Neural Computing and Applications (2022) 34:2007–2029
https://doi.org/10.1007/s00521-021-06486-6

ORIGINAL ARTICLE

Self-adaptive mobile web service discovery approach based on modified negative selection algorithm

Salisu Garba[1] · Radziah Mohamad[1] · Nor Azizah Saadon[1]

Received: 29 November 2020 / Accepted: 30 August 2021 / Published online: 20 September 2021

Abstract
This paper proposes a self-adaptive mobile web service (MWS) discovery approach based on the modified negative selection algorithm (M-NSA) to improve the effectiveness and accuracy of MWS discovery in dynamic mobile environment. The main contributions of this work are the service relevance learning model and a MWS matchmaking algorithm that it is capable of changing as soon as the discovery demonstrates the feasibility of attaining improved effectiveness or accuracy. This is achieved by transforming the two stages of modified negative selection algorithm (M-NSA) into service relevance and self-adaptive matchmaking, respectively. The proposed approach is evaluated in terms of both binary and graded relevance. After an experiment with the largest MWS dataset, the proposed approach records better results in comparison with the state-of-the-art approaches. This is owing to the self/nonself discrimination mechanism, in addition to the decent parameter analysis, and the use of more comprehensive information that covers the entire discovery space.

Keywords Self-adaptive · Mobile web service · Service discovery · Negative selection algorithm · Dynamic mobile environment

1 Introduction

Mobile web service (MWS) discovery is the most important stage in a MWS life cycle as it identifies the most relevant MWS for a particular task in accordance with the quality and context needs of the service request [1]. Web service is a reusable, language, and platform independence software paradigm proposed to facilitate the development of distributed systems from existing applications with simplicity and low cost [2, 3]. The current web service technologies have been extended to mobile web services (MWS) due to the recent incarnation of the Internet of Things (IoT) technologies. This is because of the advancement of mobile devices, the expectation that mobile internet adoption will revolutionize the broader digital ecosystem, and the possibility of mobile devices becoming the primary tool for accessing the internet is now a reality as the number of users accessing mobile services goes beyond 5 billion in 2017 [4]. The discovery of MWS normally takes place in dynamic mobile environment (DME), an environment that consists of heterogeneous mobile devices operated by numerous classes of users in highly mutable settings [5–8]. This environment makes it almost impossible to have an exhaustive model applicable in all MWS discovery scenarios. The MWS discovery has not yet realized its full potential since it has fallen short in resolving challenges that include a high influx of similar web services, unpredictable changes in DME. The existing research on MWS discovery, such as [9–11], mainly focused on applying the conventional matchmaking techniques constrained by the specific web service description standard besides inadequate adaptivity [12].

While the existing discovery techniques are constrained in DME and bound to produce less accurate matchmaking results, the renewed interest in lightweight and autonomous solutions (adaptive) is gaining popularity in solving MWS discovery problems in DME. A self-adaptive approach can

✉ Salisu Garba
Salisu.garba@graduate.utm.my

Radziah Mohamad
radziahm@utm.my

Nor Azizah Saadon
azizahsaadon@utm.my

[1] School of Computing, Faculty of Engineering, Universiti Teknologi Malaysia, 81310 Skudai, Johor, Malaysia

assess and change its behavior, at whatever point the assessment demonstrates that the outcome is not as expected or when improved functionality or performance might be conceivable [13]. In pursuance of the ultimate goal of self-adaptive MWS discovery in DME, several machine learning-based approaches show potential; unfortunately, these approaches are based on the binary classification that demands an undesirable compromise between partial scores and requires manual adjustments, recreation, and/or retraining of the models during design time which may lead to ineffectiveness and inaccuracy in discovery result [14]. In view of these shortcomings, an improved approach is required to improve the effectiveness and accuracy of MWS discovery and meanwhile ensure the stable operation of the web applications. Given that the solution to such problems exists in probabilistic or deterministic methods, probabilistic methods can deliver the best or a near-best result in nominal computational time and model the underlying probability distributions of the classes [15]. The Mammalian immune system (MIS), a distributed, multi-tiered, probabilistic, self-adaptive learning system that consists of an intricate network of tissues, organs, and chemicals, inspires the artificial immune systems (AIS) to solve various real-world computational problems [16]. The first model among the many models in the artificial immune system (AIS) is the negative selection algorithm (NSA) introduced by [17] to evaluate a given state as normal or abnormal by establishing a profile of the system and mimicking the self/nonself discrimination mechanism of MIS.

This paper presents a MWS discovery approach based on modified NSA (M-NSA). To find the most relevant MWS, the self-adaptive matchmaker (SAMM) learns the service relevance by using the detector generation stage of M-NSA. After the service relevance learning (detector generation stage) is accomplished, the self-adaptive MWS discovery (detection stage) begins in which each detector's distance in the detector set and MWS instance is measured based on a given request for self/nonself discrimination. A condition that enables the self-adaptation to occur is added in the M-NSA to update the detectors to ensure the continuous discovery of relevant MWS. In this paper, the extracted and clustered MWS are used to generate detectors. After that, the remaining MWS is plugged into the detection stage to assess the proposed approach's capabilities. The experiment first analyzes the detectors' detection rate and false alarm rate based on several parameters. The performance of the SAMM is measured using five performance criteria employed in the MWS discovery domain. The results show that the M-NAS can be easily utilized to discriminate irrelevant MWS for a given request. The modification of the traditional version of NSA in terms of parameters and the self-adaptation process helps eliminate the design-time adaption strategy that requires recreation and retraining. Moreover, the proposed approach exhibits significant improvements in comparison with state-of-the-art MWS discovery approaches. The key benefit of the proposed self-adaptive MWS discovery approach is that it can assess its parameters and change as soon as the assessment demonstrates the feasibility of attaining improved accuracy or performance. The major contributions of the proposed approach can be highlighted as follows:

- We propose a self-adaptive mobile web service discovery approach based on the modified negative selection algorithm for more effective and accurate mobile web service discovery in a dynamic mobile environment.
- We design a service relevance learning model based on the negative selection algorithm's detector generation stage for better learning of the fine-grained request-offer interaction.
- We design a self-adaptive matchmaking algorithm fitted with a reinforced learning process to adapt itself at runtime in response to self-spaced changes or when the discovery result is not as expected.
- We carry out a series of experiments using publicly available datasets to evaluate the performances of the proposed approach against state-of-the-art approaches.

The rest of this paper is organized as follows: The background and related works are discussed in Sect. 2. Section 3 presents the definition and problem description. Section 4 introduces the proposed approach. In Sect. 5, the empirical evaluation and results are presented and deliberated. Finally, the conclusions and future directions are discussed in Sect. 6.

2 Background and related work

This section reviews various research works. These include contemporary web service matchmaking techniques, self-adaptive web service discovery, and negative selection algorithms.

2.1 Web service matchmaking techniques

Web service discovery is the act of matchmaking between the advertised service and the service request to find not only the best possible web service but also the most relevant web service for a particular task. The matchmaking techniques are usually classified into four categories based on the underlying algorithm used. This includes logic-based, syntactic, and hybrid [18, 19]. Additionally, [14] recognizes a new category of matchmaking approach

called adaptive matchmaking. Description and/or rule-based logic (formal reasoning) such as case-based reasoning (CBR), in [20] and genetic algorithm (GA) in [21], formed the basis for determining the semantic relatedness between the offered MWS services and the requested MWS service in logic-based service discovery approaches. The syntactic approaches usually employ keyword-based matching. This takes advantage of data-mining techniques and information retrieval methods such as semantic similarity [22, 23] to determine the level of relatedness between the offered MWS and the requested MWS instead of formal reasoning [22, 23]. The hybrid matchmaking approaches include integrating formal reasoning and syntactic techniques to carry out the matching [24], while the adaptive matchmaking approaches adopt machine learning techniques to combine the similarity values from various approaches to ensure efficient use of resources in DME [25].

