International Journal of Multimedia and Ubiquitous Engineering
Vol. 9, No. 12 (2014), pp. 37-48
http://dx.doi.org/10.14257/ijmue.2014.9.12.04

A Study on Analysis Engine for Large-scale User Behavior based on Cloud Computing

Wei Dai*[1] and Zilong Jiang[2]

[1]*School of Economics and Management, Hubei Polytechnic University, Huangshi 435003, Hubei, China*
[2]*School of Computer Science and Technology, Wuhan University of Technology, Wuhan 430063, Hubei, China*
**dweisky@163.com (corresponding author)*

Abstract

Pointing to the demand of large scale internet websites want to improve users' loyalty and flows maintaining. A novel conception of analysis engine for massive user behavior is proposed in this paper, which combines static analysis with dynamic monitoring for user behavior. It apply some improved data mining model based on cloud computing to analysis Web log data and contextual information of the page acquired real-time. Meanwhile, using cloud database for efficient processing and storaging. The results show that this system could significantly improve the effect and efficiency of user behavior analysis.

Keywords: *User behavior, Analysis engine, Cloud computing*

1. Introduction

User behavior analysis [1] refers to trace and record user behavior of visiting internet website , and analysizes its disciplinarian to get better user experiment and improve website traffic. In the era of internet developing rapidly, we can extract features of user accessing to website and learn its characters by analysizing user behavior. On the one hand, more intelligences service can be offered by individual custom-tailor and push; on the other hand, interest and preference obtained from user behavior are helpful to improve website's architecture and relevance of pages, and it also can relieve complexity of user achieving information in the website [2].

User behavior analysis has already attract the wide attention of scholars at home and abroad. R. W. Cen [3] investigated massive information which included the query length, query modification rate, related search clicks, the first & last click on distribution information of location and query hits in user search behavior to optimize the search engine algorithm and system. Z. Y. Wang [4] intended to enhance retrieval algorithm of the search engine and the retrieval efficiency to free users from abundant disorderly search results through the distributed file system HDFS parallel computing model that supports massive log file analysis.

C.X. Tao [5] proposed an analysis engine solution of user behavior for mobile Internet, which includes the overall architecture design of system, data storage and preprocessing module and user behavior data analysis model. In addition, the foreign scholars such as Joohee Kim, Chankyou Hwang [6] proposed Hadoop analysis method of IPTV and user area's features of IPTV based on Map Reduce model.

We can know the user behavior analysis are mostly concentrate on mining WEB log at

ISSN: 1975-0080 IJMUE

present from above [7], the Web logs represent user intention, but it is not enough to describe the scene when the user visits a website, real sense of user browsing web pages can be recurred only when client's operation behavior and the context information can be collected and combined real-timely. Extracting overall user behavior trail offer data guarantee to analysize user behavior.

Hence, the paper proposes an idea of analysis engine of user behavior, which combines static analysis and real-time monitoring pointing to user behavior, experiment results show the system can significantly improve the accuracy and efficiency of user behavior analysis.

2. The Architecture of Analysis Engine

In this study, "user behavior analysis engine" is defined as: it is a system which can summarize and reason the characteristics and habit of user behavior through analysizing dynamic user behavior and historical behavior in terms of certain strategy.

The analysis engine can mine habits from large-scale user behavior informations, which architecture is shown in Figure 1. It includes:

Acquisition and preprocessing of dynamic behavior: preprocess behavior information in operation page obtained from the client, it includes data cleaning, transformation, reduction.

Storage for user behavior information based on HBase: store user behavior information from the client and the server, the dynamic and static user behavior data, analysis results.

Aggregating dynamic user behavior: filter and integrate dynamic user behavior data acquired, and then delete behavior information which is correct but invalid.

Dynamic behavior analysis: analysize dynamic behavior Based on the Markov model and Collaborative filtering method.

Static behavior analysis: mainly completes the Web log mining, and reappears the historical context information according to the time point, executes filtering, denoising, fusion operation, saves the processed results, and generate historical behavior lib.

