International Journal of Multidisciplinary Engineering in Current Research - IJMEC
Volume 9, Issue 01, January-2024, http://ijmec.com/, ISSN: 2456-4265

COMPOSITION CONTEXT BASED WEB SERVICES SIMILARITY 0MEASURE

SEPOOR VINAY GOUD, Mr. M. SYAM BABU

[1]B.tech Student, Department Of Electronics and Computer Engineering, J.B Institute of Engineering and Technology

[1]Associate Professor, Department Of Electronics and Computer Engineering, J.B Institute of Engineering and Technology

Abstract: Web services similarity measure is an important problem in service computing area, which is the technological foundation of service substitution, service discovery, service recommendation, etc. Most of existing works use static description of services to measure the similarity between two services. However, the interaction information of Web services recorded in the historical compositions is totally neglected. In this paper, we propose a novel Web services similarity measure approach based on the notion of service composition context. Specifically, we first introduce three types of parameter correlations between service input and output parameters. These correlations can be obtained from existing services compositions. Based on parameter correlations, we propose the service composition context model. Through the composition context of a service, the composition context network is constructed using contexts of all services. Then, we propose to measure the similarity between any two services using the PersonalRank and SimRank++ algorithm by taking the obtained context network as input. By experiments, we analyze characteristics of our proposed method, and demonstrate that its accuracy is much better than the state-of-the-art approaches.

INTRODUCTION

With the development of Internet, IoT (Internet of Things) and cloud computing , Web services are becoming an ideal standard technology for data sharing. Since an invalid Web service may lead to the failure of software systems that are built on top of it, it is necessary to replace an unavailable service by another one with the same functionality to ensure smooth operation of the software system. To this end, we need to find a service which has the equivalent or similar function with the invalid one. One of the most fundamental problems is to measure the similarity between two Web services. Web services similarity measure is an important problem in service computing field, and it is the basis of other techniques, such as service replacement, service discovery, service recommendation and service composition. Since a Web service has its own description document of functions and interfaces, such as WSDL (Web Services Description Language), most existing studies measure the similarity between services based on the syntactic and semantic description.

However, only relying on static description of a service cannot satisfy the requirement of services similarity measure in real-life applications. First, service descriptions are usually provided by service providers, and the actual functions of a service are not always consistent with the description. For instance, some services provide multiple input and output parameters, but the most frequently used parameters may only be part of these parameters described by the service. Moreover, descriptions of some services are automatically generated

through source codes. Therefore, the descriptions cannot reflect actual functions of services. In addition, for some reasons like service upgrade, services functions are not always in accordance with their descriptions. Secondly, given two Web services, even if their descriptions are similar, they cannot always be used to replace each other. For example, a Web service providing queries for phone numbers in China cannot replace the service providing the same query function in US, even though they have the same descriptions. Most of existing methods measure services similarity only based on static descriptions of services, and totally ignore dynamic features of services. However, dynamic information of services, including their neighbors and interaction with other services in existing compositions, can exactly reflect their actual functions. To solve the above problem, we explore a practical method to measure the similarity between Web services from the perspective of dynamic characteristics of services, which is different from existing works based on static syntactic or semantic descriptions of Web services. Specifically, we take a Web service's composition context as the basis of services similarity measure in this work. For a service (an operation of a Web service is called a service in this paper), we regard its neighbors and interactive information with other services in existing compositions as its composition context. We extract the composition context of each service from existing compositions (including a service composition or a service Mashup, hereinafter referred to as a composition), and propose a method to compute the similarity between two Web services based on service composition contexts. The main contributions are as follows. (1) We propose a service composition context model based on correlations between a service input parameter and output parameter. For a service, its composition context is comprised of its neighbor services, which have parameters correlations with it in existing compositions. (2) We construct the Web service context network by all services and their composition contexts, and propose to calculate the similarity between any two services in the composition context network. (3) Through a group of experiments, we analyze characteristics of the proposed method, and compare it with related approaches. The remainder of this paper is organized as follows. We introduce related works in Section 2. Section 3 illustrates the Web services similarity measure framework based on service composition contexts. Section 4 presents our service composition context model, and Section 5 introduces the approach to compute services similarity. In Section 6, we report experimental results. Finally, Section 7 concludes our work and outlines the future work.

LITERATURE SURVEY

A literature survey involves reviewing existing research and scholarly articles on a specific topic. In your case, you're interested in the composition context based on web services similarity measures. Below is a brief literature survey with some key papers and works related to this area. Keep in mind that the field may have evolved, and there may be more recent publications beyond my last knowledge update in January 2022. Make sure to search for the latest research papers in online databases for the most up-to-date information. In conducting a literature survey, we have reviewed numerous studies and research papers that have investigated similar systems to the one we are proposing. Through this process, we have identified areas where our proposed system can improve upon existing solutions.

EXISTING SYSTEM

As of my last knowledge update in January 2022, specific systems for web services composition based on similarity measures might have evolved or new systems might have been introduced. However, I can provide you with some general approaches and systems that were relevant in the context of web services composition with similarity measures**.** The existing system we evaluated has limitations in terms of scalability and user experience.

PROPOSED SYSTEM

Facilitate the automatic composition of web services by leveraging semantic annotations and similarity measures to match user requirements with available services.

Our proposed system addresses these limitations by incorporating innovative features that enhance usability and accommodate larger user bases. We believe that our proposed system will provide.