Many significant contributions such as [2, 10, 24, 26, 27] are recorded that mainly focus on overcoming the complexity of logic-based service discovery in a resource-constrained device and resolve the inaccuracy of keyword-based service discovery. Recently, the authors in [10] proposed a keyword-based MWS discovery approach that discovers services based on the TF-IDF. The cosine similarities between services and the request are calculated, after which the service discovery result is generated based on the similarities. However, the similarity measure used in keyword-based service discovery can only give an estimation of similarity between request and service after extensive refinement, introduce ambiguation since it is based on TF-IDF, and lack accuracy due to the high number of false positive (FP/F + ve) and the false negative (FN/F-ve) in matchmaking result [24]. In a new development, [10] proposed a goal-based MWS discovery (GSD) approach that discovers MWS by matching service goals with the goals contained in an expanded/refine query using keyword-based and topic model-based, NLP-based method is used to extract MWS goals from services' descriptions, and clusters are created based on the similarities between the extracted MWS goals. However, limiting the matchmaking to MWS goals only without considering other variables such as context may hinder the quality and accuracy of the matchmaking results by returning semi-relevant or irrelevant MWS although the use of alternative semantics expands the MWS request.

The authors in [28] proposed a web service operations discovery (OpD) approach that mines the service interface to create index libraries and use co-occurrence probability for the discovery of services with solitary or multiple operations, while [24] introduced hybrid matchmaking approach that integrates formal reasoning together with syntactical techniques to carry out the matching in an open-ended, dynamic registry. Higher priority is given to the logic-based matching, while the syntactic matching compensates for the absence of exact matching in the logic-based. However, a UDDI-based service discovery scheme is complicated and sometimes incompatible with an open and broadly decentralized environment. While the use of the popular discovery techniques may provide some benefits in terms of standard matchmaking algorithms and models, the technique is bound to produce less accurate matchmaking results owing to the ample influx of a variety of MWS, rapid and unpredictable changes that are taking place in dynamic mobile environment (DME). In addition, the renewed interest in lightweight and autonomous solutions renders the usual matchmaking difficult to apply in DME. As such the self-adaptive web service discovery technique is gaining popularity in solving MWS discovery problems in DME.

2.2 Self-adaptive web service discovery

Currently, self-adaption in web service discovery [29], organization [30], and composition [31] is gaining popularity as it can be used to take advantage of fundamental adaptivity principles so as to achieve better accuracy and to ensure efficient use of resources in DME. A self-adaptive MWS discovery framework can assess and change its own behavior, at whatever point the assessment demonstrates that the outcome is not as expected, or when improved functionality or performance might be conceivable [13]. Adaptation can either be reactive or proactive. Reactive adaptation checks the current execution to identify considerable changes and adapt, while proactive adaptation anticipates changes or deviation and adapts [25, 32]. Given that variability and lack of broad knowledge introduce uncertainty in self-adaptive web service discovery, uncertainty reduction tactics are introduced to support better adaptation decisions [33]. In [34], constraint satisfaction principles are used to create a model to solve the dynamic adaptation problems caused by changes in the execution context.

The self-adaptive mechanism is used to define alternative solutions to withdraw unavailable service when a service is predicted to be failed according to its quality values, and/or the service does not fit the required context. According to [35], the realization of a self-adaptive mechanism requires control loops that gather details from a different context and act accordingly. Furthermore, an execution plan equipped with the ability to shift from one configuration to another at runtime, strategy to effectively deal with rapid and unpredictable changes that are taking place in a DME is some of the benefits of self-adaptive web service discovery. For the self-adaptive MWS discovery approach to accomplish its intended function, the following

two operations usually occur: (i) the observation of the discovery, (ii) the application of the required strategies to adjust the discovery mechanism's behavior in light of recognized changes. The authors in [29] proposed a self-adaptive QoS-aware web service discovery (SQoSD) approach in which the discovery process computes the similarities between request and offer based on using concept, attribute, and constraint similarity. The self-adaptive discovery mechanism iteratively restructures requirement ontology trees to match web service description ontology trees until the matching result is greater than a certain threshold or the user requirement is satisfied. However, this can lead to an infinite loop, which may hinder the anticipated objectives of matchmaking in DME.

Contemporary self-adaptive approaches include neural collaborative filtering [36] and deep hybrid collaborative filtering [9] in which a deep neural network is used to obtain the nonlinear affinity among services and the pointwise loss function is used to transform the discovery operation into a classification problem. Similarly, an adaptation of probabilistic latent semantic analysis (pLSA) in which the service attributes are modeled as a heterogeneous information network (HIN) and LDA is used to obtain the essential semantics of MWS and MWS requests are proposed [10, 35]. However, the LDA-based approach is less effective as it can only capture fewer semantically consistent topics without the aid of word embedding or other query expansion strategies. The adaptation is usually achieved by enabling the discovery mechanism itself to detect the situations that warrant adjustments and execute the changes rather than manual adjustments during design time or dynamically modifying the service at runtime. The authors in [37] use the relational topic model (RTM) and factorization machines (FMs) to model the relationship between topics of MWS and use factorization machines to predict MWS for a target request. This approach is also capable of evaluating parameters under low sparsity like support vector machines (SVMs) used in [38]. In these approaches, machine learning models are used to achieve adaptation rather than abductive reasoning.

One of the glaring observations on self-adaptive approaches that employ service withdrawal, service replacement, and process restructuring is the need for additional time to perform the required action and the amount of the changes that can trigger adaptation is not evident in these approaches. The consequences of failure to determine the amount of the changes that can trigger adaptation are the accumulation of overhead cost (processor power, memory, battery, and bandwidth) due to unnecessary adaptation, especially in a resource-constrained environment (DME). The existing approaches are mostly constricted by a design-time adaption strategy that requires recreation and retraining of the aggregation model after feedback collection over a period of time (when the model became obsolete). There is a clear lack of a reinforced learning approach that can make use of feedback information to provide more relevant MWS at runtime. Moreover, the high sparsity of the interaction matrix in MTR-FM hinders the accurate classification of MWS, while retraining a model from scratch is a very costly process [38].

In pursuance of the ultimate goal of self-adaptive MWS discovery in DME, the machine learning-based approaches are the most promising; unfortunately, these approaches are based on the binary classification that demands an undesirable compromise between partial scores which may lead to ineffectiveness and inaccuracy in discovery results. Furthermore, the extensive processing and continuous loop necessary to attain better results may lead to inefficient discovery in DME. Besides, the adaptation and the execution environment are not tightly interleaved. Thus, there is a need for a discovery approach that defines alternative solutions at whatever point the assessment demonstrates that a better result might be conceivable.