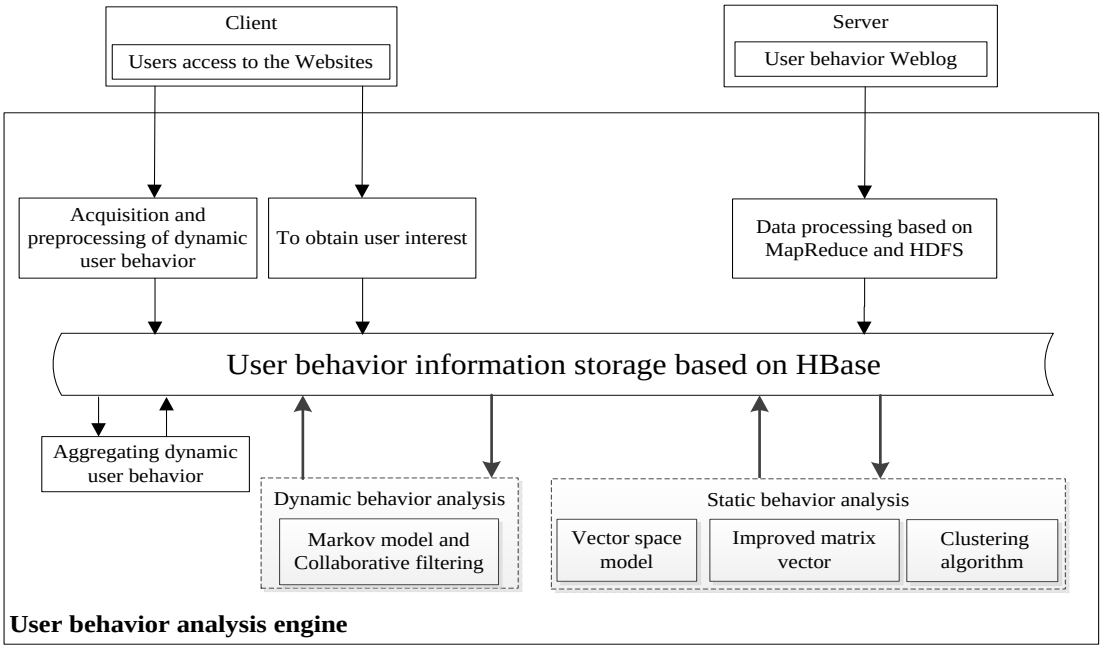


Figure 1. The Architecture of Large-scale User Behavior Analysis Engine

3. Research on Key Technology

3.1. Acquisition and Preprocessing of Dynamic User Behavior

The dynamic user behavior refers to those behavior occurred when user(including two case of login and unlogin. The logined user can be identified by user's registered account ID, while unlogined user can be identified by SessionID that is generated when unlogined user visit website.) visits pages, which includes occurrence time, the page (contains the page title and page *URL*), interrelated operation and behavior subject. Capturing them real-timely and analysis effectively has significant meaning to study the characteristics of user behavior.

The informations obtained from the client include user dynamic behavior and context information: occurrence time of behavior; the current user ID or SessionID; the current page title; the address *URL* of the current page; the current user search conditions;the number of visiting the same page; retention time of page; save the page or not; printed the page or not; add to favorites or not; copy or cut the page content and so on; machine configuration of the client; current network condition; working condition of the server etc.

MapReduce model need to used for the data scale, the course consist of filtering, eliminating duplication, deleting the auseless content, checking completeness and consistency of the information. the methods used is as the following:

1) Data cleaning: delete invalid data including incomplete data, duplicate data, pictures, pages of animation [8].

2) Data transformation: transformate operations of the pages print, collections, preservation, download into 0/1 to store in database.

3) Data reduction: standardize the data quantity and decrease the data sacle, but maintain the integrity of the data.

3.2. Mining Historical Behavior Data

The historical behavior analysis is mainly mining the Web log [9-11]. The analysis platform with single node can not meet the demand for large-scale data due to increased number of netizen. The analysis engine based on cloud platform store large-scale Web log into the distributed file system HDFS firstly, and then execute the operations of clearing, denoising, reduction to Web log with MapReduce model, finally parallelize multiple data mining algorithms and provide analysis functions with the way of web services, which include user search behavior analysis based on vector space model under cloud environment, user search behavior analysis based on improved association rules method with matrix vector, and user interest analysis based on clustering algorithm. The results are stored into user behavior lib in Hbase to reason user interest which improves analysis efficiency for large-scale log file.

