IADIS International Journal on WWW/Internet
Vol. 18, No. 2, pp. 16-28
ISSN: 1645-7641

DISCO: WEB SERVICE DISCOVERY CHATBOT

Bahareh Zarei and Martin Gaedke
Technische Universität Chemnitz. Chemnitz, Germany

ABSTRACT

An increasing number of companies offer their capabilities as Web services and publish them through public registries. Such Web services enable users to access and manipulate online data programmatically. Moreover, they can be deployed in service-based systems (SBS) or composite web applications and mashups to offer value-added services. A crucial step in deploying Web services is the Web service Discovery. Selecting the best Web service candidates to address the needs of service requesters (e.g., SBS designers) is an important task in SBSs and can influence the quality of the resulting composite web application. So far proposed solutions for addressing this challenge, which are mainly categorized into Syntactic-based and Semantic-based, are either too complex for end-users which hinders their large-scale adaptation or suffer from low precision and recall. Considering the recent trend of shifting the development activities more and more towards the end-users, we propose an approach based on chat-bot technology to allow end-users to select a set of best-fitting Web services according to their goals. Conversational Interfaces also known as chatbots aim at providing users a more natural framework to interact with web applications and devices. The advantage of DISCO compared to the state-of-the-art approaches is leveraging the natural language as the communication medium and alleviating the need for end-users to have technical knowledge about the service's structure and location. We conducted sets of experiments by recruiting 8 test subjects to query the chatbot. The results achieve high precision and recall.

KEYWORDS

Conversational Interface, Chat-bot, Web service Discovery, Service-Based Systems, Natural Language Processing, Query Expansion

1. INTRODUCTION

The last decade has witnessed an exponential growth in the number of published Web services on the web. Users are enabled to build their desired applications leveraging the Web services published in public and private registries by service developers and major Web service provider companies such as Amazon and Google (Zhang *et al.*, 2018). According to ProgrammableWeb[1]

[1] https://programmableweb.com

(PW), the number of registered Web services in this public registry as of 2019 was reported more than 22,000. Every year 2,000 new Web services and Web APIs are added to PW. The reasons behind this large-scale adoption of Web services are reducing time and effort required for building composite applications, proposed simplicity, reusability, and higher quality of the resulting system (Bano *et al.*, 2014).

Generally, the Web service life cycle consists of three phases: 1) *service publication*, in which service providers publish the Web services on registries 2) *service discovery and selection*, in which users find and select a set of best-fitting Web services and 3) *service execution*, in which the selected Web service is invoked to perform a task (Jonquet and Cerri, 2006). In this paper, we focus on the service discovery phase. Providing automated and scalable methods for identifying the most relevant Web services based on the user's description plays an important role in the quality of SBSs. Moreover it can influence the effectiveness of Web service compositions and guarantee Web service's full potential employment (Klusch, 2018, Bhardwaj, 2015).

Web service discovery approaches have been evolving during the last years. Researchers focus on various research directions to address this challenge by considering factors such as service description language or service goals. Most of the solutions can be categorized into two categories: *syntactic-based* and *semantic-based* (Zhang *et al.*, 2018). The syntactic-based solutions such as the keyword-based approaches have proven to be inaccurate. Querying the repository based on the keywords in the user's query can result in many irrelevant Web services, i.e., low precision. On the other hand, some semantically related Web services are not discovered resulting in a reduced recall value (Bhardwaj, 2015). In comparison, the semantic-based approaches exhibit better performance compared to syntactic-based. Describing Web services using Semantic Annotations for WSDL (SAWSDL) and ontology-based approaches (Fariss *et al.*, 2018) are examples of semantic-based Web service discovery solutions. However, the complexity of such approaches hinders their large-scale adaptation for end-users without prior knowledge about SBSs or Web service development.

Leveraging natural language will reduce the complexity posed on users. Natural language and *conversational interfaces* as found in personal assistants such as Siri and Amazon Alexa provide a well-known medium for user's interactions and deliver a high level of usability (David Yoffie *et al.*, 2018). Conversational interfaces a.k.a *chatbots* are computer programs able to emulate a human-like conversation with users to provide specific services (Bapat *et al.*, 2018). Chatbots bring several advantages for users such as reducing the required time and effort to perform a task up to 60% and increasing the retrieval accuracy. Moreover, Chatbots can engage users in conversations according to their conversation flow and direct them towards more informative interactions (Abdul-Kader and Woods, 2015, Fuckner, Barthès and Scalabrin, 2013).

