Personalization Mechanism for Delivering News Articles on the User's Desktop

Christos Bouras
Research Academic Computer Technology Institute,
N. Kazantzaki, Panepistimioupoli and
Computer Engineering and Informatics Department,
University of Patras, Greece
e-mail: bouras@cti.gr

Vassilis Tsogkas
Computer Engineering and Informatics Department,
University of Patras
e-mail: tsogkas@ceid.upatras.gr

Abstract—**Deploying a generalized, multi-functional mechanism that produces good results for personalizing summarized and categorized news articles, seems to be a panacea for most of the text-based, information retrieval needs. In this paper, we present the personalization algorithm, as well as a client side application that are used as an extension to an already constructed system, PeRSSonal, in order to deliver summarized and pre-categorized news articles to the user.**

Keywords-personalization algorithm, news delivery, desktop application

I. INTRODUCTION

The web information age has brought a dramatic increase in the sheer amount of information, the access to this information, as well as the intricate complexities governing the relationships within this information. Nowadays, users tend to prefer personalized information that is easily delivered, without much hassle, to them. The aforementioned facts, probe for new interconnection architectures and transferring of data. Following this path, several interconnection protocols have risen the last years: XML/XSL[1], SAX[2] and DOM[3].

Personalization aims to customize the results on a user's explicit and/or implicit interests and desires. As explained in [9], the move to personalization is no longer an option, but a necessity. Several challenges however come into place for the personalization procedure to be successful; scalability, accuracy, evolving user interests, data collection and preprocessing, intergrading multiple sources of data, as well as privacy issues are only some of the aspects that our system faces up. From an architectural and algorithmic point of view, personalization systems fall into three basic categories: rule-based, content-filtering, and collaborative filtering systems.

Personalizing news feeds is an interesting subtask of news filtering and personalization that has emerged the last years. Its target is to effectively separate interesting news articles for a user from a large amount of documents. For example in [4], the authors make use of a machine learning classification framework to filter news following the user's choices. Another technique for adaptive news, which is based on user modeling, is presented in [11], where the system maintains separate user models for each topic of news. However these systems lack serving of news summaries and use fairly trivial keyword extraction techniques. Also they expect from the users to explicitly modify their profiles which is often a tedious task. Another interesting approach is presented in [6] where the notion of information novelty is utilized. Despite the fact that the implemented algorithms perform well, they also lack automatic recording and prediction of user preferences which change frequently as in [7]. An alternative approach is researched in [8] where the system produces multi-document summaries and has the ability to adjust to the various input texts. However this system lacks categorization and personalization features that could be combined with the resulting summaries, as in our work.

PeRSSonal [5], the automatic summarization, text categorization, personalized syndication system, applies several data mining techniques through a layered infrastructure that achieves filtering of information and adaptability to the user. We are focusing on news articles that are gathered from numerous news portals from around the world and we are targeting to the alleviation of the end-user from the cumbersome task of searching through this overwhelming plethora of information. The user participates to the procedure by explicitly defining his/her preferences or by letting the system adapt automatically to his/her dynamically changing profile.

In this paper, we are dealing with the effective and adequate presentation of personalized news summaries from articles that derive from the WWW to the user's desktop. We present the personalization algorithm that is used for presenting the pre-categorized and summarized articles to the user's desktop application which is capable of exchanging information with our already mature categorization and summarization system: PeRSSonal [5]. Our personalization approach is mainly content-based with some collaborative filtering features by enhancing the algorithm with the ability to automatically adopt over time to the continuously changing user choices. Furthermore, we base our summarization procedure (enhanced by the

personalization module) on the TF-IDF term-weighting model.

The rest of the paper is structured as follows: in section 2 the architecture of the complete system is presented, whereas in section 3 some implementation issues and techniques are discussed. Section 4 gives the algorithmic outline of the personalization procedure that is followed. In section 5 we present the evaluation results for the developed mechanism and section 6 concludes this paper with some additional thoughts for future additions to the system.

II. ARCHITECTURE

The developed mechanism follows a classic n-tier architectural procedure that is depicted in Fig. 1.