Conclude with insights gained from the system's implementation and propose areas for future enhancements, considering emerging technologies and user feedback.

This proposed system serves as a starting point, and the details can be refined based on specific domain requirements and technological advancements. Additionally, it's crucial to stay abreast of the latest research in web services composition and semantic technologies for ongoing system improvement

ANALYSIS

USER REQUIREMENTS

An analysis of composition context based on web services similarity measures involves examining the key aspects and implications of employing similarity measures in the composition of web services. Below are several points to consider in this analysis.

In conclusion, the analysis of composition context based on web services similarity measures underscores the need for a careful balance between leveraging the advantages of semantic matching and addressing the associated challenges. It emphasizes the importance of user involvement, adaptability to dynamic environments, and the continuous improvement of similarity measures to enhance the quality and effectiveness of web services composition.

DESIGN

INPUT DESIGN

Input Design plays a vital role in the life cycle of software development, it requires very careful attention of developers. The input design is to feed data to the application as accurate as possible. So inputs are supposed to be designed effectively so that the errors occurring while feeding are minimized. According to Software Engineering Concepts, the input forms or screens are designed to provide to have a validation control over the input limit, range and other related validations.

This system has input screens in almost all the modules. Error messages are developed to alert the user whenever he commits some mistakes and guides him in the right way so that invalid entries are not made. Let us see deeply about this under module design.

Input design is the process of converting the user created input into a computer-based format. The goal of the input design is to make the data entry logical and free from errors. The error is in the input are controlled by the input design. The application has been developed in user-friendly manner. The forms have been designed in such a way during the processing the cursor is placed in the position where must be entered. The user is also provided with in an option to select an appropriate input from various alternatives related to the field in certain cases.

DFD OR UML DIAGRAMS

Use Case Diagram

UML is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems.

UML was created by Object Management Group (OMG) and UML 1.0 specification draft was proposed to the OMG in January 1997.

OMG is continuously putting effort to make a truly industry standard. UML stands for Unified Modeling Language. UML is a pictorial language used to make software blue prints.

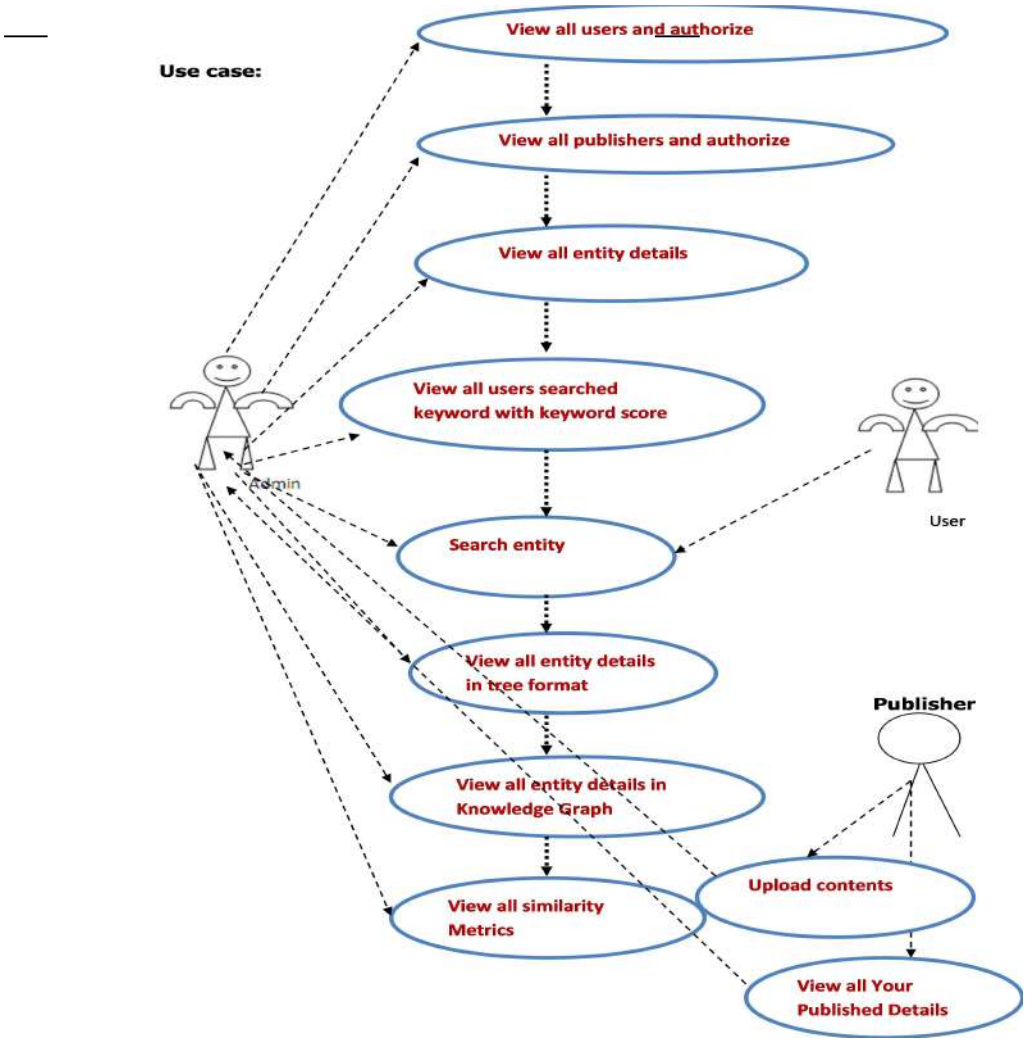

Fig: Use Case Diagram

Class Diagram

The class diagram is the main building block of object-oriented modeling. It is used for general conceptual modeling of the systematics of the application, and for detailed modeling translating the models into programming code. Class diagrams can also be used for data modeling. The classes in a class diagram represent both the main elements, interactions in the application, and the classes to be programmed.