Having those advantages in mind, we propose DISCO a chatbot capable of understanding the service requester's queries. With the help of the Natural Language Understanding module (NLU), the key parameters required to discover the intended Web services such as Web service *domain* and *operations*, are extracted from the query. In the case of incomplete or unclear inputs, DISCO initiates conversations with service requesters to guide them towards more informative input. Finally, the chatbot invokes the best candidate Web services and generates reasonable responses.

The main contribution of this research is proposing DISCO as a domain-specific conversational interface for service discovery. This solution is not reliant on the semantic annotations of the services and provides a user-friendly interface based on natural language.

Moreover, a prototype of the solution is implemented, and for demonstrating its effectiveness, sets of experiments are conducted.

The rest of this paper is structured as follows: Chapter 2 gives a detailed overview of DISCO's architecture and its comprising modules such as NLU and Dialog Manager. In chapter 3, a use case scenario is presented to showcase the chatbot's abilities in action. In chapter 4, we provide the study's evaluation. Related work is reviewed in Section 5 and finally, Section 6 concludes the paper and provides future insights.

2. DISCOVERY CHAT-BOT

The idea of using chatbots to provide a communication medium for the users is becoming an industry standard and also a promising research direction in academia (Raghuvanshi *et al.*, 2018). Chatbots are normally following a straightforward Input-Intent-Action-Response conversation model designed for modern conversational agents (Baez *et al.*, 2020). In this work, we used a modified version, by refining the *Intent* and *Action* steps, to design our domain-specific chatbot for Web service discovery (Figure 1).

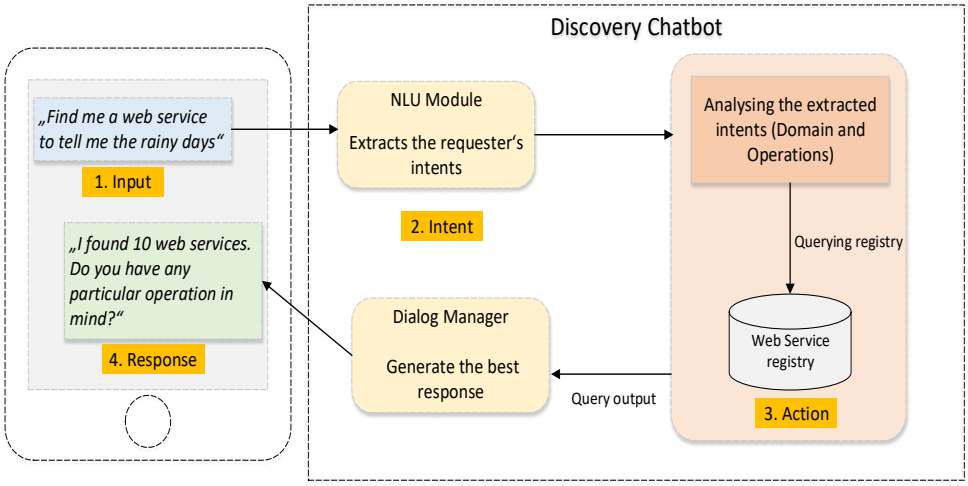

Figure 1. DISCO's Conversation model

The second step in Figure *1* which is denoted as Intent, represents the key parameters extraction such as the Web service target domain, its operations, and entities. The Action stage in this model refers to the matchmaking mechanism used to retrieve suitable Web services.

We considered the following design principles for designing the Discovery Chatbot:

- DISCO is designed as a domain-specific chatbot capable of understanding the service requester's intention, initiating related conversations, and retrieving relevant Web services. To further assist the service requesters without technical knowledge, our chatbot can answer their general questions about the Web services.

- In this work, we focus on textual service descriptions to build our service repository. The textual descriptions are more widely used in the case of RESTful services and web APIs (Zhang *et al.*, 2020).
- The services in our service registry are categorized based on the predefined domains. In the case of overlap among two categories, the service is assigned to each of the identified domains.

