Integration of Semantic Information to Predict Next Page Request While Web Surfing

Pallavi P. Patil
Department of computer science and Engineering,
Govt. Engg. College Aurangabad, Maharashtra, India

Meghana Nagori
Asst.Prof. Department of computer sceience and Engineering,
Govt. Engg. College Aurangabad, Maharashtra, India

Abstract - In this paper we use Semantic Information as a criteria for pruning states in higher order (where (K>2)) Selective Markov Models and compare the accuracy and model size of this idea with Semantic-rich Markov models and traditional Markov Models and explains how to use semantic information gathered from web application's domain ontology with transition probability matrix of lower order Markov models to predict accurate next page visited by user while web surfing by pruning states in Selective Markov Model and this provide less state and space complexity related to higher order Markov models. It also addressed the problem of ambiguous prediction of next page whole web surfing. To achieve the proposed goal paper introduce the way of generating web logs useful in generating transition probability matrix ,Semantic distance and weight matrix and also states the algorithm for next page access frequency which is useful in frequency pruning.

Keywords - Semantic distance, Domain Ontology, Transition Probability matrix, Markov Model, Web log synthesis, hit count, page access frequency

I. INTRODUCTION

Increasing popularity of the web surfing since last few years created a significant burden upon the internet traffic as is the large distributed information system for accessing shared data; hence in this internet era websites which are easily available with faster response to user's request are considered good sites and become popular. Normally web caching and web prefetching are two approaches used to reduce response time of sites. Web Caching by storing or reusing web objects that are likely to be used in the near future improve the performance of web-based systems and web prefetching deduce the client's future request for web document and make available that document to the cache before an explicit request by user. Many web servers keeps a sever access log of its users. These logs can be used to train a prediction model for future document accesses and this is useful in obtaining frequent access patterns in weblogs and mine association rules for path prediction. Thus the web servers cache performance and latency is depend on predicting users next page request while web surfing. Different web prefetching techniques such as prediction by partial match, Model based prefetching , context based web prefetching etc are available; here we used the way to model the users accessed web pages and edges representing transition probabilities between states computed from the given user sequence in the web log.

In this paper we used the semantic information with the help of domain ontology to predict next page. Integration of semantic information into Markov models for next page request prediction can be done by combining semantic information with transition probability matrix of Markov model provide a low order Markov models and also solve the problem of ambiguous prediction and provide less state, space complexity. Here we use semantic information to prune states in Selective Markov Model by generating web logs, Semantic Distance and Normalized semantic distance matrix and the combining with transition probability matrix and also by using maximum semantic distance and frequency pruning using context aware higher order model with less space and state complexity and compare such model's accuracy and size with semantic-rich Markov model and Traditional Markov models.

Further sections of this paper explains literature survey, weblog synthesis, page access frequency, Semantic distance matrix, Normalized semantic distance matrix, Experimental analysis, result and finally conclusions and future scope of this study.

II. LITERATURE SURVEY

In case of web prefetching techniques, the focus in the last few years has been on Markov Models. In this area lots of research work has been done by different researchers for accurate online prediction of accessing next page.

Pirolli and Pitkow study the prediction power and effects of using higher order Markov Model by using N-gram representation of user access paths. In addition to Borges and Levene this research leads to the use of higher order Markov Model for link prediction. The order of Markov Model corresponds to the number of prior events used in predicting a future event, so a k^{th} order Markov Model predicts the probability of the next event by looking at the past K events. Bestravos used a method that first eliminates the conditional probabilities of transitioning directly from each web page within a time TW based on server log file analysis, this is a first order Markov Model for predicting surfer path. He did not explore the effects of using larger surfer paths in the predictive model. Deshpande and Karypis provide three different criterias to prune states in the model before prediction as frequency, confidence and error pruning. Though Deshpande and karypis provide a class of Markov Model based prediction algorithms that are obtained by selectively eliminating a large fraction of states of the all k^{th} order Markov Model, resulting models have a very low state space complexity and achieve better accuracies; still require work on ambiguous prediction and state space complexity and they did not study the effect and relation of domain knowledge and semantic information on selective Markov Model. Using Markov Model for web prefetching having number of limitations such as if order increases then state space complexity also increases and reducing number of states leads to less predictive power. As a solution to this problem, the all k^{th} order Markov Model was proposed, such that if the k^{th} order Markov Model can not make the prediction then $(k-1)^{th}$ order model is tried. Deshpande and karypis provide the solution by using selective Markov model but it is not feasible for large data sets. In this paper we tried to provide solution for trade-off problem and to make Markov model feasible for large data sets by using semantic information provided by domain ontology; for this have studied web usage mining as it consist of three main tasks preprocessing, pattern discovery, and pattern analysis.