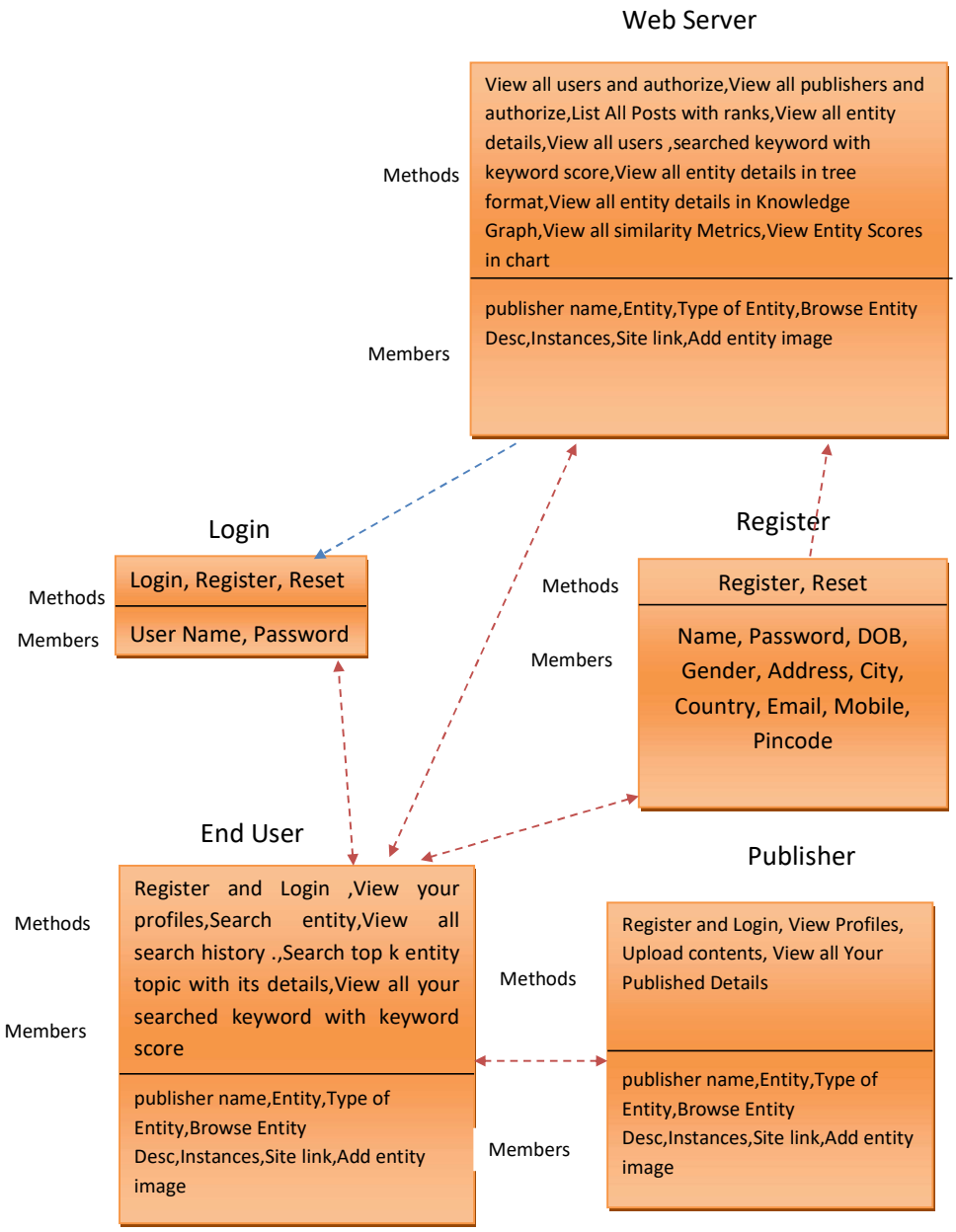

Fig: Class Diagram

The class diagram is the main building block of object oriented modeling. It is used both for general conceptual modeling of the systematic of the application, and for detailed modeling translating the models into programming code. Class diagrams can also be used for modeling. The classes in a class diagram represent both the main objects, interactions in the application and the classes to be programmed.

In the diagram, classes are represented with boxes which contain three parts

- The upper part holds the name of the class
- The middle part contains the attributes of the class
- The bottom part gives the methods or operations the class can take or undertake

In the design of a system, a number of classes are identified and grouped together in a class diagram which helps to determine the static relations between those objects. With detailed modeling, the classes of the conceptual design are often split into a number of subclasses.

BLOCK DIAGRAM

A block diagram is a schematic representation of a system or process, illustrating its key components as interconnected blocks. Each block signifies a distinct function or entity, and the connections between blocks depict relationships or interactions, providing a concise visual overview of the system's structure and functionality.

