Session Boundary Detection for Association Rule Learning Using n-Gram Language Models

Xiangji Huang[1], Fuchun Peng[1], Aijun An[2],
Dale Schuurmans[1], and Nick Cercone[3]

[1] School of Computer Science, University of Waterloo
Waterloo, Ontario N2L 3G1 Canada
{jhuang, f3peng, dale}@cs.uwaterloo.ca
[2] Department of Computer Science, York University
Toronto, Ontario M3J 1P3 Canada
aan@cs.yorku.ca
[3] Faculty of Computer Science, Dalhousie University
Halifax, Nova Scotia B3H 1W5 Canada
nick@cs.dal.ca

Abstract. We present a statistical method using n-gram language models to identify session boundaries in a large collection of *Livelink* log data. The identified sessions are then used for association rule learning. Unlike the traditional ad hoc *timeout* method, which uses fixed time thresholds for session identification, our method uses an information theoretic approach that provides a natural technique for performing dynamic session identification. The effectiveness of our approach is evaluated with respect to 4 different interestingness measures. We find that we obtain a significant improvement in each interestingness measure, ranging from a 26.6% to 39% improvement on average over the best results obtained with standard timeout methods.

Keywords: Web usage mining, language modeling, evaluation.

1 Introduction

The rapidly expanding Web contains a vast amount of data that incorporates useful information waiting to be discovered. Web usage mining is a recently established field that focuses on developing techniques for discovering usage patterns in Web log data, to better serve the needs of Web-based applications. One important Web usage mining problem is to learn interesting association rules from Web logs. Such rules can be used for reorganizing Web sites and making recommendations to facilitate users' browsing activities. However, association rules cannot be conveniently inferred from log entries directly, because these logs usually contain a large amount of irrelevant information and noise. Therefore, to facilitate association rule learning, log entries are usually first grouped into sessions that are defined as a group of user activities related to a common purpose. In this way, session boundary detection forms a useful preprocessing step that itself poses an interesting challenge in Web usage mining.

The goal of session identification is to divide a given sequence of page accesses into individual user sessions. The most commonly used session identification method is *Time Out*. Here, a user session is usually defined as a sequence of requests from the same IP address such that no two consecutive requests are separated by an interval more than a predefined threshold. In [7], experiments were conducted on two sets of Web logs: requests logs and Excite (http://www.excite.com). The requests logs from Reuters (Reuters Ltd.) contain searches on a local version of AltaVista (http://www.altavista.com). In these experiments, the session logs were initially cut with a large session interval, which was then gradually decreased while the distribution of session lengths was concurrently recorded. Based on these experiments, the authors concluded that a time range of 10 to 15 minutes was an optimal session interval length. [6] also reports the results of an experiment where a Web browser was modified to record the time interval between user actions on the browser's interface. One result was that the average time interval between each user event was 9.3 minutes, and that 25.5 minutes was subsequently recommended as the threshold for session identification. This amounts to an assumption that most statistically significant events occurred within 1.5 standard deviations (25.5 minutes) from the mean. However, the the optimal *timeout* threshold depends on the specific problem. Once a site log has been analyzed and its usage statistics obtained, a timeout that is appropriate for the specific Web site can be fed back into the session identification algorithm. Despite the application dependence of the optimal interval length, most commercial products use 30 minutes as a default timeout.

Obviously, a fixed *timeout* strategy is problematic, because users do not normally take a fixed amount of time (i.e. exactly 10 minutes or 30 minutes) for different purposes purpose. People may stay on one topic for several hours or jump to anther topic immediately. Instead of using a fixed time threshold for detecting session boundaries, we propose a new method for dynamically identifying session boundaries. In this paper, we present a method based on statistical n-gram language modeling that addresses the problem of session boundary detection. Our method is based on information theory and provides a natural mechanism for performing dynamic session boundary detection. We present experimental results on a real world dataset which demonstrates its superiority over the traditional *timeout* method.

The remainder of the paper is organized as follows. Section 2 provides a brief description of n-gram language modeling. We then describe how n-gram language models can provide a natural method for identifying session boundaries in Section 3. Third, we describe our method for mining interesting association rules from session data that has already been segmented (Section 4). The rules that are discovered will be used to evaluate our approach. We then present experimental results that demonstrate the effectiveness of our language modeling session detection technique in Section 5. Finally, we conclude in Section 6.

