Visualization of Navigation Patterns on a Web Site Using Model-Based Clustering

Igor Cadez[*] [†]
icadez@ics.uci.edu

David Heckerman[*]
heckerma@microsoft.com

Christopher Meek[*]
meek@microsoft.com

Padhraic Smyth[‡]
smyth@ics.uci.edu

Steven White[*]
j-stevew@microsoft.com

ABSTRACT

We present a new methodology for visualizing navigation patterns on a Web site. In our approach, we first partition site users into clusters such that only users with similar navigation paths through the site are placed into the same cluster. Then, for each cluster, we display these paths for users within that cluster. The clustering approach we employ is model based (as opposed to distance based) and partitions users according to the *order* in which they request Web pages. In particular, we cluster users by learning a mixture of first-order Markov models using the Expectation-Maximization algorithm. Our algorithm scales linearly with both number of users and number of clusters, and our implementation easily handles millions of users and thousands of clusters in memory. In the paper, we describe the details of our technology and a tool based on it called WebCANVAS. We illustrate the use of our technology on user-traffic data from msnbc.com.

Categories and Subject Descriptors

H.2.8 [**Information Systems**]: Database Management—*Data Mining*; I.2.6 [**Computational Methodologies**]: Artificial Intelligence—*Learning, Miscellaneous*

General Terms

Model-based clustering, sequence clustering, data visualization, Internet, Web

1. INTRODUCTION

[*]Microsoft Research, Redmond, WA, 98052

[†]Current affiliation Department of Information and Computer Science

[‡]Department of Information and Computer Science, University of California, Irvine, CA, 92717-3425

Arguably one of the great challenges for computer science in the coming century will be the understanding of human behavior in the context of "digital environments" such as the Web. Given our limited experience to date with modeling such environments, there are relatively few existing theories or first-principles to guide any analysis or modeling endeavors. On the other hand, one can readily obtain vast quantities of data from such environments. In this context, data-driven exploration of digital traces (such as server-log records) is clearly an important starting point for furthering our understanding of "digital behavior."

In this paper, we describe a novel approach to visualization and exploratory analysis of dynamic behavior of individuals visiting a particular Web site. As a test bed for our work, we use server logs of individual browsing records for many thousands of individuals or *users* at the msnbc.com site. Our approach is straightforward. First, we first partition users into clusters such that only users with similar behavior on the site are placed into the same cluster. Then, for each cluster, we display the behaviors of the users within that cluster.

The focus of our paper is on the visualization and clustering aspects of such data, rather than on the various engineering issues involved in preprocessing server-log data (identification, "sessionization," etc.). At this point, it is sufficient to assume a fairly abstract characterization of the data—that is, (a) the server-log files have been converted into a set of *sequences*, one sequence for each user session, (b) each sequence is represented as an ordered list of discrete symbols, and (c) each symbol represents one of several possible categories of Web pages requested by the user.

Figure 1 shows a sample of such sequences. The server logs from msnbc.com for a twenty-four-hour period typically produces roughly one million such sequences. The focus of the work described in this paper is the problem of exploring, visualizing, and modeling this type of large data set. There are a number of aspects of the data which make the problem non-trivial.

First, the information is inherently dynamic. Static displays (such as histograms of pages requested) will not fully capture the dynamic nature of the underlying surfing behavior. Thus, we investigate dynamic models (albeit relatively sim-

User	Sequence				
1	frontpage	news	travel	travel	
2	news	news	news	news	news
3	frontpage	news	frontpage	news	frontpage
4	news	news			
5	frontpage	news	news	travel	travel
6	news	weather	weather	weather	weather
7	news	health	health	business	business
8	frontpage	sports	sports	sports	weather
9	weather				

Figure 1: A sample of user sequences.

ple Markov models) to extract dynamic behavior at least to first order. An important point in this context is that these dynamic models serve primarily as vehicles for data exploration and we do not assume that the models necessarily represent the *true* data-generating process. In fact, there is some evidence that Web-surfing behavior may be fundamentally non-Markov in nature ([3]). Nonetheless, just as univariate histograms serve as a useful "data window" for general multivariate data analysis, our use of Markov models in this paper can be viewed as a mechanism to provide at least a partial (and useful) picture of dynamic Web behavior.