2.3 Negative selection algorithms

The artificial immune system (AIS) derives its inspiration from nature, specifically, the mammalian immune system (MIS). Currently, the negative selection algorithm (NSA), positive selection, clone selection, immune network theory, and danger theory are the most popular algorithms and theories of AIS researches [39]. The first model among the many models in the artificial immune system (AIS) is a negative selection algorithm (NSA) introduced by [17]. Subsequently, several studies have proposed various NSA with notable differences in data representation, detector representation, self-definition, and matching rule. Although there is a rapidly growing literature on NSA, which indicates that NSA is among the most widely used algorithm in many fields, the primary application of NSA has been anomaly detection (the discrimination of normal/self and anomalous/nonself state) [40]. In comparison with the positive selection algorithm, the negative selection needs fewer time and space resources [39]. The negative selection technique uses self samples to model nonself space, whereas the positive selection technique uses self samples to model self space. A given data can be classified as normal/self data or anomalous/nonself data and raise an alarm in the case of anomalous/nonself data. The NSA aims to evaluate a given state as normal or abnormal by establishing a profile of the normal state of a system. The classification accuracy is usually improved by minimizing classification error, i.e., the false positive (FP/F + ve) and the false positive (FP/F + ve). Figure 1 illustrates the basic idea of generation, detection, and classification in NSA.

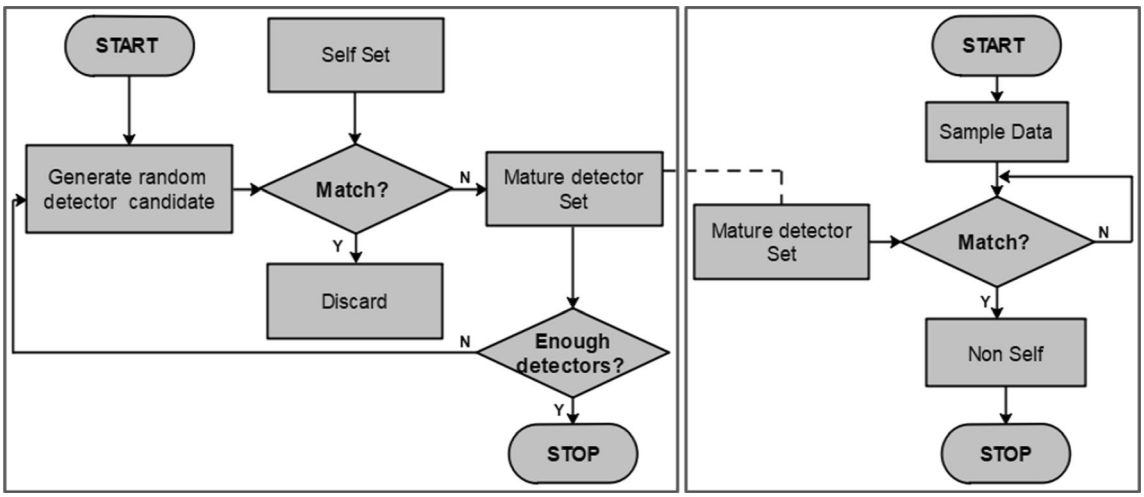

Fig. 1 The basic idea of generation, detection, and classification in negative selection algorithm

The conventional NSAs are made up of two stages [40, 41]. In the first stage (generation stage/training stage), the basic detector generation procedure comprises the random generation of detector candidates in the whole nonself space followed by a comparison between the generated detector and self. If the randomly generated detector matches self, then the detector is regenerated and a rematch with self, else, the generated detector is accepted and added to the mature/valid detector set. In addition, the regeneration and rematch process can be absconded by modifying any detector that matches the self. A criterion is used to determine the sufficiency of the mature/valid detector set generated and subsequently terminate the generation/ regeneration stage.

In the second stage (detection stage), the mature/valid detector set generated in the training/generation stage is used to determine whether the input data conform to self/ normal or nonself/anomalous. If the mature/valid detector matched the input data, then the NSA declares that an anomaly occurs in most applications [41–43]. The matching rule detector generation process involves the use of matching rules such as Euclidean or cosine distance (which implies that if a given data are inside the borders of a detector, then it fulfills the matching rule) to compare a candidate detector against any self-data. If the candidate detector matched any self-data, then the candidate detector is regenerated and a rematch with self-data, else, the candidate detector is accepted and added to a valid detector set. In addition, the matching rule is always used as a basis for classifying the test data (normal/self data or anomalous/ nonself data) in the detection process [44].

The two models of NSA are binary (BNSA) and real-valued NSAs (RVNSA). In contrast to the BNSA, there is a rapidly growing literature on RVNSA, which indicates that RVNSA is among the most promising in terms of performance and is the widely used meta-heuristic algorithm in several problem domains such as data partitioning, categorization, clustering, and classification [45, 46] due to its powerful information processing capability as it works on a unitary hypercube [0, 1]n with the end goal that every self and nonself samples have a center and a fixed radius. [47] introduce bidirectional inhibition optimization r-variable (BIORV-NSA) to tackle the detection holes and minimize the number of detectors by adapting the self-radius as well as the detectors from constant to variable. This significantly improves the detection rates with a limited number of detectors as less effective detectors are replaced with more effective detectors that cover the black holes. [48] proposed Voronoi diagrams-based NSA (Vor-NSA) in which the "Random-Discard" model of initial detector generation is replaced with the "Computing-Designated" model to reduce the time cost of the traditional NSA. Afterward, the Map/Reduce framework is integrated into the detection stage to further minimize the time cost, especially when dealing with big data.

In [49], particle swarm optimization (PSO) is used to enhance the detector generation stage of NSA and the fitness function for the detector generation is determined by a local outlier factor (LOF). The optimized detectors result in improved the accuracy of detection of spam emails. In an effort to optimize and extend the application of traditional NSA, the online adaptive learning ability is introduced to NSA by [50, 51], as such a boundary-fixed NSA (FB-NSA) is used to generate an initial set of detectors. The initial set of detectors is adapted continuously using online adaptive learning under small samples (OALFB-NSA) in the detection stage. This approach shows that the detection rate of a constant set of detectors tends to slow down gradually with an increase in false alarm rate. However, the adaptive detectors show an improved detection rate and low false

alarm rate in many instances. [52, 53] introduced the NSA as a possible solution in web service problem domains. In this case, the terms used in the operation of the natural immune system such as antigen, antibody, locus are firstly redefined to web service optimization problem, solution vector, abstract web service, etc. NSA + is proposed using the negative selection principle to gradually minimize the QoS problems in service discovery by adopting a string value to describe the offered services and assign a solution vector to each service based on its logical structure. The NSA + evaluates the QoS by its local fitness based on four quality dimensions (cost, availability, time, and reliability).

The development of NSA so far paves way for its application in many areas [52, 53]. Its theoretical and practical potential for differentiating between what belongs and what does not belongs make it ideal in solving many MWS discovery in DME. However, most of the previous studies have relied on constant-size detectors which lead to detection holes (black holes), while recent studies relied on a large number of detectors with continuous overlapping between detectors instead of identifying the best coverage of nonself using the coverage estimation which results in large time cost and complexity. Moreover, the lack of continuous adaptability is evident in most studies which gives rise to a low detection rate and high false-positive rate [50, 51]. Hence, an improved NSA that is appropriate for self-adaptive MWS discovery DME is essential.

3 Definitions and problem description

In this section, the concepts and definitions are related to the self-adaptive MWS discovery approach based on a modified negative selection algorithm as well as the research problems this paper aims to address.

3.1 Concepts and definitions

Self-adaptive MWS matchmaking algorithm is designed based on MNSA, which is derived from the artificial immune system (AIS) to search and discriminate between self and nonself. In AIS, data samples are defined as "antigen," normal samples are defined as "self," abnormal samples are defined as "nonself," and antibody is defined as "detector." The basic definitions are listed as follows.