2 *n*-Gram Language Modeling

Traditionally, the dominant motivation for language modeling has come from speech recognition. However statistical language models have recently become more widely used in many other application areas, including information retrieval [8, 10, 12], text classification [11], and now we are applying it for Web mining in this paper.

The goal of language modeling is to predict the probability of natural word sequences, or more simply, to put high probability on word sequences that actually occur (and low probability on word sequences that never occur). Given a word sequence $w_1w_2...w_N$ to be used as a test corpus, the quality of a language model can be measured by the empirical perplexity and entropy scores on this corpus [3]

$$Perplexity = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{Pr(w_i|w_1...w_{i-1})}}$$

$$Entropy = \log_2 Perplexity$$

The goal is to obtain small values of these measures.

The simplest and most successful basis for language modeling is the n-gram model. Note that by the chain rule of probability we can write the probability of any word sequence as

$$Pr(w_1w_2...w_N) = \prod_{i=1}^{N} Pr(w_i|w_1...w_{i-1}) \tag{1}$$

An n-gram model approximates this probability by assuming that the only words relevant to predicting $Pr(w_i|w_1...w_{i-1})$ are the previous $n-1$ words; that is, it assumes

$$Pr(w_i|w_1...w_{i-1}) = Pr(w_i|w_{i-n+1}...w_{i-1})$$

A straightforward maximum likelihood estimate of n-gram probabilities from a corpus is given by the observed frequency

$$Pr(w_i|w_{i-n+1}...w_{i-1}) = \frac{\#(w_{i-n+1}...w_i)}{\#(w_{i-n+1}...w_{i-1})} \tag{2}$$

where $\#(.)$ is the number of occurrences of a specified gram in the training corpus. Although one could attempt to use these simple n-gram models to capture long range dependencies in language, attempting to do so directly immediately creates sparse data problems. Using grams of length up to n entails estimating the probability of W^n events, where W is the size of the word vocabulary. This quickly overwhelms modern computational and data resources for even modest choices of n (beyond 3 to 6). Also, because of the heavy tailed nature of language (i.e. Zipf's law) one is likely to encounter novel n-grams that were never witnessed during training in any test corpus, and therefore some mechanism for assigning non-zero probability to novel n-grams is a central and unavoidable issue in statistical language modeling. One standard approach to smoothing probability estimates to cope with sparse data problems (and to cope with potentially missing n-grams) is to use some sort of back-off estimator.

$$Pr(w_i|w_{i-n+1}...w_{i-1})$$
$$= \begin{cases} \hat{Pr}(w_i|w_{i-n+1}...w_{i-1}), \\ \qquad \text{if } \#(w_{i-n+1}...w_i) > 0 \\ \beta(w_{i-n+1}...w_{i-1}) \times Pr(w_i|w_{i-n+2}...w_{i-1}), \\ \qquad \text{otherwise} \end{cases} \tag{3}$$

where

$$\hat{Pr}(w_i|w_{i-n+1}...w_{i-1}) = \frac{disc\,\#(w_{i-n+1}...w_i)}{\#(w_{i-n+1}...w_{i-1})} \tag{4}$$

is the discounted probability and $\beta(w_{i-n+1}...w_{i-1})$ is a normalization constant calculated to be

$$\beta(w_{i-n+1}...w_{i-1}) = \frac{1 - \sum_{x\in(w_{i-n+1}...w_{i-1}x)} \hat{Pr}(x|w_{i-n+1}...w_{i-1})}{1 - \sum_{x\in(w_{i-n+1}...w_{i-1}x)} \hat{Pr}(x|w_{i-n+2}...w_{i-1})} \tag{5}$$

The discounted probability (4) could be computed using different smoothing approaches including linear smoothing, absolute smoothing, Good-Turing smoothing and Witten-Bell smoothing [5]. In our experiments, we only used Good-Turing smoothing for a preliminary study, although investigating the effects of different smoothing techniques remains an interesting problem.

3 Session Detection Using n-Gram Language Models

Although the original motivation of language modeling is to estimate the probability of naturally occuring word sequences, language modeling actually provides a general strategy for estimating the probability of *any* sequence—regardless of whether the basic units consist of words, characters, or any other arbitrary alphabet. In this sense, many problems can be formulated as a language modeling problem. In Web usage mining, Web pages (or objects) are visited sequentially in a particular order, similar to the word sequences that occur in a natural language. If we consider each visited object as a basic unit, like a word or character in natural language, we can then attempt to estimate the probability of object sequences using the same language modeling tools described above.