Core ontology with axioms and Semantic Distance Matrix

Nizar R. Mabroukeh and Christie I. Ezeife proposed the integration of semantic information drawn from a web application's domain knowledge into all phases of the web usage mining process (preprocessing, pattern discovery, and recommendation/prediction).SemAware (Semantics-Aware) integrates semantic information into sequential pattern mining, this information is used during the pruning process to reduce the search space and minimize the number of candidate frequent sequences, minimizing as well the number of database scans and support counting processes.

Ddomain knowledge is available in the form of domain ontology provided by the ontology engineer during the design of the web site. A core ontology with axioms is defined by Stumme *et al.* as a structure $O = (C, \leq_C, R, \sigma, \leq_R, A)$ consisting of:

- two disjoint sets C and R whose elements are called concept identifiers and relation identifiers, respectively,
- a partial order $\leq_C$ on C, called concept hierarchy or taxonomy,
- a function $\sigma : R \rightarrow C^+$ called signature (where C^+ is the set of all finite tuples of elements in C),
- a partial order $\leq_R$ on R, called relation hierarchy, and
- a set A of logical axioms in some logical language L

During mapping of web pages to their corresponding classes, semantic distance can be computed and stored in a look up matrix, called the semantic distance matrix M.

III. SYSTEM DEVELOPMENT

A. WEBLOG SYNTHESIS

Weblog is nothing but the user's page visit patterns recorded at the server. When the user exits the session or session expires this page visit log is permanently recorded at the server for future study or analysis. The page visit log can be saved in text files or database tables. The log can be used in various ways like below:

a. Display recently viewed items in an e-commerce website.
b. Find out most searched/visited items/pages.
c. Synthesize page visit trails during bug or website crash analysis.
d. Next page visit predictions.
e. Advertising related items.

However, when this log is to be studied it needs to be cleaned so that redundancy can be reduced or filtered out from the log. This study uses various techniques to programmatically generate the clean weblog. This weblog then to be used for further analysis of the algorithm. The input for the weblog generation is Page Count. The page count needs to be defined at the start of the application. The programmatic generation of weblog uses following models/methods to generate the log.

a. Forward Model: In this model the page fetch log always starts from the first page and assumes other pages are visited after that. The forward model page log for 7 pages looks like in table below. Here the numbers from 1 to 7 represent the page numbers used to uniquely identify the web pages.

Page Fetch Log	Hit Count
1 2 3 4 5 6 7	1952
1 3 4 5 6 7	295
1 2 3 4 7	3218
1 6 7	4319

Figure 1: Forward Model weblog example

b. Reverse Model: Like in forward model, in this model the page fetch log starts from the first page but assumes that user visits back the pages visited earlier. The reverse model page log for 7 pages looks like in table below. Here the numbers from 1 to 7 represent the page numbers used to uniquely identify the web pages.

Page Fetch Log	Hit Count
1 2	4060
1 2 1	4563
1 2 3 4 5 6 5	312
1 2 3 4 5 6 7 6 5 4 3 2 1	1759

Figure 2: Reverse Model weblog example

c. Direct Model: In this model the page fetch log can start from any page in the website. This model is based on an advertising concept where items from an e-commerce site are displayed as advertise on another web site or web application and user clicks on that advertise to visit the respective web page in the e-commerce site. The direct model page log for 7 pages looks like in table below. Here the numbers from 1 to 7 represent the page numbers used to uniquely identify the web pages.

