International Journal of Scientific Research in Computer Science, Engineering and Information Technology
© 2017 IJSRCSEIT | Volume 2 | Issue 5 | ISSN : 2456-3307

Dual Access Cache Memory Management Recommendation Model Based on User Reviews

J. Sangeetha[1], Dr. V. Sinthu Janita Prakash[2], Dr. A. Bhuvaneswari[3]

[1]Associate Professor, Cauvery College for Women, Trichy, Tamil Nadu, India
[2]Head, Department of Computer Science, Cauvery College for Women, Trichy, Tamil Nadu, India
[3]Associate Professor Cauvery College for Women, Trichy, Tamil Nadu, India

ABSTRACT

With the growing number of Internet services, the prediction of relevant services based on the user opinions is the major task in the recommendation model. The inclusion of important information taken from the user-generated textual reviews (in textual comments) is the major objective in the diverse review-based recommendation. An advanced textual analysis extracts the different elements for recommendation such as reviewer review, contextual information and the comparative opinions to improve the content-based recommendation performance. The major problems in such content-based recommendation are sparsity and memory management. Hence, this paper proposes the Dual Accessing Cache Memory Management (DACMM) to alleviate the issues in the recommendation system. The provision of ratings to the products based on the user experiences and their learning contribute to recommend the interesting product to the end users. With an increase in the dimensionality of products, the dimensionality of their reviews is increased. This paper employs the feature extraction that utilizes the positive, negative reviews with the overall word count to identify the features. The assigning of the index corresponds to the user review with the positive and negative features reduce the size of products into the interesting category. The integration of query processing with the cache memory management reduces the complexity in recommendation operation effectively. The comparative analysis between the proposed DACMM with the existing clustering and cache management models in terms of query latency, hit ratio, Mean Absolute Error (MAE), computational time and precision assure the effectiveness of the proposed DACMM in the review-based recommendation.

Keywords : Cache Memory Management, Clustering, Feature Extraction, Indexing, Recommendation, Similarity.

I. INTRODUCTION

N automatic gathering of the information and the practical adaptation according to the user interests are the major issues in the Recommendation System (RS). Nowadays, the recommendation of a relevant number of services is performed based on the user preferences. The memory-based Collaborative Filtering (CF) approaches recommend the relevant services based on two categories as follows: user-based and item-based[1]. The first model employs the ratings assigned by each user in the group of likely minded users and recommends the target. The second model predicts the items based on the similarities between the current and the items already purchased by the users. The sufficient rating information is the major requirement for the collaborative filtering techniques. The lack of scalar ratings and the poor recommendation space limits the performance of CF approaches and leads to the sparsity problem. Hence, the research focuses on the user reviews by performing the comprehensive survey of attempts for the valuable information regarding any web services or the products available in such services to reduce the sparsity problem. With the substantial increase of e-commerce media, the reviews that describe the assessment of items is in the textual form. The capture of multifaceted nature of the user's reviews is used to build the fine-grained preference. Dealing with data sparsity problem, cold start problem, efficient

rating quality is the major issues in the recommendation system.

The incorporation of the peer posts with the better quality plays the major role in decision-making process. The provision of user reviews is surprisingly or technologically poor due to the difficulty in a prediction of relevant information from the massive amount of texts. The repetitive appearance of similar keywords and the reflective nature of overall rating with the product features. The identification of structured information from the informal sentences, poor spelling and grammar is the challenging task in the RS. The User Review Structure Analysis (URSA)[2]combines the language processing with the machine learning algorithms and the CF to obtain the detailed information. The utilization of the full text for the user review is the promising research area to make the prediction ratings are according to the features of the input. The rich textual information that exists in the user reviews plays the major role in the grouping of similar services for making recommendations. The extraction of the rich textual information does not only depend on the numerical rating. It also depends on the sentiments of the users.

The recognition of the Quality of Service (QoS) variations[3]with respect to the locations, on-line time complexity in memory-based CF recommender systems, visualization of ranked web services without transparency are the major problems in the memory-based CF recommender systems. The interpretation of the QoS relationships through the personalized map supports the efficient recommendation performance. The division of users based on the locations with the past experience highly contributes to better QoS prediction for the unused services. With the utilization, sparse user-contributed dataset, the prediction of similar users without sufficient knowledge is the difficult task. Hence, the correlation between the physical location of the users and the QoS properties such as response time and availability. The definition of the region containing a group of the users closely related to each other with the clear contextual information of the influences. The correlation between the QoS properties and the characteristics of the users yield the better recommendation performance. The latency or time required for the delivery of the web objects to the end user was high to recognize the QoS constraints.