Definition 1 Data sample is a collection of MWS that are yet to be classified as self or nonself. All the MWS in the feature space constitutes the set of Service Collection. SC represents the set of MWS data samples. s represents the individual members of SC. n represents the dimension of s, and the value of the s attribute i is normalized in the feature space. $\mathrm{SC} = \{s | s = (s_1, s_2, s_3, \ldots\ldots\ldots s_n),\ s_i \in [0,1],\ 1 \le i \le n\}$.

$$S' \cap \mathrm{NS}' = \emptyset, \quad S' \cup \mathrm{NS}' = \mathrm{SC} \tag{1}$$

Each s in SC has attributes i which consist of Goals $g_1, g_2, g_3 \ldots g_n$ and Context $cx_1, cx_2, cx_3 \ldots cx_n$.

Definition 2 Detectors are generated in response to data samples (MWS perceived as extraneous). Let (D) be the detector set. $D = \{d_1, d_1, d_1 \ldots\ldots\ldots d_n\}$. $d = c_d, r_d$, where d represents the instances of D, andc_d and r_d represent the center and radius of each instance, respectively.

Definition 3 Self are MWS that belong to the relevant collection which does not induce the generation of detectors. Let (S) be the set of self-samples. $S' = \{s_1, s_2, s_3 \ldots\ldots\ldots s_n\}$. $s = c_s r_s$, where s represents the instances of S, and c_s and r_s represent the center and radius of each instance, respectively.

Definition 4 Nonself are MWS that do not belong to the relevant collection which induces the generation of detectors. Let (S') be the set of self-samples. $S' = \{s'_1, s'_2, s'_3 \ldots\ldots\ldots s'_n\}$. $s = c_s, r_{s'}$, where s' represents the instances of S', $c_{s'}$ and $r_{s'}$ represent the center and radius of each instance, respectively.

Definition 5 Matching rules are what govern the detector generation phase and a matchmaking detection phase. Irrespective of representation, a matching rule M can be formally defined as $dMs \leftrightarrow$ distance measure between d and s is within a threshold τ. For example: If $\mathrm{Dist}(d,s) = \left(\sum_{i=1}^{n}(d_i - s_i)^2\right) < r_s$, then the candidate detector d will be discarded; otherwise, it will be position in the D' based on its r_d.

3.2 Problem description

The research problems this paper aims to address are defined as follows:

First, how to model the information that covers the entire discovery space (the requested MWS and the offered MWS) to learn the MWS relevance based on a given MWS request. The offered MWS and requested MWS information consist of MWS goals and context which are described below:

- *MWS goals extraction:* The objective of this task is to extract the goals that describe the functional capabilities of MWS from the RESTful description of the MWS. Goals are a set of words that describe the features of the

offered MWS and requested MWS. We assume there are goals in the offered MWS given as $g_{1,}g_{2,}g_3 \ldots g_n$.

- *MWS context identification:* The objective of this task is to extract the features that describe the nonfunctional capabilities of the MWS from the RESTful description. The context is a set of words and numerical features that distinguish between similar MWS and enable a better understanding of MWS requests. We assume there are context in the offered MWS or request given as $cx_{1,}cx_{2,}cx_3 \ldots cx_n$.
- *Relevant MWS (top-N MWS) identification:* The objective of this task is to measure the relevance by matching the MWS request and the offered MWS. This creates a two-dimensional array based on the weight vectors of length-K.

Second, how to identify, retrieve, and generate the top-N MWS of the most relevant MWS that satisfies the functional and context needs of MWS requests based on the SRL model.

4 Self-adaptive MWS matchmaker: overview

Before deliberating on the self-adaptive MWS matchmaker's technicalities, it is highly desirable to provide a digest of its functionality and the target problem of MWS matching in DME. In a typical scenario for which the matchmaker achieves the desired result, without barring its utilization under other scenarios, the following notions are taken into account: (1) The matchmaker is operation as well as user-centric as users are keenly interested in top-N MWS as opposed to discovering every MWS that is related to the request. (2) The provision of new MWS happens dynamically, and the request of any MWS happens efficiently. For any given MWS request R and MWS offers O ∈ SC described in hREST, with SC being the MWS collection of the self-adaptive MWS matchmaker, a set of relevant MWS that satisfy the users' request is returned by the matchmaker. To achieve this, the matchmaker initially learns the MWS relevance over a specified training set of positive and negative samples using M-NSA, and this is followed by matchmaking where the request is matched against the offer to obtained results based on the leaned relevance. The overview of the various stages of the proposed approach is shown in Fig. 2.

4.1 Determine the target category

Based on the work of [10], goal-oriented matching can be performed by mining functionalities of MWS described using hREST (textual descriptions). For self-adaptive MWS matchmaker, the experimentally top-performing, goal-based text similarity matching is adapted for its effectiveness. For the matchmaking to be efficient, the search space is reduced to the cluster of MWS that is most similar to a given MWS request [54]. Therefore, the most similar category or cluster of MWS request r as the target category of R is denoted by tc (r).

Given a request r, the first step of the self-adaptive MWS matchmaker is to determine the target category or cluster tc (r). of MWS request r is pre-processed by extracting the goals and generating a vector of r. Then, the cosine similarity between r and the center of categories or clusters, denoted by cc, is calculated using:

$$\text{Similarity}\left(\vec{r}, \vec{\text{cc}}\right) = \frac{\vec{r} \cdot \vec{\text{cc}}}{\left|\left|\vec{r}\right|\right| \cdot \left|\left|\vec{\text{cc}}\right|\right|} \tag{2}$$

This is used to calculate the cosine of the angle between the two vectors (MWS request vector denoted by $\vec{r}$ and the center of MWS categories or clusters denoted by $\vec{\text{cc}}$). This metric is a measurement of orientation and not magnitude. By normalizing the space, the problem with the magnitude of each word count is eliminated. Figure 3 shows an example of similarities between MWS request vector and MWS categories or clusters.

4.2 Service relevance learning (detector generation)

In order to find the most relevant MWS, the matchmaker learns the service relevance by using the detector generation stage of MNSA. The Describe the Detector Generation Stage in the proposed approach is a four-phase process, the first phase is random generation of candidate detectors based on self-sample given as Eqs. (3 and 4), the second phase is determining the affinity between the instances of the self-data and detectors computed using the Euclidean distance in Eq. (5), the third phase is the use of matching rules to determine the acceptability of the candidate detector as a matured detector, and finally, the fourth phase is the condition or criteria that stops the detector generation stage.

$$D^i = \left[d_1^i, d_2^i, \ldots \ldots \ldots d_Q^i\right] \text{where } i = 1, 2. \ldots \ldots \ldots N_d \tag{3}$$

$$S^j = \left[s_1^j, s_2^j, \ldots \ldots \ldots s_Q^j\right] \text{where} j = 1, 2. \ldots \ldots \ldots N_s \tag{4}$$

$$\text{dist}\left(D^i, S^j\right) = \sqrt{\sum_{k=1}^{Q} \left|d_k^i - s_k^j\right|^2} \tag{5}$$