2.1 Architecture

Figure 2 illustrates the DISCO's architecture. It consists of *Natural Language Unit*, *Dialog Manager* and the *Service Matchmaking Engine* (SME) as its main modules.

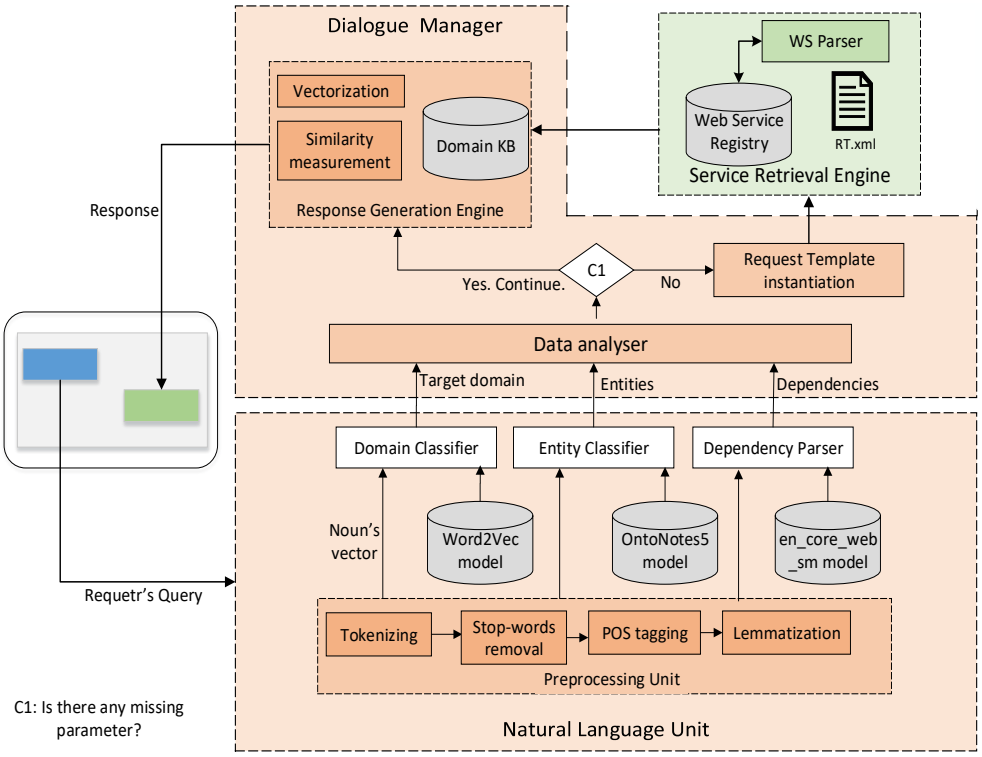

Figure 2. DISCO architecture

The NLU is responsible for understanding the requester's textual input and analyzing it to extract the key parameters (Domain, Operation, and entities). The NLU is a critical component of every chatbot, as the other modules are depending on the NLU's result (Bapat *et al.*, 2018). The extracted parameters are evaluated in the Dialog Manager that decides either to continue the conversation (in the case of incomplete queries) or to instantiate the *Request Template*.

The Request Template is defined as a generic objective description, in XML syntax instantiated during the runtime. This template will be used by SME to query the Web service registry.

2.2 Natural Language Unit

The main building blocks of NLU are the Preprocessing unit and the Domain and Entity Classifiers alongside the required datasets. Each query is preprocessed in the NLU by going through the four steps shown in figure 2. The result is then used by the classifiers for retrieving the key parameters. The classifiers play an important role in chatbots by normalizing the queries and segmenting it into logical parts (Abdul-Kader and Woods, 2015).

2.2.1 Domain Classifier

The Domain classifier assigns the requester's query into one of the predefined domains. This classification enhances the discovery process by reducing the search to the set of Web services with similar functionalities (Raghuvanshi *et al.*, 2018). To identify the target domain, the similarity between the query's core nouns and each defined domain is calculated. Therefore, for each domain, our test subjects select a list of representative nouns from a domain keyword list. This list contains the words with the highest frequency in that particular domain. Afterward, the domain with the highest similarity value is selected as the target domain. The similarity values are calculated by Word2vec algorithm using Cosine similarity between vector representation of the query nouns and domain representative words denoted respectively as Q and DR (Jin *et al.*, 2018):

$$\cos\theta = \frac{Q \cdot DR}{||Q||\,||DR||}$$

The reason behind using the core nouns for domain identification is the higher importance they have in reflecting the target domain. For instance, the two queries "*buy a house*" and "*buy a train ticket*" are referring to Real Estate and Travel domains reflected by the nouns used in both queries.