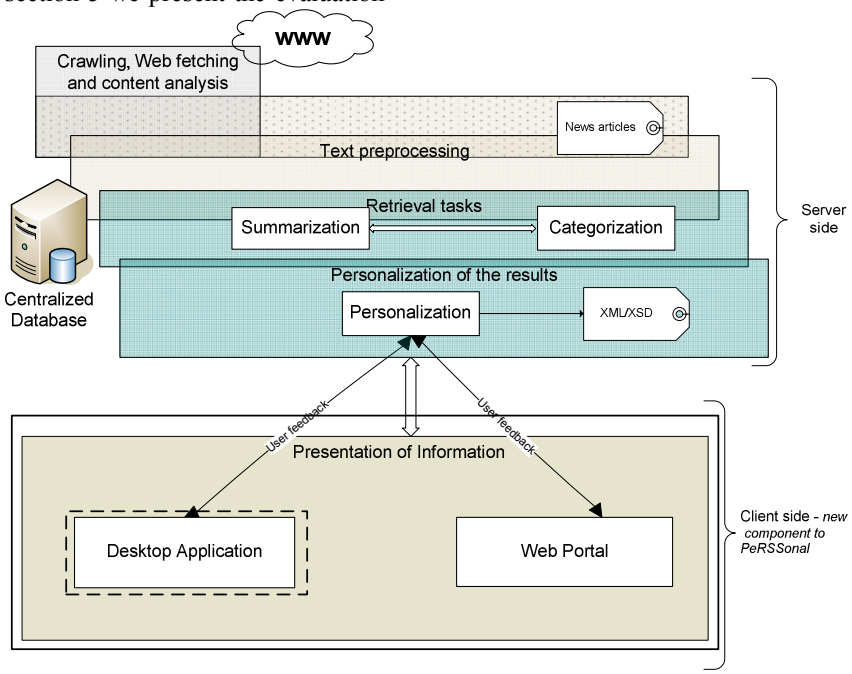

Figure 1. Architecture of the PeRSSonal Mechanism

The procedure that is followed, as depicted in Figure 1, starts with the interconnection between the mechanism and the web sources and it is where the primary tasks take place. These include: the content fetching procedure, the analysis of the downloaded content and finally the extraction of the useful information from the web content. In order to capture web pages, a simple focused web crawler is used. The crawler takes as input the addresses that are extracted from existing RSS feeds, deriving from several major news portals. These RSS feeds point directly to pages where news articles exist and they also provide us with useful information per article (e.g. article's title) or per RSS (e.g. pre-categorized feed). The crawling procedure is distributed across multiple systems which synchronize through the centralized database. Crawled html pages are analyzed and are stored without any other unnecessary page element (images, css, javascript, etc). During this analysis level, our system isolates the "useful text", meaning the main body of the article. By storing only the useful text, as well as some other page meta-data, such as URL and insertion date, the database is populated with news articles that are ready for the text preprocessing step.

The second tier of the system works on the article's title and body applying several preprocessing techniques. Text preprocessing is probably the most important preceding task of any text-based IR technique and hence our approach pays much attention to the various algorithms that are used. In particular, after the retrieval of the stored article that resides in the database, a series of inner procedures take place at this layer. Firstly the article's language is recognized. Following the sentence separation and punctuation removal steps come. Afterwards, the noun identification step takes place [5] and some common text extraction techniques follow: stopwords removal and stemming. The results of the procedures described in this layer are stemmed keywords either marked as nouns or not, their location in the text and their frequency of appearance in it. These are represented through term frequency – inverse document frequency (TF-IDF) vector statistics that are stored in the database and are utilized by the procedures of the third analysis level. The aforementioned keyword

extraction subtasks are to their majority, language dependant, meaning for example that different stopwords lists, stemming rules and noun retrieval algorithms are used for every different language.

