Web Service Recommender Systems: Methodologies, Merits and Demerits

M. Venu Gopalachari

Department of CSE
Chaitanya Bharathi Institute of Technology,
Hyderabad, India
e-mail: venugopal.m07@gmail.com

Abstract—Web services nowadays are considered a consolidated reality of the modern Web with remarkable, increasing influence on everyday computing tasks. Following Service-Oriented Architecture (SOA) paradigm, corporations are increasingly offering their services within and between organizations either on intranets or the cloud. Recommender Systems are the software agents guiding the web services to reach the end user. The aim of this paper is to present the survey of advancements in assisting end users and corporations to benefit from Web service technology by facilitating the recommendation and integration of Web services into composite services.

Keywords- *Web Services, Recommendations, Content based, Collaborative Filtering*

*****

I. INTRODUCTION *(HEADING 1)*

Recommender Systems (RSs) are defined as software applications and tools that provide recommendations to users for various items. Item in recommender systems is a general term which denotes what a system recommends to users [1]. Depending on the type of recommendation needed, the item may be any product or service such as books, music, movies, news, and so on. In fact, personalization plays an essential role in the recommendation process. By utilizing personalization, recommender systems suggest to user products and services that they might be more interested in with less effort. The main goal of recommendation techniques is to estimate ratings for items that are not previously used or rated by a user. Usually, this predicting process is based on previous ratings of the user. Once the unknown ratings are estimated for a given user, the items with the highest predicted ratings can then be recommended. To perform this task, recommendation techniques need an input which is usually provided as a user-item rating matrix, capturing users' preferences. Different techniques from approximation theory, various heuristics, and machine learning are used to estimate unknown ratings. However, according to rating estimation approach, recommender systems are classified into Content-based, Collaborative, and Hybrid approaches [2].

The idea of the content-based approach is based on the similarity of contents between two items. Therefore, to recommend item s to user c, the similarity between the content of s and other items previously rated by c is measured. For example, if news items related to economics are highly rated by user c in a news recommender system, then other news items in economics receive a higher similarity score. In content-based approach, each item is represented by a profile in which its properties are stored. Usually, the properties are extracted from the item's content before starting to calculate the similarities. Common examples of this kind are text-based items such as articles, books, and news. Although content-based recommendation approach is effective in some kinds of recommendation applications, it has significant limitations. Content-based techniques need to process items that have machine readable contents, such as text-based items, to be able to parse and extract item properties and features. It is impractical to manually assign attributes to items, for instance, a video recommendation system dealing with millions of items. These techniques lack the ability to evaluate the quality of items. For example, in article recommendations, if two articles belong to the same subject (i.e. have similar keywords), content-based recommendation methods cannot prioritize them based on the quality of writing, i.e., the well-written vs. the poor one. Over-specialization problem, in which users are only restricted to recommendations related or similar to the items they preferred in the past. That means they never get an opportunity to receive recommendations of items in other tastes which they might be interested in. Finally, cold user problem is another constraint of content-based techniques. The problem arises when a user first joins the system and has no sufficient number of ratings yet. That is because a system is unable to understand user's taste and preferences so that it can recommend relevant items to him.

Collaborative Filtering (CF) methods recommend items liked or preferred by other users who have similar preferences to a given user. To overcome the limitations of content-based methods, both academia and industry widely adopted CF methods, which is the most common approach of recommendation techniques. Collaborative filtering approaches can be classified into two general categories: memory-based (or heuristic-based) and model-based techniques. Memory-based CF methods exploit the user-item rating data to measure similarity between users or items. Typical examples of this class are User-based and Item based collaborative filtering approaches. In user-based approach, the value of an unknown rating r(c,s) by user c for item s is calculated as an aggregate of the ratings by other users (mostly similar users to user c) for items.

Despite the popularity of user-based collaborative filtering approach, there are some significant challenges such as scalability and quality of recommendations. Calculating similarities between millions of users nowadays is impractical in real time. On the other hand, reducing the number of candidate neighbors by some method assists in increasing the computational performance, yet diminishing the quality of recommendations. Therefore, a trade-off is still required, between reducing data to speed up these systems and the resulting quality of recommendations.

