This Just In! Your Life in the Newspaper

Bruno Antunes, Tiago Guerreiro, Daniel Gonçalves

Universidade Técnica de Lisboa
Av. Rovisco Pais, 49
1050-001 Lisboa, Portugal
bruno.r.t.antunes@ist.utl.pt, tjvg@immi.inesc-id.pt, daniel.goncalves@inesc-id.pt

Abstract. It is not uncommon for computer users to work on several things at once. However, to the computer, all documents, emails and applications are considered equal, regardless of why they were created or used. Little support is provided when trying to recall important information about a particular project or subject at a later time. What is more, there is no effective way to help users review their past activities to identify when a particular subject was of importance, what were their concerns at a given moment in the past, or simply review their activities during a period of time at a glance. In this paper we describe PersonalNews, a system in which users are presented with a personal newspaper, in which the news articles describe the subjects they were concerned with in a given period of time. Those articles are automatically generated from the users' documents, grouped according to their subject and analyzed for relevant passages describing them. We show that PersonalNews is able to recognize the subjects and projects the users were involved in, and even help them recall some they had forgotten about. Also, it can be used effectively to help retrieve documents on particular subjects, even when the usual hints of filename and location in the filesystem.

Keywords: Personal Information Management, Personal Document Retrieval, Newspaper Metaphor, Information Visualization.

1 Introduction

Nowadays, many aspects of our daily lives require the use of a computer. From tasks at the workplace, to eGovernment initiatives, computers increasingly pervade our lives. This results in most users having to handle several different projects at the same time, shifting their attention from one to another, and producing a wealth of information, some of it unrelated, that might need to be retrieved at a later time.

Alas, computers have no way of knowing when the users' work context changes. Instead, all documents, emails, appointments and other personal information are treated the same regardless of their subject or why they were handled or produced. Most current filesystems are the direct descendents of solutions invented when the amount of information to be handled was small, and its possible uses limited. As such, apart from their location in a hierarchy, filename and a set of dates, documents have little relevant semantics associated to them, making their organization difficult. Indeed, it has long ago been found that other more personal and semantically rich

features, such as a document's subject, purpose or appearance, are better criteria for their organization than the hints provided by filesystems. This is clear in the case of email, in which email clients are often overloaded as document-organization tools, even if not originally designed to do so. This happens because email messages, unlike documents, have associated to them a wide range of contextual information (sender, subject, etc.) that makes managing them easier [18].

What is more, information relating to a single project or activity can be spread throughout different filesystems locations or even applications, with nothing to link it as a coherent whole. This Fragmentation Problem [1] further hinders the efficient handling of personal information.

All this causes problems when users try to review their activities to find some relevant information. There is no simple way to visualize all information grouped into meaningful sets, directly derived from the users' interests and projects. Such an overview of the users' lives would help them to more easily recognize when was a certain project addressed, what was it about, and what documents resulted from it. Thus, such visualization would help users to find relevant information directly but also indirectly, by providing them with clues that can be used in traditional organization methods, such as dates and keywords.

In this paper, we describe PersonalNews, a system in which a newspaper metaphor is used to help users visualize their interests and projects, at any given period of time, at a glance. It automatically infers the different subjects that concerned the users by looking at their documents. Then, relevant passages of those documents are extracted and manipulated to create news articles that describe those subjects. Those articles are then laid out on a newspaper page according to their importance. By looking at this personal newspaper, the users are able to immediately recall relevant aspects of their lives and find relevant information for reference or re-use.

In the following section, we will describe some information visualization research that, like ours, tried to provide an overall view of personal information. We'll then describe the PersonalNews interface and, in the next section, after a description of the overall system architecture, we'll discuss how we managed to infer the different subjects the users worked on and extract relevant excerpts from documents. Afterwards we'll show how those excerpts were used to create news articles. The user evaluation of PersonalNews is described next. Finally, we'll conclude, pointing to potentially interesting future research in the area.

2 Related Work

In the last years we have witnessed many developments in personal information visualization. One of the first studies in the area was Forgot-me-not [1]. This article describes a PDA-like device where personal information is displayed, trying to help the memory of its user. For example, it allows remembering where to find a document or the name of a friend. While limited by technology existing at the time, but nevertheless interesting to us as it allows the users' lives to be visualized at a glance.