The basic goal of session identification is to group sequential log entries that are related to a common topic, and segment log entries that are unrelated. Language modeling provides a simple, natural approach to segmenting these log sequences: First, imagine a set of objects on a common topic that are frequently visited some sequence, one after another. In this case, the the entropy (or perplexity) of the sequence is low. However, when a new object is observed in the sequence that is not relevant to the original topic (but in fact indicates a shift to a new topic), the introduction of this new object causes an increase in the entropy of the sequence because it is rarely visited after the preceding objects. Such an entropy increase serves as a natural signal for session boundary detection. If the change in entropy passes a threshold, a session boundary could be placed before the new object. Put another way, the uncertainty (which is measured by entropy) within a session should be roughly constant, allowing for a fixed level of variability within a topic. However, whenever the entropy increases beyond a threshold, this presents a clear signal that the user's activity has changed to another topic. Thus, we should set a session boundary at the place where the entropy changes. The threshold on entropy change can be tuned to adjust the number of sessions generated. A general principle for setting the threshold is to generate the number of sessions whose average length is in a reasonable range (say, 30 objects). However, more principled ways for setting the threshold could be investigated.

Figure 1 shows the entropy sequence we obtained in part of our Web log dataset, which shows the entropy evolution of the first 10,000 objects. As one can see, the entropy changes radically at some points, although it remains stable in other places. This figure gives an intuition how entropy could be used for session boundary detection.

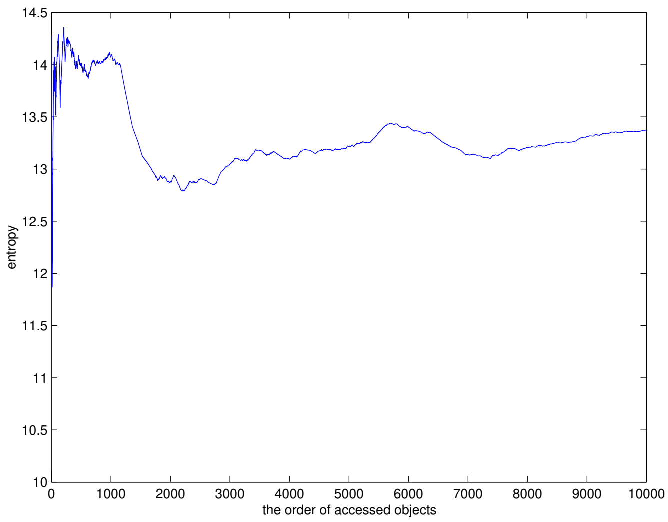

Fig. 1. Entropy evolution in our Web log dataset

4 Mining Interesting Association Rules

We implemented the Apriori algorithm [1] to learn association rules from the pre-segmented Web log data. We then used these discovered association rules to evaluate the quality of our session detection method. As in previous research on Web usage mining, the set of pages to be considered are first identified from all log entries, and then a session file is built upon the identified pages. In our experiments, we initially identified all of the pages involved in the Livelink log files provided to us. Since almost all of the pages in Livelink are actually dynamic, the number of individual pages is huge[1]. However, the problem is not in the number of pages, but in the usefulness of dynamic pages. When we analyzed the discovered patterns that describe access relationships among pages, we found that many of those patterns reveal the *programming patterns* within Livelink. For example, two pages can be found to be always accessed together because one a frame within the other, as defined by the Livelink program. Such patterns were not considered to be interesting by our domain experts. Another feature of this data is that there could be great similarity in the contents of different dynamic pages. That is, in our data two dynamic pages might be considered different, even though they contain the same set of information objects. In order to discover truly interesting and unexpected patterns, we first performed an object identification pass over the the dynamic pages caputured in the data, and then built the session file based on the objects and not the dynamic pages themselves.