Second, dynamic behavior in this general context is highly likely to be quite heterogeneous. A population of users of this size will tend to have vastly different Web-surfing patterns. To address this heterogeneity, we imagine that different users lie in different *clusters*, where each cluster has a different Markov model. Specifically, we model the data as having been generated in the following fashion: (1) A user arrives at the Web site and is assigned to a particular cluster with some probability, and (2) the behavior of that user is then generated from a Markov model with parameters specific to that cluster. We pretend this model generates the Web data we observe, except we do not get to see the actual cluster assignments. We then use a standard learning technique, the Expectation–Maximization (EM) algorithm, to learn the proportion of users assigned to each cluster as well as the parameters of each Markov model. In so doing, we assign each user to a cluster or fractionally to the set of clusters. This approach to clustering is sometimes called a *model-based* approach, and lies in contrast with more commonly used distance-based approaches. The clustering model we learn is sometimes called a *mixture model*. By using a model-based approach to clustering, sequences of different lengths may be assigned to the same cluster (e.g., sequence 1 and 5 in Figure 1 may very likely be generated from a single model corresponding to one of the clusters). This approach provides a natural and consistent mechanism for handling the problem of modeling and clustering sequences of different lengths. Full details of the model and the associated clustering algorithm are discussed [1].

The third non-trivial aspect of the data is its size. Thus, it is critical that any algorithmic technique scale in a reasonable fashion (e.g., linear or near-linear) as a function of the number of sequences N and the number of clusters K. This requirement rules out (e.g.) any direct application of standard hierarchical clustering techniques that scale as $O(KN^2)$ in both time and space complexity. An example of such an approach would be agglomerative clustering using, for example, some form of pair-wise edit-distance between sequences. In contrast, our algorithm for learning clusters of Markov chains (the EM algorithm) requires memory that is linear in N and K and has a runtime *per iteration* that is linear in N and K. In Section 3, we investigate the overall scaling behavior of our approach, and demonstrate by experiment that the *total* runtime of the algorithm (over all iterations) scales linearly in both N and K.

The paper proceeds as follows. In Section 2, we illustrate how the Markov clustering approach can be leveraged to support an interactive exploratory data analysis which provides direct insight into the heterogeneous and dynamic nature of this type of Web data. Section 3 then describes issues associated with learning the clusters, including a demonstration that the (often assumed) near-linearity of the computational complexity of EM-based algorithms holds in our approach. Section 4 concludes the paper with a summary and possible extensions of this work. Related work is discussed in [1].

2. DATA VISUALIZATION

As we have just discussed, our approach to exploratory data analysis is to cluster users and then visualize the behavior of users in each cluster. We have implemented a tool for these tasks, called WebCANVAS (Web Clustering ANalysis and VisuAlization of Sequences). In this section, we illustrate the visualization component of WebCANVAS on data from a large Web site.

The data comes from Internet Information Server (IIS) logs for msnbc.com and news-related portions of msn.com for the entire day of September, 28, 1999 (Pacific Standard Time). Each sequence in the dataset corresponds to page views of a user during that day. In particular, the behaviors of users are not broken down into finer sessions. Each event in the sequence corresponds to a user's request for a page. Requests are not recorded at the finest level of detail—that is, at the level of URL, but rather, they are recorded at the level of page *category*. The categories, developed by one of us (S.W.), are informative yet small in number (seventeen). The categories are `frontpage`, `news`, `tech`, `local`, `opinion`, `on-air`, `misc`, `weather`, `health`, `living`, `business`, `sports`, `summary`, `bbs` (bulletin board service), `travel`, `msn-news`, and `msn-sports`. Although the time of each request is known, we model only the order in which the requests are requested. Furthermore, any requests served via a caching mechanism are not recorded.

The full dataset consists of approximately one million sequences (users), with an average of 5.7 events per sequence. After a bit of experimentation, we found that a random sample of 100,000 sequences was more than adequate for purposes of building clusters useful for visualization. Using this data, we generated 100 clusters as described in Section 3.