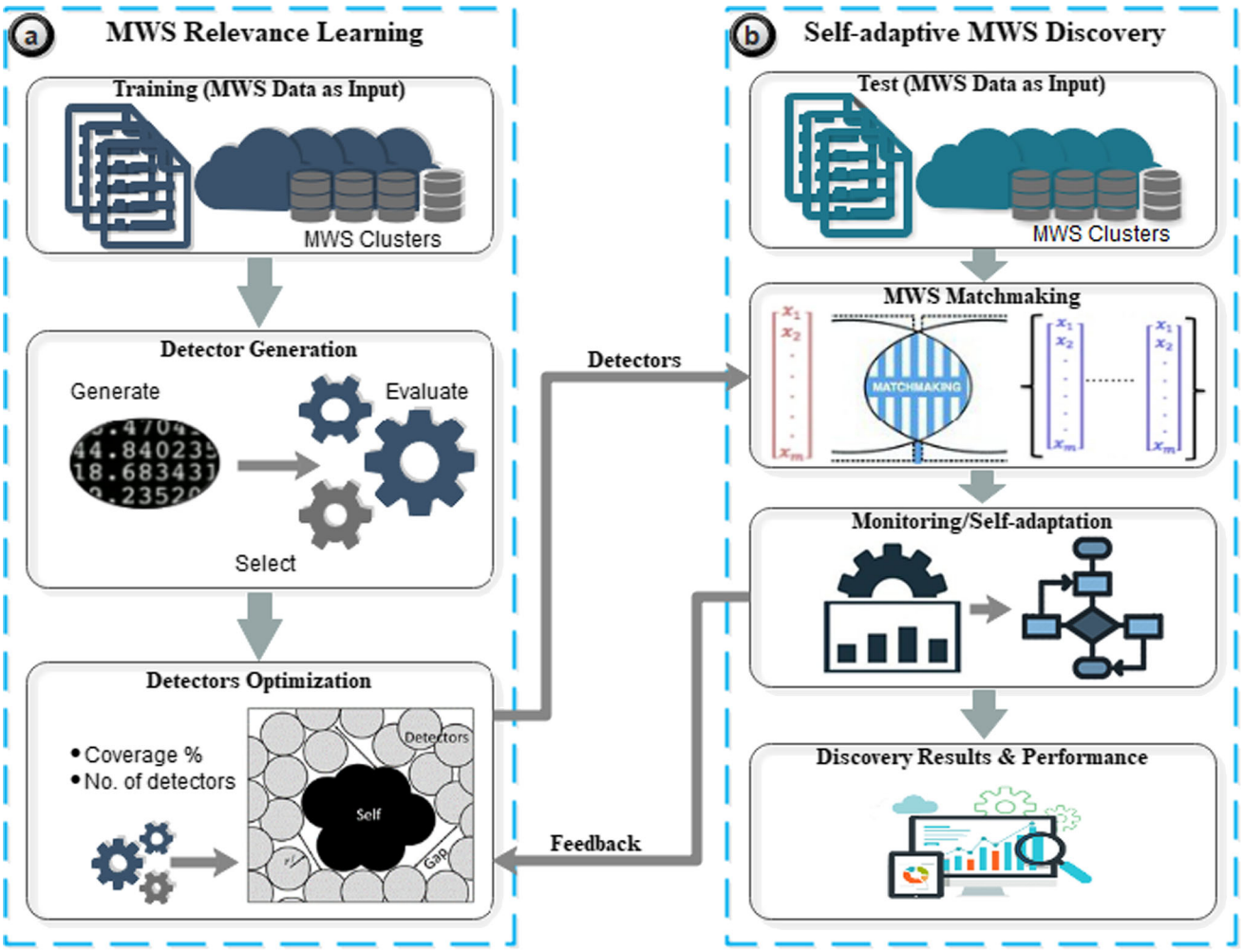

Fig. 2 Overview of the proposed self-adaptive approach

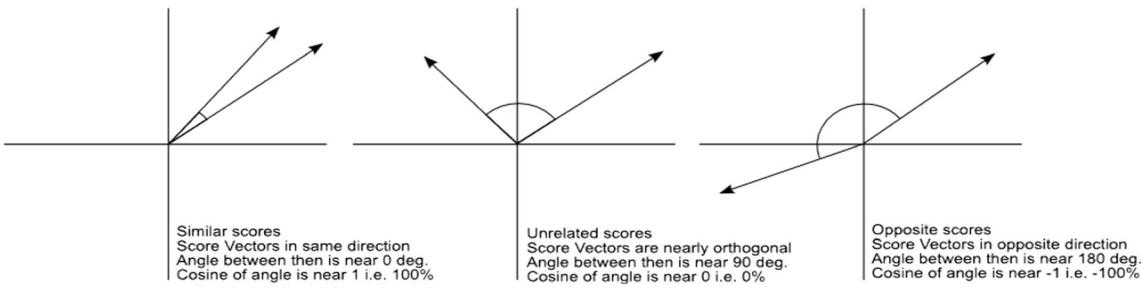

Fig. 3 The similarities between MWS request vector and MWS categories or clusters

4.2.1 Control parameters

The service relevance learning (detector generation stage) of the MNSA starts by assigning values to a number of control parameters. The control parameters are initialized parameters that are fundamental to the MNSA design and affect the behavior and output of the MNSA. The three control parameters, in this case, are self-radius *rs*, the estimated coverage *co*, and the maximum number of detectors *D*max.

Given the need to differentiate self-radius rs from the detector's radius rd, the self-radius *rs* is considered to be the most significant among the control parameters. The second control parameter is the maximum number of detectors $D_{\max}$ which is pre-set to allow the maximum number of detectors needed. The third control parameter is the estimated coverage E_c which is used to determine the

coverage based on the number the mature detectors and the sufficient number of detectors required. The estimated coverage $E_c = 1 - (1/U)$, where 1 is the full coverage, U is the number of uncovered regions, and $1/U$ is the estimated uncovered region. For example, if the number of uncovered regions is 3 out of 100, then the estimated coverage E_c is 67%.

The second and third control parameters are the determinants of the stopping criteria; the service relevance learning (detector generation stage) stops when the estimated coverage is achieved or when the maximum detectors are obtained. The elimination of other unnecessary control parameters such as the maximum age of the detector (t), makes it better to initialize the service relevance learning (detector generation stage) of the MNSA compared to other forms of NSA.

4.2.2 The size of the detectors

The detector size is significant as it can influence the performance of M-NSA, variable-sized detectors are utilized for this situation because it is not necessary to set the number of detectors ahead of time, the nonself space can be filled by a small number of variable-sized detectors, and similarly, small-scale detectors can cover the holes, thus minimizing the classification error, contrary to constant-size detectors. Figure 4 shows an example of constant-size detectors and variable-size detectors in 2-dimensional space in which the gray area signifies the self-region, which is typically specified through the self-samples/training data. The circles are the possible detectors covering the nonself region. The gaps also known as classification errors are illustrated in black.

The logical steps of the service relevance learning for SAMM are illustrated in detail in Fig. 5. Step 6 in Fig. 5 describes the two stopping criteria for service relevance learning (detector generation). Given that the main objective is to complete the detector generation at the exact or nearest to the target value (estimated coverage), therefore, the most desirable situation is to conclude based on the estimated coverage. However, an alternative criterion is setting a maximum number of detectors to handle situations when the estimated coverage is impractical. While the alternative is effective in comparison with the estimated coverage, its coverage effectiveness is far superior to a fixed-sized detector.

In step 7, the candidate detectors are generated, and since the MWS is clustered into eight (8) categories, there is a need to generate, optimize, and train eight sets of detectors corresponding to each MWS category. The self-sample from each of the eight categories is used to train the detectors from the candidate stage to the maturity stage after which the matured detectors (optimized detectors) are selected to be used in the detection stage.