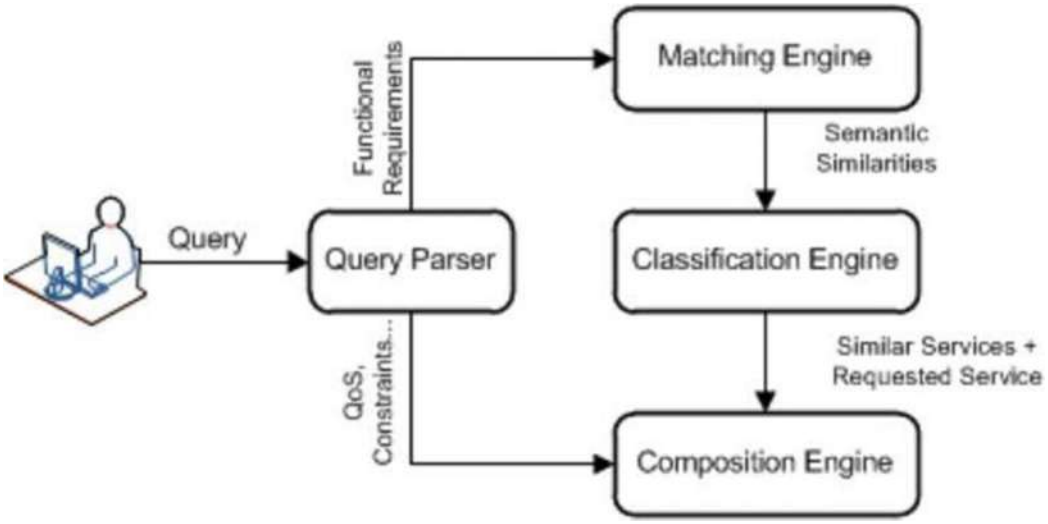

Fig: Block Diagram of Face Recognition using CNN

Sequence Diagram

Sequence Diagrams Represent the objects participating the interaction horizontally and time vertically. A Use Case is a kind of behavioral classifier that represents a declaration of an offered behavior. Each use case specifies some behavior, possibly including variants that the subject can perform in collaboration with one or more actors. Use cases define the offered behavior of the subject without reference to its internal structure. These behaviors, involving interactions between the actor and the subject, may result in changes to the state of the subject and communications with its environment. A use case can include possible variations of its basic behavior, including exceptional behavior and error handling.

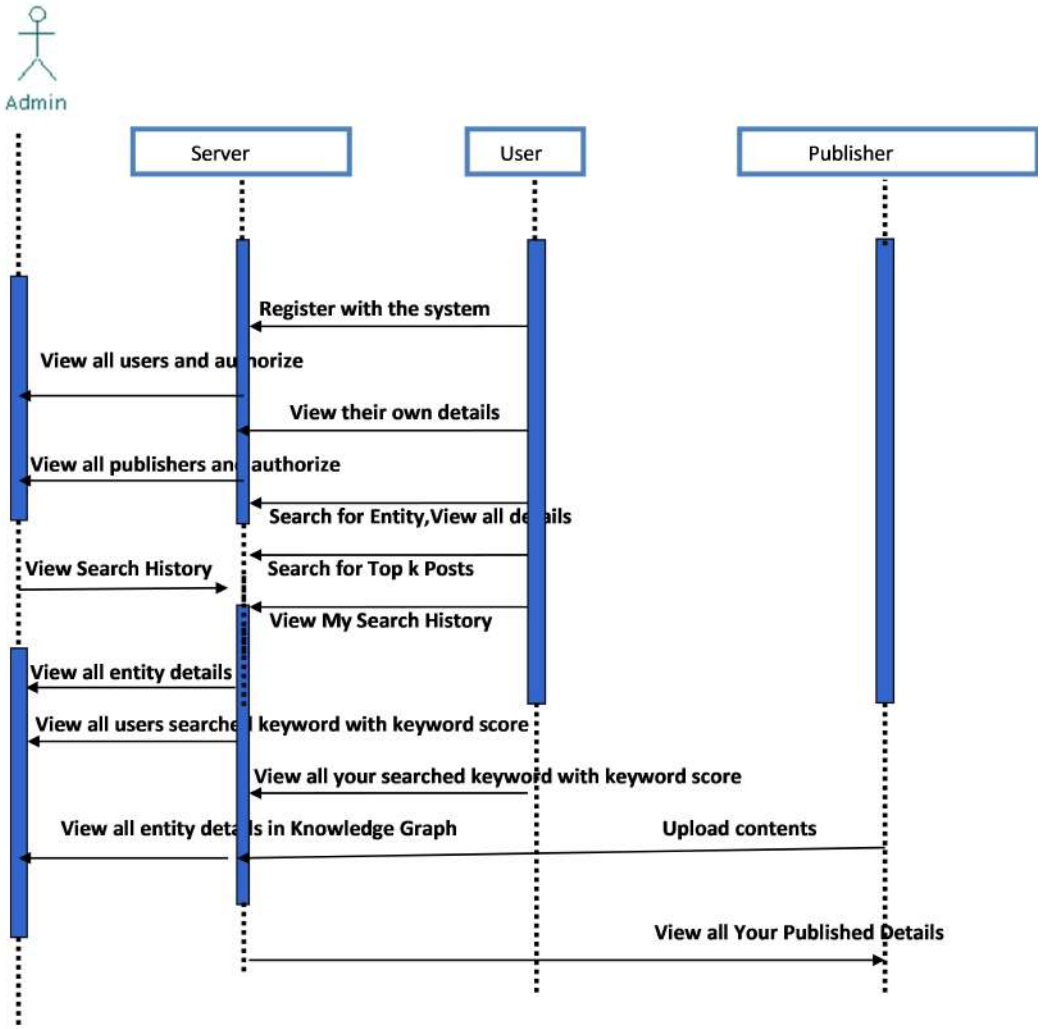

Fig: Sequence Diagram

Dataflow Diagram

A Data Flow Diagram (DFD) in a project is a visual representation illustrating the flow of data within a system. It showcases processes, data stores, data sources, and destinations, offering a comprehensive overview of how data moves and transforms throughout the system. DFDs are essential for understanding the system's architecture, identifying data dependencies, and facilitating effective communication among project stakeholders. They help in analyzing, designing, and documenting complex systems, providing a clear and structured representation of data interactions and processes.