The CF approaches are neighborhood approaches based on rating correlation. Intuitively, Neighborhood -based methods are relatively easy to implement. Only one parameter, which is the number of neighbors considered in the prediction, requires tuning. They provide a concise justification for the generated predictions. For instance, in item-based approach, the list of neighbor items along with their associated ratings, are usually presented to the user as a justification. It assists the user to understand the relevance of the recommendations better, and it could also support interactivity feature of the system where users can pick the neighbors of higher importance to them to respect in the recommendation [3]. Neighborhood based systems are efficient. Unlike model-based, they require no costly training phase, which needs to be frequently performed in real-world commercial applications. To reduce the cost of recommendation phase, the nearest neighbors can be pre-computed offline providing proximal instantaneous recommendations. Furthermore, the reduced amount of memory needed to store nearest neighbor lists enable these approaches to scale to large applications, with millions of users and items. Being non-significantly affected by the constant addition of users, items, and ratings, (typically accompanying large commercial applications), offers a high degree of stability to the system. For example, an item-based system needs not to re-train with in responding to new users' requests. Moreover, only the similarities between a newly added item and the existing ones need to be computed.

CF approaches have disadvantages along with the above mentioned advantages which may limit their effectiveness of the user. Since the similarity between two users is based on comparing their ratings for the same items, users are neighbors only if they have rated some common items. This assumption is strongly delimiting the opportunity of users who have rated only a few or no common items. Also, this excludes the items that are unrated by neighbors from recommendation, i.e., lack of coverage. The accuracy of neighborhood -based recommendation approaches suffers from the lack of availability of ratings. In fact, users typically rate only a small proportion of available items, causing the Sparsity problem. Furthermore, this leads to the cold-start problem [4] where users or items newly added to the system may have no ratings at all. With sparsity, two users or items are unlikely to have common ratings. Consequently, in neighborhood -based approaches, a very limited number of neighbors will be considered in the recommendation. Moreover, biased recommendations can also be experienced since similarity weights may be computed using only a small number of ratings.

II. ADVANCED RECOMMENDATION TECHNIQUES

Dimensionality reduction and graph-based methods are the two basic recommendation methods target to overcome the disadvantages of the traditional recommender techniques.

A. Dimensionality reduction methods

The idea of Dimensionality reduction methods is that, in order to deal with the problems of limited coverage and sparsity, users and items are projected into a reduced latent space where the most notable features can be captured. In particular, users and items are compared in a subspace of high-level features, enabling the discovery of more meaningful relations. Accordingly, even if two users have no common rated items, this approach can discover a relation between both, which results in less sensitivity to sparse data. Essentially, within recommender systems, dimensionality reduction are achieved in two ways: 1) decomposition of a user-item rating matrix [5], and 2) decomposition of a sparse similarity matrix [6].

B. Graph-based Methods

In graph-based approaches, the graph structure is used to represent the rating data where nodes can be users, items or both, while edges encode the similarities/ interactions between the users and items. In its simplest model, the data is modeled as a bipartite graph where the two sets of nodes represent users and items, and an edge connects user u to item i if u has a rating for i in the system. The edge can also be weighted by a value of its corresponding rating. Also, the nodes can represent either users or items while an edge connects two if the ratings corresponding to these nodes are sufficiently correlated. In such models, correlation prediction can be used to estimate the rating of a user u for an item i based only on the nodes directly connected to u or i, as in standard approaches. However, Graph-based approaches can include nodes that are not directly connected through allowing the propagation of information along the edges of the graph. Greater weights allow more information to pass through (property of propagation). Besides, the influence of a node on another is negatively correlated with the length of the path between them in the graph (property of attenuation) [7]. In the following section, two of widely used graph based recommendation approaches on graphs, namely: Random Walk and ItemRank approaches are presented.

Random Walk is defined as a Markov chain that describes a sequence of nodes visited by a random walker. If a random walker in state i at time t, then a random variable s(t) contains the current state of the Markov chain at time step t, such that s(t) = i. In graphs, transitive associations in graph-based methods can also be defined within a probabilistic framework, in which the similarity between users or items is evaluated as a probability of reaching these nodes in a random walk. Since the transition probabilities are based only on the current state (i.e., not on the past ones), it is considered a first-order Markov process. Formally, this can be defined by a set of n states and an n by n transition probability matrix P such that the probability of jumping from state i = s(t) to j = s(t + 1) at any time-step t is:

C. Web Service Recommendation

Although a web service can be considered as an item in recommendation approaches, they still have significant differences that should be taken into account when performing certain recommendation tasks. One difference is that Web services have more complicated representational profile than simple products. For example, a movie profile may contain simple textual information such as title, genre, a list of participating actors, production year. In contrast, a Web service profile contains functional properties (Web service operations, inputs, outputs). Further, since the Web service is a piece of code that is usually hosted in a remote server, availability and performance features, remain under no control to users. Therefore, to provide reliable and satisfying recommendations, a recommendation system needs to address a set of non-functional properties (such as quality of services, price, and execution order). It is noticeable that the term Web Service discovery is used interchangeably with WS recommendation especially when taking into account only functional properties during the search process.