Figure 2 shows WebCANVAS's initial display of sixteen of the one hundred clusters. Normally, the clusters are ordered left-to-right and top-to-bottom in descending order by the number of users in each cluster. For this display, however, the order of the clusters are scrambled so as not to reveal

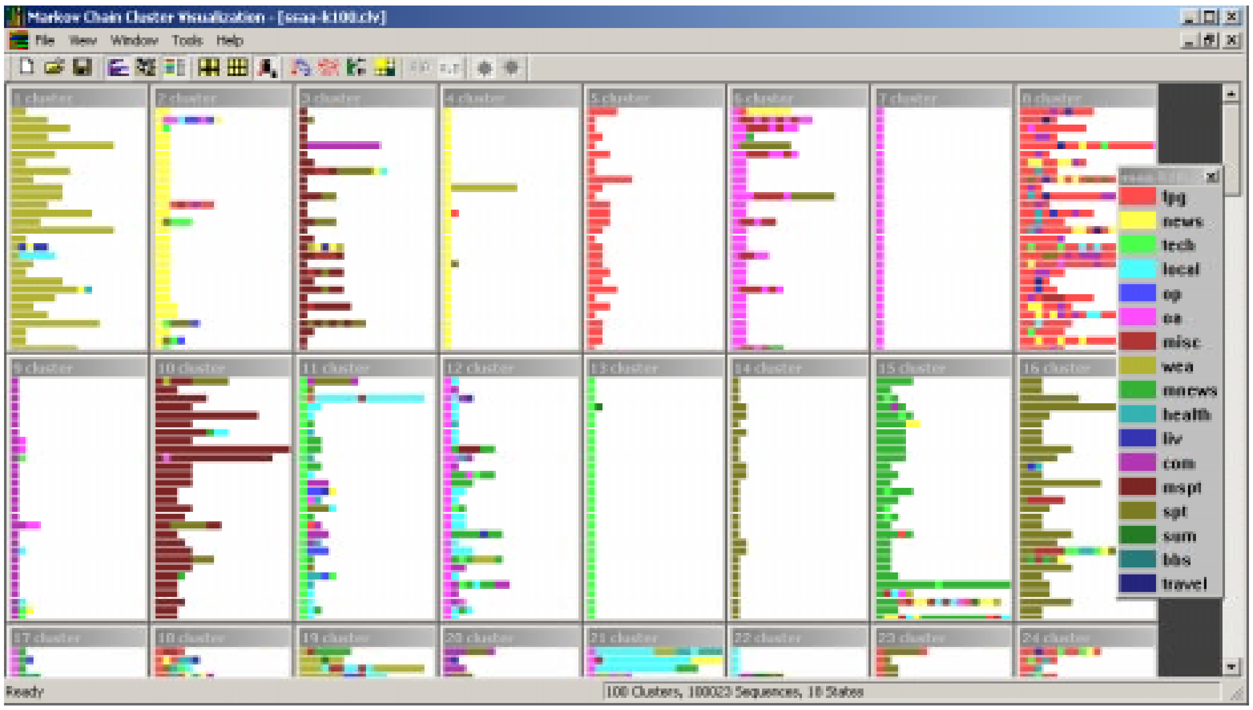

Figure 2: Initial display of msnbc.com data using WebCANVAS. Each window corresponds to a cluster. Each row in a window corresponds to the path of a single user through the site. Each path is color coded (shown here in gray scale) by category. The category legend is at the lower right of the screen.

potentially sensitive information about the msnbc.com site.

Each window corresponds to a cluster. The windows are tiled and each can be easily moved and resized. Each row of squares in a cluster corresponds to a user sequence. Each square in a row encodes a page request in a particular category encoded by the color of the square. (A category legend in shown in the right side of the screen.) For example, the second user in the second cluster has the request sequence `news`, `on-air`, `on-air`, `local`, `opinion`, `opinion`, `on-air`, `opinion`, `news`.