3.2.1. User Retrieval Behavior based on Vector Space Model: Vector space model (VSM:Vector Space Model) proposed by Salton [12-15] and others in 1970s, the basic idea is that every text and query contains some features that reflect specific content properties, every feature item represents a dimension of vector space, and a text can be represented with a set of feature items but its complex relationship between paragraphs, sentences and words in the text structure are ignored. Meanwhile, certain weigh is distributed to feature item, which reflects the significance of feature item in text content and be calculated by item statistics data such as term frequency(TF). Vector of each file is in fact a combination of all the document feature, which is named as "document-item features matrix". Every document vector can calculate similarity of each other.

Vector space model can be described as $I = (D, T, Q, F, R)$ Among them:

$D = \{d_1, d_2, \cdots, d_n\}$ as a text set, n is the number of text;

$T = \{t_1, t_2, \cdots, t_n\}$ as a feature set, m is the number of all features. A text with m features can be represented by a vector $d_i = \{w_{i1}, w_{i2}, \cdots, w_{im}\}, i = 1, 2, \cdots, n$, w_{ij} is weight of t_j in the text d_i, if the value w_{ij} of weight is 0, it indicates t_j is not appeared in d_i.

$Q = \{q_1, q_2, \cdots, q_m\}$ as the query set, a query q_r can be represented by vectors $q_r = \{q_{r1}, q_{r2}, \cdots, q_{rm}\}$, q_{rj} is a weight of t_j in query q_r, if the weight value q_r is 0, it indicates that t_j is not appeared in q_r.

Further definition:

Item frequency tf_{ij} : is the appearance frequency of feature t_j in text d_i;

Inverse document frequency idf_i :is a quantitative distribution of word in document set, its computing formula is $\log(N / n_k + 0.5)$, which N is the total number of documents, n represents the number of documents included item k.

The normalization factor: standardize every item in order to reduce the inhibitory effect of individual item with high frequency to other characteristic items with low-frequency.

Accordingly, term weight formula by above three factors is defined as:

$$w_{ik} = \frac{tf_{ik} \log(N / n_k + 0.5)}{\sqrt{\sum_{k=1}^{n} (tf_{ik})^2 \times [\log(N / n_k + 0.5)]^2}} \quad (1)$$

The similarity between text and query can be measure by the distance between two vectors. There are many kinds of calculating method of similarity,such as inner product, Dice coefficient, Jaccard coefficient and cosine coefficient. The cosine coefficient method is usually used, which the angle cosine between two vectors represent the similarity between the text and the query $Sim(d_i, q_j)$. The similarity calculation method of angle cosine is a normalization that the smaller the angle between the two vectors is, the greater the similarity of relavance between two documents is. The angle cosine between two vectors is equal to vector inner after standardizing them to unit length, which reflects the relative distribution similarity of term component between two vectors.

$$Sim(d_i, q_j) = \cos\theta = \frac{\sum_{k=1}^{n} w_{ik} \times w_{jk}}{\sqrt{(\sum_{k=1}^{n} w_{ik}^2) \times (\sum_{k=1}^{n} w_{jk}^2)}} \quad (2)$$

Algorithm design:

Input: key words of every query in visited log files.

Output: the documents that the value of similarity between query keywords vector and current database is not zero and their similarity value is sorted from big to small order.

1) Extracting keywords from each pages as the feature items, secondly, combining these feature items with query keywords, thirdly, removing repeated items and sorting by literature letter, fianlly, unite them as standard feature items set.

For example, there are page document set (*NDoc1, NDoc2, NDoc3, NDoc4*), all the feature items including every page document are represented as (*W1, W2, W3, W4, W5, W6*), and if there are the query items (*WQ1, WQ2*), and *WQ1= W4*, the standard features set is (*W1, W2, W3, W4, W5, W6, WQ2*), the matrix of page document-feature item shows in Table 1.

