

Tracking Resource Usage Using Heterogeneous Feature Spaces with Local Exceptions

Tilman Steinberg, James Ford, Yi Ouyang, Li Shen, Yuhang Wang, Wei Zheng, and Fillia Makedon

DEVLAB, Dept. of Computer Science, Dartmouth College, Hanover NH 03755, USA
{tilmann, jford, ouyang, li, wyh, zhengwei, makedon}@cs.dartmouth.edu

Abstract. We describe a method for tracking usage of resources in a web environment by representing the resources as a feature space in which usage instances appear as trails, so that common access patterns are shown as similar trails. To improve the analysis of usage data, prior knowledge about usage patterns can be specified by the server administrator as *local exceptions* to the standard dissimilarity measure, or *local dissimilarity measures*, which can be combined in non-linear ways. We outline a method to find similar trails under user-defined restrictions within this heterogeneous feature space and an associated similarity space, and to describe progressions of trails representing a web access history. This information can then be used to reassign resources, e.g. by a web site administrator, to address deficiencies or shifting priorities in accessing web-based materials.

1 Introduction

Tracking the usage of web-based information is important for the optimal allocation of resources to serve the information quickly and efficiently. However, simple log analysis can hide evolving trends by missing subtle deviations from the norm, so it is beneficial to incorporate prior knowledge in the analysis so that expected usage is de-emphasized, while new or unusual usage stands out immediately. Furthermore, it is beneficial to characterize web access by the amount of resources being used, such as the number and size of files delivered, or the number and size of database requests required to serve a page. These characteristics can be used to describe a web site as a feature space, i.e. a high-dimensional space in which *points* correspond to a combination of characteristics, and the access to pages are represented as *trails* (series of points in time) within this feature space. Similarity of these trails are given by *dissimilarity measures*, e.g. the Euclidian distance function for feature spaces homomorphic to Euclidian space. Common access patterns can now be viewed as a *cluster of trails*, while unusual accesses are *outliers* to these clusters (see Figure 1). It might even be useful to view “outlier” points of a trail as alerts: the more a user’s access deviates from the norm, the sooner an alert is raised to restrict this type of access, e.g. when accessing image data without loading the containing page, thus avoiding other items on that page such as advertising banners.

In this paper, we describe a method to easily incorporate prior knowledge about trails in feature spaces, or in this application, user access patterns for a web site. Instead of specifying instances of objects that the user would like to have considered more similar or dissimilar than the defined global dissimilarity measure (i.e. the default method of measuring similarity for all objects), we allow the user to specify regions in which custom dissimilarity measures apply, and all objects that fall within such regions are subject to these *local dissimilarity measures*. This allows for data-independent specification of prior knowledge, i.e. the prior knowledge can be saved or shared without including actual data.

Based on this prior knowledge, pattern similarities are redefined in order to hide or de-emphasize anticipated usage, and to bring out trends that are of high interest to the server administrators. This can also be used to redistribute server resources (e.g. deciding which component should be upgraded), to detect unwanted access (the above example of directly loading images), or to adjust web site behavior depending on user input (e.g. by delivering brief abstracts for users browsing through the site quickly, vs. showing detailed information to users searching with very specific keywords).

Example: Local dissimilarity can apply in the case of a software development web page that lists projects, the project leaders, and project members (see Figure 2); the web site administrator wants to distinguish between visits that go to a project page by way of the different project leaders, but not between visits to a project page following links on pages of different project members. Assuming an encoding of the web pages for each project leader and project member, this can be achieved by defining local dissimilarity measures for the pages of the project leaders and project members as follows:

- in the region containing pages of project leaders, trails become more different
- in the region containing pages of project members, trails become less different

In effect, the *similarity distance* of trails (a relation $d(t_1, t_2)$ returning values representative of the similarity between trails t_1 and t_2) between user access patterns visiting the project leaders is increased, while the similarity distance of trails for the project members is decreased. Furthermore, it is easy to make changes, e.g. to include interns or staff among project members, by simply changing the region to include the respective representations of the additional pages.

2 Related Work

Understanding user behavior on Web sites enables site owners to make sites more usable, ultimately helping users to achieve their goals more quickly. Chi, Heer and Rosien [1-3] have devised methods for categorizing user sessions to reveal users' interests and information needs. Their techniques build user profiles by combining users' navigation paths with other data features, such as page viewing time, hyperlink structure, and page content.