In our experience with WebCANVAS, a site administrator can identify useful and unexpected information after only a few moments of looking at displays such as the one in Figure 2. In this case, the site administrator (S.W.) discovered several interesting facts: (1) there are significantly large groups of people entering msnbc.com on `tech` (clusters 11 and 13) and `local` (cluster 22) pages; (2) there is a significantly large group of people navigating from `on-air` to `local` (cluster 12); and (3) there is surprisingly little navigation between `tech` and `business` sections. Each of these discoveries suggest actions to be taken to improve the site.

Each cluster window can be scrolled so that a site administrator can, in principle, view every user in each cluster. Another feature of WebCANVAS, however, provides the site administrator with a summary of the contents of each cluster. In particular, when an administrator right clicks on a cluster, the corresponding first-order Markov model for that clusters is displayed. Figure 3 shows the display when the user clicks on cluster 8.

Probabilities in the display are encoded by intensity (higher probabilities are brighter). The first column displays the *marginal distribution* of category requests. That is, the first columns shows, for each category, the probability that a user in the cluster will request a page in that category. The second column displays the probability distribution over category requests for the first event. The middle block displays the transition probabilities $p(i, j)$—the probability that a user will request category i followed by category j—for all pairs i and j. Note that, although Markov models are often encoded by conditional probabilities $p(j|i)$, we have found that joint probabilities $p(i, j) = p(j|i)p(i)$ are more informative, because they include the likelihood of first requesting category i.

The first-order Markov model shown in Figure 3 is straightforward to interpret. Users in the cluster start on `frontpage` and then may proceed to almost any other category. Once in a category, users may browse for a while, but return to `frontpage` before moving to another category. Users tend to leave the site from `frontpage` and `business`.

Finally, there will be situations where a site administrator does not care about the order in which requests are made. In this case, we cluster users using a model that reflects this lack of interest—namely, a mixture of zeroth-order Markov models (see Cadez et al., 2000).

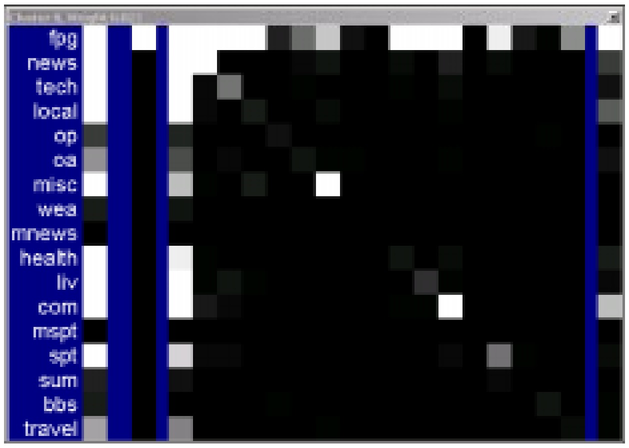

Figure 3: The first-order Markov model shown by right clicking on cluster 8 in Figure 2.

3. CLUSTERING ALGORITHMS AND ISSUES

In this section, we discuss our clustering algorithms and related issues including scalability.

As we have discussed, we construct clusters by learning a mixture of first-order Markov models (or a mixture of zeroth-order Markov models when we are interested in visit order). Each mixture component of the model corresponds to a cluster. For a prespecified number of clusters K, we can learn the parameters (MAP estimates) of the mixture model using the EM algorithm ([2]). To determine the number of clusters, we learn models for many values of K and choose the model that best predicts new or out-of-sample data. In particular, we choose the model with K clusters that minimizes

$$\text{score}(K, \mathbf{D}_{test}) = -\frac{\sum_{j=1}^{N} \log_2 p(\mathbf{X} = \mathbf{x}^j | \theta^K)}{\sum_{i=1}^{N} length(\mathbf{x}^i)} \qquad (1)$$

where θ^K are the parameters obtained from EM, and $length(\mathbf{x}^i)$ is the length of the sequence for user i. Note that this score reflects the average number of bits required to encode a category request made by the user.