Page Fetch log	Hit Count	Page Fetch Log	Hit Count	Page	Hit Count
1	59155	2 7 6 5 4 3 2 1	4557	7 6 7	4915
1 3	1165	6 7 6 5 4 3 2 1	4661	6 5	2904
1 3 2 1	525	3 4 3 2 1	1892	7 3 4 5 6 7	3945
2 4	2726	6 4 3 2 1	1207	7 2	24
7	280	7 1	8533	7 3 2 1	4251

Figure 3: Direct Model weblog example

d. Alternate Model: This model is based on assumption like user visits pages with only even or odd page numbers. The alternate model page log for 7 pages looks like in table below. Here the numbers from 1 to 7 represent the page numbers used to uniquely identify the web pages.

Page Fetch Log	Hit Count
1 3 5 7	2970
2 4 6	1616

Figure 4: Alternate Model weblog example

e. Random Model: This model is based on an assumption that user may click any related or unrelated page from any page. Two random number generators are used to generate a page visit sequence.

Ideally if all the above mentioned models are used to generate weblog the random model should not have any record as all the page visit logs generated by two random number generators might have already covered and added to the weblog. It is good to have zero records generated in this model because it is a confirmation that all the page visit sequences are considered and no combination is missed the evaluation. Thus, the 7 page website example considered here does not have any record in this model.

$$Record\text{-}Count\ in\ Weblog = (Pc \times Pc \times 3) + (Pc \times (Pc - 6)) + 3$$

....where Pc is Page Count

Below table represents the weblog record count for various page counts.

Page Count	Weblog Record Count
6	111
9	273
25	2353
70	19183
500	997003

Figure 5: Page Count v/s Weblog Record Count illustration

This case study synthesizes three different datasets using the same models as above but not all. Below table gives information about the models used in the three datasets synthesized in this case study.

Data Set	Forward Model	Reverse Model	Direct Model	Alternate Model	Random Model
DS 1	✓	✓	✓	✓	✓
DS 2		✓		✓	
DS 3	✓		✓		

Figure 6: Models used for dataset generation

As the dataset 2 and 3 does not use all the models the weblog record count is less than the count generated using the above formula.

B. SEMANTIC DISTANCE MATRICES AND NORMALIZATION

A random number generator is used to dynamically populate semantic distance between two pages. The maximum semantic distance is a user defined value acting as a limit on semantic distance between two webpages. This study ignores the semantic distance if it is more than the maximum semantic distance defined by the user at start of application. Maximum semantic distance is inversely proportional to the maximum level of relatedness a user would allow between two concepts. The effect of usage of maximum semantic distance on the algorithm is that it removes least used states/relations from the Markov matrix and minimizes the error factor caused due to least used visit logs.

An example of semantic distance matrix for the 7 page website having maximum semantic distance value 6 can be as below. Where the cell value represents semantic distance between any two pages indicated as row header and column header of the table.

	P1	P2	P3	P4	P5	P6	P7
P1	0	3	1	1	1	3	2
P2	2	0	3	5	3	1	1
P3	3	3	0	3	2	4	3
P4	4	2	1	0	1	2	1
P5	2	5	1	4	0	1	2
P6	3	5	5	4	3	0	3
P7	5	5	4	5	1	4	0

Figure 7: Semantic Distance Matrix example

The semantic distance matrix needs to be normalized so that the weight matrix can be calculated. The normalization process is done as below:

$$\frac{M_{p_i,p_j}}{\sum_{k=1}^{n} M_{p_i,p_k}} \quad , \; M_{p_i,p_j} > 0$$

$$0 \quad , \; M_{p_i,p_j} = 0$$

Using the equation above the normalized semantic distance matrix for given example becomes like below:

	P1	P2	P3	P4	P5	P6	P7
P1	0	0.27272727	0.09090909	0.09090909	0.09090909	0.27272727	0.18181818
P2	0.13333333	0	0.2	0.33333333	0.2	0.06666667	0.06666667
P3	0.16666667	0.16666667	0	0.16666667	0.11111111	0.22222222	0.16666667
P4	0.36363636	0.18181818	0.09090909	0	0.09090909	0.18181818	0.09090909
P5	0.13333333	0.33333333	0.06666667	0.26666667	0	0.06666667	0.13333333
P6	0.13043478	0.2173913	0.2173913	0.17391304	0.13043478	0	0.13043478
P7	0.20833333	0.20833333	0.16666667	0.20833333	0.04166667	0.16666667	0

Figure 8 : Normalized Semantic Distance Matrix example

C. TRANSITION PROBABILITY & WEIGHT MATRICES

The transition probability matrix is combined with normalized semantic distance matrix to calculate the weight matrix. The transition probability is constructed for the given pattern at runtime by evaluating the hit count for the given pattern. First of all the summation of hit count for all visit logs starting from the given pattern is calculated as Pattern Frequency (P_{freq}). Then visit logs are evaluated to get the probable next pages and related hit counts in a table, say, NextPageLogs. Summation of hit count in NextPageLogs is multiplied by 100 and divided by P_{freq} to calculate the probability of transition to the next page.

For example, for given pattern '1 2 3' the transition probability is calculated as below:

a. P_{freq} is calculated using query below:

```
DECLARE @PatternFrequency INT
SELECT @PatternFrequency = SUM(HitCount) FROM WebLog WHERE PageFetchLog like '1 2 3%'
```

This results in value 93066

b. The NextPageLogs table is filled using queries below:

```
SELECT LTRIM(SUBSTRING(PageFetchLog, CHARINDEX('1 2 3', PageFetchLog) + LEN('1 2 3'), LEN(PageFetchLog))) AS SubPattern, HitCount INTO #SubPatternLog FROM WebLog WHERE PageFetchLog like '1 2 3%'

SELECT CASE WHEN CHARINDEX(' ', SubPattern) > 0 THEN RTRIM(LTRIM(SUBSTRING(SubPattern, 0, CHARINDEX(' ', SubPattern)))) ELSE SubPattern END AS NextPage, HitCount INTO #NextPageLog FROM #SubPatternLog WHERE LTRIM(RTRIM(SubPattern)) != ''
```

s

These queries results in table as below:

Next Page	Hit Count
4	3361
5	4684
2	4647
4	3370
4	2899
6	4530
7	3236

Figure 9: Next Page search weblog example

c. The final transition to next page probability is calculated using query below:
SELECT NextPage, SUM(HitCount) * 100 / @PatternFrequency AS ProbabilityPerCent FROM #NextPageLog GROUP BY NextPage ORDER BY NextPage DROP TABLE #NextPageLog DROP TABLE #SubPatternLog

This query results into the final transition probability for the given pattern. Rest of the missing page transitions are assumed as zero.

Next Page	Probability%
2	7
4	76
5	5
6	4
7	3

Figure 10: Next Page probability example

d. This Probability Percentage is further divided by 100 for the calculations and displayed in the Transition Probability matrix as below:

Pattern	P1	P2	P3	P4	P5	P6	P7
1 2 3	0	0.07	0	0.76	0.05	0.04	0.03

Figure 11: Transition Probability Matrix example

The Weight matrix is calculated by combining the transition probability matrix with semantic distance matrix. Each non-zero entry in the normalized semantic distance is subtracted from 1.
The weight matrix for pattern in discussion here is as below:

Pattern	P1	P2	P3	P4	P5	P6	P7
1 2 3	0.833333	0.903333	0	1.593333	0.938889	0.817778	0.863333

Figure 12: Weight Matrix example

Thus, the weight matrix finally gives the probable next page user is going to request. The highest the weightage the highest the probability. So, based on this, server can fetch the page and cache it so that it will be served is much lesser wait time.