The storing of web objects close to the clients by maintaining web cache improves the recommendation performance with minimum latency[4]. The operating stages of web caching are the browser, proxy and web server. Under different web cache policies, the numerous factors affecting the caching in browser stage. To implement the novel mechanisms on cache management based recommendation, space is required for the storage of new items arrived. The web caching policy utilizes the replacement mechanism during the cache memory is full to remove the old items that leave the space for a new item. The making of replacement decision depends on the following five conditions: frequency, fetching cost, modification time and the expiration time. The policies[5]to govern the cache replacement are First-In-First-Out (FIFO), Least Recently Used (LRU), Least Frequently Used (LFU), random (RAND) and the Greedy-Dual Size (GD-size) for better recommendation performance. The sparsity and the scalability are still the major issues that affect the recommendation adversely. The division of data into a number of segments and their classification depends on the suitable clustering process. The maximization of similarity within the group and the minimization of similarity between the object groups are the basic principles for clustering process. The customer score and the demographic variables (age, gender, and occupation) play the major role in the classification of purchased data during the recommendation process. The brief review of approaches in this section conveys that the lack of extract suggestions to the queries, effective prediction of recommendation system and the memory management are the major issues to be solved for a better recommendation. This paper proposes the three novel algorithms to alleviate the issues in existing recommendation approach. The technical contributions of proposed DACMM are listed as follows:

The introduction of cache services and their effective management through random and sequential access (dual) by the user queries resolves the memory management problem effectively

The integration of query processing with the inclusive similarity-based clustering plays the major role in the effective recommendation.

The proposed dual accessing model with the effective feature extraction and indexing in a periodical manner

reduces the cache memory consumption for the relevant results to the user queries.

The paper organized as follows: The detailed description of the related works on the methods involved in relevant recommendation of services discussed in section II. The implementation process of Dual Access Cache Memory Management (DACMM) is described in section III. The comparative analysis of proposed approach with existing methods provided in section IV. Finally, the conclusions about the application of DACMM on the input data presented in section V.

II. Related Work

This section discusses the related works on the various recommendation models available traditionally. Recently, the RS are complement to conventional query-based services that offered the proactive information discovery. Esparsa et al [6] explored the fragmented noisy snippets that were directly used in recommendations. The validation of whether the Real-Time Web (RTW) services were used as the basis for the recommendation and the performance with the traditional systems. The relationship between the web services and the providers described by the two-dimensional form called the user-item matrix. But, the matrix model cannot reveal that relationship accurately. Cao et al [7] presented the cube model to describe the relationship among the services and providers. The matrices which represent the cube model were consumer-service QoS matrix, binary matrix and consumer-provider matrix. Based on the status of the cube model, the Standard Deviation (SD) and the Inverse Consumer Frequency (ICF) based filtering approaches to assure the effective recommendation. Zhang et al [8] enhanced the recommendation system performance by fusing the virtual ratings derived from the user reviews. They identified the self-supervised sentimental classification models with high-precision and recall under proximity evaluation. The unstructured and semi-structured review patterns in the vast number of reviews made the tracking as difficult task. Daoud et al [9] developed the diverse recommendation methodologies to alleviate the challenges such as overload of online shoppers. They utilized the text mining approach to mine the product opinions, features and their semantic similarity regarding the opinion sources. The reliable extraction of the product adopter mentions from the noisy review in large scale and the mapping of adopter mentions with the demographic feature space was the major issue in the recommendation model. Zhao et al [10] developed the unsupervised bootstrapping method to automatically derive the patterns corresponding to adopter mentions that were grouped into six categories. The assessing and provision of customer satisfaction with the business intelligence were the recent research are in recommendation.

The sentences having the mixed review comments such as positive and negative. Hence, the extraction of relevant opinions to the particular feature and classification of them required complete sentence and the overall opinion. Addepalli et al [11] aimed to examine and demonstrate the various issues in the e-commerce websites. The acknowledgement of semantic analysis and the machine learning applied to the large scale consumer reviews. The information overload was the major problem that limits the accuracy and causes the data sparseness. Liu et al [12] proposed the novel recommendation algorithm that incorporated the two methods such as opinion mining and recommendation based on the similarity between the user ratings. The explicit ratings and the implicit opinions were considered to address the data sparseness problem. The generation of useful summary by using the classification based on the polarity of the opinion about each feature.Htay et al [13] used the part-of-speech tagger to identify the phrases, adjectives, nouns and phrases. They considered the written review as the input and summary overview as the output. The identification of product feature, extraction of opinion words and phrases and the generation of summary were the major tasks performed for recommendation. The mining of large volume of unstructured texts in the movie reviews and the devising the algorithms related to the tasks was the difficult task. Singh et al [14] explored the new scheme called SentiWordNet based scheme in two levels such as document and aspect level. The utilization of linguistic features supported the effective document-level classification. The proposed SentiWordNet scheme located the opinionated text around the feature and computed the sentiment orientation. The identification of preference similarity among the reviewers was the major issue in the review-based recommendation. Chen et al [15] proposed the novel clustering method on the basis of Latent Class Regression model (LCRM) that considered both overall

ratings and the feature-based opinions. The extension of LCRM model with the weighing features preferences in the cluster and reviewer level.