Lifestreams [5] [6] organizes personal documents as a time-ordered stream that functions as a diary of user's electronic life. It uses a simple organizational metaphor

(document streams). The tail of the user's stream contains documents from the past. Moving away from the tail and toward the front, we find more recent documents. Its disadvantage is that it only focuses on documents and gives time a central role, while other, more semantically rich features (subject, purpose, etc.) are neglected.

LifeLines [11] is a general-purpose technique for visualizing personal history record summaries. They provide a complete visualization environment offering overviews, zooming, filtering and details on demand. An overview is always visible while providing easy access to the details. Its problem is that it handles personal data in limited fairly structured contexts, such as medical or court records.

Another interesting application is FacetMap [11], an interactive visualization system guided by queries. It tries to show information efficiently, considering constraints such as display resolution, the number of items and their attributes. It uses faceted search with the topics used to organize the information (time, kind, author, etc.), represented by ovals that can be used to filter search results. However user tests showed them to be confused by the interface and with the several facets.

Stuff I've Seen [17][1] is a system with a built-in search engine that can index all the information that a user saw over a period of time. This information can refer to Web pages, emails and documents, among others. Search results are presented with the help of a timeline, following an overview+detail approach. The timeline is annotated by personal landmarks (pictures, tasks, etc.) and public events (reports, vacations, holidays, etc.). This system is limited to simple, direct searches providing no in-depth semantic analysis of the available information.

More recently, Themail [17] creates a visualization of the information preserved in its users' email files. It shows the relation between the users and their email contacts, across the time, by highlighting the most important keywords in the messages exchanged with them. The interface is attractive and efficient. However, it is limited to the information contained in email messages. Another problem of this application is to treat all the messages similarly, not taking into account their relative importance.

All these works provide a way to visualize personal information. However they ignore some important aspects considered by PersonalNews, such as presenting information allowing the immediate perception of semantically relevant patterns. Also, the information is shown in different ways according to its relative importance, making this evident to the eyes of the user, and helping guide its exploration.

3 The PersonalNews Interface

The PersonalNews interface was implemented as a Web Application whose front-end is a web page with the traditional look of a periodical. A locally-installed dedicated web server allows it to run locally and access the users' documents. However, it would also run remotely, if necessary. This makes our solution amenable to be used in tandem with the increasingly common web-office applications and remote document storage solutions, in which the users' documents no longer reside in their hard drives. The use of HTML and CSS gives us the versatility to format the different news in heterogeneous, dynamic ways. Furthermore, using this technology makes it possible for anyone who has ever surfed the web to use our application.

