Desktop Context Detection Using Implicit Feedback

Paul - Alexandru Chirita, Stefania Costache, Julien Gaugaz, and Wolfgang Nejdl
L3S Research Center/ University of Hannover
Deutscher Pavillon, Expo Plaza 1
30539 Hannover, Germany
{chirita,costache,gaugaz,nejdl}@l3s.de

ABSTRACT

The personal information stored on the desktop usually reaches huge dimensions nowadays. Its handling is even more difficult, taking into account complex environments and tasks we work with. An efficient method of identifying the present working context would mean an easier management of the needed resources. In this paper we propose a new way of identifying desktop usage contexts, based upon a distance between documents, which also takes into account their access timestamps. We investigate and compare our technique with traditional term vector clustering, our initial experiments showing promising results with our proposed approach.

1. INTRODUCTION

Nowadays, our personal information is mostly in electronic form, spanned over several physical locations, as desktop computers, PDAs, mobile phones, digital cameras, etc. The capacity of our hard drives today, combined with more and more powerful computers and explosion of communications in electronic form – be they emails or documents downloaded from the web – allows us to accumulate on our desktop an overwhelming quantity of personal information coming from several locations. All this information is useless unless we are able to search it at the time we *need* it. As outlined within the PIM'05 Workshop, a "better PIM means a better use of our precious resources (time, money, energy, attention) and, ultimately, a better quality to our lives." This is where Information Retrieval can play a role. In most existing cases, the way the user queries her information is by typing some keywords in a simple search box. The outcome is a long list of, hopefully, ranked results. Yet this also comes with a challenge: To fastly locate the needed information within those results.

People tend to remember of information in terms of associations and context [14]. In this paper we argue that devising specialized algorithms for context detection would allow for a more efficient retrieval. Imagine that among the top-ranked documents of the results list of a search query, there is an email the user knows having sent in correlation with the information she is searching, but the email gives no clue of what or where the sought information is. Being able to retrieve the other documents belonging to the same context as the email would allow to find the searched information faster, thus saving time. This functionality allows us also to no more search for a precise document, but for one or more contexts, which reduces the search space, and allows us to save attention, since context is more natural than content.

Most research on context detection is incorporated in the broader field of desktop search systems, but not focused on context detection as a separate domain. In this paper we investigate several approaches specifically targeted at detecting contexts on the PC Desktop. We start with a brief review of some of the relevant context detection background in Section 2. We then analyze in Section 3 both traditional text-based clustering techniques, as well as new techniques based on usage analysis clustering. Initial results presented in Section 4 are very promising, grouping together files that would have otherwise needed manual processing. However, more tweaking of the parameters determining the output number of clusters is necessary. Finally, we conclude and discuss our on-going work in Section 5.

2. RELEVANT BACKGROUND

Even if the implicit feedback that we receive from various tools was proved to be as accurate as other predicting methods about the importance that one resource has for its user, there is almost no work at all dedicated to using these importance measures to cluster the documents on one's desktop, and use them as working contexts. We present previous work exploring text based clustering, but also try to give a broader overview of the measures that can be extracted about the activities of the user when benefiting of her personal resources.

2.1 Text Based Context Detection

[1] uses text from productivity applications (like word processors, browsers, etc.) to extract keywords representative of the task the user is performing – i.e. the context. Those keywords are then used to pro-actively present the user with documents in relation with her current task, as opposed to our approach which is a static context detection, based on usage activity. Their type of clustering, based on similarity of results' titles and URLs, allows to group very similar documents, such that the user is not overwhelmed with numer-

ous poorly distinct results. We also perform text clustering but with standard algorithms, based on word vectors.

Scatter/Gather is a browsing method for results from a search, presented in [5]. It uses the Fractionation algorithm to perform text clustering on search results and automatically organizes them into a given number of topic-coherent groups. The user can then choose a cluster or a set of them to display only documents belonging to those clusters. Even though [3] presents an innovative browsing technique, it relies on the traditional word vectors which are used as a base for clustering. We could even think of using our new activity distance in a way for Scatter/Gather to take into account the time dimension.

[11] builds upon the idea that if a user organizes her desktop as a "filer", the type of organization can still be seen as a "piler", since the hierarchies built for storing the files on a computer are too complex, so the user would no longer be able to manually locate her data. They suggest using clustering based upon the term vectors of documents to help in reorganizing the resources already stored on the desktop, but also for giving hints where a newly created file can be classified. In our approach, we aim at clustering desktop resources for the purpose of detecting the contexts in which activities are performed on the computer, and use a different cluster identification technique, based upon the access behavior of the user.