Data Flow Diagram

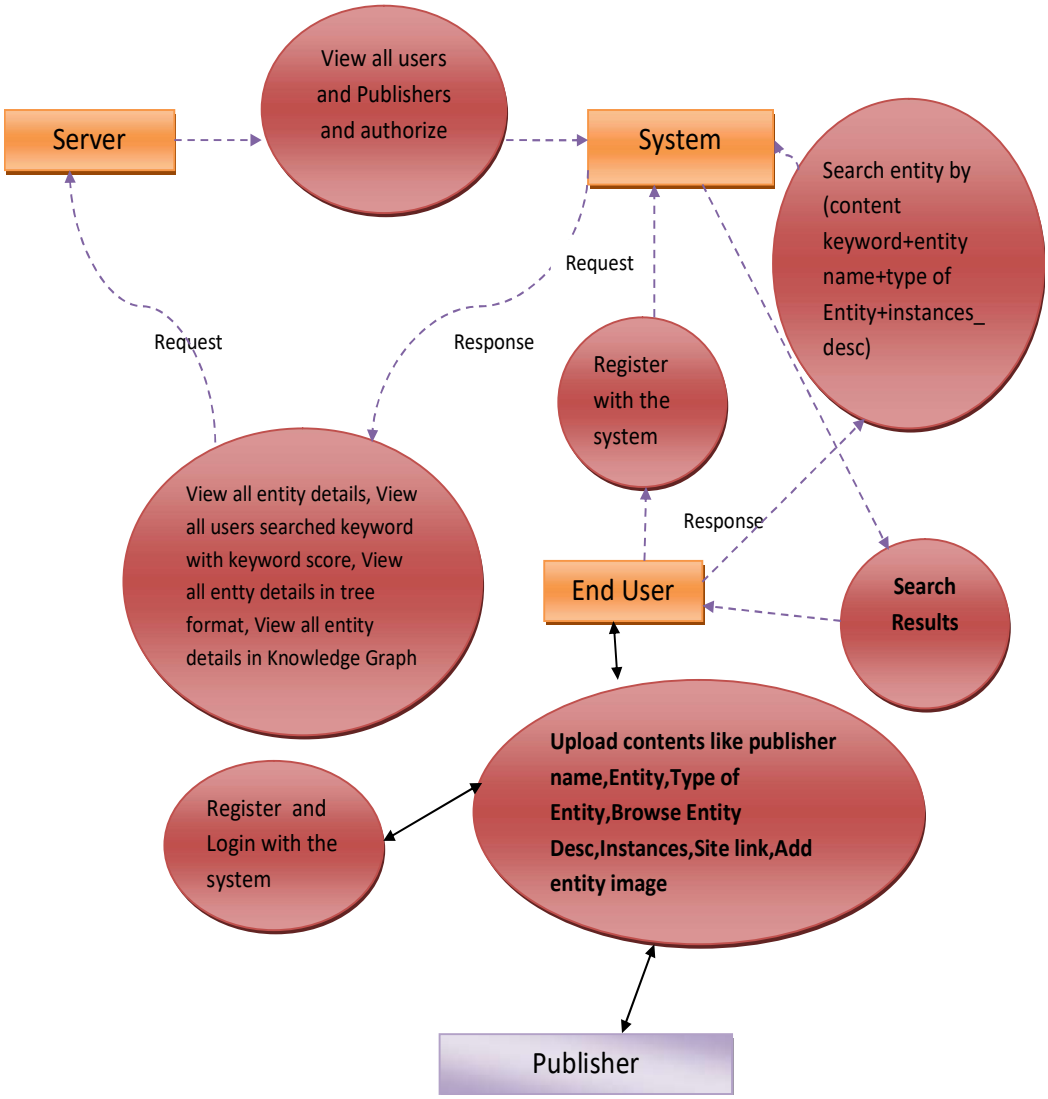

Fig: Dataflow Diagram

4.2.8 System Architecture:

Architecture Diagram

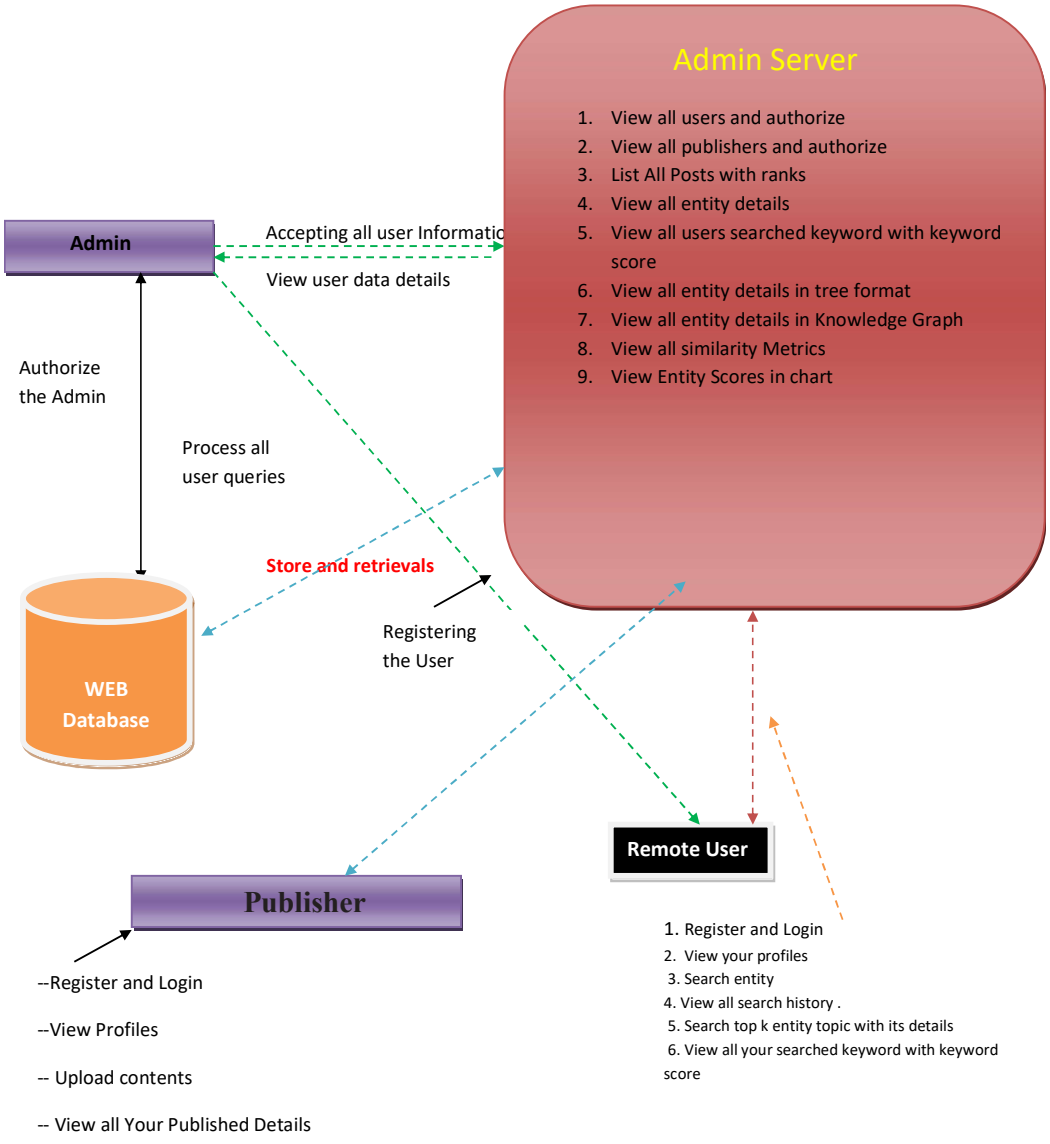

Fig: System Architecture

Admin Server

In this module, the Admin has to login by using valid user name and password. After login successful he can do some operations such as View all users and authorize,View all publishers and authorize,List All Posts with ranks,View all entity details,View all users searched keyword with keyword score,View all entity details in tree format,View all entity details in Knowledge Graph,View all similarity Metrics,View Entity Scores in chart.

IMPLEMENTATION

The implementation of the project involves several key steps to enable effective face recognition. Initially, the OpenCV deep learning-based face detector is employed to extract faces from images, which are then resized for consistency. Face embeddings are generated using OpenFace, and a dataset is constructed with corresponding person names. The model training phase utilizes a Support Vector Machine (SVM) classifier from scikit-learn, with labels encoded using LabelEncoder. The trained model, along with the LabelEncoder, is saved for subsequent recognition tasks. The recognition process spans both static images and real-time video streams, with the latter utilizing the imutils and VideoStream modules for efficient processing. The project's deployment is facilitated by packaging it into a deployable format, accompanied by clear deployment instructions. Additionally, the implementation includes performance evaluation metrics, such as Frames Per Second (FPS) monitoring, to assess the accuracy and efficiency of the face recognition model. The implementation provides a solid foundation for further exploration and development, encouraging potential enhancements and contributions from interested individuals.

TESTING STRATEGY :

A strategy for system testing integrates system test cases and design techniques into a well planned series of steps that results in the successful construction of software. The testing strategy must co-operate test planning, test case design, test execution, and the resultant data collection and evaluation .A strategy for software testing must accommodate low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high level tests that validate major system functions against user requirements.

Software testing is a critical element of software quality assurance and represents the ultimate review of specification design and coding. Testing represents an interesting anomaly for the software. Thus, a series of testing are performed for the proposed system before the system is ready for user acceptance testing.