The discovered association rules describe the association relationships between the information objects. For example, an association rule

$$\langle o1, o2, o3\rangle \rightarrow \langle o4, o5\rangle \; [support = 0.01 \; confidence = 0.6]$$

means that 1% of the sessions contain objects $o1, o2, o3, o4$ and $o5$, and that 60% of the sessions containing $o1, o2$ and $o3$ also contain $o4$ and $o5$. The number of association rules that are discovered depends on the *support* and *confidence* thresholds. For our dataset, we found that the number of rules generated is not significantly affected by changing the confidence threshold. However, changing the support threshold affects the number of retrieved rules substantially. Table 1 shows how the number of rules varies with the support threshold. From this

Support threshold	0.02	0.01	0.008	0.005	0.003	0.0028	0.0025	0.002	0.001
Number of assoc. rules	2	14	39	88	723	4,556	74,565	4,800,070	>1,000,000,000

Table 1. Number of generated association rules (confidence threshold = 0.5)

table, one can see that a large number of rules can be discovered if the support threshold is set very low. For evaluation purposes, to find interesting rules from

[1] We identified nearly 200,000 pages from the two-month data.

a large number of discovered patterns, we rank the discovered rules according to their interestingness measures and prune out redundant rules based on the structural relationship among rules.

We considered four interestingness measures for the purpose of evaluating our new session detection method. These measures were used to measure the interestingness of an association rule $A \rightarrow B$ [2], as shown below. Using support and confidence to measure the interestingness of a discovered rule is straightforward. However, no interesting rules have been found in our experiments by using support as the interestingness measure. So we choose the confidence as one of interestingness methods for evaluation. The reason why we choose the measures IS, MD and $C2$ is that they are among the best interestingness measures according to our earlier work in [9].

1. $C2$ [4]. The C2 formula measures the agreement between A and B. It has been evaluated as a good rule quality measure for learning classification rules [2]. It can be defined as

$$C2 = \frac{P(B|A) - P(B)}{1 - P(B)} \times \frac{1 + P(A|B)}{2}.$$

2. *Confidence (CS)*. The confidence of a rule or pattern can be expressed as $P(B|A)$. For association rules, $P(B|A)$ means the probability that objects in B occur in a session conditioned on the occurrence of objects in A. With this measure, rules are ranked according to their confidence value as the main key and their support value as the secondary key. Therefore, this measure is denoted as CS.
3. IS [13]. Derived from statistical correlation, the IS measure is defined as

$$IS = \sqrt{\frac{P(AB)P(AB)}{P(A)P(B)}}.$$

IS is designed to be better suitable for the scenario in which the support value of the rule is low.
4. *Measure of Discrimination* (MD) [2]. The MD measure was inspired by a query term weighting formula used in information retrieval and has been used to measure the quality of classification rules [2]. We adopt the formula to measure the extent to which an association rule $A \rightarrow B$ can discriminate between B and $\overline{B}$:

$$MD = \log \frac{P(A|B)(1 - P(A|\overline{B}))}{P(A|\overline{B})(1 - P(A|B))}.$$

All the above-listed measures except MD and $C2$ have been used to measure the interestingness of association rules. MD and $C2$ have only been used to measure classification rules. The values from the *MD and C2* measures can

[2] In association rule $A \rightarrow B$, A and B are sets of objects.

be zero or negative, indicating A and B are not correlated or they are negatively correlated, respectively. In our learning programs, rules with this kind of interestingness values are considered uninteresting and are pruned.

The use of an interestingness measure can help identify interesting association rules by ranking the discovered rules according to the measure. However, it cannot be used to identify redundant rules. By redundant rules we mean that the same semantic information is captured by multiple rules and hence some of them are considered redundant. We use four pruning methods proposed in [9] for pruning redundant association rules (details omitted here).

5 Empirical Evaluation

We now empirically evaluate the effectiveness of our language modeling based session detection method on the Livelink dataset. We first describe the Livelink dataset in Section 5.1 and how the raw data is preprocessed in Section 5.2. Then in Section 5.4 we present the results of association rule learning given segmentations produced by both the traditional *timeout* and language modeling techniques. We then analyze the results in Section 5.5.

5.1 The Data Set

The log files used in our experiments were extracted from Livelink access data over a period of two months (April and May 2002). Livelink is a Web-based system[3] that provides automatic management and retrieval of a wide variety of information objects over an intranet or extranet. The size of the raw data is 7GB. The data set describes more than 3,000,000 requests made to a Livelink server from around 5,000 users. Each request corresponds to an entry in the log files, where each entry contains: 1, the IP address the user is making the request from; 2, the cookie of the browser the user is making request from, which can be as long as 5,000 bytes; 3, the time the request is made and the time the required page is presented to the user; 4, the name of the request handler in the Livelink program; 5, the name of the method within the handler that is used to handle the request; 6, the query strings that can be used to identify the page and the objects being requested, and some other task relevant information, such as URL addresses for error-handling. A sample log entry of Livelink is shown in Figure 2. For privacy and security reasons, some of the lines are removed.