We applied this approach to samples of our msnbc.com data. We learned models using a training set of 100,023 sequences sampled from the original one million. We then evaluated the models using the out-of-sample score in Equation 1 on a different sample of 98,687 sequences drawn from the original data. EM was run until the relative change in log likelihood across successive iterations fell bellow 0.0001. All runs, including those described in the next section, were performed on a desktop PC with a Pentium III Xeon processor running at 500MHz with enough memory to avoid paging. In our largest runs with $N = 200,000$ and $K = 200$, only 11.5Mb of memory were used.

Figure 4 shows the out-of-sample scores for first- and zeroth-order Markov models for various values of the number of clusters K. We see that the best first-order model by our criterion is a model with $K = 40$ components. In contrast, the

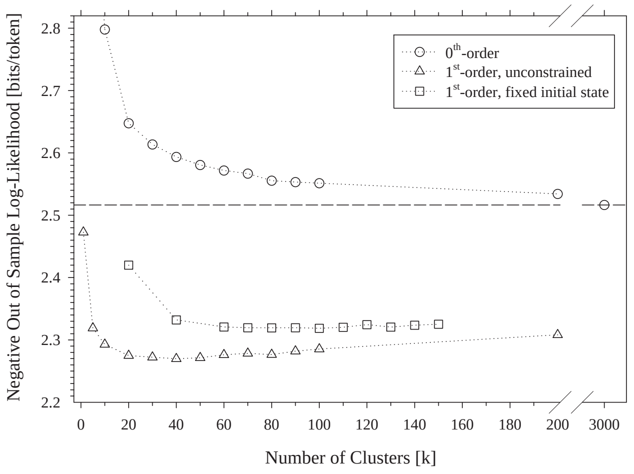

Figure 4: Number of bits (on average) needed to encode an out-of-sample event versus number of clusters K for a mixture of first-order Markov models, the same model constrained so that every user in the cluster has the same first page-category request, and a mixture of zeroth-order Markov models.

zeroth-order models continue to do better as K increases. Of course, when K reaches some value less than N, the number of model components will exceed the number of distinct observations. At this point, the out-of-sample score cannot increase any further with K. One last observation about these two curves is that, for values of K of practical interest ($K < 3000$), the best zeroth-order model is worse at predicting out-of-sample data than the worst ($K = 1$) first-order model.

In learning these models, we encountered a phenomenon that, to our knowledge, has not been reported in the literature. Namely, when learning a mixture of first-order Markov models using this data, we found that two or more clusters often were encoded by a single mixture component. For instance, consider two clusters: a cluster of users who initially request category a and then choose between categories b and c, and a cluster of users who initially request category d and then choose between categories e and f. These two clusters can be encoded in a single component of the mixture model, because the sequences for the separate clusters do not contain common elements.

The presence of multi-cluster components does not affect the predictive power of a model. Nonetheless, when used in conjunction with our visualization tool, the existence of such components are problematic. Specifically, the behaviors of users from more than one cluster are presented in the same window, often confusing or distracting the site administrator. Consequently, there is a need to produce models without multi-cluster components. One method for do so is to run the EM algorithm and then post process the resulting model, separating any multi-cluster components found. A second method is to allow only one state (category) to have a non-zero probability of being the initial state in each of the first-order Markov models.

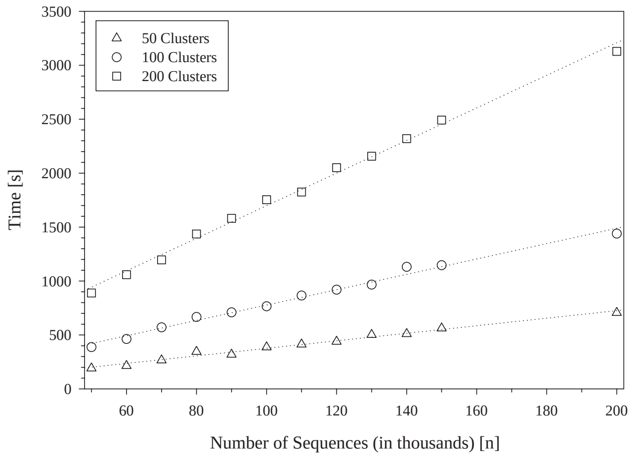

Figure 5: Running time of the first-order Markov Chain clustering algorithm. Dotted lines (linear fit) are included for reference.