The information retrieval tasks of our mechanism are located in the third analysis level, where the summarization and categorization algorithms are applied. The main scope of the categorization module is to assist the summarization procedure by pre-labeling the article with a category and has proven in [5] to be providing better results. Following the IR tasks of the system, personalization algorithms take place. The personalization module that is described in this paper, is easily adaptable to the user, meaning that small changes to the user's preferences, as expressed by his/her browsing behavior, are detected adjusting thus the profile. Our personalization algorithm uses a variety of user-related information in order to filter the results presented to the user.

Finally, the content is delivered to the user using XML formatting for data content and XSD schemas for the transmitted data. The focusing of this paper are the last parts: the personalization procedure on the server-side, assisted by the feedback information from the user, as well as the delivery of the summarized content to the user's desktop through the client-side application.

III. IMPLEMENTATION ISSUES

In order to implement the client-side application we utilized the latest version of the Qt libraries and corresponding toolkit [10]. The advantages of the selected approach are wide availability for different platforms, and high performance. Availability is a key feature for any network-based application to succeed. The selected choice also gives our client-side application a nice performance boost over the web interface approach. The latest originates from the fact that the application a) implemented in C++ and b) is threaded suitably for the user needs: each thread takes care of the transferring of needed information with the PeRSSonal server using pre-fetching techniques. In this manner, the client side is becoming much more responsive and user friendly over the web interface of PeRSSonal.

As already mentioned, XML formatting is utilized for transferring of the data between the client application and the server. The XML files are created dynamically by the PeRSSonal server when asked from the user and the corresponding XSD schemas are also available. In this way, the serving capabilities of PeRSSonal greatly increase: any connected client can communicate with PeRSSonal using the supplied XSD schemas and the dynamically created XML files. Future extensions for this are also available since for example a mobile phone client version can be implemented using the described API.

An important aspect of the mechanism that dynamically generates the XML responses on each user request is that it gets benefit from caching techniques that are used on the server side. More information about the caching system is however beyond the scope of the current paper.

IV. ALGORITHM ANALYSIS

In this section we are presenting a novel personalization algorithm that is utilized for the presentation of information by our system. We are explaining how the user's profile is generated, the way that it is dynamically updated and, finally, how the presented results depend on it.

```
Update_profile(a, b, c){
 Get_articles(a,b)
 for each article{
  if (full article)
  if (time_viewed > Rar_thr1 && time_viewed <
Rar_thr2){
     Keywords_positive = top 5 frequent keywords
     Update_list(Positive, Keywords_positive)}
 else
    if(time_viewed> Rsum_thr1 && time_viewed<
Rsum_thr2){
    Keywords_positive = top 5 frequent keywords
    Update_list(Positive, Keywords_positive)}
 Get_articles(c)
 for each article{
    Keywords_negative = top 5 frequent keywords
   Update_list(Negative, Keywords_negative)
}
Get_article(lists){
//Recovers the browsed articles and the amount
//of time spent reading the full article or its
summary (a,b).Recovers the negated articles (c)
}
Update_list(list, keywords){
    for each (keyword in keywords)
    if (keyword not in list[])
        list.add(keywords[keyword])
    else
        list.update_freq(keywords[keyword])
                        }
```

Figure 2. Personalization algorithm that utilizes user feedback

A. Personalization using user feedback

The developed mechanism is based on the continuous feedback from the user. The steps that are followed by the personalization procedure are presented in Fig. 2. When a new user is registering to the PeRSSonal service, he/she states the keywords of his/her preference as well as the scores that describe this preference initializing thus his/her profile.

This procedure is trivial and can be avoided altogether since the personalization subsystem keeps track of the user's choices and browsing history, and so the user's preferences are updated on each visit. The user's profile consists of two keyword lists: a positive

one, where the user-preferred keywords are placed, and a negative one where uninteresting keywords for the user are kept. By using these lists, the summarization procedure can personalize the summaries with exceptional results.