Steps 8 through 10 in Fig. 5 describes the matching process, while steps 12 through 22 describe the condition that enables the addition and counting of the detectors already generated. The choice of including more detectors or not is complicated as the detector coverage is changing throughout the detector generation. This is because more detectors would lead to unnecessary detectors and fewer detectors would lead to a poor detection rate. Therefore, a fixed sample size is used for the coverage estimation and the determination of the maximum number of detectors which mostly keeps the cost of the service relevance learning under control.

4.3 Self-adaptive MWS discovery (detection stage)

After the service relevance learning (detector generation stage) is accomplished, the self-adaptive MWS discovery (detection stage) begins. Each instance MWS is introduced to the number of detectors that are generated (detector set). The affinity metric in Eq. 8 is used to measure the distance of each detector radius (rd) in the detector set, and if the distance is less than rd for any detector, the instance MWS is classified as nonself; otherwise, it is classified as self. By the rationale of the MNSA, if an instance of MWS is recognized by any detector, it is classified as nonself. On

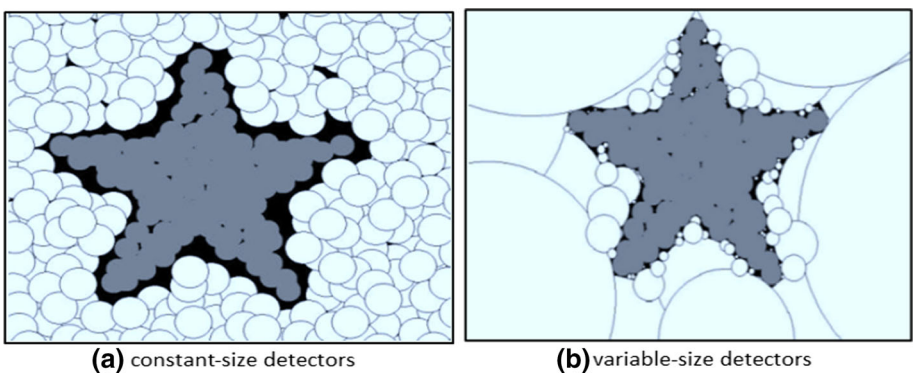

Fig. 4 An example of constant-size and variable-size detectors in 2 dimensional

Algorithm 1: Service relevance learning (initial detector generation)

Input: Training Set TS'
r_s: Self-radius
E_c: Estimated coverage
D_{max}: Maximum number of detectors
Output: Initial Detector Set DS'
while $(C < E_c) \,||\, (D < D_{max})$
  Generate a random Detector candidate $d\ for\ C_1, C_2, C_n$
  Calculate minimum distance to any self points, using Equation 2:
  $Dist(s,d) = \sqrt{\sum_{i=1}^{n}(s_i - d_i)^2}$
  if $(dist\_min < r_s)$
    Return to top
  else
    if $(D = 1)$
      Store detector as d_i and $dist\_min = r_{di}$
      Increment i + 1
    else
      Calculate minimum distance for each previous detector, *dist_min2,*
      if $(dist\_min2 < r_d)$
        $C = C + 1$,
      else
        Store detector as *di* and *dist_min2 = rdi,,*
        Increment C + 1
        $C = 0$,
      endif
    endif
  endif
end while

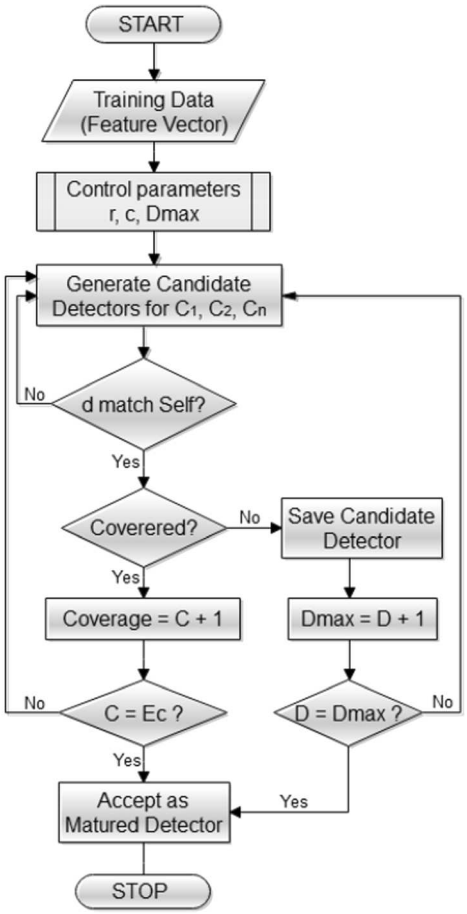

Fig. 5 Service relevance learning (initial detector generation)

the off chance that no detectors are fit for recognizing the instance MWS, it is classified as self.

4.3.1 Adoption of distance metrics

Matching rules are based on a based distance measure and both play the most important role in NSA. Matching rules are used both in the service relevance learning (detector generation stage) and in the self-adaptive MWS matchmaking (detection stage). The domain and the dataset representation are used as the criteria to determine the most appropriate matching rules. r-contiguous bit, r-chunks, and Hamming distance are more compatible with binary data representation, while Euclidean distance, Minkowski distance, and Manhattan distance are more compatible with real-valued data representation.

While there are a few control parameters that might be changed to influence the performance of M-NSA, choosing the most suitable distance measure between Euclidean distance shown in Eq. (8), Minkowski distance shown in Eq. (6), and Manhattan distance shown in Eq. (7) is critical to the performance of the M-NSA algorithm. This is because the number of detectors generated, the shape of a detector in an n-dimensional space, and the assessment of detector coverage are all determined by the distance measure in real-valued data representation. Likewise, during the detection stage, the decision rule executed to classify the unknown incoming data instance as either self or nonself is based on a distance measure. Therefore, the Euclidean distance is adopted as the matching rule of M-NSA as it is more effective on real-valued data compared to Minkowski distance and Manhattan distance [55].

$$\text{Minkowski Dist}(X,Y) = \left(\sum_{i=1}^{n}|x_i - y_i|^p\right)^{1/p} \tag{6}$$

$$\text{Manhattan Dist}(X,Y) = \sum_{i=1}^{n}|x_i - y_i| \tag{7}$$

$$\text{Euclidean Dist}(X,Y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \tag{8}$$

where p is the positive integer 1 in the Manhattan distance and p is the positive integer 2 in the Euclidean distance, and x_i, y_i are the coordinates of x and y, respectively.

4.3.2 Matchmaking strategy

The matchmaking strategy is founded on a practical principle of request capabilities and offered capabilities of MWS to satisfy the users; hence, a logic-based approach is integrated into the M-NSA, and the strategy entails a series of steps that are followed to get the most relevant MWS weighting each of the matching cases as shown in steps 8 through 19 in Fig. 6. In detail, users may request MWS with certain capabilities; if these capabilities are identical or similar to the offered capabilities by the service providers ($R_s \equiv O_s \vee R_s = O_s$), it is of greater use compared to any other case. However, if the offered capabilities are more generic ($R_s \supset O_s$), it is still more desirable compared to a situation where the offer can fulfill only a certain part of the request ($R_s \subset O_s$). Moreover, if the offered capabilities and the requested capabilities share common features ($R_s \cap O_s$) it is still desirable compared to the absence of any positive relationship between the request and the offer ($R_s \neq O_s$).