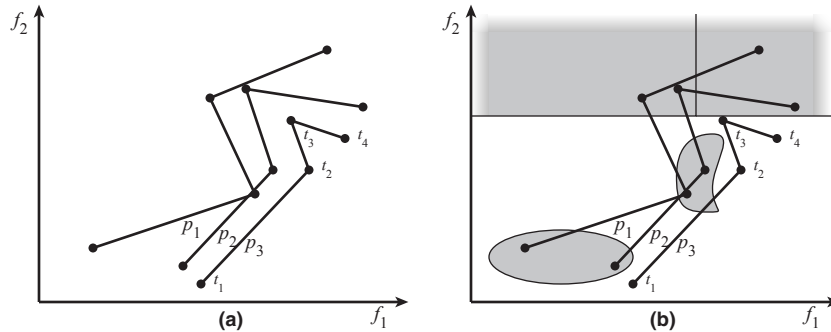


Figure 1. Local exceptions to dissimilarity measures can change similarity between trails. In (a), trails p_2 and p_3 are similar (i.e. they have the smallest distance between individual points), and denote a common pattern whereas p_1 is considered an outlier. In (b), where prior knowledge is added, trails p_1 and p_2 are more similar as their points each fall into local dissimilarity regions (gray areas) which emphasize similarity (dissimilarity measure returns smaller values), whereas p_3 falls outside.

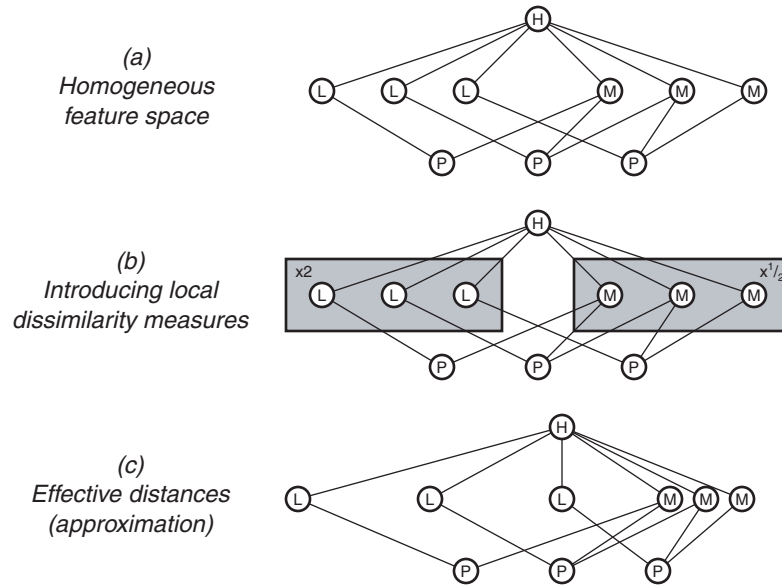


Figure 2. Defining different local dissimilarities for multiple regions. Project leaders (L) are effectively made more dissimilar, while project members (M) are made more similar. Since the dissimilarity is defined for a region, new objects are automatically made more similar or dissimilar when placed within such a region.

Previous work on high-dimensional similarity searches has taken into account that practical applications of high-dimensional feature spaces (i.e. representations of diverse sets of concurrent observations) can have *local heterogeneities*. The primary solution to this problem has been to split up the feature space into locally manageable areas, or to find subspaces where local rules can apply.

Many efforts have been put on mining various patterns concerning Web access [4] [5], which can be viewed as sequential access activities as captured by Web server logs. Previous efforts have tried to employ techniques of *sequential pattern mining* to discover Web access patterns from access analysis [6].

Agrawal and Srikant [6] defined *sequential pattern mining*, a technique to discover frequent patterns in a sequence database, where each sequence is a list of transactions ordered by transaction time, and each transaction consists of a set of items. The goal is then to find all sequential patterns with a user-specified minimum support, where the support is the number of data sequences that contain the pattern. In [7], Agrawal and Srikant included time constraints, sliding time window, and user-defined taxonomy. In [7], GSP, a *generalized sequential pattern mining* algorithm, is proposed to mine sequential patterns by scanning the sequence database multiple times. However, GSP becomes inefficient and has difficulties with extremely long sequences.