In Web service recommendation, many solutions have been proposed to assist users in retrieving their interested services. However, several problems have been identified and tackled such as service selection, service execution, service recommendation, and web service composition. Existing approaches analyze different aspects of Web services, such as service descriptions, execution conditions, and service behaviors. Both functionality properties and non-functionality properties have also been examined. Several Web service presentations such as XML-based, semantic web, graph-based, have also been explored.

III. Related Work

In order to recommend the most relevant web services matching certain user's needs, different approaches have been proposed. They exploit different types of information, i.e., web service descriptions, QoS of services, semantic concepts, and usage patterns. In this section, brief descriptions of some of the existing works under each group are given.

A. Text-based Approaches

In one of the earliest works, Dong et al. [8] proposed a similarity search engine for Web services. The engine analyzes Web service descriptions to extract similarities between inputs, outputs, and operations of a given set of services. It first splits the names of inputs, outputs and operations then cluster them in different concepts. Each input/output is represented by a vector of three elements i = (pi; ci; op), where i is the input, op is a Web service operation, pi is a set of input parameter names, and ci is a set of concepts associated with the parameter names. Whilst, each operation op is denoted by a vector op = (w; f; i; o), where w is the textual description of the Web service, f is the textual description of op; i and o identify input and output parameters respectively. To estimate the similarity between inputs, outputs orWeb service descriptions, TF-IDF measure is computed between their corresponding concepts. Finally, the similarity of Web service operations is computed based on the similarities of vector elements.
Similarly, Platzer et al. [9] proposed an approach to match user's query string with Web service descriptions. First, they collect Web service descriptions in WSDL format from different resources, such as user's uploading, links from websites, or references from UDDI (Universal Description, Discovery, and Integration) repositories. Next, each Web service description _le is parsed to generate a description vector whereby each value indicates the number of times that the word appears in the description. Likewise, user's query string is represented as a vector.
After weighting each term by TF-IDF, the similarity between a query-vector and a document-vector is then computed by Vector Space Model (VSM). Finally, the system recommends the Web services whose descriptions have the highest similarity values with a given query string. Figure 3.1 shows the system architecture. Blake et al. [10] attempted to utilize user's daily routine when recommending Web services. First they capture the text data that a user is working on such as HTML _les, Word documents, File systems, messages, such as SOAP (Simple Object Access Protocol) messages. Second, from the captured data, they extract text strings, which are compared to the operations of available Web services in the third step. They captured four naming tendencies that software designers/developers used to name a Web service. To compute similarities, they applied Levenshtein Distance (LD) and Letter Pairing (LP). In the last step, Web services with highest similarity values are recommended to a user.
The demerits of text based approaches includes polysemy and synonym problem which occurs with many meanings of one word and many expressions of single meaning of the word. It is not guaranteed that service description text is always correct and meaningful known as Web service description accuracy problem. This approach cannot flexibly address cases when a user either types incorrect words or abbreviated words in the query. As most Web service documents are currently described in English, users are also restricted to write their queries in English. Using translation tools to switch to English may add another layer of inaccuracy to user's query and therefore it is not recommended in Web service recommendation.

B. Quality of Service based Approaches

QoS features in Web service recommendation are widely addressed in the literature, as well as Web service search engines, Web service recommender system, Web service reliability prediction, Web service selection, optimal service composition and so on. In [11], Zhang et al. proposed a search engine (WSExpress) that considers both functional and non-functional features of Web services. In functional features, they represent each Web service description by an operation, input, and output. Authors in [12] considered only QoS values of Web services to build a Web service Recommender System. Initially, they built a useritem matrix, where rows represent user IDs and columns represent Web service IDs. Each entry is a vector of QoS values that are observed by the corresponding user for the corresponding Web service. The QoS metrics are round-trip-time and failure-rate of eachWeb service. Once the matrix is created, the similarity between Web services and users using Pearson Correlation Coefficient is then computed.
The probability that user a will probably use a Web service i is finally formulated based on the computed similarities. To enrich the recommendation process, the authors in [14] extended the prediction function used in [13] to a hybrid one that consists of item-based and user-based prediction sub-functions. To control both sub-functions, a parameter _ is added to balance the weight of item-based and user-based prediction functions.