Table 1. An Instance on Vector Space of Page Document

	W1	*W2*	*W3*	*W4*	*W5*	*W6*	*WQ2*
Q1	0	0	0	0	0	0	1
NDoc1	14	21	33	0	0	0	0
NDoc2	0	11	15	0	0	22	0
NDoc3	8	0	0	14	15	17	0
NDoc4	0	8	9	12	0	15	0

2) Calculating the frequency tf_{ij} of t_j in page text d_i; secondly, calculating inverse document frequency idf_i of item by formula $\log(N / n_k + 0.5)$; thirdly, calculating the weight of every feature items in every page document vector by equation (2) to generate every page document vector in vector space.

3) Calculating the similarity $Sim(d_i, q_j)$ of each document vector and the query vector, as follows in equation (2). Intercepting articles that the value of similarity greater than 0.2000, and then returning results from high to low.

3.2.2. Association Rules based on Improved Matrix Vector: Association rule [13, 14] mining is used to find correlation among the attributes in databases.

The initial motivation is to solve shopping basket analysis problem that finding the association rules of different commodities in transaction database, mining valuable and useful knowledge that emboding correction of data items. These knowledge and rules represent shopping behavior and mode, and can conduct merchant to reasonablely arrange and design goods shelves.

The form of association rule is a kind of rule like this: "to customers that buy milk and bread, 90% of whom bought butter", namely"(milk, bread)→ butter" issue.

Supposing the $I = \{i_1, i_2, \cdots, i_m\}$ is set of items, data *D* of related task is a set of database transactions, thereinto every transaction *T* is a set of items, namely $T \subseteq I$. Every transaction can be identified by *TID*. Supposing A is a items set, transaction T contains A only when $A \subseteq T$. Association rules are contain formula like $A \Rightarrow B$, and $A \subset I, B \subset I$, $A \cap B = \varnothing$. Rule $A \Rightarrow B$ exists in transaction set *D*, and has support *s*, which is percentage of transactions in *D* contain $A \cup B$, namely *P* (*A*∪ *B*). Rule $A \Rightarrow B$ has confidence *c*, which is percentage of transactions that contain A in *D* also contain *B*, namely *P* (*B* |*A*).

$$Support(A \Rightarrow B) = P(A \cup B) \tag{3}$$

$$Confidence(A \Rightarrow B) = P(B \mid A) \tag{4}$$

Support and confidence are two important concept that represent association rule, the first is used to measure the statically significance of association rule in whole dataset, and it denotes frequency of rule; the later is used to measure creditable degree, and it denotes strength of rule. General speaking, only these association rules with high support degree and creditable degree are user-interested and useful. Association rules mining is mainly realized by the two steps:

Step 1: Finding all the frequent item sets in the database *D* according to the minimum support degree.

Step 2: Generating association rules according to the frequent item sets and minimum confidence.

The task of step one is to quickly and efficiently find all frequent item sets in *D*, is the central problem of the association rule mining algorithm, is a standard of measuring the standard association rules mining algorithm; Currently all association rules mining algorithm is designed for the first step due to step two is relatively easy.

Definition 1: Boolean matrix: giving a Boolean matrix representation pointing to transactions set and item sets in the database. The detailed method is that arranging every transactions set as a row and every item *s* set as a column. Namely transaction is as row vector, and project is as column vector. If item *i* is in transaction *j*, the value of row *j* and column *i* in matrix is 1, otherwise 0. The matrix is named as Boolean matrix of database.

Definition 2: Vector inner product: pointing to any two n dimensional vector

$\alpha = \{x_1, x_2, \cdots, x_n\}, \beta = \{y_1, y_2, \cdots, y_n\}, < \alpha, \beta >= \sum_{i=1}^{n} x_i y_i$ is defined as the inner product of α and β:

Lemma 1: Any item of C_k is a superset of certain k-item in C_{k-1}

Lemma 2: The inner product results between corresponding n dimensional row vector of every frequent items set in transaction database *D* and every row vector of Boolean matrix R is less than or equal the number of items in frequent items set.

Definition 3: column vector counting: is the summation of every elements in a certain column.

Lemma 3: If a column vector count in the Boolean matrix is less than the minimum support count, the column should be deleted. (transaction compression)

Lemma 4: In a *k*- frequent item set, if a row vector count in the Boolean matrix is less than *k*, the row should be deleted. (Project compression)

Definition 4: The method of calculating γ_i: to a row vector α_i in R *(i=1,2,...,m* and the scanning sign $a \neq -1$) calculate the number of γ_i that $< \alpha_i, \alpha_j >=< \alpha_i, \alpha_i >$ (*j*=1,2,…,*m and j≠i*).