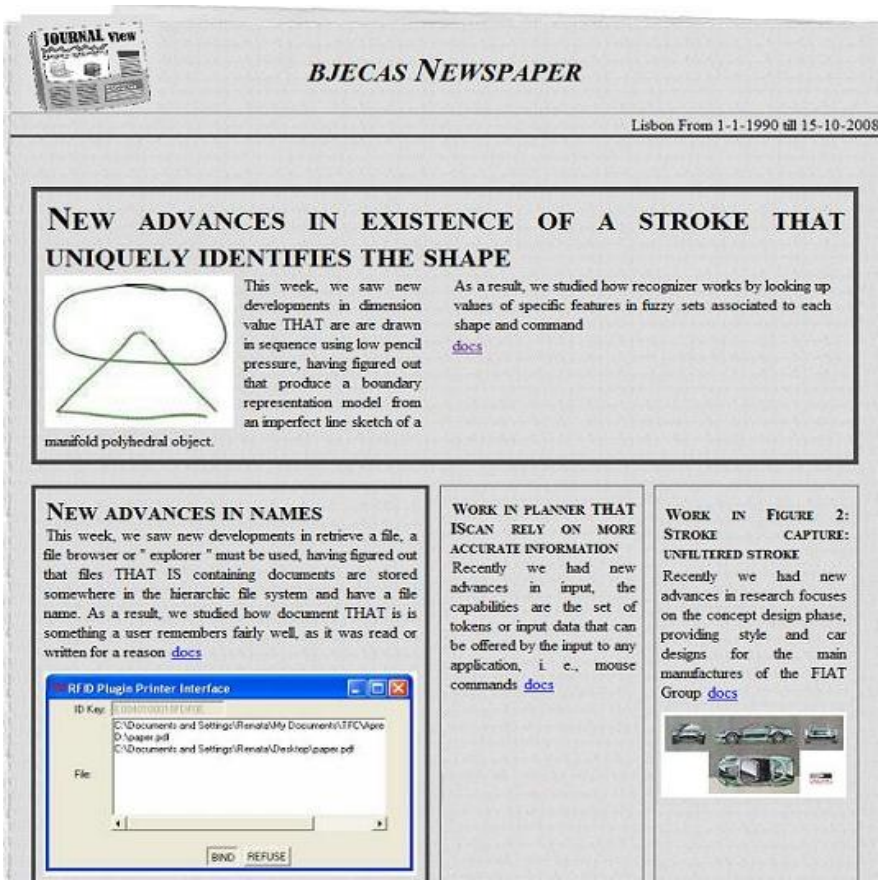

Fig. 1. The PersonalNews Interface

In the generation the newspaper page we used different font types, sizes and weights, to give it an appearance closer to that of an actual newspaper. Also, we resorted to other graphical elements such as horizontal and vertical lines to separate the articles, whenever relevant, as if the newspaper had sections. Also, when indexing the users' documents, we extract images in them. Those images are used to illustrate the articles. This helps users to recognize their subject by capitalizing on their visual memories.

When the users launch the application for the first time, it asks them to configure the application, most importantly, to give a name and the location of the documents to index. The users can also control the newspaper generation with other parameters. However, we provide good enough defaults, as described in the next section.

The main application page shows the personal newspaper edition for a selected time period (Figure 1). At the top is the newspaper heading, including its title and the dates which it refers to. Below appear the news articles, placed in the page according to their relative importance (as inferred from the number of personal documents underlying each article). More important articles are shown, as for real newspapers, with bigger headings and occupying a larger area. Each article has, at its bottom, a

link which the users can follow to reach a list documents that were used to create the article. This gives PersonalNews the potential to be used as a document-retrieval tool.

To select a time span the interface provides a browseable, zoomable timeline. It displays previously created newspaper editions, which are cached to make re-visiting them more efficient. To create an edition reporting on a new time period it is enough to select a time interval by direct manipulation of the timeline.

3.1 Newspaper Layout

The placement of the articles on the page is automatically determined based on the number of news to be displayed in the newspaper (depending on the number of relevant subjects in the time-period) and their importance (based on the number of documents that gave rise to them). An algorithm of successive subdivision of space is used to accomplish this. The page is first divided in two vertically, and the topmost part is reserved for the article of bigger importance. The remaining of the page then is divided to accommodate the news of the next importance levels. If each importance level has up to three articles, they will be placed side-by-side, in different columns. Otherwise, three will be placed side-by-side and the remaining in the next vertical division, and so on. The size reserved for each sub-division depends on the importance of the news to place there, relatively to the others. This ensures that more important news (corresponding to more important subjects) are displayed more preeminently and closer to the top.

The images extracted from the users' documents to illustrate the news are placed close to them. The decision of whether or not to use an image is governed by two criteria: the template used to create the article (as described later on this paper) might suggest its use; or they will be used to produce a more aesthetically pleasing result.

4 Creating Personal News Articles

In order to create the news articles, it is necessary to analyze the documents and the information therein. First it is necessary to identify the documents' subjects and group them accordingly. Then, documents in the same group are analyzed, resulting in a set of excerpts that meaningfully represent the group's subject. Using those excerpts, and taking into account their morphology, it is possible to create news articles.

4.1 Inferring Document Subjects

Given a set of personal documents, in order to infer which subjects those documents relate to it was necessary to group them according to their content. Since it is impossible to know beforehand what subjects might arise, some kind of unsupervised technique is necessary. Hence, chose to do this by resorting to a clustering algorithm.

Clustering Documents. One of the better known clustering algorithms is k-means, and its variants [18]. However, it requires the desired number of clusters to be know beforehand. It is impossible to know this as it will vary greatly from user to user and time-span to time-span. There are strategies in which different cluster numbers are tried and the best result is chosen. However, they are not well suited for an interactive system, since they are time-consuming and computationally intensive.

For the those reasons, we chose the QT-Clust algorithm [7]. It does not require the number of clusters to be known beforehand. Instead it needs a clustering radius. While this value might be hard to assess for the general case, it is still easier to do so than to arbitrarily predetermine the number of subjects. Furthermore, this radius can be used to fine-tune the granularity with which document subjects are considered. A large radius might result in all HCI papers to be placed in the same cluster, while a smaller one could separate between usability, user models, and other sub-topics.

For QT-Clust to work, it is necessary to have some kind of measure of how closely related documents are. We chose as its basis sets of relevant keywords extracted from the documents using the tf-idf algorithm [15]. This gives us keywords that appear frequently in a document but rarely in others, thus discriminating it. Stopwords were removed and the remaining stemmed, using the Porter Stemmer [13], before tf-idf.