Previous research on frequent pattern mining mainly originates from an apriori paradigm. In this paradigm, a set of candidate patterns are tested to see if each candidate has sufficient support in the database. When the length of a sequence pattern increases, this paradigm will have difficulties in pattern mining. But in reality, it is often the case that Web accessed occur in a long access sequence, which limits the application of the apriori paradigm.

Zaïane et al. [5] presented a knowledge discovery tool named WebLogMiner, which can be used to mine web server log files in three steps: (1) database construction from log files, including cleaning and filtering, (2) multi-dimensional web log data cube construction and manipulation, and (3) data mining on web log data cube and database. Every dimension represents a field with all possible values described by attributes, such as server domain, directory, and file. OLAP (On-Line Analytical Processing) and the data cube structure supply a powerful data mining environment. This work focuses on time-series analysis, which include network traffic analysis, event sequence and user behavior pattern analysis, transition analysis, and trend analysis.

Huang et al. [8] described a cube model to represent Web access sessions for data mining. The session data is organized into three dimensions: component, attribute, and session. The component dimension represents a session as a set of ordered components $\{c_1, c_2, \dots, c_p\}$, in which each component indexes the i th visited page in the session, and the attributes dimension define the attributes associated with each component. The advantages of the cube model are that, after representing sessions in a regular data structure, many existing data mining methods can be applied, while the order of page sequences are maintained, and more page attributes for different analysis purposes can be easily included. The k -modes algorithm (a variant of the k -means algorithm for clustering categorical

data) is used to cluster categorical sessions; this employs a transition frequency matrix to validate session clusters.

Fu et al. [9] proposed attributed-oriented induction to generalize sessions according to the page hierarchy and a hierarchical method to cluster generalized sessions based on the observation that the web pages are not randomly scattered but have a hierarchical structure. Replacing pages with their corresponding general place in the page hierarchy, the attribute-oriented induction of sessions can reduce the dimensionality of a session. Generalized sessions are then clustered by using the BIRCH algorithm.

Chakrabarti and Mehrotra [10] developed a technique for applying existing local dimensionality reduction methods that takes into account local correlations in the data, noting that global dimensionality reductions can have problems using local correlations, either yielding wrong results or not using the local correlations to their full advantage (for example, data is correlated along two independent axes). They also used an indexing structure using the local correlations to support range and nearest-neighbor queries.

Other previous research work for time-series data in multidimensional feature spaces include: Puuronen et al. [11] use strategic splitting of the feature space to identify the best feature subset for each instance, and use decision trees with local feature selections. Apte et al. [12] use the dissimilarity measures for a feature space to split it into regions with distinct characteristics. Vlachos et al. [13] describe non-metric similarity functions based on longest common subsequence matching techniques for object trajectories in two- and three-dimensional space. Gionis et al. [14] reduce dimensionality by applying “locally sensitive” hashing functions to points in feature space, grouping those points that are close within the feature space. Keough et al. [15] create an approximation of the original data by replacing actual time series data with simpler series of mean-point and end-point tuples that can then be used for indexing.

Gordon [16] surveys methods to specify prior knowledge in the form of constrained classification where class membership of objects is based on similarity. Klein et al. [17] aim to take spatial clusters into account when adding constraints, noting that with specification of spatial constraints, outperforms instance-based constraint clustering.

3 Translating Characteristics of Web Resources into Feature Spaces

In this section, we represent a site and its associated resources as a feature space, i.e. a high-dimensional space in which *points* correspond to a combination of characteristics of the site and its resources. User access to a site is then represented as a *trail* of points going through this feature space.

3.1 Definitions

We describe a user’s visit to a web site as a time series $T = \{t_1, t_2, \dots, t_n\}$ of events, each characterized by a number of k features $\{f_1, f_2, \dots, f_k\}$ whose values

may change from event to event. These features are the basis for a *feature space* $F = (f_1, f_2, \dots, f_k)$, a k -dimensional space where each dimension describes one feature or characteristic, or a part of a feature (e.g. size of a file, number of database requests, time to assemble a page). The user’s visit and the time series T then become a *trail*, a time series of points in feature space, corresponding to the measurements taken from observations at successive time points.

A *dissimilarity measure* is a mapping from two points $x, y \in F$ or trails $T_1, T_2 \subset F$ to a value describing how dissimilar the two points or trails are. (For trails of different length, the minimal dissimilarity between equal length sections of the trails is taken.) We distinguish between the *global dissimilarity measure* $d_g(x, y)$, which is the default mapping for all point or trail pairs, and user-defined *local dissimilarity measures* $d_R(x, y), x, y \in R$, which are defined for pairs within a user-specified region R only. For pairs with only one point in R , the global dissimilarity measure applies.