2.2.2 Entity Classifier

Entities are real-world objects such as people, Nationalities, and Companies that can be used as service parameters. The entity classifier labels each entity with the proper identifier from the knowledge base. For entity recognition, we use the spaCy[2] open-source library written mainly in Python and Cython. To train models, the OntoNotes5 (https://catalog.ldc.upenn.edu/LDC2013T19s) corpus is used. This corpus contains various genres of text, such as news and conversations.

2.2.3 Dependency Parser

Web services provide functionalities known as operations. For an efficient Web service discovery, those operations should be considered (Zhang *et al*, 2018). We formulate a Web service operation as $SO =< AV, AN, P >$ with AV as an *action verb* describing the service operation and AN as an *action noun* that is affected by the operation. P denotes the optional

[2] https://spacy.io/

parameters and entities. To generate such triple, it is necessary to extract the grammatical structure of the query. Generally, the grammatical structure of a sentence can be expressed by part-of-speech (POS) tags and word dependencies. A dependency describes the grammatical relation between two words in a sentence (Zhang *et al.*, 2020). For example, the query "*upload music*" has a dobj (upload, music) dependency that represents the direct object relation between two nouns. To extract the full meaning from queries, we use nsubj (nominal subject), dobj (direct object), and prep (preposition) as the main dependencies according to the Universal Dependencies (https://universaldependencies.org/). Table 1 presents 3 example queries and the generated service operations according to the word dependencies.

Table 1. Dependency extraction

Example query	Service operation	Dependency
"the hotel information in New York can b retrieved"	$SO =< \text{retrieve}, \text{hotel information}, \{-\} >$	nsubj
"A service to calculate the tax rate and insurance for international goods"	$SO =< \text{calculate}, \text{tax rate}, \{-\} >$	dobj
"An API to send the weather forecast by geographical GPS location "	SO=<send, weather forecast,{GPS location, -}	prep_by

2.3 Dialogue Manager

Dialogue Manager is the chatbot brain, responsible for keeping the current state of the conversation, assessing the processed queries, and finally generating the best response according to the conversation logic (Raghuvanshi *et al.*, 2018). As shown in the DISCO architecture (Figure 2), the Dialogue Manager consists of *Data Analyzer* and *Response Generation Engine.*

The Data Analyzer assesses the key parameters extracted by the NLU for completeness and correctness. Upon every input from the NLU, the Data Analyzer triggers the Response Generation Engine to ask the user for confirming the information (i.e., target domain) or providing a more descriptive query. The Request Template is instantiated containing the extracted parameters once the user confirms. By involving the user in the parameter extraction process we guarantee the preciseness of the retrieved services.

The responses are generated by the Response Generation Engine according to conversation logic know as *conversation flow*. The conversation flow determines the possible paths that a conversation between user and chatbot might lead to. It provides designers a comprehensive list of responses and events that are triggered during the conversation (Candello *et al.*, 2017).