Using the second method can have the unfortunate consequence that a cluster of users that have different initial states but similar paths after the initial state are divided into separate clusters. Nonetheless, this potential problem was fairly insignificant for our msnbc.com data. In particular, Figure 4 shows the out-of-sample score for mixture models constrained to have the same first request. We see that these constrained models have a predictive power almost equal to that of the unconstrained models. Of course, when introducing this constraint, more components are needed to represent the data than in the unconstrained case. For this particular data, a constrained model with 100 components has almost the predictive power of an unconstrained model with 40 components. For our visualization in Section 2, we used the constrained model with $K = 100$ to assign users to clusters.

Finally, we turn to issues of scalability. As we have mentioned, the memory requirements of the EM algorithm are linear in the number of sequences N and the number of clusters K. In fact, our implementation can process millions of users and thousands of clusters—all in memory—with RAM sizes that are typical for today's personal computers.

The runtime of the algorithm *per iteration* is linear in N and K. Nonetheless, it is difficult to mathematically characterize how the number of iterations required to reach convergence depend on N and K. When the number of sequences and particularly number of clusters increase, the *shape* of the likelihood surface changes, and new local maxima or saddle points can appear. As a result, the number of iterations required for the algorithm to converge may increases with N or K.

To address this issue, we measured the *overall* runtime of the EM algorithm applied to our dataset for various N and K. We varied the number of sequences from 10,000 to 200,000 and the number of clusters from 10 to 200. Each data point in the graphs represents the time required for our EM algorithm to converge. Figure 5 shows the overall runtime as a function of K and N. The dotted lines represent linear fits through the corresponding data points. These results demonstrate that, at least for the msnbc.com dataset, the EM algorithm scales linearly with N and K.

4. SUMMARY AND FUTURE WORK

We have developed a simple approach for visualizing user behavior on a Web site, and implemented our method in a tool called WebCANVAS. In our approach, we first cluster user behaviors using a mixture of first-order Markov models. We then display the behavior of all users in each cluster along with the first-order Markov model for each cluster. In using first-order Markov models for clustering, we are able to take into account the order in which each user requests pages. In our experience with data from the msnbc.com site, these methods have revealed numerous interesting insights that suggest improvements in the site.

Our use of model-based clustering has the advantage that, we easily can choose statistical models to reflect the criteria by which a site administrator wishes to cluster users. For example, if order of page request is not important, we can cluster users using a mixture of zeroth-order Markov models. In addition, using a model-based approach combined with an EM algorithm for learning parameters, our approach has a memory requirement and runtime that scales linearly in the number of sequences N and the number of clusters K. This scalability makes our approach suitable for applications on extremely large data sets.

Finally, there are several straightforward extensions to our technology. We have already mentioned one: use higher-order Markov models for model components. This extension allows us to model longer range dependencies in the sequence. Another extension is to model the *duration* of each visit. This extension can be achieved by using any number of duration models (e.g., exponential decay) as the components of a mixture model. In another extension, we can use page visits at the URL level, in addition to category information, to cluster users. In particular, we can use Markov models to characterize the transitions among categories as well as the transitions among pages within a given category.

5. ACKNOWLEDGMENTS

We thank Max Chickering for his comments on an draft of this paper. The dataset for this paper was provided by msnbc.com. The work of Igor Cadez and Padhraic Smyth was supported in part by NSF award IRI-9703120.

6. REFERENCES

[1] I. Cadez, D. Heckerman, C. Meek, P. Smyth, and S. White. Visualization of navigation patterns on a web site using model based clustering. Technical Report MSR-TR-00-18, Microsoft Research, Redmond, WA, 2000.

[2] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, B 39:1–38, 1977.

[3] B. Huberman, P. Pirolli, J. Pitkow, and R. Lukose. Strong regularities in World Wide Web surfing. *Science*, 280:95–97, 1997.