Other approaches also handled QoS of composite Web services. However, their goal is to optimize Web service compositions by selecting the best Web services that satisfy given QoS constraints. The authors in [15] proposed an approach consisting of two models: a combinatorial model and a graph model. The first
defines the problem as a multi-dimension multi-choice knapsack problem (MMKP), while the second model defines the problem as a Multi-Constrained Optimal Path (MCOP) problem. In both models, they use a user-defined utility function of some system parameters in order to optimize application-specifc objectives. By utilizing a Directed Acyclic Graph (DAG), the authors in [16] presented an approach that follows service execution paths in the graph. They addressed the service selection process as an optimization problem which can be solved by linear programming methods.

The authors in [17] proposed an extensible, preference-oriented, open and fair QoS model for Web service selection. Their goal is to allow providers to flexibly add new specifc domain criteria for evaluating the QoS of Web service without changing the underlying computation model. In addition, it can provide means for users to accurately express their preferences without resorting to the complex coding of user profiles. Furthermore, the approach manipulates the QoS information collected from both parties, i.e., provider's site such as service privacy and user's site such as feedbacks. Other approaches focus on transactional QoS criteria, such as Haddad et al. [18]. To create a composite service, a set of Web services are selected based on some transactional properties. For instance, a composite service must support atomic transactions, compensable transactions or it has to terminate after a finite number of execution steps. In general, QoS-based approaches to Web service recommendation consume explicit knowledge, i.e., throughput, bandwidth, execution time, failure rate, privacy or feedback, etc.

C. Semantic-based Approaches

Manikrao et. al. [19] proposed a Web service recommender system based on semantic matching and rating prediction. To describe Web services, they used ontologies written in DAML (DARPA1 Agent Markup Language, recently Web Ontology Language, OWL-S). By matching all semantic attributes ofWeb services, they determine the similarity degree between two services while considering a given threshold. Given a user, the system can predict the rating of a Web service based on user's ratings on similarWeb services in the past. They decide that two services are more similar if their average ratings are less different. Another work that uses DARPA Agent Markup Language to semantically describe Web services is presented in [20]. The authors attempted to address the weakness of WSDL documents in representing some semantic features. To do so, they proposed to match input/output concepts of a service request to the corresponding concepts of service advertisements. They also set specific matching degrees, namely exact, plug in, subsume and fail. Moreover, the output concepts match is given a higher priority than the input concepts match. That means, services whose advertisements are better matched on output concepts are preferred for the recommendation rather than on input concepts.

The authors in [21] applied SVD (singular value decomposition) and LSI (Latent semantic indexing) and conducted some experiments based on precision and recall metrics. Particularly, they represented Web service documents in a matrix of documents and terms. Then, they decomposed the document-term matrix to an approximate matrix using SVD. Finally, to find the closest documents to a given query, the query terms and the document rows in the decomposed matrix are matched. In their proposed approach, authors start by linking a Web service request and Web services descriptions to high-level ontology concepts, before decomposing and matching them. The steps involved are pre-processing service requests and determine the overall search category of Web services and based on high-level ontology concept of the Web service request, they retrieve relevant indexed service descriptions from UDDI. Then the associated low-level ontology concepts are retrieved from an ontology framework based on the terms of a given request. They expand the request and transform the web service descriptions into term-document matrices, With the retrieved concepts. The resulted matrices are decomposed using SVD and hey match the matrices with the request vector in order to infer a final similarity score.

The authors in [22] applied AI planning techniques to help generate semantically based Web service compositions. They use a collection of Web services and composition templates that are both described in OWL-S. First, they translate the OWL-S process descriptions and composition template to Hierarchical Task Network (HTN) domains and initial task networks, respectively. In order to compute Preference-Based Web service compositions, they use the best-first algorithm with two heuristics namely, Optimistic Metric Function (OM) and the Look-ahead Metric Function (LA). The functions are used to perform branch-and-bound pruning in order to effectively discard all unpromising nodes from the search space. Their work is aimed at enhancing the quality of compositions and the speed of generation by applying optimization techniques within the composition process.

Although using ontologies in representing Web services provides better utilization of functional and non-functional properties, semantic-based approaches still suffer from significant limitations in reality. Specifically, due to the need of domain experts, the process of creating and publishing ontology annotated content remains time-consuming and error-prone. Another limitation that might face rating-based solutions such as [19] is the new item problem. In which new items that have not yet rated are not recommended; in contrast, items with higher ratings are always recommended. By exploiting latent semantics hidden in Web service descriptions, other semantic-based approaches can generate recommendations without creating ontologies or asking any effort from users.