The most relevant keywords for each document aren't enough to adequately group them. Related documents might not share enough keywords to be deemed similar. For instance, a document might refer the "clustering of documents" and another to the "grouping of documents". We needed to, somehow, abstract from the keywords into the underlying subjects. To accomplish this, we used Latent Semantic Analysis.

Latent Semantic Analysis. Latent Semantic Analysis (LSA) [3] abstracts from keyword frequencies to semantic descriptions of the documents' subjects and requires no a priori knowledge of those subjects. It is, thus, of special interest to us as it is able to go beyond keywords appearing in documents when assessing their semantics.

LSA is based on a matrix operation called Singular Value Decomposition. It decomposes a matrix of term frequencies per document X, into three matrixes, U, S and V, such that $X=U.S.V^T$. By construction, the less important terms-document influences are grouped together to the right of the S matrix. This allows us to replace the weights of the less influential terms with 0, thus removing much "noise" and in effect reducing the semantic dimensionality of the data. The use of tf-idf to choose only the most relevant keywords in the documents allows us to control the computational complexity of LSA. The matrix resulting from LSA can be seen as giving us the coordinates of documents in the semantic space (the vector of the weighs given to each term for each document). The Euclidean distance between them establishes how closely related documents are, a necessary input for QT-Clust.

Join Clusters. We found that, while the clustering algorithm produced good results, it tended to generate a number of clusters larger than the number of actual subjects in the documents. A closer inspection of the results showed that several of those clusters were nevertheless very close together in the semantic space. Increasing the clustering radius, however, quickly led to the creation of too general clusters. Instead, we opted for merging clusters whose centers are closer than a pre-determined threshold. This maintains semantically related documents in the same groups.

Tuning the Parameters. In this process, several parameters are relevant: the *clustering radius*; the *join distance* used to merge nearby clusters; the *dimension reduction* of in LSA; and the *number of keywords* used. It will be hard for users to know which values produce good results. Hence, we performed a set of tests to estimate which values function well in an average or general situation.

We performed our tests resorting to two different document sets, pre-classified into different subjects. The first test set, TS1, was composed of 116 scientific papers produced in our research group[1], in eight different but related areas: *eLearning*, *Accessibility*, *Mobile*, *CG*, *Improve*, *Multimedia Information Retrieval*, *Sketch-Based* and *Narrative Based Document Retrieval*. The second test-set, TS2, is composed of papers downloaded from the IEEE and ACM digital libraries, on eleven subjects: *Collaborative Tagging*, *Consensus Algorithm*, *Data Mining*, *Embedded Computation*, *Intelligent Environment*, *LSA*, *Network Protocols*, *Parallel Computation*, *Speech Synthesis*, *Use cases* and *Sketch Based Interfaces*.

To assess the quality of the results, we used two different measures: the percentage of correctly grouped documents and the Silhouette Coefficient (SC) [8] that gives us an estimate of how independent the clusters are from each other. Values between 0.7 and 1.0 indicate an excellent cluster separation, values between 0.5 and 0.7 a medium separation, and lower values show that the clustering process yielded poor results.

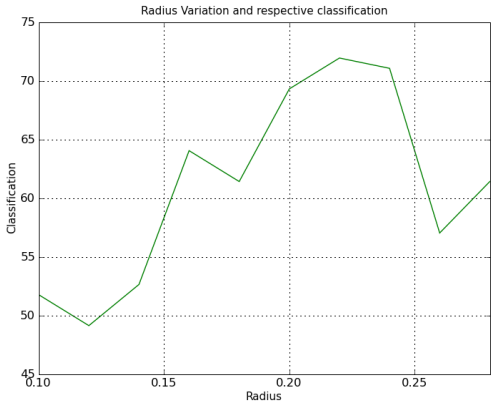

Fig 2. Classification % vs Clustering Radius

In terms of the *clustering radius*, we performed the clustering with all possible values between 0 and 1.0, with a 0.01 step (this step was used for all tests). The results, for both test sets, were similar, and are exemplified, for TS2, in Figure 2. As can be seen, the best classification is found for a radius between 0.20 and 0.25. In terms of SC, the best values (around 0.65, indicating an average cluster separation) also appear in that interval, for both test-sets. We see that a good value for the radius is 0.22.