Algorithm Design: In terms of the definition and lemma above, improved association rules algorithm with matrix vector for the user purchase behavior is designed.

Input: User behavior log information for a certain period

Output: Association rule is defined as: commoName1_commoName12→ commoName13. The front is these commodities that have been purchased; the later is some commodities that the system recommended to the user.

Step 1: Generating all frequent item sets with improved matrix vector method

1) Firstly, clear and preprocess user behavior information, secondly, extract the fields including user IP, date, shopping goods name, and then combine a format like this: "*user IP_ date →goods1_ goods2_...*", thirdly, form a transaction records set *D*, and store such a record into a path in the HDFS file system;

2) Establishing transaction Boolean matrix R : establish the corresponding transaction Boolean matrix $R_{m \times n}$ in memory through scanning the transaction record set *D*; and gives the minimum support degree S_{min} .

3) Getting frequent 1-items set: computing its inner product for every column in the R

columns, and set the scanning position of the column -1 if the value of inner product is less then $S_{\min}$. The result shows that the corresponding item of the column does not exist in frequent 1- items set, while the other items consist of frequent 1-items set.

4) Computing γ_i value to every row vector α_i (i=1,2,…,m ; and the scanning sign of the row ≠-1) in R. If the value of $\gamma_{i+1} \geq S_{\min}$ (with the constrain conditions that γ_{i+1} includes α_i itself) is true, switch to (5), otherwise to (6).

5) Currently α_i is likely row vector that contains frequent items. Finding other row vectors in R that row vectors counting is less than α_i row vector counting, and set row scanning sign -1. If ++i≤m, switch to (4), otherwise to (7).

6) Setting row scanning sign -1. If ++i≤m, switch to (4), otherwise to (7).

7) Finding those rows that row scanning sign is not -1, and then extracting those items with value 1 to get frequent items set.

8) Getting all frequent subsets in terms of these frequent sets. The results are all frequent sets.

Step 2: Generating rules: extracting those rules that meeting confidence from all frequent items set--items set Y is divided into two subsets X and Y-X with more than 0 elements, meanwhile $X \rightarrow Y - X$ need to meet confidence degree, confidence degree can calculate by formula $\sigma(\{X \cup \{Y - X\}\}) / \sigma(\{X\})$. k-item set can generate $2^k - 2$ rules.

From the entire frequent item sets extraction with confidence required by the rules. Association rules are extracted: set Y is divided into 2 nonempty subset of X and Y-X, at the same time $X \rightarrow Y - X$ must satisfy the confidence; confidence can be $\sigma(\{X \cup \{Y - X\}\}) / \sigma(\{X\})$ calculated by formula. We can from the k-item set generation 2^k-2 rules, $\varnothing \rightarrow Y$ and $Y \rightarrow \varnothing$ neglected. According to the above rules, for each frequent item sets with the way Map Reduce parallel computing association rules of the respective energy generation, and stored in HBase database.

3.2.3. User Behavior Analysis of Contextual Aware based on Markov Chain and Collaborative Filtering: Markov model is based on Markov process; it regards a whole random as a series of statement transfer. The state T at the moment t is represented with $q_{(t)}$, which is any one of n kinds of state set $S = \{s_1, s_2, \ldots s_n\}$. The Characteristics of Markov model are represented with transfer probability, and the probability of later state is decided by the order of foregoing states, namely the appearance probability of state $q_{(t)}$ is $P(q_{(t)} \mid q_{(t-1),} q_{(t-2),} \ldots, q_{(1)})$. Markov model estimates the coming state probability distribution in terms of state transfer matrix.

The state transition matrix: transition probability $a[i][j] = P(s_j \mid s_i)$ is a probability from state i to state j. $a[i][j]$ has N*N value because both i and j have N states. They can be represented as follows: $A = a[i][j]$, $\sum a[i][j]$ (j=1…N).