5.2 Data Preprocessing

The objective of data preprocessing is to transform the raw log data into a form that can be used for learning patterns. The following steps are performed to preprocess the data in our investigation: 1, the user is identified from each log file entry; 2, the requested information objects are identified from each entry; 3,

[3] Developed and sold by Open Text Corporation.

```
Wed Apr 10 19:22:52 2002 CONTENT_LENGTH = '0' func = 'll'
GATEWAY_INTERFACE = 'CGI/1.1' HTTPS = 'on' HTTPS_KEYSIZE = '128'
HTTPS_SECRETKEYSIZE = '1024' HTTPS_SERVER_ISSUER = 'C=US, O="RSA
Data Security, Inc.", OU=Secure Server Certi fication Authority'
HTTPS_SERVER_SUBJECT = 'C=CA, S=Ontario, L=Waterloo, OU=Terms of
use at www.cibc.com/verisign/rpa (c)99,
 OU=Authenticated by CIBC, OU="Member, VeriSign Trust Network", O=Open Text Corporation, OU=Network and
 Online Services, CN=intranet.opentext.com'
HTTP_ACCEPT = '*/*' HTTP_ACCEPT_ENCODING = 'gzip, deflate'
HTTP_ACCEPT_LANGUAGE = 'en-us' HTTP_CONNECTION = 'Keep-Alive'
HTTP_COOKIE = 'WebEdSessionID=05CAB314874CD61180FE00105A9A1626; LLInProgress=%2FIOPiEOOD4iNz4iMzk3Py8vIA;
 LLCookie=%2FIOPiEOOD4iNz4iMzk3MHhvZHd%2Fb28hbW9uaWVifyEkaWFuYW8gAA; LLTZCookie=3600'
HTTP_HOST = 'intranet.opentext.com' HTTP_REFERER =
'https://intranet.opentext.com/intranet/livelink.exe?func=doc.Vie
wDoc&nodeId=12856199' HTTP_USER_AGENT = 'Mozilla/4.0 (compatible;
MSIE 5.01; Windows NT 5.0)' objAction = 'viewheader' objId =
'12856199' PATH_TRANSLATED = 'C:\Inetpub\wwwroot' QUERY_STRING =
'func=ll&objId=12856199&objAction=viewheader' REMOTE_HOST =
'24.148.27.239' REQUEST_METHOD = 'GET' SCRIPT_NAME =
'/intranet/livelink.exe' SERVER_NAME = 'intranet.opentext.com'
SERVER_PORT = '443' SERVER_PROTOCOL = 'HTTP/1.1' SERVER_SOFTWARE =
'Microsoft-IIS/5.0' _REQUEST = 'llweb' Wed Apr 10 19:22:52 2002 -
638968       Func='ll.12856199.viewheader'   Timing:.140
0A<1,0,'Number_of_Dele
te_Statements'=0,'Number_of_Insert_Statements'=0,'Number_of_Other_Statements'=0,'Number_of_Select_Statemen
ts'=7,'Number_of_Update_Statements'=0,'OutputTime'=47,'Total_Execute_Time'=16,'Total_Fetch_Time'=31,'Total
_SQL_Statements'=7,'Total_SQL_Time'=47> 04/10/2002 19:22:52
Done with Request on socket 069DC4B0 04/10/2002 19:22:57
Processing Request on socket 09A87EF8
```

Fig. 2. A Livelink log entry

noisy entries are removed (which request no interesting objects); and finally, 4, the log file entries are grouped into sessions according to our language modeling based method outlined above. In this experiment, we use IP addresses to stand for users of Livelink. Even though the same user can log into Livelink through different IP addresses, most often a user accesses Livelink from the desktop in his/her office, and therefore most of the accesses are associated with a fixed IP address. This is actually a safer assumption than using cookies to identify users, because cookies are often disabled. Identifying objects from the large number of dynamic Livelink pages is an unique part of the problem. An object could be a document (such as a PDF file), a project description, a task description, a news group message, a picture and so on. Different types of objects have different domains of identities. Based on Livelink domain knowledge we can extract the identities of the objects being requested from the the query string of the log entry. Most entries contain exactly one object, although some entries contain no objects or multiple objects. We ignore all entries that contain no information objects. The total number of different objects identified from the two-month Web log data is 38,679.