As the SC values indicate, there are several close-by clusters that might be merged together. To control this, we studied the best values for the join distance. We found that the SC value improves significantly in the 0.2-0.7 interval. Of course, as the SC

[1] Visualization and Intelligent Multimodal Interfaces group of INESC-ID, http://www.vimmi.inesc-id.pt

improves, the number of clusters reduces, as does the percentage of correctly classified documents. This value starts worsening at 0.3, so we conclude that a good value for the join distance is in the 0.2-0.3 range. This makes sense, as it is comparable to the clustering radius. By choosing a similar value we are merging together clusters that the algorithm might have grouped in the first place if not for some quirk in its working. We also tested both variables at once. The results corroborate our previous findings.

In terms of the number of keywords used to infer the documents' subjects, we found that the value that produces the best results is between 20 and 30. The results were similar but not uniform across test-sets. This was to be expected, as the keywords vary greatly according to the domain. For TS2 there were two peak classifications, at 20, 50 and 80 keywords. The values for those numbers were, however, very similar, allowing us to assume a good compromise at 20 keywords.

Finally, regarding the dimension reduction in LSA, we saw, for both test-sets, that reducing the dimension by more than 10 produces a sharp drop in classification quality. As the time it takes to process the data increases with the dimension reduction, we chose to fixate this parameter at 1.

We conclude that the best values for the different parameters are: *clustering radius=0.22*; *join distance=0.22*; *dimension reduction=1*; and *number of keywords=20*. We used those values in all the user tests described below.

We also looked at the actual documents placed in each cluster. For the most part, all clusters strongly relate with a particular subject. They either contain documents of a single subject, or most of them are. The results were better, as expected, for TS2, as its subjects were much more independent than those of TS1. Indeed, while the documents have been classified into subjects, it is not impossible for them to mention things of others. In TS2, for instance, several Multimedia Retrieval documents were classified as Sketch-Based Interaction. A closer look at the documents showed them to describe a sketch-based retrieval tool. Hence, while the results aren't numerically perfect, in semantic terms they are more than adequate. It can be argued that the algorithm, in some cases, performs a classification which is better than the one done by hand. Furthermore, when the sets of documents are further processed in order to create the news articles, this some of the wrongfully classified documents will be filtered out, as will be shown below.

Excerpt Extraction. From sets of semantically-related documents, we wished to create news articles succinctly describing those subjects. To do so, we extracted relevant excerpts from the documents.

As a starting point, we considered, for each cluster, the keywords previously extracted from the documents in the cluster using the tf-idf algorithm. We took into account the possibility of applying the algorithm again to only the documents in the cluster, but this would have yielded worse results. The existing keywords describe the document in terms of how unique it is when compared with all of the users' documents. A new set of keywords would compare the documents inside the cluster, producing keywords that stand out in it. If, for instance, a cluster has 95% of documents on a particular subject and the remaining 5% were erroneously placed there, the most likely outcome would be for keywords actually describing the clusters'

subject not to be chosen (as they are bound to be very common inside it), and keywords regarding the 5% misses to be selected, thus polluting the results.

Given the keywords, we repeat the following algorithm for all documents in the cluster: after removing the stopwords and stemming the document, we look for a keyword in the text. If we find it, we look for other keywords inside a two word radius from it. We repeat this process if new keywords are found inside the radius. At the end, we will have identified an excerpt in which several keywords occur and that is likely to be relevant to the document's subject.

If when a keyword is identified no others are found inside the two-word radius, this radius is increased by one (up to a limit of five). While close-by keywords are preferable, we found it better to have poorer excerpts than none. Even so, we defined a quality metric for the excerpts: the number of keywords in the excerpt divided by the radius used to obtain it. Excerpts with closer keywords will be better, as it is more likely they are semantically related. The method described above offers no guarantees regarding whether entire phrases are extracted, or that the extracts are understandable in their entirety. Some post-processing is done to minimize these problems.

It would be extremely difficult to accurately use a full-fledged parser, such as a chart parser, to try to understand if the phrases are complete and make sense. Such a parser requires a well-defined grammar and, as one of our premises is that the excerpts most likely *won't* be syntactically correct, such an approach would fail. Less constrained parsers such as a chunk parser might do better, but their unstructured nature could make it difficult to properly suggest missing sentence elements. Thus, we chose to define a set of simplification / correction heuristics that detect the most common problems and correct them.