The idea of collaborative filtering is to find the nearest neighbors set for target user through computing the similarity between the target user and other users. Neighbors set $N_u = \{N_1, N_2, \ldots, N_i\}, u \notin N_i$ descending by similarity $sim(u, v)$ are generated. The top N users are selected according to the predetermined number N of neighbors.

Algorithm design:

Input: All dynamic user behavior information and the query *URL* of user

Output: Recommending next probably *URL* set

1) Cleaning and preprocess all the dynamic user behavior and context information, which shows in Table 2, extracting the fields of *UserID, DanymicBehaviorTime, SearchURL, PageStayTime, SavePage, PrintPage, Favorites* in each record;

2) Establishing the Markov sate transfer matrix: firstly, selecting *UserID* to identify user; secondly, reorganize the user's Markov sequence of *SearchURL* order by the time sequence according to user query record, and then build row and column of matrix in terms of *SearchURL*, statistic the times of switch from current *SearchURL* to other for every user, and the ratio of the times of switch and overall switch times for the user is the value of corresponding place in matrix, the Markov sate transfer matrix for every user can be built. Row head of every transfer matrix is *UserID_SearchURL*, column head of every transfer matrix is *SearchURL*, and each Markov sate transfer matrix is stored in table UserShiftMatrix in Hbase.

3) Weighting for every Markov state transition matrix: firstly, *PageStayTime* is selected as one of weight computing condition for an element in state transition matrix, if *PageStayTime* ∈ （0, 30）, the corresponding element value in matrix is set 1; if *PageStayTime*∈（30, 60）, the corresponding element value in matrix must be multiply with （1+1/20）; if *PageStayTime*∈（60, ∞）, the corresponding element value in matrix must be multiply with （1+2/20）; secondly, *SavePage, PrintPage, Favorites* are selected as one of weight computing condition for an element in state transition matrix, if any one parameter value is 1, the corresponding element value in matrix must be multiply with（1+2/20）.

4) Calculating the similarity by the cosine factor method: according to interest parameters Table 3, interest keywords that user u selected in register information is regarded as vector u, the similarity between u and v can be achieved by formula:

$$sim(u,v) = \cos(u,v) = \frac{u * v}{|u||v|} \tag{5}$$

5) Recommended results: pointing to the user input of query *URL*, find the next possible *URL* in Markov state transition matrix of every user, and the value that is more than certain threshold(such as 0.1000) is the value met demand. And then carrying out the same operation on top *N* users with high similarity, reasonable recommend set can be achieved.

Table 2. User Behavior Parameters

Table name	*DanymicBehavior_T*			
Primary key	*DanymicBehaviorTime*			
Field name	Data type	Null or not	the field description	Default value
DanymicBehaviorTime	Datetime	No	the occurrence time	
UserID	Varchar(10)	No	the current user ID	
PageTitle	Text	No	the current page title	
SearchURL	Text		search url for the current user	Null
PageStayTime	Int		page stay time	0
SavePage	Char(1)		save the page or not	0
PrintPage	Char(1)		the print page or not	0
Favorites	Char(1)		add to Favorites or not	0
Copy or Cut Content	Text		copy or cut the page content	

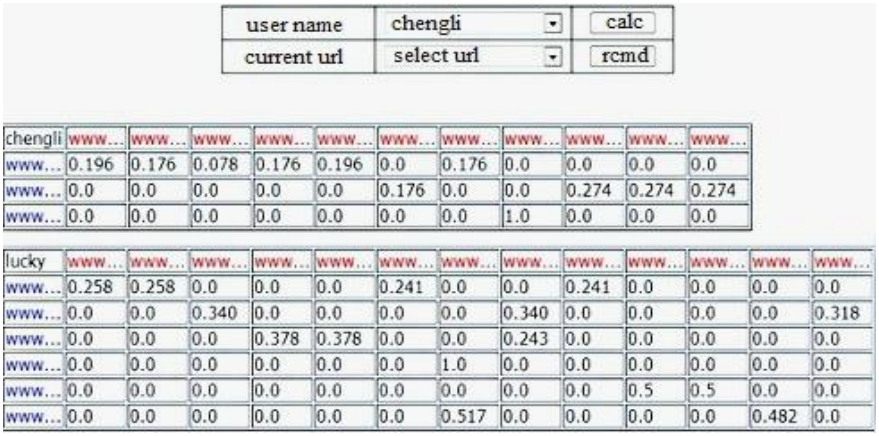
Table 3. Interest Model Parameters

Table name	*FavoritesModel_T*			
Primary key	*UserID*			
Field name	Data type	Null or not	The field description	Default value
UserID	Varchar(10)	Not	Online user ID	
FavoriteField	Text		User's interest domain	
UserField	Text		User field	
Deduction	Text		User interest keywords	