Locating desired services to fulfill the requirements and the replacement of services.Zhou et al [16] located the desired services by proposed cluster Data Providing (DP) by Fuzzy-C-Means (FCM) algorithm. The DP services vector assignment to one or multiple clusters within certain degree with the consideration of relationship between the DP service elements. The identification of similar neighbors, inability to produce the recommendation list to the cold start users were the major shortcomings in RS. The employment of trust statements for reduction of malicious attacks and the resistance against the attacks was the necessary stage in such approaches Moradi et al [17]proposed a model-based recommendation method that utilized the graph-based algorithm to cluster the items in three stages such as problem space creation, grouping of similar items and the prediction of top-N interested items for recommendation. The lack of additional attributes related to the grouping of products or users yielded the poor recommendation performance. Liao et al [18] proposed the clustering-based approach called self-constructing algorithm to reduce the dimensionality. The grouping of similar products in the same cluster and the dispatch of similar products were achieved. They utilized the correlation graph that showed the inter-relationship creation among the groups.Korfiatis et al [19] presented the demographic RS architecture based on the user-defined hierarchy of system quality indicator importance. The utilization of algebraic approach to determine the preferences for large size datasets handling. The absence of real-case user evaluation scenario in the extension of RS evaluation stage. Cho et al [20] proposed the efficient purchase pattern clustering method based on Self-Organizing Map (SOM) for ontology recommendation system. The Recency, Frequency and Monetary (RFM) factors using SOM network and the analysis of them considered that segmented by using the purchase-pattern clustering. The optimization of search engine was the major requirement to reduce the random accesses to disks which was the expensive task.

Wang et al [21]utilized the medium called Solid State Drive (SSD) that replaced the Hard Disk Drive (HDD) in their infrastructure point of view. The property of SSD was random access latency alternate to the sequential access latency. They conducted the series of experiments governing SSD impact on the search engine cache management. The grouping of similar services together in a single cluster increased the data size in the large factors. Besides, they compared the proposed SDD with the Static-Query Result Cache (S-QRC) and Dynamic Query Result Cache (D-QRC) in terms of hit ratio and average query latency under Collision Avoidance (CA) scheme. Besides, the performance of proposed D-QRC is investigated in terms of the Least Frequently Used (LFU) and Least Recently Used (LRU).Hu et al [22] proposed the Clustering-based Collaborative Filtering (ClubCF) with the aim of recruiting the similar services exist in the same cluster for collaborative recommendation. The operational speed of the recommendation system was highly dependent on the size of the index structure. Formoso et al [23] studied the impact of compression and coding techniques that reduced the matrix size into 75 % and hence the operating speed was high. They proposed the novel identifier reassignment technique to achieve the high compression rate. Gupta and Moharir [24] proposed the Markovian request model for capturing the time correlation with the user requests. The content delivery through distributed network of servers through geographically co-located servers reduced the latency effectively. Chaurasia and Satsangi [25] implemented two different techniques such as Bayesian classification and ID3 decision tree algorithm to analyze the patterns. The memory consumption, MAE, accuracy, precision and recall were considered as the major parameters to achieve the high recommendation performance.

III. Dual Access Cache Memory Management (DACMM) Recommendation Model

This section discusses the implementation details of proposed dual access cache memory management model for the recommendation of the web services. Fig. 1 shows the workflow of DACMM to reduce the computation time and latency.

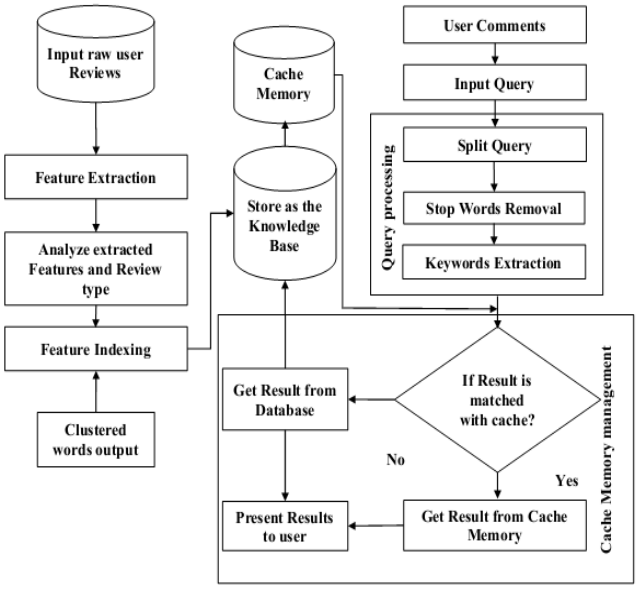


Figure 1. Workflow of proposed DACMM

Initially, the raw user reviews are given to the feature extraction model. This block extracts the relevant features for a recommendation that reduces the dimensionality and thus leads to the reduction in time complexity. The indexing based on the similarity measure and the storage of clustered words to the knowledge base are the sequential steps to the feature extraction process. The query of product and the user emoticons are collected in parallel to manage the cache memory effectively. The large size phrases are split up into the various segments in the query processing and this is considered as the initial stage in query processing. The removal of stop words and the relevant keywords extraction are the successive steps in the query processing stage. The results from the query processing are matched with the contents of the cache memory. Then, check whether the relevant results (services) are available in the cache memory or not. If they are available means the results are extracted from the cache memory otherwise the results are extracted from the database and stored in the database effectively. The major processes in proposed work to reduce the time consumption are listed as follows:

- Query processing
- Feature Extraction
- Feature Indexing
- Cache Memory Management

The detailed description of each process in proposed work is presented in next sub-sections. Table I presents the variables used in proposed algorithm.