Most notably, looking for commas and periods offers a good way to understand how far a phrase should extend to make sense. For instance, if an excerpt's last word are preceded by a period, it will most likely belong to another sentence and can be removed. Similarly, if the excerpt ends just one word before a period or comma, that word needs, most likely, to be included in the excerpt. We took care to recognize special cases, such as commas separating enumerated list elements. Pronouns such as "that" and "which" usually separate the main part of a sentence from a relative clause that, while clarifying some meaning, is often not central to the overall subject of the sentence. Such cases, when detected, are subject to a similar treatment.

Creating News Articles. In the possession of a set of meaningful excerpts describing the subject of a cluster of documents, it is now necessary to use those excerpts to create readable texts describing that subject which can be used as a news article. This is not simply a matter of juxtaposing several excerpts, as it would yield incoherent texts. Furthermore, we can take advantage of the article creation process to select the excerpts with greater quality, and include additional information in the articles, gleaned from other sources, if the need arises.

We created a set of news article templates, in which different patterns for possible articles are defined, with blank areas to be filled with document excerpts. Our templates are expressive enough to accommodate a wide range of situations. This is done by allowing the specification of alternate texts to be used depending on properties of the excerpts. For instance, different texts can be chosen depending on the tense of the verb in the excerpt chosen to fill in the following blank. Also, it is

possible to specify, in the excerpt declaration itself, what properties it should have, such as its size, and whether it begins with a noun phrase, verb phrase, etc. All this expressivity allows us to build rich templates that adapt to the wide range of possible excerpts that might arise. Whenever an excerpt is needed to fill in a blank, all excerpts for the cluster that satisfy the constraints specified in the template are considered. The better classified will be chosen. In case of a tie, the excerpt is chosen randomly.

Even with all the versatility we've just described, sometimes the border between the text in the template and that of the excerpts is a little awkward, making the article hard to understand. So, as a final step in the article generation phase, the text resulting from the template is post-processed to improve its quality.

The first step in that process is the use of heuristics to polish the text. For instance, when a noun from the template immediately precedes a verb from an excerpt, it is often the case that inserting *that* or *that is* improves the results. This would transform "We implemented improvements increased the retrieval rate" into "We implemented improvements *that* increased the retrieval rate"

Even after these heuristics are applied, sometimes there are situations still to be solved. So, the second step is to compare the sentences' structures to that of structures known to be valid English sentences, correcting them where necessary. We created a list of possible sentence structures by analyzing all sentences in 18000 news articles from Reuters. A chunk parser was used to find the category of the different words in those sentences. From this data we created a list of sentence structures, ordered by frequency. Whenever a news article needs to be polished, its own words are classified according to their type, and then the Levenshtein distance [10] is used to select structures that more closely match it. If any words are missing, they will be chosen among those in the documents from where the excerpts were taken by looking at where they originally were located and their grammatical type. This yields imperfect but readable texts. One example of a (short) news article created by PersonalNews is:

Work in mesh quality and mesh accuracy

Recently we had new advances in partitioning techniques that are the most popular methods for rendering implicit surfaces, by creating a polygonal mesh.

---

5 Evaluation

To prove the adequacy of our approach and the quality of the personal newspaper, we performed a set of user tests. A total of 18 users tested PersonalNews. Their own personal documents were used. To do so, we visited them at home or working place and, after installing the system on their computers, indexed their documents. As finding users for which large numbers of English language documents accurately describe their interests is not easy, in Portugal, we focused on college students, for which this is true: most, if not all, books, papers, and other materials are provided in English to those students. Each user was asked to perform four different tasks:

1. *Setting-up and launching the application:* the user must run PersonalNews for the first time, configure it and let it index all relevant documents;

	Week			Month			Year		
	$\bar{x}$	σ	%	$\bar{x}$	σ	%	$\bar{x}$	σ	%
num. remembered	3.67	2.11		3.78	2.05		4.33	2.44	
% rem. found PN	3	1.49	**81%**	3.27	1.53	**86.7%**	3.78	1.85	**87.2%**
# found PN only	1.72	1.07		1.56	0.98		1.67	0.77	
Import. rememb.	3.45	1.17		3.44	1.08		3.73	1.02	
Imp.not found PN	2.5	1		2.56	1.01		2.5	1.08	

Table 1. Completeness Results

2. *Identify subjects from different time spans:* the user will start by choosing a particular week, and then writing down all subjects he or she remembers working on at that time, classifying each according to its perceived relevance from 1 (less relevant) to 5. Then PersonalNews is asked to create a newspaper edition for the particular time period and the subjects of the news articles are identified by the user and compared to those previously recorded.
3. *Search for a particular subject:* the user starts by choosing a subject and then requests PersonalNews to create an edition for a time when it is thought that the subject might have been of interest. The articles are then scanned for news about that subject.
4. *Search for a document:* after choosing a particular document, the user tries to employ PersonalNews to find that document by generating an edition in which one of the news might have been created using that it.