The logical strategy is extended further by converting any request that did not yield the desired offer into a detector. The affinity binding between detectors generated in the service relevance learning stage and the new service request R_s is compared against the threshold τ; if the affinity is greater than or equal to the threshold, then the new service request R_s is qualified to be a candidate detector, else, the position of the detectors will be only

Algorithm 2: The self-adaptive matchmaking algorithm

1. **Input:** The set of Offered MWS (O);
2. R_s: Requested MWS (R);
3. D_d: Initial set of detectors and the threshold τ
4. **Output:** The top-N S that best fit the request
5. $S \leftarrow \emptyset$
6. **for each** *service offer* $s \in O$ **do**
7. *Calculate the similarity of* s *with every detector* d *in* $D, \forall\, d \in D$ *using* $Dist(s,d) = \sqrt{\sum_{i=1}^{n}(s_i - d_i)^2}$
8. *Check the Distance* $Dist(s,d)$;
9. **if** $R_s \equiv O_s \vee R_s = O_s$ // Exact Match
10. *weight* ← 1 // *Assign weight +=1*
add the new s to Top-N S;
11. **else if** $R_s \supset O_s$ // Plug-In Match
12. *weight* ← 0.75 // *Assign weight +=0.75*
add the new s to Top-N S;
13. **else if** $R_s \subset O_s$ // Subsumption Match
14. *weight* ← 0.5 // *Assign weight +=0.5*
add the new s to Top-N S;
15. **else if** $R_s \cap O_s$ // Intersection Match
16. *weight* ← 0.25 // *Assign weight +=0.25*
add the new s to Top-N S;
17. **else if** $R_s \neq O_s$ // Undesirable/Fail Match
18. *weight* ← 0 // *Assign weight +=0*
19. **end if**
20. **end do**
21. **for each** *service request* R_s **do**
22. **if** $Dist(R_s, D) \geq \tau$ //Updating the detector set
23. *Return Top-N S*
24. **else** *Update D*
25. *Generate new d using Use new* R_s
26. *Add new d to D*
27. *Go to 7*
28. **end if**
29. **end do**
30. **end for**
31. **end for**

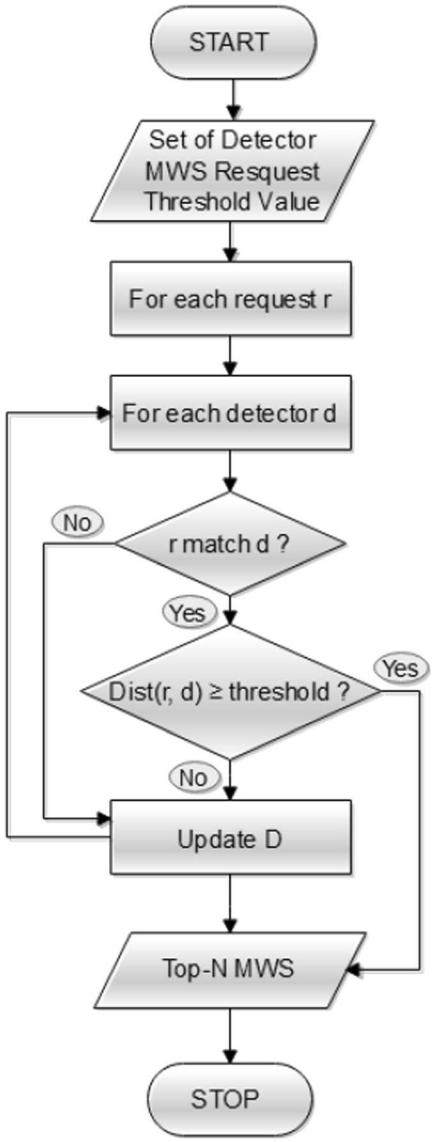

Fig. 6 The self-adaptive matchmaking algorithm

adjusted as other detectors of equal importance are already available.

Overall, the matchmaking algorithm is designed to continually recognize the user's desired MWS; therefore, five matching cases are considered in defining the degree of match (DOM). Exact match ($R_s \equiv O_s \vee R_s = O_s$), plug-in match ($R_s \supset O_s$), subsumption match ($R_s \subset O_s$), intersection match ($R_s \cap O_s$), and fail/nonmatch ($R_s \neq O_s$). Since the MWS data and detectors are represented by real-valued data, the proposed matchmaking strategy always deals with similarity as a real value in [0, 1]. After the examination of several methods that can be used to represent the DOM, weights between 0 and 1 are adopted to reflect the real-valued representation of the MNSA. Consequently, each of the five DOM (Exact, Plug-in, Subsumes, intersection, and Fail) carries a weight of 1, 0.75, 0.5, 0.25, and 0, respectively. The empirical evaluation supports the assertion that this, in addition to the service relevance learning (detector generation), curtails and compensates for the high number of false negatives associated with strict logic matchmaking strategy.

The logical steps of the SAMM are illustrated in Fig. 6. SAMM adjust adaptively the detectors generated during service relevance learning according to the MWS discovery result. Steps 22 through 27 in Fig. 6 describes the condition that enables the self-adaptation to take place. This is because when too many FN occurs, the detectors need to be updated to ensure the discovery of relevant MWS. The two contributing factors of FN are holes in the self-space that the generated detector cannot cover and the too-large self-radius that cover nonself space. Moreover, when the set of self-elements changes (offered MWS) as a result of the addition of new or unavailability of MWS, adjusting the detectors becomes necessary to respond to the changes.

5 The empirical evaluation

This section covers different experiments that are conducted to evaluate the proposed self-adaptive mobile web service discovery approach based on a modified negative selection algorithm. The purpose of the experiments is to answer two questions: (1) What is the performance of the proposed SRL model in terms of DR and FAR? (2) What is the performance of the matchmaker despite the changes in DME? The proposed model as well as the matchmaker was developed and implemented using Python 3.7 version in Anaconda Navigator 1.9.2 and carried out on a PC with Intel Core$^{(TM)}$ CPU i5-5200U, @2.20 GHz, and 8 GB RAM, running the Windows 10 Home Edition 64-bit OS. The source code for the proposed approach is readily and freely accessible in GitHub [56] for testing. This section also discusses the rationale behind experimental decisions, and choices regarding test collections, settings, evaluation metrics, baseline algorithms, and evaluation results are also discussed.

5.1 Dataset description

In order to ensure impartial results and analysis, a great deal of experimental norms is taken into consideration, and the training, testing, and comparison with related algorithms are all conducted under the same experimental conditions. According to [57], the ProgrammableWeb dataset is one of the contemporary, consistent, and largest RESTful web service collections from different domains publicly accessible for evaluation; in addition, its daily evolution set it apart from other test collections within the Semantic Service Matchmaker Evaluation Environment (SME2). Therefore, it is employed as a testbed to evaluate the proposed self-adaptive MWS matchmaking algorithm.

As shown in Table 1, the crawled dataset from PW consists of 13,520 MWS and 5769 Mashups from more than 400 different domains (categories) such as video, social, messaging, photographs, games. Given that the objective constraint of this experiment is MWS, a large dataset is required for service relevance learning (detector generation stage) and the ProgrammableWeb dataset is described using hREST, and information such as name, tags, description is extracted. It also undergoes through pre-processing that involves filtration, normalization, and organization to obtain the feature vectors for training and testing which serve as the inputs of the service relevance learning and MWS discovery.