4. Experiment Results

The analysis engine platform is running on the Ubunt12.10, the software mainly includes: jdk-1.7.0_11, Jena-2.6.4, Myeclipse-8.0, Hive-0.10.0, HBase-0.94.4, Hadoop-1.0.4, Tomcat-6.0, Jquery-1.6, Spring-3.0, Struts2-2.2.1, the browser is above IE8.0. An example is given applied context aware behavior analysis.

The system calculates the middle Markov matrix, as shown in Figure 2.



chengli	www...	www...	www...	www...	www...	www...	www...	www...	www...	www...	www...
www...	0.196	0.176	0.078	0.176	0.196	0.0	0.176	0.0	0.0	0.0	0.0
www...	0.0	0.0	0.0	0.0	0.0	0.176	0.0	0.0	0.274	0.274	0.274
www...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0

lucky	www...	www...	www...	www...	www...	www...	www...	www...	www...	www...	www...	www...	www...
www...	0.258	0.258	0.0	0.0	0.0	0.241	0.0	0.0	0.241	0.0	0.0	0.0	0.0
www...	0.0	0.0	0.340	0.0	0.0	0.0	0.0	0.340	0.0	0.0	0.0	0.0	0.318
www...	0.0	0.0	0.0	0.378	0.378	0.0	0.0	0.243	0.0	0.0	0.0	0.0	0.0
www...	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0
www...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.5	0.5	0.0	0.0
www...	0.0	0.0	0.0	0.0	0.0	0.0	0.517	0.0	0.0	0.0	0.0	0.482	0.0

Figure 2. Markov Matrix

Pointing to current *URL* for a user, system uses Markov model and collaborative filtering to list next possible *URL*, as shown in Figure 3, Figure 4.

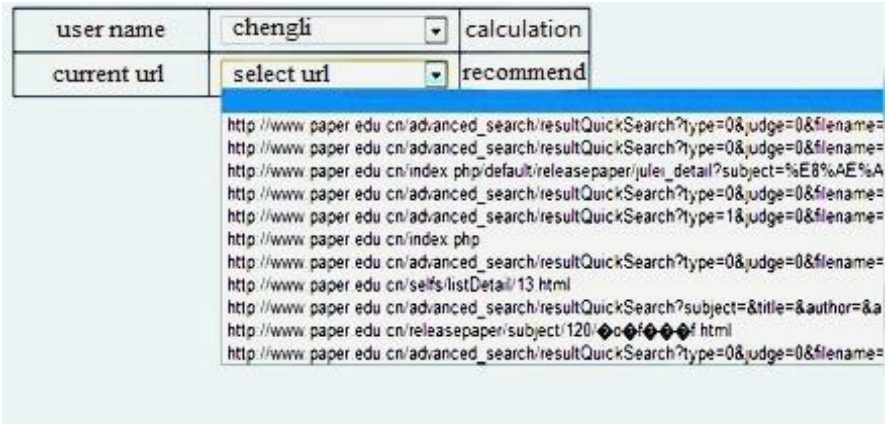


Figure 3. The Selection of Current URL

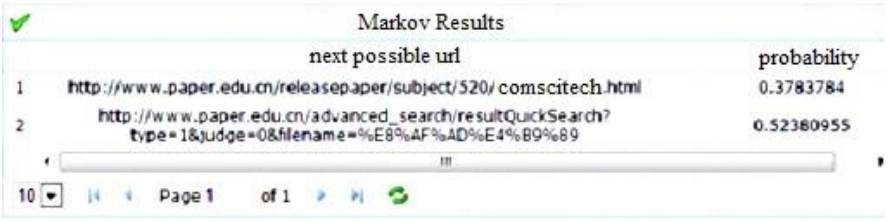


Figure 4. Markov Results

5. Conclusion

The analysis engine of user behavior adopts various data mine algorithms based on MapReduce model and HBase cloud storage capacity to analyze the information of WEB log, dynamic user behavior and context information, and achieves efficiently interest information for a user by static user behavior analysis and dynamic monitor.