While Task 1's main purpose was to introduce the users to the system, the other three allow us to analyze its *completeness* by checking if all subjects found relevant by the user are identified and result in the creation of news articles in which the subject is identifiable. Also, we look at how PersonalNews can help the users *remember subjects* they might have forgotten about. In this capacity, the system will act as a proxy memory for the user. Finally, we evaluate how PersonalNews can work as a *personal document retrieval tool*, to which users can resort to find their documents, based on their subject.

Task 1 – Acquaint users with PersonalNews. The installation was, in all cases, straightforward. The users had only to provide a folder where their personal documents were stored. (often "My Documents"). All users used the default values for the parameters.

Task 2 – Evaluating Completeness. The results for this task are summarized in Table 1. For a week-long interval, the users remembered between 1 and 8 different relevant subjects (avg=3.67, st.dev=2.11). Of those, an average of 3 were identified by the application (sd.dev=1.49). This represents a success rate of 81.81%. Furthermore, the application found up to 4 subjects (avg=1.72, st.dev=1.07) not previously remembered by the users, nearly half of the ones they recalled alone.

Similar results can be found for monthly and yearly periods. As expected, the number of subjects remembered by the users is greater and we found them to be broader in scope for longer periods of time. The percentage of subjects found by the application grows from 81.81% (week) to 86.7% (month) and 87.2% (year). Broader

and more important subjects are more easily inferred by PersonalNews. It is also important to note that while the number of retrieved subjects is around 81%-87%, most of those not found by PersonalNews are the least relevant for the user. Figure 3 depicts this clearly: all subjects of importance 5, and nearly all of 4, are found.

We conclude that PersonalNews is able to find most of the subjects the users find relevant, especially the most important.

Task 3 – PersonalNews as a Memory Proxy. When asked to find a news article from a subject of the users' liking (decided before using the system), we found that this was possible 88.9% of times. Only in two situations there wasn't a relevant article in the newspaper edition generated by the user.

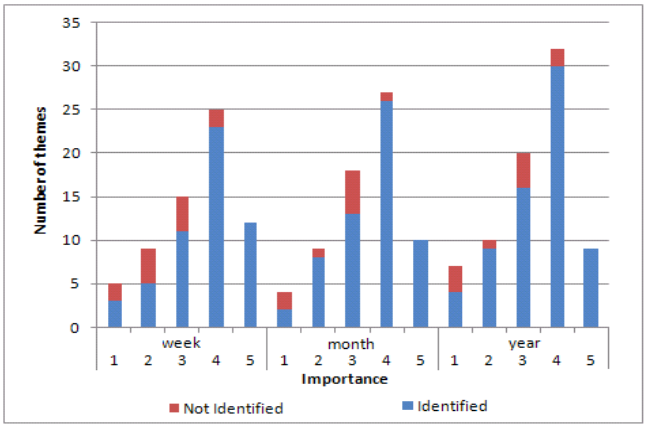

Fig. 3. Subjects found by personal news, by importance

Task 4 – Retrieving Documents. After choosing a particular document, the users then tried to find it using PersonalNews, which entailed creating a newspaper edition for the right period of time, identifying the appropriate article in it, checking the list of documents that were used to create it, and finding the target document in that list. All users managed to do so, for a success rate of 100%. Ten users did it on their first attempt, six had to check the document lists for two articles, and two had to create two newspaper editions to do so.

5.1 Discussion

PersonalNews identifies the vast majority of subjects that users remember and, of those, all they find more relevant. Furthermore, it can even help users remember forgotten information. Also, we see that the users have little trouble understanding the news articles and their subjects. Thus, while is clear that some of the sentences used in the news articles are not perfect, they are nevertheless readable and their contents easily understood. We can state that PersonalNews provides an effective way for users to have a meaningful overview of their lives and interests.

After performing the aforementioned four tasks, the users were then asked to fill in a satisfaction questionnaire. Each question was classified with a score ranging between 1 (Bad) to 5 (Excellent). The users were, mostly, pleased with PersonalNews. Noteworthy are the users' opinions regarding the completeness of the application, (avg=4.17, stdev=0.86), corroborating the results obtained from the four tasks. Users also found it easy to retrieve documents (avg=4.56, stdev=0.92) and, overall, they think the system is useful (avg=4.17, stdev=0.38). To be improved is article legibility (avg=3.67, stdev=0.68), probably due to a less than perfect sentence quality (avg=3.11, stdev=0.58). The users found it adequate but not excellent.