5.3 Session Identification

After the users and objects have been identified from the log entries, we grouped the requests into sessions. In our application, a session is an ordered sequence object sets requested by a user during a single visit to Livelink. In most cases, a session is defined as a group a of actions requested by a single user, where that no two consecutive requests are separated by an interval more than a predefined threshold during a limited time of period for a purpose. The method using this definition for identifying sessions is called the *timeout* session detection method.

There are two kinds of session detection methods used in the experiments. The first one uses the *timeout* method to identify sessions, in which we set the fixed time thresholds to be 5, 10, 15, 20, 25, 30, 35 and 40 minutes in the experiments. The second one uses an n-gram language modeling based method to identify sessions. In the experiments, we set n to be 1, 2 and 3 respectively and the corresponding thresholds are set to be 0.005, 0.003 and 0.0025. We will evaluate these two session detection methods by comparing the number of discovered interesting rules in the top 10, top 20 and top 30 lists generated from the two session detection methods.

5.4 Experimental Results

As shown in Table 1, the number of generated rules greatly depends on the support threshold. At low support regions, a very small change in support threshold can lead to a super exponential growth in the number of rules. To avoid missing interesting rules or generating too many rules, we carefully chose the the support and confidence thresholds for each method in the experiments. The value of confidence is set to be 0.5 for all the language modeling based methods and the *timeout* methods at different time interval. For example, we set the support and confidence thresholds to be 0.0028 and 0.5 for the *timeout* session detection method at the 10 minutes time threshold in the experiments. The number of rules generated under this setting is 4,556. The support thresholds for the standard methods at the other time interval are set to be values that lead to generation of a similar number of rules.

Results of Timeout Method: Our baseline model is the *timeout* approach, which is the standard method currently used in many Web mining research investigations. For this method, we conducted experiments on time out thresholds of 5, 10, 15, 20, 25, 30, 35 and 40 minute thresholds. The results for the top 10, top 20 and top 30 are shown in Table 2, 3, and 4 respectively. The first row in Table 2 is the number of sessions generated under each threshold. The entries in Table 2, 3 and 4 represent the number of interesting association rules discovered by each interestingness measure with different thresholds among the top 10, 20 and 30 [4]. The last two rows are the total number of interesting rules discovered by the 4 interestingness measures and the percentage of interesting rules discovered, which is computed as the number of total interesting rules discovered divided by the total number of generated rules.

[4] All the discovered association rules were evaluated by our domain experts in Open Text.

time intervals	5 min.	10 min.	15 min.	20 min.	25 min.	30 min.	35 min.	40 min.
C2	3	6	6	6	6	6	6	6
CS	2	1	1	4	4	4	4	4
IS	3	5	5	4	5	6	5	5
MD	6	7	7	10	10	8	8	8
Total	14	19	19	24	**25**	24	23	23
Percentage	35%	47.5%	47.5%	60%	**62.5%**	60%	57.5%	57.5%

Table 2. Top 10 results with timeout method for session boundary detection

time intervals	5 min.	10 min.	15 min.	20 min.	25 min.	30 min.	35 min.	40 min.
C2	4	12	11	11	12	11	12	12
CS	3	5	4	8	5	6	6	6
IS	5	10	8	8	8	10	10	12
MD	10	14	14	16	18	18	18	18
Total	22	41	37	43	43	45	46	**48**
Percentage	27.5%	51.25%	46.25%	53.75%	53.75%	56.25%	57.5%	**60%**

Table 3. Top 20 results with *timeout* method for session boundary detection

The best performance obtained in top 10, top 20 and top 30 are **62.5%**, **60%** and **65.83%** under time thresholds 25, 40 and 40 minutes.

Results of Language Modeling Based Method: For the language modeling based methods, we experimented with 1-gram, bi-gram, 3-gram models using Good-Turing smoothing. A different threshold is set for each model to generate roughly the same number of sessions, which is 0.0005, 0.0003, 0.00025 respectively. The results are shown in Table 5, 6 and 7 respectively. The first row of the table is the models used (for example, GT1.5 means 1-gram language model

time intervals	5 min.	10 min.	15 min.	20 min.	25 min.	30 min.	35 min.	40 min.
C2	9	17	16	15	17	19	21	22
CS	5	8	8	12	12	12	12	12
IS	11	18	16	24	18	18	20	19
MD	18	24	24	30	26	26	26	26
Total	43	67	64	74	73	75	**79**	**79**
Percentage	35.83%	55.83%	53.33%	61.67%	60.83%	62.5%	**65.83%**	**65.83%**

Table 4. Top 30 results with *timeout* method for session boundary detection

with Good-Turing smoothing and the entropy change threshold is 0.0005). Other rows are of the same meaning of Table 2. The values of support for association rule learning are set to be 0.0042, 0.0036 and 0.00336 for the models GT1.5, GT2.3 and GT3.25 respectively. Under this setting, a similar number of rules can be generated for all the three models.