The *similarity space* is a companion high-dimensional space created from a feature space such that the similarity space has desirable properties (e.g. homogeneity) suitable for advanced similarity searches, when the feature space lacks these properties.

3.2 Defining the Feature Space

First, we need to create a feature space that represents web resources. To make this feature space meaningful, we consider a set of *attributes* for each page or resource (e.g. images) requested by a user:

- an identifier for each page (e.g. spatial coordinates, in order to group related pages, such as “products” vs. “service”, according to the site’s structure)
- an identifier for the referring page (to ensure proper usage of the site)
- characteristics for each page or component (type, size, value, etc.)
- number of files that are part of the page request and sent to the user
- size of files (individual, average, total) of the page request
- number of requests to secondary resources (e.g. a database server) and the size of request yields from secondary resources
- total time needed to serve the complete page, including requests to secondary resources (this is especially useful for dealing with access during peak times)
- time since last request from this user (i.e. reading time)
- time since this page was last requested by this user
- number of times this page was requested by this user

A sample encoding for such attributes is given in Table 1.

Most of these attributes can be extracted from standard web server logs; however, others require additional logging facilities, e.g. tracking use of secondary resources such as databases would necessitate additional code in the server scripting language (e.g. PHP [18]) to calculate and save the relevant information.

Defining the feature space often needs a normalization step so that values for different features are of the same order of magnitude, e.g. consider “number of files” vs. “size of files” where small changes in the latter would outweigh changes in the former even though the former are more significant.

Table 1. Sample attributes for feature space and their encoding.

Attribute	Description of Feature	Attribute Type
page_id	page identifier	2-dim. array of integer
page_size	number of bytes sent	integer
page_delay	time until next page accessed	float
created_time	time server needed to create page	float
page_cost	value of page to site owner	float

3.3 Tracking Access

User access is monitored by capturing values for each of the above listed attributes, and thus creating a trail in feature space that describes which pages were accessed and how many resources were used to create these pages. In doing this, it is useful to have a mechanism built into the web server to identify the same user (e.g. using session information as generated by PHP or JSP); furthermore, for using secondary resources such as a database, it is helpful to create detailed logs that list how many such resources were used to serve each page.

We assume that by identifying the user by IP address and reading the referrer information, we can get an accurate representation of the user’s access of the site and thus one trail in feature space. The exact mechanism would depend on the site, e.g. by requiring a login on each visit, bookmarks or using the “back” button become less of an issue; while an open site would have to deal with visits starting at random pages. Other attributes that may put the user’s access into a resource-consumption context include e.g. the time of day and served load.

4 Local Exceptions: Incorporating Prior Knowledge

In this section, we describe how a web resource owner can customize the tracking process by defining regions in feature space where similarities are defined differently, effectively emphasizing or de-emphasizing differences in trails going through these regions. The net result is that common and adequately covered requests can be grouped together, whereas unusual requests will stand out more.

4.1 Encoding Prior Knowledge as Local Dissimilarity Measures

Following our paradigm, the website administrator can define regions in feature space for which a *local dissimilarity measure* is used instead of the *global dissimilarity measure*. Examples for regions are:

1. *ranges or intervals* across one or more features, e.g. page size is larger than 10KB, time between requests is between one second and five minutes
2. *individual selection* through boolean functions with features as inputs, corresponding to arbitrarily-shaped regions of feature space

3. *user-specified regions*, e.g. all pages in the “project members” section which contain references to more than three projects

Local dissimilarity measures map any two points within their region to a value describing the differences between the attributes of the two points. For pairs of points that do not both lie within the same region, the global dissimilarity measure applies, such as the Euclidian distance.

In order to distinguish among different accesses to a software project page, we first represent the web pages by their location in a hierarchical tree arrangement (see Figure 2 (a)). We then define two regions L and M , one for the project leaders and one for the project members (see Figure 2 (b)), and for each region, a local dissimilarity measure.