An interesting idea on how text clustering can be used on the desktop comes from [13]. They propose to exploit besides vector space models for lexical clustering, the idea of lexical chains, that is the possibility to identify the part where a context is active inside a text document. They also rely on the WordNet lexical reference vocabulary to disambiguate senses of words in different contexts. One of the described applications of their system, QUESCOT, tries to detect topic changes in documents but also, more interesting to us, how is a document relevant to a query from the topics perspective, by determining a context of occurrence for each query concept.

2.2 User Activity and Implicit Feedback

In [10], a broad discussion has been made about the different types of behavioral patterns that can occur on the desktop. Their experiments relied on implicit feedback given by various tools, and mainly based their observations on analyzing the reading time durations, the time that the user spent on one resource. Of course that a more accurate measure would be to monitor attention focus only on specific units within documents, since for example, we would find the relevant background of a paper interesting and not the whole. Most of the examples given rely upon previous work performed in information filtering research, as they have proved that in general there is a strong correlation between the reading time and the importance of a document in the interests area of the user, since the recommendations based on this measure, were very accurate [9, 8, 2] (mostly done for browsing behavior on the web). This helps us in supporting our opinions, since the reading time is the difference between close and open time and we actually log and use these access times. Oard and Kim conclude with the idea that behavior evidence is hardly taken into account in present research, and even less effort is put into combining this with content-based representation, the approach that we envision in this paper.

All these observations were reinforced once again by almost the same type of studies performed in a non-laboratory environment and presented in [4]. They also demonstrated by their web search experiments that the time that a user spends on one document is the most relevant in order to predict a certain satisfaction, as in our case it will be able to predict a stronger appertaining to a context or another. As Fox et al. also mentioned, these observations can be further improved by logging the rare but very meaningful facts that occur on one's desktop, that the user also printed out a copy of the present document as it might mean it is more important to the user, as [7] also suggested, and also that a certain link was added in the bookmarks. A good summarization on implicit feedback measures, categorization and usage can also be found in [6].

Implicit feedback is what [12] uses by analyzing file activities to deduce links between them based on temporal locality – i.e. when they were accessed. These links are then used to provide ranking using three different algorithms, namely Basic-BFS, HITS, and PageRank. We also use implicit feedback but to detect context instead of ranking.

3. CONTEXT DETECTION ON THE DESKTOP

Looking at the previous work, text based clustering is a common method used for context detection. In this paper we propose to also exploit usage analysis (i.e., access timestamps of documents) in the pursuit of this goal. This section will start with a description of the textual clustering methods we considered for desktop context detection, then it will introduce a new activity based document similarity metric, and finally it will exploit this metric for grouping resources on the PC desktop.

3.1 Text Based Context Detection

Using term frequencies coordinates to construct a representational space leads to a good measure of similarity between documents, and we thus considered them as an appropriate input source for desktop context detection. We used the cosine distance to compare the term vectors representing two documents, and then clustered them using both K-means and an agglomerative algorithm with average-link and complete-link distances for the similarity between clusters (three approaches). We chose only these approaches (i.e. we rejected single-link), as they yield more dense clusters.

PC Desktops were defined as containing all data stored by a single user on a personal machine. This includes personal files (HTML, DOC, PDF, PPT, XML, etc.), web cache history, messenger history, emails, but also program generated indexable files (i.e., containing some form of textual content). Upon their full text, a standard text preprocessing technique was taken: tokenization, removal of stop words, and then stemming. Finally, the obtained word list needed to be further pruned, since it would have been computationally too expensive to apply clustering on a term vector space with so many dimensions, the word list reaching almost 300,000 stems. This is also partially due to the fact that the text extractors generated a lot of noisy words (e.g., by sticking some words together). We thus followed the approach of Yang and Pedersen [15], and pruned our word list by document frequencies, keeping only the words with the

highest number of occurrences, as they were proved to be better for aggressive dimension reduction than those with low document frequencies. We used 15 as the minimum DF for the pruning threshold, which reduced the list of words to around 1,000 entries. The newly obtained list was used to compute the TF vectors for each document, the input for the clustering algorithms. The YALE[1] system provided us with the functionalities and algorithms implementations needed to perform the clustering. The results of these methods will be presented in Section 4.

