best two algorithms per generic approach. Let us inspect the first one in more detail. While some of the generated annotations can also be identified with previous work methods (e.g., "search", as it has a high term frequency in the input URL), many others would have not been located by them either because they are not named entities and have a low frequency in the input page (e.g., "schema" or "proximity search"), or simply because they are *not* contained in the starting URL (e.g., "retrieval" or "malleable"

Algorithm	P@1	P@2	P@3	P@4
TF+TF	**0.67**	**0.68**	**0.68**	**0.67**
TF+DF	0.67	0.59	0.57	0.56
TF+LC	0.55	0.51	0.47	0.46
TF+SS	0.66	0.65	0.60	0.60
DF+TF	**0.52**	**0.62**	**0.66**	**0.62**
DF+DF	0.67	0.59	0.57	0.54
DF+LC	0.62	0.53	0.47	0.49
DF+SS	0.54	0.50	0.53	0.49
LC+TF	**0.58**	**0.63**	**0.63**	**0.63**
LC+DF	0.67	0.59	0.56	0.57
LC+LC	0.48	0.51	0.46	0.47
LC+SS	0.63	0.63	0.58	0.56
SS+TF	**0.72**	**0.69**	**0.63**	**0.64**
SS+DF	0.67	0.59	0.57	0.55
SS+LC	0.61	0.54	0.52	0.51
SS+SS	0.68	0.62	0.61	0.59

Table 9: P@1-4 for hybrid extraction for large Web pages.

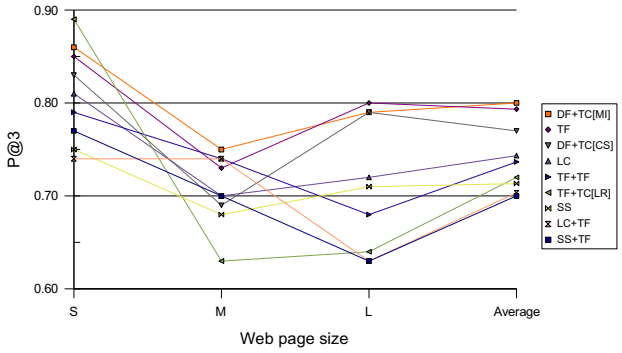

Figure 1: Precision at the first three output annotations for the best methods of each category.

Algorithm	1^{st} Annot.	2^{nd} Annot.	3^{rd} Annot.	4^{th} Annot.
http://citeseer.ist.psu.edu/542246.html				
TF	Search	Information	User	System
LC	Proximity Search	Relational Databases	Banks System	Foreign Key
DF+TC[MI]	Schema	Search	Web	-
DF+TC[CS]	Schema	Search	Retrieval	-
TF+TF	Search	Query	Malleable	Schema
LC+TF	Database	Query	Probabilistic	Networks
http://en.wikipedia.org/wiki/PageRank				
TF	Web	Search	Information	Pages
LC	Search Engine	Web Pages	Authority Transfer	Link Structure
DF+TC[MI]	Web	Information	Conference	-
DF+TC[CS]	Web	System	Conference	-
TF+TF	Web	Page	Links	Semantic
LC+TF	Web	Search	Semantic	Information
http://www.l3s.de/ chirita/resume.htm				
TF	Bucharest	University	Search	Computer
LC	Computer Science	Information Retrieval	Personalized Web Search	Technical Report
DF+TC[MI]	Retrieval	Search	System	-
DF+TC[CS]	Retrieval	Search	Information	-
TF+TF	Search	Retrieval	Web	Ranking
LC+TF	Search	Web	Pages	Semantic

Table 10: Examples of annotations produced by different types of algorithms.

– both major research interests of our subject, highly related to the annotated page; or "Banks system" – a database search system very similar to the one presented in the given Web page; or "probabilistic", etc.).

Practical Issues. As we discussed earlier, the approach we propose is highly scalable: Our annotation algorithms are highly efficient, and utilize client processing power to annotate the Web pages. Users don't need to produce annotations for all pages they visit, but only for a sample of them (possibly randomly selected). The server collecting the data is free to limit the amount of incoming connections in order not to become overloaded.

From an implementation perspective, the algorithms proposed here can be easily integrated into browser toolbars already distributed by the major search engines. In fact, these toolbars already communicate with logging servers in order to send statistical browsing information[5], utilized for enhancing the search results ranking function. Thus, only a small additional communication cost is necessary.

5. APPLICATIONS

The approach we presented here enables a range of applications, from the most obvious, such as personalized Web search or Web recommendations, to ontology learning and Web advertising.

Personalized Web Search. An interesting application is Web search personalization [8]. One could exploit such keyword extraction algorithms to generate term based profiles from each user's collection of personal documents. Upon searching some external collection (e.g., the Web), the output results could be biased towards those pages residing closer to the user profile in the terms hyperspace.