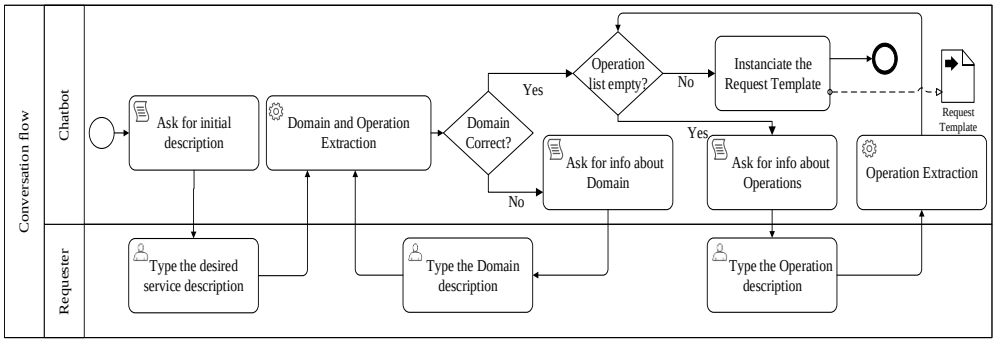

Figure 3. DISCO conversation flow

To enable DISCO to respond to domain-specific questions, we create a knowledge base using a corpus of textual data. This corpus contains texts about Web service technology from Web service related articles such as scientific publications and Wikipedia pages. The chatbot knowledge base highly influences the quality of the responses (Abdul-Kader and Woods, 2015). For response generation, we used a straightforward TF-IDF (Term Frequency-Inverse Document Frequency) method. This method calculates the similarity between the query Q and the sentences in the knowledgebase D:

$$\mathrm{S(Q,D)} = \sum_{w} tf_{wq} \cdot \frac{tf_{wD}}{tf_{wD} + \frac{k|D|}{\mathrm{avg}|D|}} \cdot \log \frac{|C|}{df_w}$$

The summation of term frequencies of word w in the query Q (denotes as tf_{wq}), indicates the fact that the term frequency has a direct relation with the term's importance. The second part of the formula reduces the importance of repetition in the case of short documents. On the other hand, rare words should have a higher weight compared to common words. Therefore, the IDF part is added to the end of the formula, where $|C|$ is the number of documents while df_w is the number of documents containing word w.

2.4 Service Retrieval Engine

The Service Retrieval Engine parses the textual description of Web services in the registry and extracting a set of service operations by using the query expansion techniques. Query expansion is the process of enhancing the given query to capture and extend its meaning. Query expansion can be done by adding new meaningful terms and phrases to the initial query (Selvaretnam and Belkhatir, 2012). In this work, we use the existing grammatical dependencies to obtain new ones and expand the original query. This approach is similar to the one proposed by Zhang *et al.* (2018) for extracting goals from the service descriptions.

According to Figure 4, based on the expanded dependencies a list of service operations is generated. For each query, the desired operations extracted by the dependency parser (cf. section 2.2.3) are matched with each Web service's operations in the target domain. The services with the highest number of operations in common will be retrieved. An overview of the service operation retrieval process is shown in Figure 4.

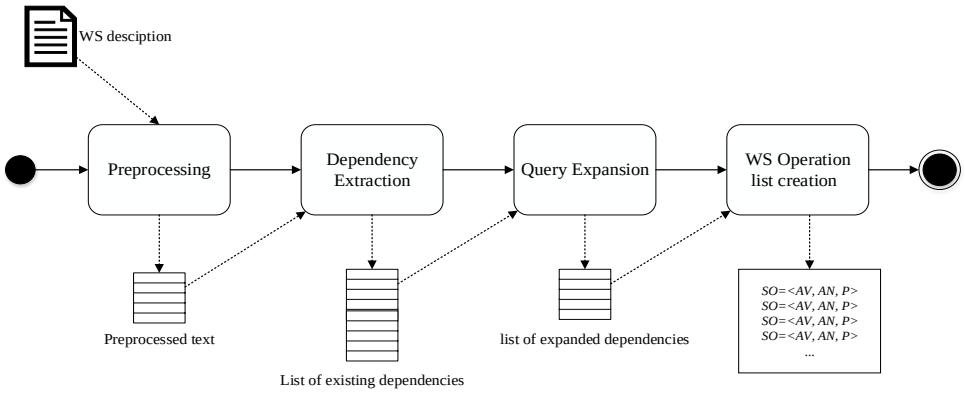

Figure 4. Web service description parsing

3. USE CASE SCENARIO

In this section, we go through an example to demonstrate the DISCO's functionality in detail. Figure 5 shows the beginning of a conversation with the requester. This conversation is based on the conversation flow in Figure 3.

DISCO: Hello, please give me a description of your desired Web service.
Requester: retrieve information about number of Coronavirus cases

DISCO: so, you are looking for a Web Service in Covid-19 domain (Yes/No)?
Requester: Yes.
...

Figure 5. Conversation's beginning fragment

Until this part of the conversation, the target domain is identified as COVID-19 and one operation $SO_1 =< retrieve, Coronavirus\ Cases, \{-\} >$ is extracted. Since extracting more operations improves the result, DISCO asks the user if he/she has any other operation in mind.