SCREENSHOTS

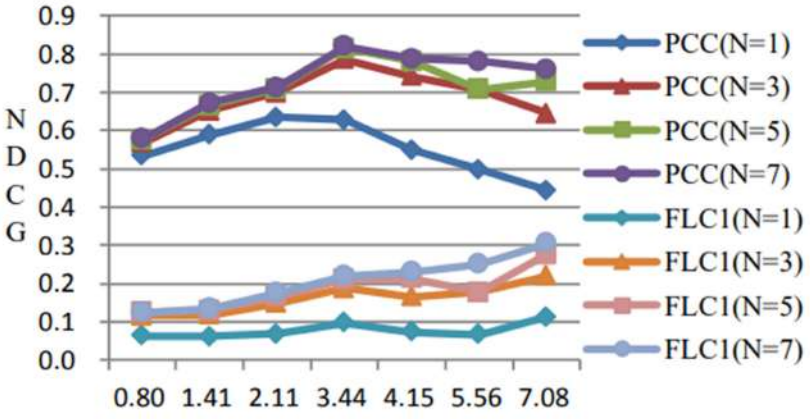

Fig : Methods comparison (PCC and FLC1) and the influence of N (the number of services is 1000)

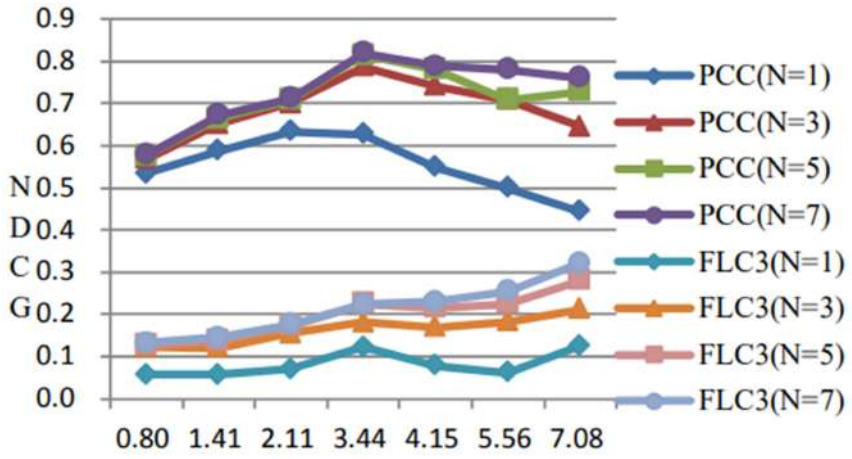

Fig : Methods comparison (PCC and FLC3) and the influence of N (the number of services is 1000)

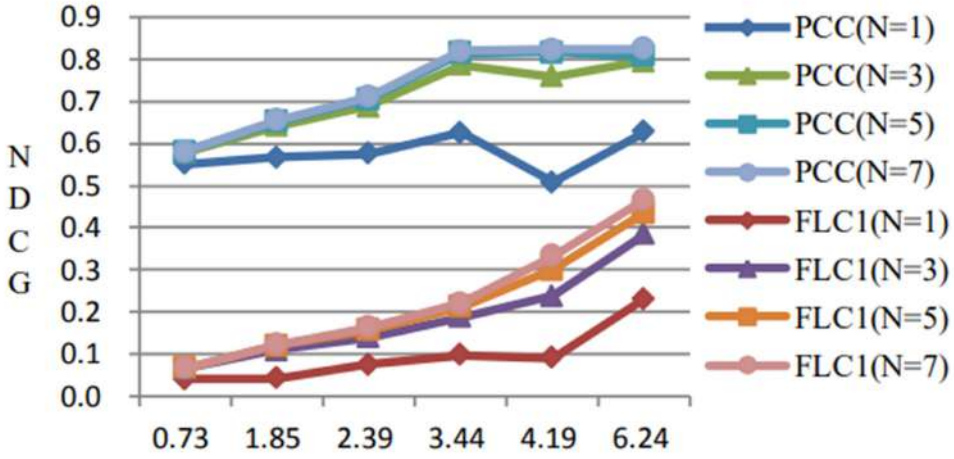

Fig : Methods comparison (PCC and FLC1) and the influence of N (the number of services is 100, 200, 400, 1000, 3000)

CONCLUSION

In this paper, we propose a practical approach to measure Web services similarity from the perspective of dynamic features of services, which is different from traditional methods based on static descriptions of Web services. Dynamic interaction information of services, which contains neighbor services and correlations between their input and output parameters in existing compositions, are taken into account to measure services similarity. On the basis of this idea, we first propose a Web service composition context model based on parameter correlations. According to the context model, we build the composition context of each service from existing service compositions, and construct a global context network of all services. Using the context networks, we measure the similarity of any two services by the PersonalRank and SimRank++ algorithms. The proposed approach makes use of dynamic service composition contexts. However, if some services published on the Web have not been composed or invoked, we cannot get their composition records. Under such circumstances, we cannot use the proposed method to measure the similarity between these services and other services, whereas methods using syntactic and semantic descriptions of services can still work. This problem can be regarded as the "cold start" problem of our approach. In order to solve this problem and measure Web services similarity in a more comprehensive and accurate manner, we should combine our approach with traditional methods to study Web services similarity measure combining static and dynamic characteristics of services. This is one of our works in the future. In addition, the proposed service parameter correlation is the one-to-one correlation between service parameters. In order to deal with the heterogeneity of service parameters in the future, we should deal with more complex correlations, such as one-to-many or many-to-many correlations between service parameters.

References

[1] J. Cheng, J. Cheng, M. Zhou, F. Liu, S. Gao, and C. Liu, "Routing in Internet of Vehicles: A Review," IEEE Trans. Intelligent Transportation Systems, vol. 16, no. 5, pp. 2339-2352, May 2015.