TABLE I
SYMBOLS AND DESCRIPTIONS

List Of Variables	Description
$Feat_{list}$	Feature List
Pos_{Set}	Positive Dictionary List
Neg_{Set}	Negative Dictionary List
R_{list}	Review List
$Feat_{ID}$	Feature Index
W_n	No of Words in each Review
W_{Tp}	Word Type
$Clus_{id}$	Cluster Id
R_{tp}	Review Type
R_{id}	Review id or index
Ch_{list}	Cache list
Q_{list}	List of the words in the Query
MF_{list}	Matched Feature List
Db_{list}	Database List , Set of the Results available in Database
$Query_{res}$	Matched result from the Database is the Query result.

The user emotions are passed to the preprocessing stage prior to the keyword extraction. The preprocessing removes the stop words and the punctuation in the reviews.

A. Query Processing

The next step is the relevant keyword extraction to reduce the time consumption. The next step is the Part Of Speech (POS) tagging by using the POS tagger. Then the linguistic filter is applied to extract the noun phrases. The major reason to extract the noun phrases is the keywords for DACMM model are mostly noun related. Once the phrases are extracted from the review reports, then they are subjected to the stop word removal process.In another stage, the raw text is split up-to-the sentences and each sentence is divided into words using tokenizer. Then, each sentence is tagged with POS tags to identify the entity. In entity detection, the potentially interesting sentences are searched. Finally, the relation between most likely sentences is identified. The user reviews regarding the product contain both positive and negative comments that determine the product quality. The analysis of these reviews inherit the feature extraction as the major step for recommendation.

B. Feature Extraction

The extraction of features is categorized into two according to the status of domain specific knowledge as follows: i) feature extraction in absence of domain specification and (ii) feature extraction in presence of domain specification. In the first case, the list of potential features in the review is made. The features considered in this model are nouns in the POS tagged sentences. E.g. Multimedia, firmware, color, etc. Initially, the nouns which are treated as the features are added to the feature list. Let us consider the review as follows:

"I have an ipod and it is a great buy but I'm probably the only person that dislikes the iTunes software."

The feature set corresponds to the above review are regarded as $F = (\text{ipod}, \text{buy}, \text{person}, \text{software})$. The terms in this set are considered as the initial feature set corresponds to mobile domain. The pruning of the initial feature set through the relationship estimation identifies the strongly relation or merging. Hence, the feature buy is merged with the ipod only if the target feature is ipod and the features (person, software) are pruined. Similarly, if the target feature issoftware, then person is merged with software and the feature set (ipod, buy) is pruined.

In the second model of in presence of domain knowledge, the feature extraction process is based on the reviews of the user for each product. The algorithm to extract the features about the product is listed as follows:

Feature Extraction

Input: User reviews (R_n)
Output: Features extracted of the Products from the user reviews

For each review (r_i)// i=1: n, n – Number of product reviews
    PR= Preprocess (r_i);
End For;
Initialize first, second, current;
For each word from review;
    If (word starts with (+) || word starts with (-)) && if (word length=2)
Feature= first + second;
Set feature id, review id, feature and value;
Update$Feat_{list}$;
Else if (word starts with second)
First= second;
Second=current;
Else
First= {};
Second=current;
End if
End for

The feature set is updated with the POS tags $(+, -)$ and the word count. If the word in the specified length is equal to two, then check whether the word starts with positive or negative tag. If this condition is satisfied, then the combination of first and second are added to the feature. The ID to represent the feature, review with the numerical values. Alternatively, if the word starts with the second means, the positions are changed as follows: First->second and second->current. The word in second position is regarded as the first one and the word in current position is regarded as the second one. Finally, if the word is not started with the positive, negative or second means, the first position is kept as empty and the word in current position is regarded as second. In this way, the features relevant to the recommendation are extracted successively.

C. Feature Indexing

The index assigning to the positive and negative group of words in the review context speed up the execution and this plays the major role to manage the cache memory effectively. The Inclusive Similarity-based Clustering (ISC) [26] of words is the prior process to feature indexing. The clustered words, positive and negative word set are given as the input to the indexing algorithm. The algorithm to perform the indexing is listed as follows:

Feature Indexing

Input: clustered words ($Cluster_{list}$), words in raw review ($Clus_{list}$), positive word set (Pos_{Set}), negative word set (Neg_{Set}), Feature list ($Feat_{list}$)
Output: Indexed Features

For i=1 to n // n- No of words in cluster
    If(Pos_{Set} contains W_n)
        // W_n-No. of words in the review
    Update$W_{Tp} = 'Positive'$; // W_{Tp} – Word type
    Update$Clus_{id}$;
    Else if(Neg_{Set} contains W_n)
    Update $W_{Tp} = 'Negative'$
    Update$Clus_{id}$;
        End if
End for i;
For j=1 to m; // m-No. of reviews in cluster
Compare list of words in reviews with $Clus_{list}$
If ($Type(W_n)$ in $clus_n$=' $Positive$')