D. Usage-based Approaches

In this scheme, the past usage data is used as a rich resource exposing users' interests. However, there are still few attempts that take these data into account for Web service recommendation. Birukou et al. [23] proposed a Web service recommender system that utilizes the past usage data. They introduced a new concept called 'Implicit Culture', in which if a user has a similar request with other members of the community, he will be suggested operations that were used by those members. Therefore, they record the usage of users together with their requests. When a new request is received, they will recommend the requester Web service operations that are used by other users with similar requests. VSM (Vector space model), TF-IDF, and WordNet-based semantic similarity are used to compute the similarities. In fact, the authors only used past usage data in their proposed rule-based theory. The similarity between requests is computed using text-based approaches. Hence, they miss the correlation between users and Web services. Also, the shortcomings of text-based approaches are met while matching users' requests.

IV. Graph-based Recommendation

In previous sections, WS recommendation techniques are presented in terms of the type of information they handle. In this section, the focus is on the design aspect of the recommendation approach. Therefore, this section mainly presents some works that follow graph data model which our work is based on as an underline recommendation model. Usually, a graph-based recommendation approach includes two steps: constructing a graph from available data and ranking item nodes for a given user. Bogers et al. [24] proposed a Movie Recommendation approach using Random Walks over the Contextual Graph. Their approach uses original random walk on graph by considering the user and item context (genre, director, actor, etc.). They presented the recommendation process as a browsing process of a user on a movie database website. In fact, their approach ignores an important factor of recommending movies which is the selection context, where a user chooses to watch a movie. In other words, different user preferences can be realized depending on where and with whom the user selects a movie. However, the author does not show any validation of the proposed approach.

Lee et al. [25] propose a bipartite graph to model the interactions between users and items. They define a recommendation factor set, F, in which each factor f2F is a combination of multidimensional data. Then they connect item nodes with each node that corresponds to a certain recommendation factor. Although their work utilizes contextual information, there are no clear principles of how to define such a crucial recommendation factor set F.

Based on the framework, they proposed three recommendation methods: Co-occurrence Graph-based Method, Contextual Bipartite Graph-based Method, and Heterogeneous Graph-based Method. To validate their methods, they compared them with other existing methods in terms of recommendation accuracy. However, they only assessed their work regarding recommendation performance (i.e. accuracy) and ignore other criteria such as recommendation effectiveness, diversity, novelty, etc.

Finally, although using graph based approach is considered an effective technique to address the recommendation process, the construction of the graph, its nodes and edges, in addition to the ranking method of nodes, considering their direct and indirect relations, needs to be carefully dealt with. That is to ensure obtaining high recommendation accuracy of the final results along with overcoming essential recommendation problems such as data sparsity.

V. Conclusion

This paper presented the review on web service recommendation approaches. It first overviewed some of the most popular approaches used in recommender systems, including content-based, collaborative filtering and hybrid methods. As collaborative filtering methods are widely used, their strengths and limitations are also presented with an insight to address some of their limitations in this thesis. The problem addressed in the adopted approach is one of a larger set of problems that make up the Web service recommendation problem. Then, this paper also presented an overview of more advanced recommendation techniques including dimensionality reduction methods and Graph based methods. In Web service recommendation, existing approaches examine different properties of Web services, i.e., descriptions and requests, QoS, semantic descriptions and historical usage data. In general, text-based approaches suffer from synonymy and polysemy problems; QoS-based approaches focus only on realization properties, semantic-based approaches are error-prone and time-consuming and existing usage-based approaches ignore the correlation between users and web services. In graph based recommendation approach, the construction of the graph, its nodes and edges, besides the ranking method of nodes, featuring direct and indirect relations, need to be carefully dealt with. That is to obtain high recommendation accuracy for the final results and also to overcome essential recommendation problems such as data sparsity.

References

[1] Francesco Ricci, Lior Rokach, and Bracha Shapira, “Introduction to recommender systems handbook. In Recommender Systems Handbook”, pages 1-35, Springer, 2011.

[2] Gediminas Adomavicius and Alexander Tuzhilin, “Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions”, IEEE Transactions on Knowledge and Data Engineering, 17(6), Pp:734-749, 2005.

[3] Robert Bell, Yehuda Koren, and Chris Volinsky, “Modeling relationships at multiple scales to improve accuracy of large recommender systems”, In proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining, pages 95-104, ACM, 2007.