The profile update procedure, running constantly at every user's visit, takes note of the following aspects: a) the browsed articles (the ones that the user selected to view), b) the amount of time a user spends viewing the summary or the full text of a specific article, c) the articles that the user avoids viewing (either their summary or their full text); the above derives from the simple logical assumptions that follow. A user will most likely spend an amount of time above a certain threshold, Rar_thr1 or Rsum_thr1, reading an article's full text or its summary respectively, that is of interest for him/her (factor a). However, an upper bound, Rar_thr2 and Rsum_thr2, should be used for these metrics since we don't want the mechanism to mistake forgotten browsed articles for the really interesting ones. The thresholds that are used for Rar_thr1 and Rar_thr2 are 30 seconds and 3 minutes respectively defining thus which article's keywords should be added (or have their weight increased) in the user's positive keywords list. The summary viewing thresholds are calculated in an analogous way:

$$\text{Rsum\_thr1} = \text{Rar\_thr1} * \text{Sratio} \quad (1)$$

$$\text{Rsum\_thr2} = \text{Rar\_thr2} * \text{Sratio} \quad (2)$$

Where Sratio expresses the summarization "compression ratio":

$$Sratio = \frac{\#\ words\ (summary)}{\#\ words\ (fulltext)} \quad (3)$$

Moreover, most of the times, a user will select to browse articles of a topic that he/she finds interesting (factor b) as advertised by the article's title and/or summary. Lastly, a user will probably avoid visiting articles that he/she finds uninteresting and thus the keywords that represent those articles should be receiving a lessened or negated weight (factor c).

From the above factors, the personalization algorithm keeps track of the keywords that the user has expressed preference to, and thus, the articles (containing these keywords) that he/she is likely willing to read in the future. The parameter that depicts the user's preference for a keyword according to the aforementioned factors (a-c) is Uwi and is based on the relative frequency that the keyword has on the list, a frequency that is constantly modified by the user's choices. Uwi derives from the following equation:

$$Uwi = rel(fr(kwi)) * (1 + Tkwi) \quad (4)$$

Where Tkwi is the normalized total time spent on the specific keyword if it belongs to the positive list; however if the keyword is in the negative list, Tkwi is set to 0 since no time is actually spent of these keywords by the user. Furthermore, we expect that when the user profile reaches its steady state, the mean times of the keywords preferences to be correct, hence depicting the overall user preferences.

The overall personalization factor for each keyword i, named k4, is:

$$k4_i = B * Uwi \quad (5)$$

Where, for the parameter B: if the keyword belongs to the positive keyword list, then B>1; whereas if the keyword belongs to the negative keyword list, then B<1. The norm of the B parameter can take any value that we desire, thus increasing or decreasing at will the effect that personalization and dynamic profile generation have on the sentence weighting procedure. From the previous, k4 can be positive, negative or zero if there is no information about the user's preference of the specific keyword.

The overall weighting formula that is introduced in [5] becomes as follows.

$$W_i = \sum (1 + rel(fr(kw_{k,i})))(k_1 + k_2 + N)k_3 k_4 \quad (6)$$

B. Client Side Application and Content Delivery

According to the personalization features of the mechanism that were described earlier, the server transmits the responses to the client using the XML communication protocol derived from the existing XSD schemas. Following the delivery of the content, a number of features are available to the user. Users can browse through the 4 basic information "channel" tabs: recent/latest, user-preferred, most read and featured, selecting the articles that they are mostly interested in viewing. The summaries of the selected articles are loaded and send to the application screen. The whole application is fully configurable working is a modular manner, since the GUI is dynamically created using the existing XSD schema. It is also important to note that information is loaded in a threaded manner by the client. Having available information about the user's profile and browsing habits, we are able to pre-fetch articles, summaries and relativity information before the user actually asks for it. This implementation offers a significant boost to our application over the web interface of PeRSSonal, since it makes it more responsive and usable.

V. EXPERIMENTS

In order to evaluate the possible enhancement of the proposed personalization mechanism on the system's procedures, and more specifically, on the summarization subsystem, we conducted a set of experiments. We also experimented in order to determine whether the client side desktop application is actually a nice equivalent to the web interface.