Let $d_g(x, y)$ be the global dissimilarity measure, e.g. Euclidian distance for two pages x and y . We want to emphasize the differences between project leaders, so for region L , we define the local dissimilarity measure d_L as

$$d_L(x, y) = 2d_g(x, y) . \quad (1)$$

On the other hand, we want to de-emphasize differences between project members, so for region M ,

$$d_M(x, y) = \frac{1}{2}d_g(x, y) . \quad (2)$$

Figure 2 (c) shows the effect of the local dissimilarity measures with regards to distances between trails through project leaders and project members.

4.2 Finding Access Patterns via Similarity Search in Feature Space

An *access pattern* can be viewed as a “tunnel” in feature space through which trails representing user accesses run. For example, going from an introduction page to the main menu does not leave much variation (even with a “static” page, possible sources of variation remain, such as the time spent on the page).

To define regions and local dissimilarity measures, the web site administrator can create representative trails by performing predefined visits of the site and specifying how much variation is expected along each step. Each such sample creates a region in feature space, for which the administrator would then specify a dissimilarity measure mapping all pairs of trails within the region to a smaller difference. All user visits that follow a representative trail are then treated as being very similar for the purpose of clustering.

Instead of creating complete sample visits, the web site administrator can also create partial trails to capture usage of particular sections of the web site.

5 Experimental Results

The proposed framework is being demonstrated with experiments involving a popular website [19] with about 500,000 page requests from approximately

80,000 visitors in 2003, to show how certain access patterns can be emphasized over others. Further experiments to demonstrate how this system handles retrieval is still work in progress.

From the logs, we created a rudimentary set of trails by assuming that requests from the same IP address within a time window (page requests on the same day) came from the same user and constituted a distinct visit. (The more detailed custom log data detailed above, using session information and details on database requests, was not yet available as this would have required rewriting crucial parts of the website.) From the access log, we extracted the identifier of the page (translated into coordinates in a hierarchical tree representation) and, for subsequent page requests, the time since the last request.

We grouped the visited pages into two categories: index pages (e.g. those named “index.php”, “home.php”, or with paths ending in “/”) and content pages (all other pages ending in “.php”). For the purpose of this analysis, we assumed that a visitor is reading a page if the next page request is made within one and fifteen minutes (i.e. the page is left up on the visitor’s browser window long enough to allow for more than just quick browsing, but is not “abandoned” because the visitor is doing something else). To determine these visits, a region was defined for the page identifiers and the time since the last request, and all points within this region defined to be very similar. The query trail for finding the visits as described is then a trail of three points, all located inside the region. The global and local dissimilarity measures in this example are:

$$d_g(x, y) = d(x, y) \tag{3}$$

$$d_R(x, y) = \frac{1}{1000} d_g(x, y) \tag{4}$$

Figure 3 shows the associated feature space and the region for the content pages. Figure 4 shows two plots of sorted visit lengths (number of page accesses on y -axis, visits on x -axis): the top plot is not distinguishing between pages and time between page requests; the bottom plot shows only those visits where the user spent between one and fifteen minutes on at least three consecutive pages categorized as “content pages”. As Figure 4 shows, the number of (desirable) visits with users reading content pages is lower than the raw numbers indicate, but still a significant fraction.

5.1 Logging a Web Site for Features

For web sites driven by scripting languages such as JSP or PHP, it is usually easy to insert code to create additional logging information:

- time to assemble each page (read current time in microseconds at the beginning and the end of the page, and log the difference)
- sum of the file sizes of included or embedded media
- number and size of database requests (ideally, database requests are handled in wrapper functions to be independent of the actual database system used; these can easily be extended to update counter and size variables)