$R_{tp} = Positive$; // R_{tp} – Review type
Else if($Type(W_n)$ in $clus_n$=' $Negative$')
$R_{tp} = Negative$;
Else
$R_{tp} = Neutral$;
End if
End forj;
Fork=1 to H; // H – Size of feature set $Feat_{list}$
For l= 1 to R_n
Compute the review index (ID);
If ($Feat_k$ is present in R_{ID})
Update $Feat_{ID}$ in $Feat_{list}$
End if
End for l;
End for k;

The clustered words, raw review, positive / negative word set and feature list are passes to the indexing algorithm. For each word in the cluster, check whether the positive set contains the relevant word means, the word type is specified as "positive" and the negative set contains the relevant word means the extracted word is identified as "negative". Otherwise, they are regarded as the neutral. Then, the algorithm compares the list of words in the review set with the clustered list. If the type of word is positive, then the corresponding review type is positive otherwise the review type is negative. Then, the review index is computed for each feature in the feature list. If the extracted feature is present in the list corresponding to the review index, then the feature ID is updated accordingly.

D. Cache Memory Management

The indexed features stored in the knowledge base are serve as the base for memory management. The output from the query processing also plays the major role in the cache memory management.

Cache Memory Management

Input; Ch_{list}, User Query, $Feat_{list}$
Output; Results for user query

Get query from user;
Split the words in query and update Q_{list}
Remove stop words from and update Q_{list}
For x =1 to Q_{list} size;
If (Q_x matches with $Feat_{list}$)
Update MF_{list};
End if;
End for x;
If (MF_{list} size > 0)
Sort Ch_{list} by LastModifiedTime;
For g=1 to size of MF_{list}
If (Ch_{list} contains MF_{list})
Return result from Ch_{list};
Break loop;
Else continue;
End if
End for g;
Else if (MF_{list} size = 0) // No features are mentioned by the user in the user query.
Sort Ch_{list} by LastModifiedTime;
For h=1 to size of Q_{list}
If (Ch_{list} contains contents of Q_{list})
Return result from Ch_{list};
Break loop;
Else continue;
End if
End for h;
End if;
If (Ch_{list} not matched with Q_{list}) // If query is not matched with any of the cached results
// accessing database for results.
Load results in the database to Db_{list}
Let $Query_{res}$ be the results from Db_{list}
Sort Ch_{list} by LeastModifiedTime;
If $Ch_{\text{LeastModifiedTime}}$ is empty;
Fill $Ch_{\text{LeastModifiedTime}}$with $Query_{res}$;
Else
modify $Ch_{\text{LeastModifiedTime}}$ to $Query_{res}$
End if;

The list available in the cache, feature list and the user query are the major inputs to the memory management process. The stop words removal, split up of words and the update of query using POS taggers are the initial stages in memory management process. The queries from the user are arranged in list format and the comparisons of each query with the feature list and then corresponding product is recommended. Then the feature list is updated as (Modified feature list MF_{list}). The cache list is also updated with the features and relevant user query. Then the elements in the cache list are arranged in ascending order based on the time value. Comparatively, the query is checked with the results stored in the database if the contents in the cache are not matched. The sorting of contents in the cache memory according to $Ch_{\text{LeastModifiedTime}}$ and the sequential update of cache list reduces the searching time efficiently.

IV. Performance Analysis

This section discusses the effectiveness of proposed system by comparing with the existing cache design methodologies[21] and the clustering methodologies[22] in terms of query latency, hit ratio,

computation time and Mean Absolute Error (MAE). The dataset to validate the proposed system is customer review regarding the product in [27].

Dataset

The customer reviews are gathered from the Amazon.com regarding the five products as follows: Canon G3 (digital camera), Nikon coolpix 4300 (digital camera), Nokia 6610 (cellular phone), creative labs Nomad Jukebox Zen Xtra 40 GB (Mp3 player) and the Apex AD2600 progressive –scan (DVD player). The customer reviews regarding the Apex DVD player is taken for example to validate the performance of proposed DACMM. Table II illustrates the positive, negative and neutral reviews regarding the DVD player.

TABLE II
COUNT OF REVIEWS

Types	No of Reviews
Positive	44
Negative	39
Neutral	16

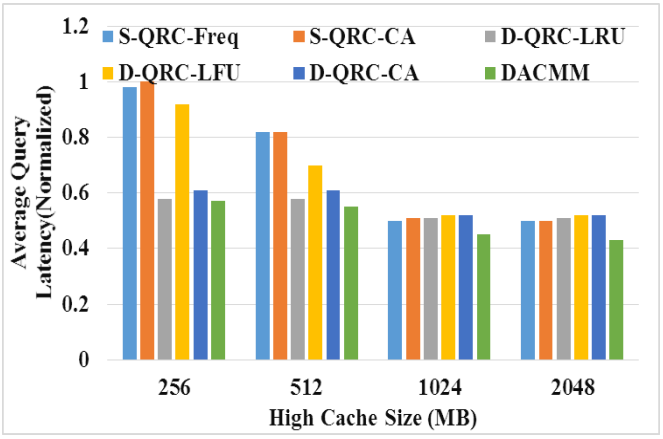
The proposed DACMM is simulated in JAVA to test the performance of system corresponding to the various parameters. Table III presents the precision variations for extracted features. The dual access cache memory management topology yields the better precision performance effectively.