LM models	GT1.5	GT2.3	GT3.25
C2	9	9	9
CS	7	2	3
IS	8	10	8
MD	10	10	10
Total	**34**	31	30
Percentage	85%	77.5%	75%

Table 5. Experimental top 10 for language modeling methods

LM models	GT1.5	GT2.3	GT3.25
C2	18	18	18
CS	13	9	4
IS	16	18	18
MD	20	20	20
Total	**67**	65	60
Percentage	**83.75%**	81.25%	75%

Table 6. Experimental top 20 for language modeling methods

In the language modeling based methods, the results obtained for top 10, top 20 and top 30 are **85%**, **83.75%** and **83.33%**.

5.5 Analysis and Discussions

Effects of Different Thresholds in Timeout Method: The standard *time-out* session detection method obviously depends on the time threshold. We find that generally time thresholds between 25 minutes and 40 minutes are good. A threshold that is too small (say, 5 minutes) leads to poor performance. Figure. 3 illustrates the influence of different time thresholds in top 10 results.

Effects of Language Modeling based Method: Table 8 shows the improvements made by the language modeling based method compared to the standard

LM models	GT1.5	GT2.3	GT3.25
C2	27	28	26
CS	15	14	4
IS	26	28	28
MD	30	30	30
Total	98	**100**	88
Percentage	81.67%	**83.33%**	73.33%

Table 7. Experimental top 30 for language modeling methods

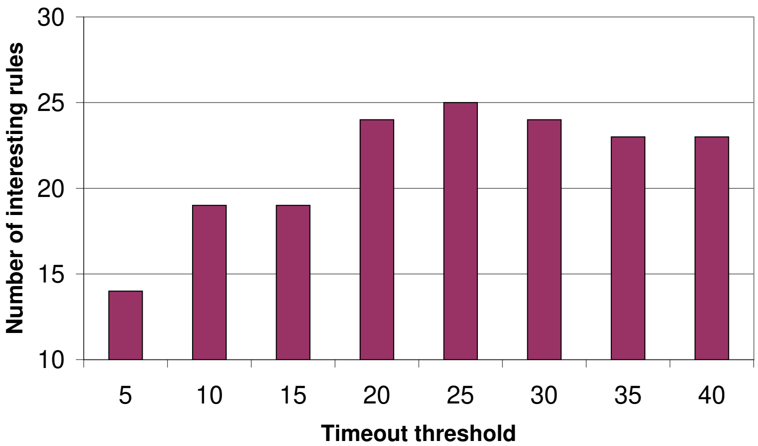

Fig. 3. Timeout method comparison at different thresholds

timeout method. We choose the best result from each method. We observe that a significant improvement can be made for top 10, 20 and 30 results.

It is also interesting to notice that performance of each language modeling based method is much better than the best one obtained in the *timeout* methods. Figure 4 shows the the comparison among the language modeling methods and the best *timeout* method in top 10 results. By looking into each interesting measure, we find that the language modeling based approach consistently outperforms all of the *timeout* methods at the top 10, 20 and 30.

All of these results demonstrate that the language modeling approach is effective at identifying session boundaries for association rule learning.

Effects of different order of n-gram language models: We find that in language modeling based approach, better results are obtained on 1-gram and

	Timeout based	LM based	Improvements
top 10	62.5%	85%	36%
top 20	60%	83.75%	39.6%
top 30	65.83%	83.33%	26.6%