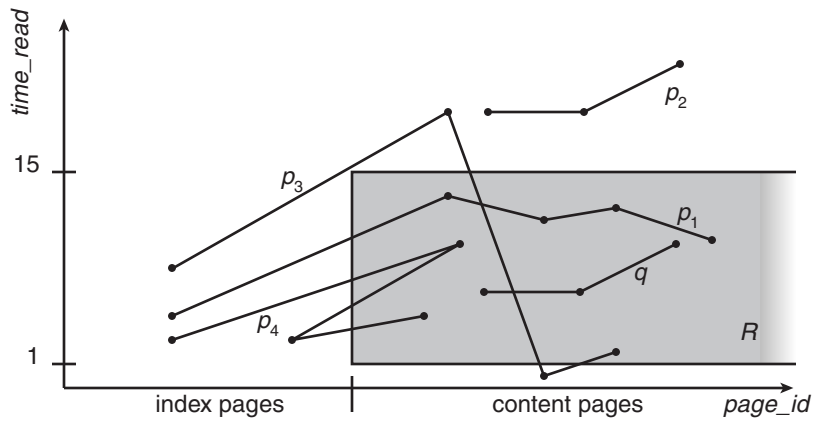


Figure 3. Feature space including page identifier and time spent reading a page, region R (shaded) for read content pages, and sample trails p_1 through p_4 that represent visits to a web site. q is the query trail that represents 3 consecutive accesses of content pages with at least 1 minute and at most 15 minutes between requests. Only trail p_1 is similar to q as it alone has three points inside R . While p_2 is a trail visiting the same pages as q , the reading time is outside the defined boundaries, and consequently p_2 is not considered a match.

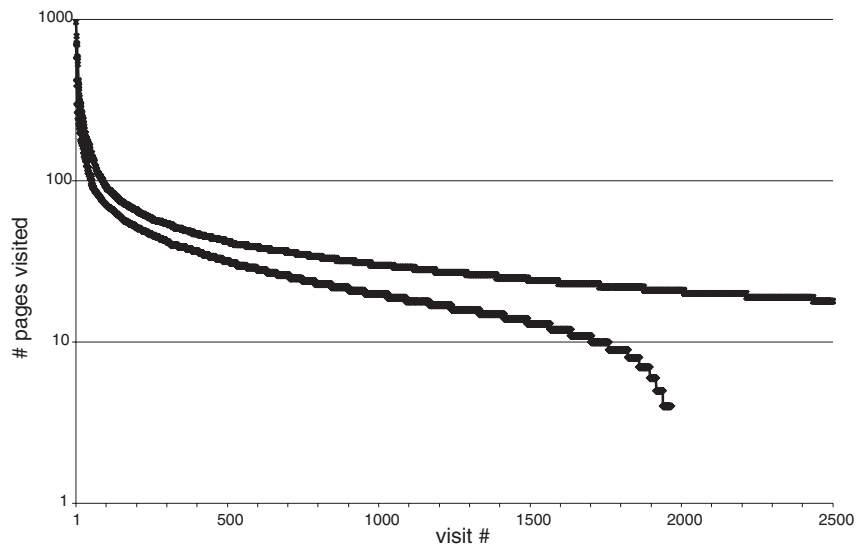


Figure 4. Visits ordered by number of pages visited. The top line shows all visits, the bottom line only those visits that include at least 3 consecutive accesses of pages with content and between 1 and 15 minutes between requests for these pages.

- for variable length pages: the number of components or items that the page contains (e.g. number of products found in a search)
- pages marking the start or end of a session (i.e. login and logoff) create new visit identifications, so that subsequent sessions are treated as multiple visits

6 Conclusion and Future Work

We have described a framework for customizable feature spaces that can be used for tracking time series data, such as users’ access to a web resource. The advantage of this approach is the simplicity with which prior knowledge can be expressed and applied for refining similarity searches (and their converse, outlier searches). Ideally, web site administrators could combine known characteristics of their web site with instances of user access trails they deem representative for specific access patterns, i.e. the regions and corresponding local dissimilarity measures would need to be computed from these instances.

This framework has originally been developed for tracking medical conditions [20], and ongoing work includes experiments to verify the applicability and efficiency of similarity queries under various conditions, and extending our method to allow for trails of different lengths and with missing or extra points.

The current framework assumes that the regions of local dissimilarity measures are disjoint. This can be extended to handle overlapping regions. In that case, we maintain a tree consisting nodes representing each region. The level of the node then corresponds to their priority. When two regions overlap, only the local dissimilarity measure of the region with higher priority is applied. To compute the distance of two data points in such feature space, we traverse the tree to find the applicable dissimilarity measure.

6.1 Using Ontologies to Build a Feature Space

Ontologies are a standard formalism for describing concepts of data objects. By building an ontology of the web site’s pages, it is possible to convert a series of page accesses to a chain of concepts that the user followed. The ontology represents a categorization of the web site’s pages, so that pages with a common purpose fall into the same category. As categories are labeled with concepts (e.g. product information vs. product ordering vs. software download vs. personal information), web pages formerly distinct by URL now have additional attributes as assigned by the ontology and can be grouped accordingly.