TABLE III
PRECISION ANALYSIS

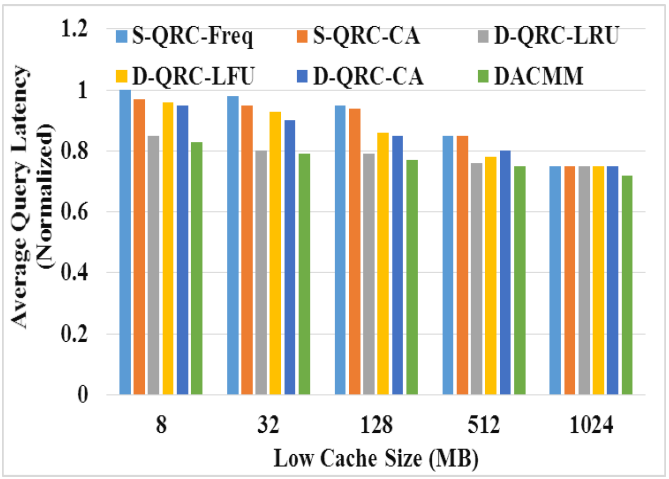
Parameters	Precision (%)
Frequency of Occurrence	80
Useful Reviews	98
Extra Reviews	90
Sufficient Reviews	95
Feature Attributes	95

A. Average Query Latency

This section illustrates the average query latency variations with respect to the cache size variations in two dimensions such as high and low. Fig. 2 shows the graphical illustration of average query latency variations with respect to cache size variations.



(a)



(b)

Figure. 2 Average Query Latency Analysis for (a) High Cache size and (b) Low cache size

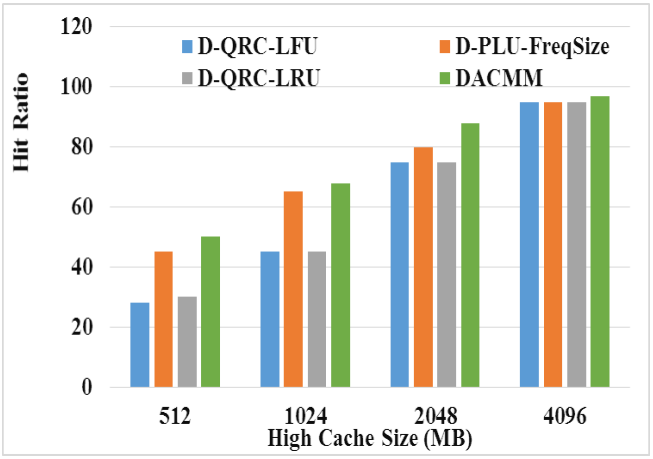
From Figure. 2 it is observed that the proposed DACMM offers minimum average query latency compared to the existing methods for low and high cache size variations respectively. For high cache size (256 to 2048 MB), the average query latency of the proposed DACMM are 0.57 to 0.43 which is 1.72 to 15.69 % better than D-QRC-LRU approaches. Similarly, it offers 4 % reduction in latency values for the size 1024 MB in low cache size variations

B. Hit Ratio

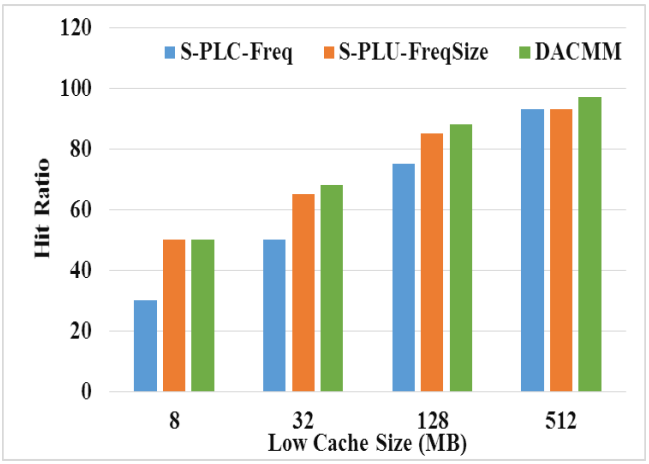
The metric to evaluate the performance of cache on the large size query and recommendations is called hit ratio. Higher value of hit ratio indicates the effectiveness of proposed work. Fig. 3 (a) and (b) shows the graphical illustrations of variation of hit ratio with respect to the high and low cache size variations respectively.

For high cache size (512 to 4096 MB), the average query latency of the proposed DACMM are 50 to 91 %

which is 10 to 2 % better than D-PLU-freqsize approaches. Similarly, it offers 4.12 % improvement in hit ratio compared to the S-PLU- Freq size variations respectively.



(a)



(b)

Figure 3. Hit Ratio Analysis for (a) High Cache size and (b) Low cache size

It is observed that the indexing and relevant feature extraction in proposed DACMM yields the better performance compared to the traditional methods respectively.