[4] Andrew I Schein, Alexandrin Popescul, Lyle H Ungar, and David M Pennock, “Methods and metrics for cold-start recommendations”, In Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval, pages 253-260. ACM, 2002.

[5] Gene H Golub and Charles F Van Loan, “Matrix computations”, volume 3. JHU Press, 2012.

[6] Ken Goldberg, Theresa Roeder, Dhruv Gupta, and Chris Perkins, “Eigentaste: A constant time collaborative filtering algorithm”, Information Retrieval, 4(2),Pp:133-151, 2001.

[7] Christian Desrosiers and George Karypis, “A comprehensive survey of neighborhood-based recommendation methods”, In Recommender systems handbook, pages 107-144. Springer, 2011.

[8] Xin Dong, Alon Halevy, Jayant Madhavan, Ema Nemes, and Jun Zhang, “Similarity search for web services”, In Proceedings of the Thirtieth international conference on Very large data bases,Volume 30, pages 372-383, 2004.

[9] Christian Platzer and Schahram Dustdar, “A vector space search engine for web services”, In Third IEEE European Conference on Web Services, pages 9, 2005.

[10] M Brian Blake and Michael F Nowlan, “A web service recommender system using enhanced syntactical matching”, In IEEE International Conference on Web Services, pages 575-582. IEEE, 2007.

[11] Yilei Zhang, Zibin Zheng, and Michael R Lyu, “Wsexpress: A qos-aware search engine for web services” In IEEE International Conference on Web Services, pages 91-98. IEEE, 2010.

[12] Zibin Zheng, Hao Ma, Michael R Lyu, and Irwin King, “Wsrec: A collaborative filtering based web service recommender system”, IEEE International Conference on web services, Pp 437-444, 2009.

[13] Yechun Jiang, Jianxun Liu, Mingdong Tang, and Xiaoqing Liu, "An effective web service recommendation method based on personalized collaborative filtering", IEEE International Conference on web services, Pp:211-218, 2011.

[14] Zibin Zheng and Michael R Lyu, "Collaborative reliability prediction of service oriented Systems", In Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering, Volume 1, Pp:35-44, 2010.

[15] Tao Yu, Yue Zhang, and Kwei-Jay Lin, "Efficient algorithms for Web services selection with end-to-end QoS constraints", ACM Transactions on the Web (TWEB), Vol.1, 2007.

[16] Liangzhao Zeng, Boualem Benatallah, Anne HH Ngu, Marlon Dumas, Jayant Kalagnanam, and Henry Chang, "QoS-aware middleware for web services composition", IEEE Transactions on Software Engineering, Vol.3(5), Pp:311-327, 2004.

[17] Yutu Liu, Anne H Ngu, and Liang Z Zeng, "QoS computation and policing in dynamic web service selection", In Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters, Pp: 66-73, 2004.

[18] Joyce El Haddad, Maude Manouvrier, Guillermo Ramirez, and Marta Rukoz, "QoS-driven selection of web services for transactional composition", International Conference on Web Service, Pp:653-660, 2008.

[19] Umardand Shripad Manikrao and TV Prabhakar, "Dynamic selection of web services with recommendation system", International Conference on Next Generation Web Services Practices, 2005.

[20] Massimo Paolucci, Takahiro Kawamura, Terry R Payne, and Katia Sycara, "Semantic matching of web services capabilities", The Semantic Web, pages 333-347, 2002.

[21] Chen Wu, Vidyasagar Potdar, and Elizabeth Chang, "Latent Semantic Analysis, The Dynamics of Semantics Web Services Discovery", Advances in Web Semantics, Pp 346-373, Springer, 2009.

[22] Shirin Sohrabi and Sheila A McIlraith, "Preference-based web service composition: A middle ground between execution and search", In The Semantic Web, Pp: 713-729, 2010.

[23] Aliaksandr Birukou, Enrico Blanzieri, Vincenzo D'Andrea, Paolo Giorgini, and Natallia Kokash, "Improving web service discovery with usage data", Software, IEEE, Vol. 24(6), Pp:47-54, 2007.

[24] Toine Bogers, "Movie recommendation using random walks over the contextual graph", In Proceedings of the 2nd workshop on context-aware recommender systems, 2010.

[25] Sangkeun Lee, Sang-il Song, Minsuk Kahng, Dongjoo Lee, and Sang-goo Lee, "Random walk based entity ranking on graph for multidimensional recommendation", In Proceedings of the fifth ACM conference on Recommender systems, RecSys '11, pages 93-100, 2011.