D. PAGE ACCESS FREQUENCY

Access frequency of each page is determined using formula below and used in further analysis of the algorithm where the frequency pruning is done. The frequency pruning is explained in upcoming section.

$$F_{req}(P_i) = 100 \times (\sum \text{hit count of all web logs having } P_i) / \sum (\text{all hit count})$$

…, where P_i is any page in the weblog.

An example of page access frequency can be as below:

	Page Access Frequency
P1	74.0323944
P2	69.17289
P3	67.77826
P4	64.212944
P5	59.14268
P6	67.9342346
P7	76.4108

Figure 13: Page Access Frequency example

IV.EXPERIMENTAL ANALYSIS

a. System Test

This section describes how to use the application to ensure the results of the algorithm.

BasicInputs

Page count and Maximum Semantic Distance are two basic inputs required to start the algorithm analysis. After specifying page count, maximum semantic distance probable values are calculated based on following equation.

If (PageCount / 2) > 5 Then

PageCount – 1 >= Max (SemDist) >=5

Else

PageCount – 1 >= Max (SemDist) >= PageCount / 2

…, consider only quotient and ignore reminder of division PageCount/2

After selecting appropriate web page count and max semantic distance user has to click on the "Generate Web Log and Semantic Distance Matrix" button to generate the web log, semantic and normalized-semantic distance matrices, transition probability and weight matrices as well.

Result Summary section gives result of the analysis done against the selected weblog dataset. The summary includes comparison of Traditional Markov model, Semantic Markov Model and Frequency Pruned Semantic Markov Model. The result summary is displayed as below:

b. Manual Test

During the manual test user is given chance to specify a page visit pattern so that the algorithm can find out the next probable page that will be visited based on the weblog analysis.

The Semantic Distance, Normalized Semantic Distance and Page Frequency matrices remains almost same during manual test.

V. PERFORMANCE COMPARISON FOR DIFFERENT INPUT SIZES

Below are some examples of the performance test of the algorithm. Here, for first three tests the Maximum Semantic Distance is approximately 75% of the total number of pages. And for later three tests the Maximum Semantic Distance is less than 50% of the total number of pages.

Pages = 10 & Max Semantic Distance = 8

	Model	Accuracy Data Set 1	Size Data Set 1	Accuracy Data Set 2	Size Data Set 2	Accuracy Data Set 3	Size Data Set 3
1	First Order TMM	22.91667	48	100	9	17.5	40
2	First Order SMM	29.16667	48	100	9	22.5	40
3	First Order Frequency Pruned SMM	27.08333	48	100	9	10	40
4	Second Order TMM	60.41667	48	100	9	57.5	40
5	Second Order SMM	62.5	48	100	9	60	40
6	Second Order Frequency Pruned SMM	52.08333	48	100	9	55	40
7	Third Order TMM	91.11111	45	100	9	81.57895	38
8	Third Order SMM	91.11111	45	100	9	81.57895	38
9	Third Order Frequency Pruned SMM	91.11111	45	100	9	81.57895	38
10	K Order TMM	92.63158	190	95.65218	69	100	118
11	K Order SMM	92.63158	190	95.65218	69	100	118
12	K Order Frequency Pruned SMM	87.36842	190	95.65218	69	100	118

Pages = 20 & Max Semantic Distance = 15

	Model	Accuracy Data Set 1	Size Data Set 1	Accuracy Data Set 2	Size Data Set 2	Accuracy Data Set 3	Size Data Set 3
1	First Order TMM	27.80269	223	100	37	17.87709	179
2	First Order SMM	29.59641	223	100	37	17.31844	179
3	First Order Frequency Pruned SMM	27.35426	223	100	37	16.75978	179
4	Second Order TMM	59.19283	223	100	37	53.63129	179
5	Second Order SMM	31.83857	223	100	37	55.86592	179
6	Second Order Frequency Pruned SMM	13.45292	223	100	37	48.60335	179
7	Third Order TMM	97.69585	217	100	37	96.53179	173
8	Third Order SMM	97.69585	217	100	37	83.81503	173
9	Third Order Frequency Pruned SMM	89.40092	217	100	37	71.6763	173
10	K Order TMM	99.18734	2338	95.94595	666	99.25017	1467
11	K Order SMM	99.18734	2338	89.63964	666	99.25017	1467
12	K Order Frequency Pruned SMM	97.4337	2338	81.23123	666	43.69461	1467