C. Mean Absolute Error

With the variations of size of clusters, the analysis of Mean Absolute Error (MAE) variations for existing clustering methods and proposed DACMM is graphically depicted in Figure 4.

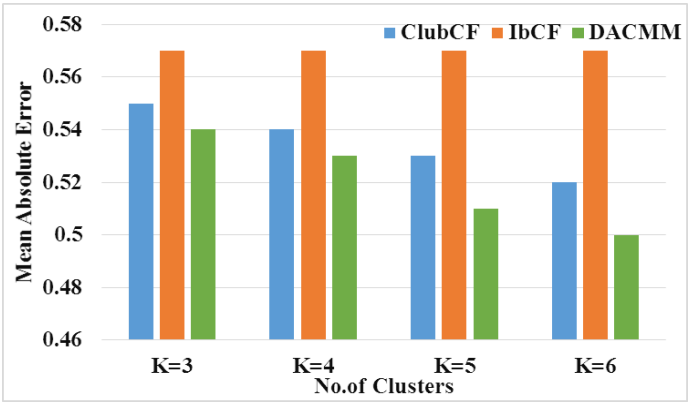


Figure. 4 Mean Absolute Error Analysis

It is observed that the indexing and relevant feature extraction in proposed DACMM yields the better performance compared to the traditional methods respectively. For minimum number of clusters (3), the MAE of the DACMM model is 0.54 and it is 0.5 for maximum clusters (6). The comparison between the proposed DACMM with the existing clustering topologies showed that the proposed DACMM offers 5.26 and the 12.28 % reduction compared to the existing ClubCF approaches.

D. Computational Time

With the variations of size of clusters, the analysis of computational time variations for existing clustering methods and proposed DACMM is graphically depicted in Figure 5.

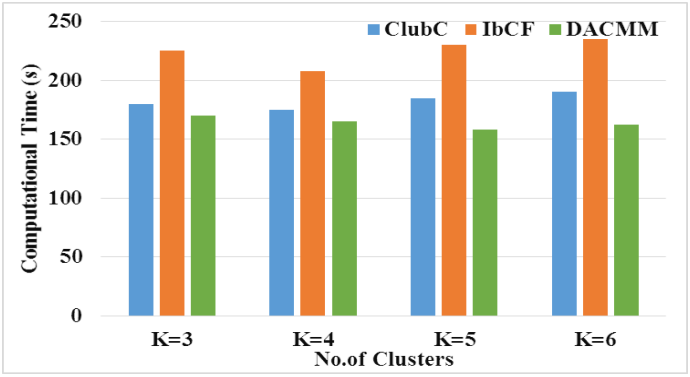


Figure 5. Computational time analysis

It is observed that the indexing and relevant feature extraction in proposed DACMM yields the better performance compared to the traditional methods respectively. For minimum number of clusters (3), the computational time of the DACMM model is 170 and it is 162 secs for maximum clusters (6). The comparison between the proposed DACMM with the existing topologies showed that the proposed DACMM offers

5.56 and the 14.74 % reduction compared to the existing ClubCF approaches respectively.

V. Conclusion and Future Work

The prediction of relevant services based on the user opinions is the major task in the recommendation model. This paper discussed the various limitations such as sparsity, memory management and the cold start in recommendation models. Hence, this paper proposed the Dual Accessing Cache Memory Management (DACMM) to alleviate the issues in the recommendation system. The provision of ratings to the products based on the user experiences and their learning are highly contributed to recommend the interesting product to the end users. With increase in dimensionality of products, the dimensionality of their reviews is increased. This paper employed the feature extraction that utilized the positive, negative reviews with the overall word count to identify the features. The assigning of the index corresponded to the user review with the positive and negative features reduced the size of products into the interesting category. The integration of query processing with the cache memory management reduced the complexity in recommendation operation effectively. The comparative analysis between the proposed DACMM with the existing clustering and cache management models in terms of query latency, hit ratio, Mean Absolute Error (MAE), computational time and precision assured the effectiveness of the proposed DACMM in the review-based recommendation. In future, the proposed work can be extended into investigate the authorization of the user by applying the key-generation/validation methodologies with the optimal cache management concepts.

VI. REFERENCES

[1]. L. Chen, G. Chen, and F. Wang, "Recommender systems based on user reviews: the state of the art," User Modeling and User-Adapted Interaction, vol. 25, pp. 99-154, 2015.

[2]. G. Ganu, Y. Kakodkar, and A. Marian, "Improving the quality of predictions using textual information in online user reviews," Information Systems, vol. 38, pp. 1-15, 2013.

[3]. X. Chen, Z. Zheng, X. Liu, Z. Huang, and H. Sun, "Personalized qos-aware web service recommendation and visualization," IEEE Transactions on Services Computing, vol. 6, pp. 35-47, 2013.

[4]. W. Yasin, H. Ibrahim, N. Hamid, and N. Udzir, "Windows web proxy caching simulation: A tool for simulating web proxy caching under windows operating systems," Journal of Computer Science, vol. 10, pp. 1380-1388, 2014.

[5]. Y. S. Cho, S. C. Moon, S.-p. Jeong, I.-B. Oh, and K. H. Ryu, "Clustering method using item preference based on RFM for recommendation system in u-commerce," in Ubiquitous Information Technologies and Applications, ed: Springer, 2013, pp. 353-362.