DISCO: Great! Let's now talk more about the details. Do you have any other functionalities in mind?
Requester: track the updates

DISCO: I will find you the most related service based on your provided information

Figure 6. conversation fragment

Figure 6 shows the rest of this conversation as the second operation $SO_2 =< track, updates, \{-\} >$ is retrieved. SO_1 and SO_2 are matched against the Web service operations in the target domain based on the semantic similarity. A list of top-three retrieved services for this example is shown in Table 2.

Table 2. Top three retrieved services

Name	Description	Category
Coronavirus (COVID-19) in the UK	This API returns COVID-19 information in the UK including testing capacity, case count, ...	COVID-19
CoronaTracker	CoronaTracker functions as a portal to keep track of latest news development about the COVID-19. ...	COVID-19
Apple Exposure Notification	Contact Tracing interface that provides interoperability between Android and iOS devices …	COVID-19

4. EVALUATION

We collected the descriptive data including the name, category, and textual description of 1100 web APIs from the public PW repository. The APIs belong to six domains: *Transportation* (120), *Mapping* (230), *Weather* (60), *Mobile* (240), *Social* (350), and *COVID-19* (100). The domains are selected according to their popularity and scale in the PW. For each API, a list of service operations is generated by the Service Retrieval Engine.

We recruited eight test subjects including undergraduate and postgraduate students as well as developers. The subjects were asked to perform several tasks such as building a representative word set for each domain. To eliminate the impact of the subject's understanding of the domain on the result, we asked them to read selected short articles about each domain to familiarize themselves with the terms and vocabulary of that particular domain. Afterwards, the subjects had to construct a set of experimental queries from the six domains with no limitation on the length. Collectively 60 queries were constructed by our subject requesters. Finally, the subjects were asked to evaluate the identified domain, operations and the list of suggested APIs.

4.1 Evaluation Result

For evaluating the NLU performance, subjects grade the identified target domain and extracted operations. The operations were graded separately on a scale of 0-3 where "0" indicates irrelevant operations that are not meaningful or important. Such operations are mainly extracted from general phrases. The grade "1" represents poorly relevant operations that are meaningful but are not specific enough to match the user's intention. The operations with the grade "2" are relevant, while the grade "3" indicates the highly relevant operations with specific parameters. Examples of evaluated operations by the subjects are presented in Table 2. According to the evaluation result, for 60 queries, 97% of the target domains matched with the intended domains.

Moreover, the extracted operations were evaluated by subjects as follows: 20% of total operations were evaluated as irrelevant and poorly relevant (10% each), 20% relevant and 60% highly relevant.

Table 3. Subject's evaluation example

Example query	Operations	Evaluation
"Find an API for Attaching the image to a Facebook post and write post in my Facebook"	<find, API, {}>	1
	<attach, image, {}>	2
	<write, post, {Facebook}>	3
"provides the photographic functions to map a picture to geographical point on the map"	<provide, photographic functions, {}>	2
	<map, picture, {}>	1

The two popular metrics to measure our approach performance are precision and recall, defined as follows:

$$\text{Precision} = \frac{|WS_{rt} \cap WS_{rl}|}{WS_{rt}} \quad \text{Recall} = \frac{|WS_{rt} \cap WS_{rl}|}{WS_{rl}}$$

Where WS_{rt} and WS_{rl} denote retrieved and related services respectively. The precision and recall are computed based on the subject's evaluation result. The precision of our approach in retrieving the intended Web services is 0.78 and the recall value is computed as 0.80.

4.2 Threats to Validity

Two types of threats can target the validity of our evaluation result. The *internal threats* target the evaluation repeatability. In our experiments, the subjects were responsible for building the domain representative set and experiment queries as well as evaluating the result. If the subject is not familiar with the domain and the procedure, the results might be affected negatively. As already stated, to resolve this issue the subjects were educated about the procedure and domains in advance.

The *external threats* affect generalizing the results to other situations and environments outside the experiment scope. The PW repository is dependent on the inputs from API providers, therefore the API popularity and scale change throughout the time (such as the newly added COVID-19 domain to this repository). To overcome this issue, we have to consider the other repositories for our evaluation as well.