3.2 Activity Based Context Detection

In order to apply a standard clustering algorithm exploiting implicit feedback, we first need to define a distance between documents which exploits this information. The main idea of our activity-based distance is that if two files are *often* accessed in a *small window of time*, the distance between them should be small. One obvious parameter of the distance is then the time $t(a_f, b_g) = |t(a_f) - t(b_g)|$, elapsed between two file accesses a_f and b_g to files f and g respectively. We also consider an additional parameter, namely the number of steps $s(a, b)$ between accesses a and b. If we note s_x the position of access x in the file access sequence, then $s(a_f, b_g) = |s(a_f) - s(b_g)|$. For example, if we consider the sequence of file accesses $a_f \rightarrow c_h \rightarrow b_g$, then $s(a_f, b_g) = 2$. We therefore propose an activity-based distance between file accesses, and define it as following product:

$$d(a_f, b_g) = t(a_f, b_g)s(a_f, b_g)$$

Before exploiting this file access distance to measure distances between files, let us first observe that if two files f and g are accessed one time, separated by a given number of steps, they should have a distance lower than if they were accessed a larger number of times separated by the same number of steps. Furthermore, a file can obviously be accessed more than one time. For example, let us consider two distinct accesses a_f and b_f to the same file f. If we use Equation 3.2 as a distance between files, then $d(f, f) = d(a_f, b_f) = t(a_f, b_f)s(a_f, b_f) > 0$, which is therefore not a valid distance. To avoid this, we introduce a new function $d'_i(f, g, \sigma)$, defined as follows: Consider the log of all file accesses. Let us assume, without loss of generality, that among f and g, f appears first in the log. Also, let us denote a_f^i as the i^{th} access to file f if $i \leq n_f$, or $a_f^i = N$ if $i > n_f$; where n_f is the total number of accesses to file f, and N is the total number of access logged. We can now define:

$$d'_i(f, g, \sigma) = \begin{cases} d(a_f^i, a_g) & \text{if } \exists\, a_g | s(a_f^i) \leq s(a_g) < s(a_f^{i+1}) \\ & \wedge\, s(a_f^i, a_g) = \sigma \\ 0 & \text{otherwise} \end{cases}$$

After defining the function $occ(f, g, \sigma)$ as the number of times file g is accessed σ steps after f, we can finally derive our activity based distance, which is the sum of the closest file access distances weighted by the inverse of their occurrences:

$$D(f, g) = \sum_{\sigma} \left[occ(f, g, \sigma)^{-1} \sum_{i} d'_i(f, g, \sigma) \right]$$

[1] `http://yale.cs.uni-dortmund.de/`

Having this distance in place, we applied an agglomerative clustering algorithm with both complete- and average-link. We could then cut the tree at different heights in order to obtain any number of non overlapping clusters representing contexts.

4. EXPERIMENTS

We evaluated how the newly introduced activity based context detection methods perform when compared to text based one. The experiments were performed only on a small scale (i.e., the authors of this paper), within the research environment of L3S. All file accesses (open, create, etc.) on the desktop were logged during several months of normal activity, with an installed activity logging tool[2]. In the end, the log files were used to cluster their resources based on our new methods.

We experimented on several values of the number of clusters that were supposed to exist on the desktop. In order to make sure the resulted contexts are humanly apprehensible, we fixed the average size of a cluster at 20 documents, which yielded a fixed number of clusters. As only a low percentage of the total number of files on the desktop are actually touched by the user, the activity based methods yielded a lot less clusters. Therefore, in order to increase the quality of our experiments, we also experimented with limiting the text clustering only to the user accessed files. To summarize, we used the following methods:

1. Text clustering over all documents
2. Text clustering over touched documents
3. Activity clustering over touched documents

We first analyze the text clustering performed on the whole set of documents on user's desktop, namely Method 1 as presented above. When manually inspecting the obtained clusters, we could easily see that the agglomerative clustering output one or two huge clusters and the rest were grouped in smaller ones (1-5 documents), which would not be the expected outcome, a relatively uniform distribution of resources in clusters. The K-means version, produced a slightly better repartition among clusters. In the larger clusters however, we could observe that there were several clearly outlined sub-clusters that were grouped together. We believe this to be the consequence of imposing the number of clusters (to K), and thus better results might be obtained after a better definition of K.