Table 8. Effects of LM based methods

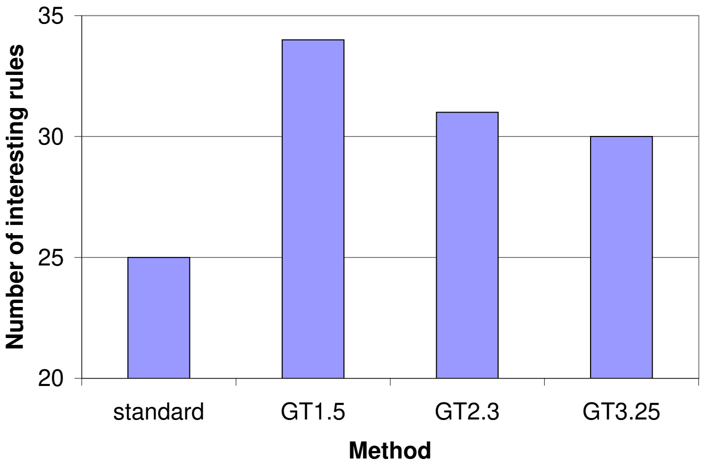

Fig. 4. Comparison of language modeling and timeout methods

2-gram models. One reason is that sparse data problems begin to dominate for n-gram language modeling with longer context (see Section 2). Although we do not have to cope with unseen events in this domain (because we are detecting boundaries on the training set) the statistics one obtains from limited training data is not reliable when there are too many parameters being estimated.

6 Conclusions and Future Work

We have proposed a novel approach for dynamic session boundary detection based on statistical n-gram language modeling. This approach is based on information theory and is intuitively understandable. Experiments on learning interesting association rules from the Livelink dataset show that we obtain consistent improvements over the traditional ad-hoc *timeout* methods when preprocessing the data for association rule discovery.

Our future work includes investigating the optimal order of n-gram language models, the influence of different smoothing techniques in language modeling, and the effect of this approach for other Web usage mining problems, such as sequential pattern mining.

7 Acknowledgements

We would like to thank Gary Promhouse for spending a lot of time evaluating the discovered association rules in the experiments. Without his help and useful feedback, this research cannot be fully conducted. We also would like to thank Open Text Corporation for supporting this research and providing us with its Livelink Web log datasets.

References

1. Agrawal, R. and Srikant, R.; (1994). Fast Algorithms for Mining Association Rules, *Proc. of the 20th International Conference on Very Large Databases*, Santiago, Chile.
2. An, A. and Cercone, N.; (2001). Rule Quality Measures for Rule Induction Systems: Description and Evaluation, *Computational Intelligence*, Vol. 17 No. 3.
3. Bahl, L., Jelinek, F. and Mercer, R.; (1983). A Maximum Likelihood Approach to Continuous Speech Recognition *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(2), pp. 179-190.
4. Bruha, I.; (1996). Quality of Decision Rules: Definitions and Classification Schemes for Multiple Rules. In *Nakhaeizadeh, G. and Taylor, C. C. (eds.): Machine Learning and Statistics, The Interface*. Jone Wiley & Sons Inc.
5. Chen, S. and Goodman, J.; (1998). An Empirical Study of Smoothing Techniques for Language Modeling. *Technical report*, TR-10-98, Harvard University.
6. Catledge, Lara D. and Pitkow, James E.; (1995) Characterizing Browsing Strategies in the World Wide Web, *Proceedings of the 3rd International World Wide Web Conference*, April 1995, Darmstadt, Germany.
7. He, D. and Goker, A.; (2000). Detecting session boundaries from Web user logs, *Proceedings of the 22nd Annual Colloquium on Information Retrieval Research (ECIR)*, April 2000, Sidney Sussex College, Cambridge, England.
8. Hiemstra, D.; (2001). Using Language Models for Information Retrieval. Ph.D. Thesis, Centre for Telematics and Information Technology, University of Twente.
9. Huang, X., An, A., Cercone, N. and Promhouse, G; (2002) Discovery of Interesting Association Rules from Livelink Web Log Data. In *Proceedings of the IEEE International Conference on Data Mining (ICDM)*, December, 2002, Maebashi TERRSA, Maebashi City, Japan.
10. Lafferty, J. and Zhai, C.; (2001). Document Language Models, Query Models, and Risk Minimization for Information Retrieval. In *Proceedings of 24th ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*.
11. Peng, F. and Schuurmans, D.; (2003). Combining Naive Bayes and n-Gram Language Models for Text Classification. submitted to *The 25th European Conference on Information Retrieval Research (ECIR)*.
12. Ponte, J. and Croft, W.; (1998). A Language Modeling Approach to Information Retrieval. In *Proceedings of ACM Research and Development in Information Retrieval (SIGIR)*, 275-281.
13. Tan, P. and Kumar, V.; (2000). Interestingness Measures for Association Patterns: A Perspective, *Technical Report TR00-036*, Department of Computer Science, Univ. of Minnestota.