5. LITERATURE REVIEW

A great deal of recent studies has been focused on conversational interfaces and chatbots. Chatbot technology is covering a wide range of applications and has a great potential for evolving in brand new domains. By leveraging natural language, chatbots provide a medium for end-users to perform tasks without prior knowledge and as a result improving usability (Abdul-Kader and Woods, 2015). Considering these advantages, we proposed a domain-specific Discovery Chatbot for Web service discovery applications. To justify the novelty of our approach we review some of the existing practices in this domain.

Researchers have proposed a variety of Web service discovery solutions considering factors such as functional and non-functional requirements or Quality of Service (QoS). Our focus is on the matchmaking techniques due to their relevance to our approach. From this point of view, solutions are categorized as syntactic-based, semantic-based, and context-aware (Jalali *et al.*, 2014).

Syntactic-based solutions mainly rely on information retrieval (IR) techniques and vector space model to retrieve the relevant services. These solutions such as keyword matching, are simple, more familiar for users, and are used as a standard in already established UDDI. However, they have low precision and cannot be used in automatic processing (Bhardwaj and Sharma, 2016). On the other hand, the Semantic-based solutions, attempt to overcome the aforementioned drawbacks by retrieving the semantically similar Web services. Depending on the reasoning method the semantic-based solutions are categorized into *logic-based* and *non-logic-based* approaches. The common methods in non-logic-base solutions are text similarity measurements, schema matching, and graph matching. On the other hand, in logic-based solutions, logical reasoning is performed on the service description using ontologies and Semantic Annotations for WSDL and XML Schema (SAWSDL) (Klusch, 2014). Ontology-based approaches allow automatic discovery and have a higher precision since they are providing an accurate description of the services. However, the ontology specification and maintenance require high effort that prevent large-scale adoption. Moreover, the different ontologies used by users and providers can impose new challenges for logic-based solutions (Zhang *et al.*, 2018).

The idea of using conversational interfaces and natural language for service discovery was practiced by Fuckner et.al. (2013). In this solution, the user interacts with a dialogue-based multi-agent platform for invoking services. Authors used mainly a keyword-based approach to match the concepts from the user's query. Our work is different in the sense that we used the service's textual description as the source of information and instead of keyword-based we used query expansion technique and semantic similarity to invoke the related Web services.

The two solutions proposed by Zhang et.al. (2018) and (2020), use query expansion to mine the service goals and match them with user's queries. The authors extract the linguistic structure of the service description using the Stanford parser and match them with each query's goal. In contrast, DISCO retrieves services according to the conversation with the requester.

6. CONCLUSION AND FUTURE WORK

This research proposed a method based on chatbot technology to address the usability issues of Web service discovery solutions. The low precision and recall and high technical skill required to adopt the solutions are among the main issues regarding the existing approaches. Our approach retrieves services based on the extracted parameters from the requester's query such as domain, operations and entities. Upon analyzing each query, the response generation engine decides the conversation direction according to our conversation flow. By integrating the chatbot in the Web service discovery process, we improved the retrieval rate and provided users a natural-language-based interface that does not require them to have prior knowledge about Web service's structure. Also, our approach can benefit from several improvements in the NLU by extending the query expansion to cover wider range of dependencies. The idea of integrating conversational interfaces into the Web service life cycle introduces new research directions in the field of mashup and SBSs such as requirement extraction chatbot for SaaS applications.

ACKNOWLEDGEMENT

This work has been supported by the ESF and the Free State of Saxony with grant number 1000235478.

REFERENCES

Abdul-Kader, S. A. and Woods, J. (2015) 'Survey on Chatbot Design Techniques in Speech Conversation Systems', *(IJACSA) International Journal of Advanced Computer Science and Applications*, 6(7), pp. 72–80. Available at: www.ijacsa.thesai.org (Accessed: 30 December 2019).

Baez, M., Daniel, F. and Casati, F. (2020) 'Conversational Web Interaction: Proposal of a Dialog-Based Natural Language Interaction Paradigm for the Web', in *The 3rd International Workshop on Chatbot Research (Conversations 2019)*. Springer, Cham, pp. 94–110. doi: 10.1007/978-3-030-39540-7_7.

Bano, M. *et al.* (2014) 'What makes service oriented requirements engineering challenging? A qualitative study', *IET Software*, 8(4), pp. 154–160. doi: 10.1049/iet-sen.2013.0131.