Due to the constraints of time and energy, this platform currently integrates only three data mine model. Aggregating and integrating more data mine model for certain scene based on the cloud platform is the future work.

Acknowledgements

The work is supported by Humanities and Social Sciences Youth Fund Project of Ministry of Education of China (No.13YJCZH028).

References

[1] Y. Chen, Y. Yu and W. Zhang, "Analyzing User Behavior History for Constructing User Profile", IT in Medicine and Education, **(2008)**, pp. 343-348.
[2] I. Ullah, G. Doyen, G. Bonnet, *et al.*, "A Survey and Synthesis of User Behavior Measurements in P2P Streaming Systems", Communications Surveys & Tutorials, IEEE, no. 99, **(2011)**, pp. 1-16.
[3] C. W. Ceng, Y. Q. Liu, M. Zhang, *et al.*, "Search Engine User Behavior Analysis Based on Log Mining", Journal of Chinese Information Processing, vol. 24, no. 3, **(2010)**, pp. 49-54.
[4] Z. Y. Wang and L. Guo, "An Analysis of the Search Engine User Behaviors Based on Hadoop", Computer Engineering & Science, vol. 33, no. 4, **(2011)**, pp. 115-120.
[5] C. X. Tao, X. J. Xie and K. Chen, "Design of Mobile Internet Big Data User Behavior Analysis Engine Based on Cloud Computing", Telecommunications Science, vol. 29, no. 3, **(2013)**, pp. 27-31.
[6] J. Kim, C. Hwang, E. Paik, *et al.*, "Analysis of IPTV User Behaviors with MapReduce". International Conference On Advanced Communication Technology, **(2012)**, pp. 19-22.
[7] X. L. Bai and F. Chen, "The Prediction of Web User's Behavior Based on Web Usage Mining", ETP/IITA Conference on Telecommunication and Information, **(2010)**, pp. 32-35.
[8] S. K. Kim, "Enhanced user experience design based on user behavior data by using theory of Inventive Problem Solving", in Industrial Engineering and Engineering Management, **(2010)**, pp. 16-23.
[9] W. H. Liao and C. P Chueh, "Analysis and Interpretation of e-Reader User Logs: A Case Study of High School Students' User Behaviors", in Intelligent Context-Aware Learning and Teaching Environment, **(2011)**, pp. 106-112.
[10] T. M. Li, W. J. Hou and H. Pan, "Put context-aware in Virtual Assembly: a study of ontology-based architecture and an arithmetic", Joint Conferences on Pervasive Computing, **(2009)**, pp. 316-322.
[11] H. R. Hasani, N. Movahhedinia and B.S.Ghahfarokhi, "Evaluation of user perceived quality based on user behavior", Proceeding of 12th IEEE in Computational Intelligence and Informatics, **(2011)** , pp. 341-348.
[12] G. Salton, A. Wong and C.S. Yang, "A vector space model for automatic indexing", Communications of the ACM, vol. 75, **(1975)**, pp. 613-620.
[13] R. Agrawal and R. Srikant, "Mining association rules between sets of items in large databases", International Conference on Management of Data, Washington DC, **(1993)**, pp. 207-216.
[14] R. Agrawal and R.Srikant, "Fast algorithms for mining association rules", Proceedings of the 20th International Conference on Very Large Database, **(1994)**, pp. 487-499.

[15] J. A. Whittaker and M. G. Thomason., "A Markov Chain Model for statistical software testing", IEEE Transactions on Software Engineering, vol. 30, no. 10, **(1994)**, pp. 812-824.

Authors

Wei Dai, he received his M.S.E. in Computer Science and Technology (2006) and Ph. D in Computer Application Technology (2012) from Wuhan University of Technology. His current research interests include different aspects of Intelligence Computing and Information Systems.

Zilong Jiang, he received his M.S.E. in Computer Science and Technology (2006) and now he is a Ph. D in Computer Application Technology of Wuhan University of Technology. His current research interests include Machine Learning, Internet advertising, HPC.