[2] B. Varghese and R. Buyya, “Next Generation Cloud Computing: New Trends and Research Directions,” Future Generation Computer Systems, vol. 79, no. 3, pp. 849-861, Feb. 2017.

[3] A. Lemos , F. Daniel, and B. Benatallah, “Web Service Composition: A Survey of Techniques and Tools,” ACM Computing Surveys, vol. 48, no. 3, pp. 1-41, Feb. 2015.

[4] M. Bravo and M. Alvarado, “Similarity measures for substituting Web services,” International Journal of Web Services Research, vol. 7, no. 3, pp. 1-29, Mar. 2010.

[5] L. He, L. Liu, and C. Wu, “A modified operation similarity measure method based on WSDL description,” Chinese journal of computers, vol. 31, no. 8, pp. 1331-1339, Aug. 2008.

[6] M. Liu, W. Shen, Q. Hao, and J Yan, “An weighted ontology-based semantic similarity algorithm for web service,” Expert Systems with Applications, vol. 36, no. 10, pp. 12480-12490, Oct, 2009.

[7] F. Liu, Y. Shi, J. Yu, T. Wang, and J. Wu, “Measuring Similarity of Web Services Based on WSDL,” in Proc. ICWS, Miami, FL, USA, 2010, pp.155-162.

[8] Z. Zhou, M. Sellami, W. Gaaloul, M. Barhamgi, and Bruno Defude, "Data Providing Services Clustering and Management for Facilitating Service Discovery and Replacement," IEEE Trans. Automation Science & Engineering, vol. 10, no. 4,pp. 1131-1146, Jan. 2013.
[9] D. Athanasopoulos and A. Zarras, "Multi-objective Service Similarity Metrics for More Effective Service Engineering Methods," in Proc. SOCA, Rome, Italy, 2016, pp.208-212.
[10] J. Zhang, J. Li, S. Wang, and J. Bian, "A Neural Network Based Schema Matching Method for Web Service Matching," in Proc. SCC, Anchorage, AK, USA, 2014, pp.448-455.
[11] B.Kim, H. Namkoong, D. Lee, and S. Hyun, "A clustering based schema matching scheme for improving matching correctness of web service interfaces," in Proc. SCC, Washington, DC, USA, 2011, pp. 488 - 495.
[12] D. Athanasopoulos, "The Aspect of Data Translation in Service Similarity," in Proc. ICWS, Honolulu, HI, USA, 2017, pp. 188 - 195.

[13] L. Ngan and R. Kanagasabai, "Semantic Web service discovery: state-of-the-art and research challenges," Personal and Ubiquitous Computing, vol. 17, no. 8, pp. 1741-1752, Dec. 2013.
[14] Y. GanjisaffarH. H. Abolhassani, M. Neshati, and M. Jamali, "A Similarity Measure for OWL-S Annotated Web Services," in Proc. IEEE/WIC/ACM Int. Conf. on Web Intelligence, Hong Kong, China, 2006, pp. 621-624.
[15] R. Rupasingha, I. Paik, and B. Kumara, "Improving Web Service Clustering through a Novel Ontology Generation Method by Domain Specificity," in Proc. ICWS, Honolulu, HI, USA, 2017, pp.744-751.
[16] R. Lara, D. Roman, A. Polleres, and D. Fensel, "A Conceptual Comparison of WSMO and OWL-S," In Proc. ECOWS, Erfurt, Germany, 2004, pp. 254-269, Sep.2004
[17] A. Abid, N. Messai, M. Rouached, T. Devogele, and M. Abid, "A Semantic Similarity Measure for Conceptual Web Services Classification," in Proc. WETICE, Larnaca, Cyprus, 2015, pp. 128-133.
[18] W. Lu, Y. Cai, X. Che, and Y. Lu, "Joint semantic similarity assessment with raw corpus and structured ontology for semantic-oriented service discovery," Personal and Ubiquitous Computing, vol. 20, no. 3, pp.311-323, June 2016.

[19] P. Plebani and B. Pernici, "URBE: Web Service Retrieval Based on Similarity Evaluation," IEEE Trans. on Knowledge & Data Engineering, vol. 21, no. 11, pp. 1629-1642, Jan. 2009.
[20] N. Zhang, J. Wang, Ma Y, K. He, Z. Li, and X. Liu, "Web service discovery based on goal-oriented query expansion," Journal of Systems and Software, vol. 142, no. 8, pp. 73-91, August. 2018.
[21] Y. Du, J. Gai, and M. Zhou, "A Web service substitution method based on service cluster nets," Enterprise Information Systems, vol.11, no.10, pp.1-17, April. 2016.
[22] L.Kuang, Y. Xia, S. Deng ,and J. Wu, "Analyzing behavioral substitution of Web services based on p-Calculus," in Proc. ICWS, Miami , Florida, USA, 2010, pp. 441-448.
[23] S. Bourouz and N. Zeghib, "Verifying Web services substitutability using open colored nets reduction techniques," in Proc. ICMSAO, Hammamet, Tunisia, 2013, pp. 1-5.

[24] H. REN and J. LIU, “Service substitutability analysis based on behavior automata,” Innovations in Systems and Software Engineering, vol. 8, no. 4, pp. 301-308, Dec. 2012.

[25] X. Wang, C. Huang, X. Luo, R. Nie, Y. Tang, and X. Mei, “Determining substitutability of cloud services supported by semantically extended type theory,” Journal on Communications, vol. 37, no.2, pp.20-30, Feb. 2016.