Web Recommendations for Desktop Tasks. It is quite common for people to search the Web for currently existing content to assist their present work tasks. It is possible to use the approaches we proposed in this paper in order to *automatically* suggest such pages [9, 22]. More specifically, upon editing a Desktop document, relevant keywords would be extracted from the already written content and utilized to search on the Web for additional, useful resources on the same topic. Then, these automatically located pages could be displayed in a condensed format (e.g., title and URL) using a small discreet window placed for example in the bottom-right corner of the screen.

Ontology Learning. In the scenario of ontology-driven annotations, where an underlying ontology (customizable for each user) provides the terminology for such annotations, it might be necessary to enrich the ontology with user-specific concepts. These can be provided from the user's context, represented by keywords extracted from her Desktop environment. For instance, the initial Personal Information Management ontology might lack a concept relevant to a specific user, for instance the class "Research Interests" as subclass of "Interests".

An analysis of keywords detected by the Lexical Compounds method is particularly valuable for an Ontology Learning approach, which we intend to investigate in future work. Based on the metrics described in this paper, the term "Research Interests" could be suggested as a keyword, and adopted as a class in the user-specific layer of the ontology. Relevant multiword expressions detected in such a way, sharing the same head noun, may be proposed as classes and organized hierarchically, while a hypothesis analysis for collocation over the Desktop corpus will enable us to distinguish between a simple modified head noun and a multiword expression bearing more meaning than the sum of its parts. Further knowledge may be extracted by considering sentences containing the keywords from the set determined by the Sentence Selection algorithm, which are often of descriptive nature. Considering the example above, it would be straightforward to identify "information extraction", "natural language processing" and "knowledge representation" as instances of the concept "Research Interests", given for example the sentence "His research interests include information extraction[..]" and given the assumption that variations of this sentence will be found in documents on the Desktop and on the Web. Finally, this strategy will enable us to extract both instances and relevant relations between the proposed classes.

[5]Only with user's consent.

Other Applications. If we move away from using personal Desktops, we can identify quite a lot of other applications of the same algorithms, some of them even already investigated. Due to space limitations we note here only one very important example: Web advertising. Keywords, as extracted from Web pages with the algorithms we presented, could be used to better match advertisements, as well as to propose better bidding expressions for the owner of each input Web site.

6. CONCLUSIONS AND FURTHER WORK

We have described a novel approach for scalable automatic personalized generation of annotation tags for Web pages. To the best of our knowledge there is no algorithm that does the same. Our technique overcomes the burden of manual tagging and it does not require any manual definition of interest profiles. It generates personalized annotation tags for Web pages by building upon the implicit background knowledge residing on each user's personal Desktop. Also, in contrast to keyword extraction algorithms that can only propose terms which actually appear on the input Web page, our system proposes a more diverse range of tags which are closer to the personal viewpoint of the user. The results produced provide a high user satisfaction (usually above 70%). Thus, the greatest benefit of P-TAG is the high relevance of the tags for the user, and therefore the capacity of the tag to describe a Web page and to serve for a precise information retrieval.

The current implementation of P-TAG is intended to assist Web search engines. For the near future we therefore plan to implement a shared server approach that supports social tagging, in which the system would know about personal annotations from other users and would be able to identify the most popular annotations, e.g., the ones with the highest score. This would also enable the sharing of the automatic generated personal annotations in a collaborative environment, and would simply automatically create, apply and share tags dynamically.

For such a server based approach we envision the following advantages:

1. *Diversity*: Keywords are generated from millions of sources, and thus cover various user interests.
2. *Scalability*: The annotation server can choose from which machines to collect the annotations, as well as from how many machines.
3. *High Utility for Web Search, Analytics, and Advertising*: One can easily mine the dominating interests of the persons browsing a given Web page or set of pages. Similarly, the Web page could be made searchable for an extended set of keywords, including the annotations generated with our algorithm.
4. *Instant Update*: We do not have to worry about the high volatility of the Web; newly created Web pages get annotated automatically, as they are visited by users.

We believe that P-TAG provides rather intriguing possibilities which can lead to a considerable high amount of annotated Web pages by automatic personalized tagging, thus lowering the barrier for the application of more semantics on the Web.

7. REFERENCES

[1] H. Alani, S. Kim, D. E. Millard, M. J. Weal, W. Hall, P. H. Lewis, and N. R. Shadbolt. Automatic ontology-based knowledge extraction from web documents. *IEEE Intelligent Systems*, 18(1):14–21, 2003.

[2] P. G. Anick and S. Tipirneni. The paraphrase search assistant: Terminological feedback for iterative information seeking. In

Proc. of the 22nd Annual Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval, 1999.

[3] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. ACM Press / Addison-Wesley, 1999.

[4] C. H. Brooks and N. Montanez. Improved annotation of the blogosphere via autotagging and hierarchical clustering. In *Proc. of the 15th World Wide Web Conference*, 2006.

[5] J. Budzik and K. Hammond. Watson: Anticipating and contextualizing information needs. In *Proceedings of the Sixty-second Annual Meeting of the American Society for Information Science*, 1999.