Here we can notice that semantic-rich 1st-order Markov models have the same model size as regular 1st-order models, this is because no pruning is used in semantic-rich models. While these semantic-rich models solve the problem of contradicting prediction, they also provide very close accuracy to that of regular 1st-order models. This accuracy differs depending on the nature of the sessions in the data sets. It can be found that the Frequency Pruned Semantic Markov Models are almost accurate with less number of states examined.

VI. CONCLUSIONS

1. Conclusions

Semantic information provided by domain ontology used with Markov transition probability matrix to generate semantic weight matrix given solution to ambiguous prediction of next page while web surfing and by using page access frequency and idea of maximum semantic distance; this paper provide the pruning criteria for higher-order Markov models. This paper discussed different methods used in weblog synthesis and compared performance of semantic-rich Markov models for different input sizes and maximum semantic distances. Idea used in this paper provide better accuracy and solution to ambiguous prediction problem as well as less state space complexity than traditional Markov models.

2. Future Scope

Future work includes the need to develop the way of providing strong semantic aware input that will be easy than semantic distance.

Also there is need to overcome the lower limit of order of the Markov Model used. The algorithms must be expanded to use highest order possible, so that, hypothetically, there will be no contradictions at all.

REFERENCES:

[1] . Nizar R. Mabroukeh and c. I. Ezeife "**A Taxonomy of Sequential Pattern Mining Algorithms"** University of Windsor.

[2] . Azer Bestavros (best@cs.bu.edu) " Using Speculation to Reduce Server Load and Service Time on the WWW" cComputer Science Department, Boston University,Boston, MA 02215 February 21, 1995.

[3] Nizar R. Mabroukeh and Christie I. Ezeife "**Using Domain Ontology for Semantic Web Usage Miningand Next Page Prediction",** School of Computer ScienceUniversity of Windsor 401 Sunset Ave.Windsor, Ontario N9B 3P4mabrouk@uwindsor.ca

[4]. Jose Borges and Mark Levene ," Data Mining of User avigation Patterns", Department of Computer Science University College London Gower Street London WC_E _BT U_K_ fjborges_ mleveneg_csuclacukAbs

[5]. James Pitkow and Peter Pirolli," MINING LONGEST REPEATING SUBSEQUENCES TO PREDICT WORLD WIDE WEB SURFING", Proceedings of USITS' 99: The 2nd USENIX Symposium on Internet Technologies & Systems Boulder, Colorado, USA, October 11–14, 1999

[6] Mukund Deshpande and George Karypis, "Selective Markov Models for Predicting Web-Page Accesses", University of Minnesota, Department of Computer Science/Army HPC Research Center Minneapolis, MN 55455 fdeshpand,karypisg@cs.umn.edu

[7]. Rakesh Agrawal ,Tomasz Imielinski, Arun Swami,," Mining Association Rules between Sets of Items in Large Databases", IBM Almaden Research Center 650 Harry Road, San Jose, CA 95120

[8]. Gerd Stummea, , Andreas Hotho a, Bettina Berendt," Semantic Web Mining State of the art and future directions", a Knowledge and Data Engineering Group, University of Kassel, D-34121 Kassel, Germany b Institute of Information Systems, Humboldt University Berlin, Spandauer Str. 1, D-10178 Berlin, Germany Received 20 May 2005; accepted 2 February 2006

[9] Peter Pirolli," Distributions of Surfers' Paths through the WWW", Xerox Palo Alto Research Center 3333 Coyote Hill Road Palo Alto CA 94304 USA