7 Acknowledgements

This work has been supported in part by NSF IDM 0083423 (Mining Human Brain Data: Analysis, Classification, and Visualization), NSF IDM 0308229 (Data Management of Protected Information for Data Sharing and Collaboration), NSF IIS 0312629 (ITR: A System for Data Integration and Pattern Discovery in Multimodal Spatio-Temporal Data: Lesion Analysis and Data Sharing), and ISTS (Institute for Security Technology Studies) funding.

References

1. Heer, J., Chi, E.H.: Identification of web user traffic composition using multi-modal clustering and information scent. *Proc. of the Workshop on Web Mining*, SIAM Conf. on Data Mining, pp. 51–58, Chicago, IL, 2001.
2. Chi, E. H., Rosien, A., Heer, J.: LumberJack: Intelligent discovery and analysis of web user traffic composition. *Proc. of WEBKDD 2002*, Lecture Notes in Computer Science 2703, pp. 1–16, Edmonton, Canada, 2002.
3. Heer, J., Chi, E. H.: Mining the structure of user activity using cluster stability. *Proc. of the Workshop on Web Analytics*, SIAM Conf. on Data Mining, Arlington, VA, 2002.
4. Cooley, R., Mobasher, B., Srivastava, J.: Data preparation for mining world wide web browsing patterns. *Knowledge and Information Systems*, 1(1), pp. 5–32, 1999.
5. Zaïane, O. R., Xin, M., Han, J.: Discovering web access patterns and trends by applying OLAP and data mining technology on web logs. *Advances in Digital Libraries*, pp. 19–29, 1998.
6. Agrawal, R., Srikant, R.: Mining sequential patterns. *11th Int'l. Conf. on Data Engineering*, pp. 3–14, Taipei, Taiwan, 1995.
7. Srikant, R., Agrawal, R.: Mining quantitative association rules in large relational tables. *Proc. of the 1996 ACM SIGMOD Int'l. Conf. on Management of Data*, pp. 1–12, Montreal, PQ, Canada, 1996.
8. Huang, J. Z., Ng, M. K., Ching, W.-K., Ng, J., Cheung, D. W.-L.: A cube model and cluster analysis for web access sessions. *WEBKDD 2001*, pp. 48–67, 2001.
9. Fu, Y., Sandhu, K., Shih, M.: Clustering of web users based on access patterns. *Proc. of the 1999 KDD Workshop on Web Mining*, Springer: San Diego, CA, 1999.
10. Chakrabarti, K., Mehrotra, S.: Local dimensionality reduction: a new approach to indexing high dimensional space. *Proc. of 26th VLDB*, Cairo, Egypt, 2000.
11. Puuronen, S., Tsymbal, A., Skrypnik, I.: Advanced local feature selection in medical diagnostics. *13th IEEE Symposium on Computer-Based Medical Systems (CBMS'00)*, Houston, TX, June 23–24, 2000.
12. Apte, C., Hong, S. J., Hosting, J., Lepre, J., Pednault, E., Rosen, B.: Decomposition of heterogeneous classification problems. *Intelligent Data Analysis*, 1998.
13. Vlachos, M., Kollios, G., Gunopulos, D.: Discovering similar multidimensional trajectories. *Proc. of 18th ICDE*, pp. 673–684, San Jose, CA, 2002.
14. Gionis, A., Indyk, P., Motwani, R.: Similarity search in high dimensions via hashing. *Proc. of 25th VLDB*, Edinburgh, Scotland, 1999.
15. Keogh, E., Chakrabarti, K., Mehrotra, S., Pazzani, M.: Locally adaptive dimensionality reduction for indexing large time series databases. *Proc. of ACM SIGMOD Conf. on Management of Data*, pp. 151–162, Santa Barbara, CA, 2001.
16. Gordon, A. D.: A survey of constrained classification. *Computational Statistics & Data Analysis*, 21, pp. 17–29, 1996.
17. Klein, D., Kamvar, S. D., Manning, C. D.: From instance-level constraints to space-level constraints: making the most of prior knowledge in data clustering. *Proc. of the 19th Int'l. Conf. on Machine Learning*, Sydney, Australia, 2002.
18. PHP: Hypertext Preprocessor, <http://www.php.net/>.
19. The Ancient Olympic Games Virtual Museum. <http://www.greecom.org/olympics/>.
20. Steinberg, T., Ford, J. C., Wang, Y., Makedon, F. S.: Similarity searches in heterogeneous feature spaces. *Proc. of Mathematical Methods and Computational Techniques in Electrical Engineering*, Athens, Greece, 2003.