[6] L. Carr, W. Hall, S. Bechhofer, and C. Goble. Conceptual linking: ontology-based open hypermedia. In *Proc. of the 10th Intl. Conf. on World Wide Web*, 2001.

[7] C.-H. Chang and C.-C. Hsu. Integrating query expansion and conceptual relevance feedback for personalized web information retrieval. In *Proc. of the 7th Intl. Conf. on World Wide Web*, 1998.

[8] P. A. Chirita, C. Firan, and W. Nejdl. Summarizing local context to personalize global web search. In *Proc. of the 15th Intl. CIKM Conf. on Information and Knowledge Management*, 2006.

[9] P. A. Chirita, C. S. Firan, and W. Nejdl. Pushing task relevant web links down to the desktop. In *Proc. of the 8th ACM Intl. Workshop on Web Information and Data Management held at the 15th Intl. ACM CIKM Conference on Information and Knowledge Management*, 2006.

[10] P. Cimiano, S. Handschuh, and S. Staab. Towards the self-annotating web. In *Proceedings of the 13th World Wide Web Conference*, 2004.

[11] P. Cimiano, G. Ladwig, and S. Staab. Gimme' the context: context-driven automatic semantic annotation with c-pankow. In *Proc. of the 14th World Wide Web Conference*, 2005.

[12] H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan. Gate: A framework and graphical development environment for robust nlp tools and applications. In *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics*, 2002.

[13] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391–407, 1990.

[14] S. Dill, N. Eiron, D. Gibson, D. Gruhl, and R. Guha. Semtag and seeker: Bootstrapping the semantic web via automated semantic annotation. In *Proceedings of the 12th World Wide Web Conference*, 2003.

[15] T. Dunning. Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*, 19:61–74, 1993.

[16] H. P. Edmundson. New methods in automatic extracting. *Journal of the ACM*, 16(2):264–285, 1969.

[17] E. N. Efthimiadis. User choices: A new yardstick for the evaluation of ranking algorithms for interactive query expansion. *Information Processing and Management*, 31(4):605–620, 1995.

[18] D. Fetterly, M. Manasse, M. Najork, and J. Wiener. A large-scale study of the evolution of web pages. In *Proc. of the 12th Intl. Conf. on World Wide Web*, 2003.

[19] S. Gauch, J. Wang, and S. M. Rachakonda. A corpus analysis approach for automatic query expansion and its extension to multiple databases. *ACM Transactions on Information Systems*, 17(3):250–250, 1999.

[20] S. Handschuh and S. Staab. Authoring and annotation of web pages in cream. In *Proc. of the 11th Intl. World Wide Web Conf.*, 2002.

[21] M. A. Hearst. Untangling text data mining. In *Proc. of the 37th Meeting of the Association for Computational Linguistics on Computational Linguistics*, 1999.

[22] H. Holz, H. Maus, A. Bernardi, and O. Rostanin. From Lightweight, Proactive Information Delivery to Business Process-Oriented Knowledge Management. *Journal of Universal Knowledge Management. Special Issue on Knowledge Infrastructures for the Support of Knowledge Intensive Business Processes*, 0(2):101–127, 2005.

[23] J. Kahan, M. Koivunen, E. Prud'Hommeaux, and R. Swick. Annotea: An Open RDF Infrastructure for Shared Web Annotations. In *Proc. of the 10th Intl. World Wide Web Conf.*, 2001.

[24] M.-C. Kim and K. Choi. A comparison of collocation based similarity measures in query expansion. *Information Processing and Management*, 35:19–30, 1999.

[25] A. Kiryakov, B. Popov, D. Ognyanoff, D. Manov, A. Kirilov, and M. Goranov. Semantic annotation, indexing, and retrieval. In *International Semantic Web Conference*, 2003.

[26] A. M. Lam-Adesina and G. J. F. Jones. Applying summarization techniques for term selection in relevance feedback. In *Proc. of the 24th Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval*, 2001.

[27] D. Maynard, K. Bontcheva, and H. Cunningham. Towards a semantic extraction of named entities. In *Recent Advances in Natural Language Processing*, 2003.

[28] G. Miller. Wordnet: An electronic lexical database. *Communications of the ACM*, 38(11):39–41, 1995.

[29] J. Rocchio. Relevance feedback in information retrieval. *The Smart Retrieval System: Experiments in Automatic Document Processing*, pages 313–323, 1971.

[30] V. Vinay, K. Wood, N. Milic-Frayling, and I. J. Cox. Comparing relevance feedback algorithms for web search. In *Proc. of the 14th Intl. Conf. on World Wide Web*, 2005.

[31] S.-C. Wang and Y. Tanaka. Topic-oriented query expansion for web search. In *Proc. of the 15th Intl. Conf. on World Wide Web*, pages 1029–1030, 2006.

[32] J. Xu and W. B. Croft. Query expansion using local and global document analysis. In *Proc. of the 19th Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 4–11, 1996.