6 Conclusions

One of the problems computers users nowadays face is to deal with the growing amounts of personal information in their computers. Particularly, it is hard to have an overall view of what took place at any given point in time, to both remember and retrieve relevant information.

We presented PersonalNews, a system in which a personal newspaper allows users, at a glance, to have an idea of their interests in a particular time span. We've shown that PersonalNews correctly identifies up to 87% of all subjects of interest to a user, and even helps them find an average of 1.65 subjects they had already forgotten about. Also, it can be used as a personal document retrieval tool and was, in general, well accepted by the users.

In the future, it would be interesting to resort to other information sources, such as emails, instant messaging logs or datebooks to create richer news articles. Also, a better use of natural-language understanding techniques would greatly improve the results. New ways to lay out the articles would merit further study, moving to other technologies that, unlike HTML+CSS, give us better positioning metrics control.

References

1. Bergman, O., Beyth-Marom, R. and Nachmias, R. 2006. The project fragmentation problem in personal information management. In Proceedings CHI 2006, pp. 271-274. Montreal,. ACM Press.
2. Brooks, C., H. and Montanez, N. 2006. An Analysis of the Effectiveness of Tagging in Blogs Proc. AAAI Conf.
3. Deerwester, S., Dumais, S., Furnas, G., Landauer, T., Harshman, R. (1990). "Indexing by Latent Semantic Analysis". *Journal of the American Society for Information Science* 41 (6): 391–407.
4. Dumais, S., Cutrell, E., Cadriz, JJ., Jancke, G., Sarin, R., Robbins, C. D. 2003. Stuff I've seen: A System for personal information retrieval and re-use. Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval.

5. Fertig, S., Freeman, E., Gelernter, D. 1996. Lifestreams: an alternative to the desktop metaphor, Conference companion on Human factors in computing systems.
6. Freeman, E., Gelernter, D. 1996. Lifestreams: a storage model for personal data, ACM SIGMOD.
7. Heyer, L., J., Kruglyak, S., Yooseph, S. 1999. Exploring expression data: identification and analysis of coexpressed genes. Genome Res.
8. Kaufman, L., Rousseeuw, P., J. 1990. Finding Groups in Data: An Introduction to Cluster Analysis. Wiley Series In Probability and Statistics. John Wiley and Sons.
9. Lamming, M., Flynn, M. 1994. "Forget-me-not" Intimate computing in Support of Human Memory. In Proceedings of FRIEND21
10. Levenshtein, V. Binary codes capable of correcting deletions, insertions, and reversals. Soviet Physics Doklady 10 (1966):707–710.
11. Liu, Y., Ciliax, B. J., Borges, K., Dasigi, V., Ram, A., Navathe, S., Dingledine, R. 2004. Comparison of two schemes for automatic keyword extraction from MEDLINE for functional gene clustering. In Proc. IEEE Computational Systems Bioinformatics Conf.
12. Plaisant, C., Milash, B., Rose, A., Widoff, S., Shneiderman, B. 1996. LifeLines: visualizing personal histories, Proceedings of the SIGCHI conference on Human factors in computing systems: common ground.
13. Porter, M.F. (1980) An Algorithm for Suffix Stripping, Program, 14(3): 130-137
14. Ringel, M., Cutrell, E., Dumais, S. and Horvitz, E. 2003. Milestones in time: The value of landmarks in retrieving information from personal stores. To appear in the Proceedings of Interact 2003
15. Salton, G. and M. J. McGill (1983). Introduction to modern information retrieval. McGraw-Hill. ISBN 0070544840.
16. Smith, G., Czerwinski, M., Mayers, B., Robbins, D., Robertson, G., Tan, D. S. 2006. FacetMap: A Scalable Search and Browse Visualization. In IEEE Transactions on visualization and computer graphics vol. 12
17. Viégas, F. B., Golder, S., Donath, J. 2006. Visualizing Email Content: Portraying Relationships form Conversational Histories. In ACM CHI 2006
18. Whittaker, S., Sidner, C. 1996. Email overload exploring personal information management of email. In *Conference proceedings on Human factors in computing systems*, pages 276-283, ACM Press.
19. Xu, R., Wunsch, D. 2005. Survey of clustering algorithms. Transactions on neural networks.