[6]. S. G. Esparza, M. P. O'Mahony, and B. Smyth, "Mining the real-time web: a novel approach to product recommendation," Knowledge-Based Systems, vol. 29, pp. 3-11, 2012.

[7]. J. Cao, Z. Wu, Y. Wang, and Y. Zhuang, "Hybrid Collaborative Filtering algorithm for bidirectional Web service recommendation," Knowledge and information systems, vol. 36, pp. 607-627, 2013.

[8]. W. Zhang, G. Ding, L. Chen, C. Li, and C. Zhang, "Generating virtual ratings from chinese reviews to augment online recommendations," ACM Transactions on intelligent systems and technology (TIST), vol. 4, p. 9, 2013.

[9]. M. Daoud, S. Naqvi, and A. Ahmad, "Opinion Observer: Recommendation System on ECommerce Website," International Journal of Computer Applications, vol. 105, 2014.

[10]. W. X. Zhao, J. Wang, Y. He, J.-R. Wen, E. Y. Chang, and X. Li, "Mining Product Adopter Information from Online Reviews for Improving Product Recommendation," ACM Transactions on Knowledge Discovery from Data (TKDD), vol. 10, p. 29, 2016.

[11]. S. L. Addepalli, S. G. Addepalli, M. Kherajani, H. Jeshnani, and S. Khedkar, "A Proposed Framework for Measuring Customer Satisfaction and Product Recommendation for Ecommerce," International Journal of Computer Applications vol. 138, 2016.

[12]. H. Liu, J. He, T. Wang, W. Song, and X. Du, "Combining user preferences and user opinions for accurate recommendation," Electronic Commerce Research and Applications, vol. 12, pp. 14-23, 2013.

[13]. S. S. Htay and K. T. Lynn, "Extracting product features and opinion words using pattern knowledge in customer reviews," The Scientific World Journal, vol. 2013, 2013.

[14]. V. K. Singh, R. Piryani, A. Uddin, and P. Waila, "Sentiment analysis of movie reviews: A new feature-based heuristic for aspect-level sentiment classification," in 2013 International Mutli-Conference on Automation, Computing, Communication, Control and Compressed Sensing (iMac4s), 2013, pp. 712-717.

[15]. L. Chen and F. Wang, "Preference-based clustering reviews for augmenting e-commerce recommendation," Knowledge-Based Systems, vol. 50, pp. 44-59, 2013.

[16]. Z. Zhou, M. Sellami, W. Gaaloul, M. Barhamgi, and B. Defude, "Data providing services clustering and management for facilitating service discovery and replacement," IEEE Transactions on Automation Science and Engineering, vol. 10, pp. 1131-1146, 2013.

[17]. P. Moradi, S. Ahmadian, and F. Akhlaghian, "An effective trust-based recommendation method using a novel graph clustering algorithm," Physica A: Statistical Mechanics and its Applications, vol. 436, pp. 462-481, 2015.

[18]. C.-L. Liao and S.-J. Lee, "A clustering based approach to improving the efficiency of collaborative filtering recommendation," Electronic Commerce Research and Applications, vol. 18, pp. 1-9, 2016.

[19]. N. Korfiatis and M. Poulos, "Using online consumer reviews as a source for demographic recommendations: A case study using online travel reviews," Expert Systems with Applications, vol. 40, pp. 5507-5515, 2013.

[20]. Y. S. Cho, S. C. Moon, S.-p. Jeong, I.-B. Oh, and K. H. Ryu, "Efficient purchase pattern clustering based on SOM for recommender system in u-commerce," in Ubiquitous Information Technologies and Applications, ed: Springer, 2014, pp. 617-626.

[21]. J. Wang, E. Lo, M. L. Yiu, J. Tong, G. Wang, and X. Liu, "Cache design of ssd-based search engine architectures: an experimental study," ACM Transactions on Information Systems (TOIS), vol. 32, p. 21, 2014.

[22]. R. Hu, W. Dou, and J. Liu, "ClubCF: A Clustering-Based Collaborative Filtering Approach for Big Data Application," IEEE Transactions on Emerging Topics in Computing, vol. 2, pp. 302-313, 2014.

[23]. V. Formoso, D. Fernández, F. Cacheda, and V. Carneiro, "Using rating matrix compression techniques to speed up collaborative recommendations," Information retrieval, vol. 16, pp. 680-696, 2013.

[24]. S. Gupta and S. Moharir, "Request Patterns and Caching for VoD Services with Recommendation Systems," arXiv preprint arXiv:1609.02391, 2016.

[25]. M. Chaurasia and C. Satsangi, "Enhancing Proxy Server cache Management using Log Analysis and Recommendations," International Journal of Computer Applications, vol. 113, 2015.

[26]. D. V. S. J. P. J. Sangeetha, "An Efficient Inclusive Similarity Based Clustering (ISC) Algorithm for Big Data," World Congress on Computing and Communication Technologies, 2016.

[27]. Available: https://www.cs.uic.edu/liub/FBS/sentiment-analysis.html