Bapat, R., Kucherbaev, P. and Bozzon, A. (2018) 'Effective crowdsourced generation of training data for chatbots natural language understanding', *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 10845 LNCS, pp. 114–128. doi: 10.1007/978-3-319-91662-0_8.

Bhardwaj, K. C. (2015) 'Machine Learning in Efficient and Effective Web Service Discovery', *Journal of Web Engineering*, 14(3), pp. 196–214.

Bhardwaj, K. C. and Sharma, R. K. (2016) 'Ontologies: A Review of Web Service Discovery Techniques', *International Journal of Energy, Information and Communications*, 7, pp. 1–12. doi: 10.14257/ijeic.2016.7.6.01.

Candello, H. *et al.* (2017) 'Evaluating the conversation flow and content quality of a multi-bot conversational system', *Extended Proceedings of the 16th Brazilian Symposium on Human Factors in Computing Systems.*, 9(October), pp. 60–61.

David Yoffie, P. B. *et al.* (2018) *Voice War: Hey Google vs. Alexa vs. Siri*. Available at: www.hbsp.harvard.edu. (Accessed: 6 January 2020).

Fariss, M., Allali, N. El and Asaidi, H. (2018) 'Review of Ontology Based Approaches for Web Service Discovery : Methods and Protocols Review of Ontology Based Approaches for Web Service Discovery', in Springer, C. (ed.) *International Conference on Advanced Information Technology, Services and Systems*. Springer International Publishing, pp. 78–87. doi: 10.1007/978-3-030-11914-0.

Fuckner, M., Barthès, J.-P. and Scalabrin, E. E. (2013) 'Web Service Discovery and Execution Using a Dialog-Based Approach', in *WEBIST 2013: Web Information Systems and Technologies*. Aachen, pp. 103–118. doi: 10.1007/978-3-662-44300-2.

Jalali, M., Pakari, S. and Kheirkhah, E. (2014) 'Web Service Discovery Methods and Techniques: A Review Re-Routing Based on QoS Requirements Changing in WSN View project Information Dissemination in Social Networks View project Web Service Discovery Methods and Techniques: A Review', *International Journal of Computer Science, Engineering and Information Technology (IJCSEIT)*, 4(1). doi: 10.5121/ijcseit.2014.4101.

Jin, X., Zhang, S. and Liu, J. (2018) 'Word Semantic Similarity Calculation Based on Word2vec', *ICCAIS 2018 - 7th International Conference on Control, Automation and Information Sciences*, pp. 12–16. doi: 10.1109/ICCAIS.2018.8570612.

Jonquet, C. and Cerri, S. A. (2006) 'Characterization of the Dynamic Service Generation concept', (06007). Available at: http://www.lirmm.fr/~jonquet/publications/documents/RR-LIRMM-06007-Jonquet-feb2006.pdf.

Klusch, M. (2014) 'Service Discovery', *Encyclopedia of Social Network Analysis and Mining*, pp. 1707–1717. doi: 10.1007/978-1-4614-6170-8_121.

Klusch, M. (2018) 'Service Discovery', in Alhajj, R. and Rokne, J. (eds) *Encyclopedia of Social Network Analysis and Mining*. New York, NY: Springer New York, pp. 2474–2484. doi: 10.1007/978-1-4939-7131-2_121.

Raghuvanshi, A., Carroll, L. and Raghunathan, K. (2018) 'Developing Production-Level Conversational Interfaces with Shallow Semantic Parsing', in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Brussels, pp. 157–162.

Selvaretnam, B. and Belkhatir, M. (2012) 'Natural language technology and query expansion: Issues, state-of-the-art and perspectives', *Journal of Intelligent Information Systems*, 38(3), pp. 709–740. doi: 10.1007/s10844-011-0174-3.

Zhang, N., Wang, J., Ma, Y., He, K., Li, Z. and Frank, X. (2018) 'Web service discovery based on goal-oriented query expansion', *The Journal of Systems & Software*, 142, pp. 73–91. doi: 10.1016/j.jss.2018.04.046.

Zhang, N., Wang, J. and Ma, Y. (2020) 'Mining Domain Knowledge on Service Goals from Textual Service Descriptions', *IEEE Transactions on Services Computing*, 13(3), pp. 488–502. doi: 10.1109/TSC.2017.2693147.