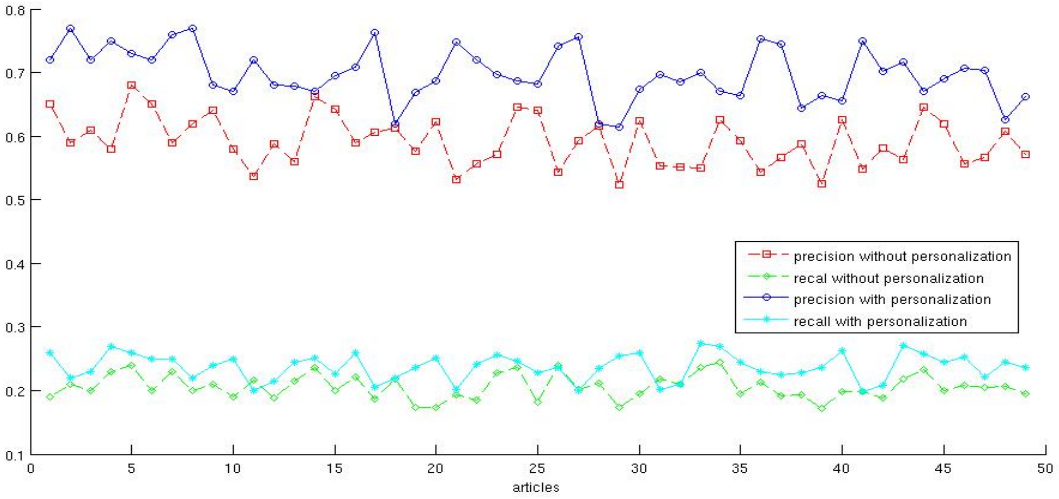

Figure 3. Precision and Recall Results

For our summarization evaluation approach, we used classic precision-recall metrics and 30 human system users from various fields of interests. We first asked our test users to register to the system and use it (through its web interface and the desktop application) for one month's period so that the personalization algorithm fully adapts their profile to their preferences. Afterwards, we provided them 50 full text articles that were matching their created profiles and we asked them to rate some sentences of these articles for a suitable summary of the article. We also produced the personalized as well as the "generic" summaries of these articles (without utilizing the personalization features explained in this paper) and compared them with the users' choices in terms of precision and recall. The results, presented in Figure 3, are depicting the rates with and without the personalization factor. Extracted summaries are then compared with the sentences selected by the users and precision as well as recall metrics are evaluated.

From Fig. 3 it is deducted that the appliance of our new personalization scheme has provided a significant benefit to the overall summarization performance of the mechanism as far as precision and recall metrics are concerned. We measured this increase to be around 17% for precision and 14% for recall. It is important however to note that the statistics are based on the user choices and are subjectively biased by nature. On the other hand though, there are no real objective criteria for the extraction of a summary from a text and it is this "bias" that the proposed personalization mechanism is trying to estimate.

Moving to our second set of evaluation, we asked our users to utilize both the web interface and a beta version of the client side application of PeRSSonal for a period of some days. We then asked them to rate both of the presentation systems in terms of: a) usability (is the system serving enough and good summaries?) and user-friendliness, b) performance and interactivity (are response times good?), c) efficiency and briefing in content representation. The rating was done with a scale from one to ten, with ten being the best.

The results that are presented in Fig. 4, express the user's overall preference in favor of the desktop client side application for the presentation subsystem of PeRSSonal. It is clear though that some users think low of the desktop application in terms of its usability since they often got puzzled on its use especially as far as discovering all the features that it provides. This is however expected since a) to our knowledge this is the first desktop application that is focusing on such information retrieval and personalization tasks, b) taking into consideration its compact representation of quite a big amount of information.