The activity clustering method performed almost the same as K-means applied with Method 1, meaning that the documents in the test set were distributed more equally among the clusters than text clustering, and the larger clusters also had one or two sub-clusters inside. If we compare Methods 2 and 3, clustering on the set of the touched documents, we could observe an encouraging behavior of both clustering methods. They managed to cluster files that normally were not so easy to connect, such as papers with the presentations about them, etc.

We plan to present at workshop time more results in order to sustain the idea that the activity clustering performs similarly to the textual one. This is very promising, as the activity clustering does not receive any kind of information about the content of the files. More, the activity clustering brings several advantages, as it filters out uninteresting

[2] We thank Leo Sauermann from DFKI for implementing this logger.

documents from the user's work space, while also reducing the dataset considerably. Finally, combinations of these two sources of information will probably result in significant improvements over each of them taken alone.

5. CONCLUSIONS AND FUTURE WORK

In this paper we proposed to use usage analysis as an input source for clustering desktop documents. We defined a distance between documents, taking into account the number of steps between consecutive accesses of files and a time window in which they occur, and applied to it agglomerative with complete and average link clustering methods. Our initial experiments showed promising results, our proposed usage analysis based approach performing similarly to textual clustering, even though it has *no* information about the textual content of documents.

As future work, we plan to derive time coordinates for the files based on their activity distances, allowing us to apply clustering algorithms necessitating also coordinates (e.g. K-means). We also intend to apply more complex clustering algorithms, such as a fuzzy K-means approach, which would allow for one file to belong to different clusters in a certain proportion. Finally, combining text and activity based clustering would also be an interesting idea to investigate, as well as performing a more semantic-based clustering (e.g., based on the locations of these files on the PC Desktop).

6. REFERENCES

[1] J. Budzik, K. J. Hammond, and L. Birnbaum. Information access in context. *Knowl.-Based Syst.*, 14:37–53, 2001.

[2] M. Claypool, P. Le, M. Wased, and D. Brown. Implicit interest indicators. In *Proc. of the 6th ACM IUI Intl. Conf. on Intelligent User Interfaces*, 2001.

[3] D. R. Cutting, J. O. Pedersen, D. R. Karger, and J. W. Tukey. Scatter/gather: A cluster-based approach to browsing large document collections. In *SIGIR*, 1992.

[4] S. Fox, K. Karnawat, M. Mydland, S. Dumais, and T. White. Evaluating implicit measures to improve the search experience. In *Workshop on Implicit Measures of User Interests and Preferences, SIGIR 2003*, 2003.

[5] M. A. Hearst and J. O. Pedersen. Reexamining the cluster hypothesis: Scatter/gather on retrieval results. In *SIGIR*, 1996.

[6] D. Kelly and J. Teevan. Implicit feedback for inferring user preference: a bibliography. *SIGIR Forum*, 37:18–28, 2003.

[7] J. Kim, D. Oard, and K. Romanik. User modeling for information access based on implicit feedback. In *Technical Report*, 2000.

[8] J. A. Konstan, B. N. Miller, D. Maltz, J. L. Herlocker, L. R. Gordon, and J. Riedl. Grouplens: applying collaborative filtering to usenet news. *Commun. ACM*, 40:77–87, 1997.

[9] M. Morita and Y. Shinoda. Information filtering based on user behavior analysis and best match text retrieval. In *SIGIR '94: Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, 1994.

[10] D. W. Oard and J. Kim. Modeling information content using observable behavior. In *Proceedings of the 64th Annual Meeting of the American Society for Information Science and Technology*, 2001.

[11] D. E. Rose, R. Mander, T. Oren, D. B. Poncleon, G. Salomon, and Y. Y. Wong. Content awareness in a file system interface: implementing the "pile" metaphor for organizing information. In *SIGIR '93: Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*, 1993.

[12] C. A. N. Soules and G. R. Ganger. Connections: using context to enhance file search. In *SOSP*, 2005.

[13] M. A. Stairmand. Textual context analysis for information retrieval. In *SIGIR '97: Proceedings of the 20th annual international ACM SIGIR conference on Research and development in information retrieval*, 1997.

[14] J. Teevan, C. Alvarado, M. S. Ackerman, and D. R. Karger. The perfect search engine is not enough: A study of orienteering behavior in directed search. In *In Proc. of CHI*, 2004.

[15] Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In *ICML*, 1997.