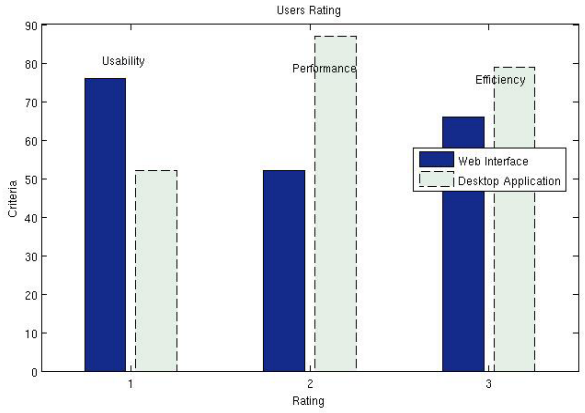

Figure 4. Users evaluation of the presentation subsystem

Furthermore, as far as performance and interactivity is concerned, the desktop application outruns the web interface. This originates from the caching and pre-fetching techniques that the desktop application makes use. Lastly, efficiency and briefing in content representation, a key target of the PeRSSonal system is an aggregation of the previous factors and shows that despite the fact, that the client-side application is still under development, the novel features that is provides are considered overall useful by the users.

After the text edit has been completed, the paper is ready for the template. Duplicate the template file by using the Save As command, and use the naming convention prescribed by your conference for the name of your paper. In this newly created file, highlight all of the contents and import your prepared text file. You are now ready to style your paper.

VI. Conclusions and Future Work

We have outlined the personalization algorithm that our system utilizes for presenting pre-categorized and summarized articles to the user. Furthermore, we presented the communication channel that PeRSSonal uses for delivering content to the end users. Our personalization approach is mainly content-based with some collaborative filtering features adopting over time to the continuously changing user profile.

We conducted experimental procedure in order to evaluate the overall improvement of PeRSSonal's summarization capabilities with the appliance of the new personalization algorithm. We used real system users and even though the evaluation of a summarization system is a difficult and subjective task, we discovered a significant amendment. Moreover, we evaluated the developing client side desktop application and explored the significance of developing such a communication infrastructure from the eye of the end user. The results are encouraging and express the users' need for efficiency and interactivity from an application that is targeted not as a replacement but as a complement of the Web 2.0 technologies that are utilized by the system.

As future work, we are focusing on a stable version of our desktop application and on a wider evaluation of PeRSSonal. As far as the core procedures of the system are concerned, we are intending to incorporate multilingual support as well as support for multimedia and improved caching features.

References

[1] Extensible Markup Language (XML) - W3C, http://www.w3.org/TR/1998/REC-xml-19980210.

[2] Simple API for XML (SAX), http://www.saxproject.org/

[3] Document Object Model (DOM) – W3C, http://www.w3.org/DOM/

[4] E. Banos, I. Katakis, N. Bassiliades, and G. Tsoumakas, "PersoNews: A Personalized News Reader Enhanced by Machine Learning and Semantic Filtering," Lecture Notes in Computer Science, Vol. 4275, 2006 pp. 975.

[5] C. Bouras, V. Poulopoulos, and V. Tsogkas, "PeRSSonal's core functionality evaluation: Enhancing text labeling through personalized summaries," Data and Knowledge Engineering Journal, Elsevier Science, Vol. 64, Issue 1, 2008, pp. 330 – 345.

[6] C. Bouras and V. Tsogkas, "Improving text summarization using noun retrieval techniques," Lecture Notes in Computer Science. Knowledge-Based Intelligent Information and Engineering Systems, Vol. 5178/2008 pp. 593-600.

[7] E. Gabrilovich, S. Dumais, and E. Horvitz, "Newsjunkie: Providing Personalized Newsfeeds via Analysis of Information Novelty," Proceedings of the 13th international conference on WWW, pp. 482-490 2004 ACM.

[8] K. McKeown, R. Barzilay, and D. Evans, "Columbia multi-document summarization: Approach and evaluation," Proceedings of the Workshop on Text Summarization, 2001 ACM SIGIR Conference.

[9] O. Nasraoui, "World Wide Web Personalization," Invited chapter in Encyclopedia of Data Mining and Data Warehousing, J. Wang, 2005 Ed, Idea Group.

[10] Qt Cross-Platform Application Framework, http://trolltech.com/products/qt/

[11] C. Wongchokprasitti, and P. Brusilovsky, "NewsMe: A Case Study for Adaptive News Systems with Open User Model," Proceedings of the Third International Conference on Autonomic and Autonomous Systems, 2007.