Data splitting strategy is employed to ensure the resulting model's generalization ability is as high as possible, and the intensity of the computation, a balance between the variance of parameter estimates and performance statistics (underfitting and overfitting) should be considered before data splitting decision [58]. Therefore, the test collection is split into 80:20 (80 for training and 20 for validation) parts to ensure reasonable computation time and reduce biases.

Table 1 The statistics of the ProgrammableWeb dataset

SN	Attributes	Value
1	Number of MWS	13,520
2	Number of categories	466
3	Description language	hRESTS
4	Number of composite MWS/mashups	5769
5	Avrg. No. of MWS per composite MWS/mashups	2
6	Avrg. No. of token per MWS	43

5.2 Performance criteria and evaluation metrics

Two performance criteria determine the success of the proposed matchmaking algorithms. First is the ability of the service relevance learning (detector generation) algorithm to have a high detection rate (DR) and low false alarm rate (FAR). The DR is similar to precision as shown in Eq. 9, while the FAR is the MWS incorrectly label as irrelevant (false positives) divided by the total irrelevant MWS (false positives plus true negatives) as shown in Eq. 12. Second is the matchmaking algorithm's ability to differentiate between MWS that are relevant to a given request and irrelevant MWS to a given request. This is usually known as the binary test, followed by the ability of the algorithm to return with the correct estimation of the degree of relevance. This is usually known as a graded relevance test.

The second performance criteria involve the use of F-Measure to obtain the test's accuracy. F-measure is a combined metric (a weighted average of the precision and recall or harmonic mean of precision and recall) firstly introduced in information retrieval. 1 represents the best value, while 0 represents the worst value of F-measure. While recall expresses the ability of the algorithm to find all the relevant MWS within a test collection, precision expresses the proportion of the MWS instances the algorithm says was relevant actually were relevant. The detailed definition of precision is the number of MWS correctly label or discovered as relevant (true positives) divided by the relevant MWS that should be discovered (true positives plus false positives). The detailed definition of recall is the number of MWS correctly label or discovered as relevant (true positives) divided by the total relevant MWS label or discovered (true positives plus false negatives). True positives (TP), false negatives (FN), true negatives (TN), false positives (FP) are described in Table 4. The evaluation metrics for precision, recall, and F-measure are shown in Eqs. 9, 10, and 11, respectively.

$$\begin{aligned}\text{Precision} &= \frac{\text{MWS correctly label or discovered as relevant}}{\text{Total relevant MWS discovered}} \\ &= \frac{\text{True Positives (TP)}}{\text{True Positives (TP) + False Positive (FP)}}\end{aligned} \tag{9}$$

$$\begin{aligned}\text{Recall} &= \frac{\text{MWS correctly label or discovered as relevant}}{\text{Total relevant MWS discovered}} \\ &= \frac{\text{True Positives (TP)}}{\text{True Positive (TP) + False Negatives (FN)}}\end{aligned} \tag{10}$$

$$F - \text{Measure} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \tag{11}$$

$$\begin{aligned}\text{FAR} &= \frac{\text{MWS incorrectly label as irrelevant}}{\text{Total irrelevant MWS}} \\ &= \frac{\text{False Positives (FP)}}{\text{False Positive (FP) + True Negatives (TN)}}\end{aligned} \tag{12}$$

Rank-aware evaluation metrics such as MAP in Eq. 13 and NDCG in Eq. 14 are used to ensure relevant MWS is at the upper part of the list of discovered MWS. The precision considers the whole list as a set of MWS and treats all the errors in the discovery list equally. MAP (binary relevance) is able to give more weight to errors that happen high up in the list of discovered MWS. NDCG (graded relevance) is able to use the fact that some MWS are "more" relevant than others. Highly relevant MWS should come before medium relevant MWS, and so on (Table 2).

$$\text{MAP} = \frac{\sum_{q=1}^{Q} \text{Ave}P(q)}{Q} \tag{13}$$

$$\text{NDCG} = \frac{1}{Z}\sum_{k=1}^{N} \frac{2^{\text{rel}_k}}{\log_2(k+1)} \tag{14}$$

5.3 SRL model parameter turning

To ensure that the service relevance learning (detector generation) can produce the best set of detectors, the influence of three important parameters (self-radius r_s, estimated coverage E_c, and the maximum number of detectors D_{max}) needs to be tested in various settings. The self-radius r_s is an important parameter in any NSA, since small r_s may lead to false-positive outcomes, while a large r_s may lead to false-negative results. The selection of mature detectors is subject to the initial and final fitness values for candidate detectors, generating variable-sized candidate detectors maximum diversity and minimum overlap between detectors to ensure that a reduced number of detectors coverage the entire search space to lessen the amount of computation. Therefore, both the estimated coverage E_c and the maximum number of detectors D_{max} are used as the termination conditions to ensure a sufficient number of detectors as well as full coverage. To validate the impact of the SRL method for the MWS discovery, we apply various settings to our MNSA model accordingly:

- *MNSA-1*: This variant is intended to analyze the effect of the self-radius r_s. In this case, the self-radius r_s is set between 0.01, 0.3, and 0.05, while the maximum number of detectors D_{max} and the estimated coverage E_c were not given much consideration during the model training.
- *MNSA-2*: This variant is intended to analyze the effect of the maximum number of detectors D_{max}. In this case, the maximum number of detectors D_{max} is set between

Table 2 The four probable results of the test

Variables		Description (number of)
TP	True positive	MWS correctly label or discovered as relevant
FP	False Positive	MWS incorrectly label or discovered as relevant
TN	True negative	MWS correctly label or discovered as irrelevant
FN	False negative	MWS incorrectly label or discovered as irrelevant

100, 200, and 300, while the self-radius r_s and the estimated coverage E_c were not given much consideration during the model training.

- *MNSA-3*: This variant is intended to analyze the effect of the estimated coverage E_c. In this case, the estimated coverage E_c is set between 80%, 90%, and 99%, while the self-radius r_s and the maximum number of detectors D_{max} were not given much consideration during the model training.

The evaluation metrics discussed previously (detection rate DR and false alarm rate FAR) are used to measure the service relevance learning (detector generation) effectiveness in the experiments.

5.4 Results analysis of the SRL model

Experimental results are shown in Tables 3, 4, and 5. Table 3 demonstrates the comparisons of detection rates and false alarm rates of the three MNSA (MNSA-1, MNSA-2, MNSA-3) in terms of self-radius under the equivalent experimental conditions (estimated coverage E_c of 99% and the maximum number of detectors D_{max} 100). It is noticeable that the DR and the FAR are higher when the self-radius r_s is smaller, while the DR and the FAR are smaller when the self-radius r_s is higher.

Table 4 demonstrates the comparisons of detection rates and false alarm rates of the three MNSA (MNSA-1, MNSA-2, MNSA-3) in terms of the maximum number of detectors under the equivalent experimental conditions (estimated coverage E_c of 99% and self-radius r_s 0.01). It is noticeable that the increase in the maximum number of detectors does not result in a significant increase in the DR, but there is an increase of the FAR; however, the maximum number of detectors works side-by-side with the size of the training set.

Table 5 demonstrates the comparisons of detection rates and false alarm rates of the three MNSA (MNSA-1, MNSA-2, MNSA-3) in terms of estimated coverage under the equivalent experimental conditions (self-radius r_s 0.01 and the maximum number of detectors D_{max} 100). It is noticeable that the DR gradually increases as the estimated coverage increases, while the FAR gradually decreases as the estimated coverage increases.

Figure 7 shows the comparisons of the maximum number of detectors for MNSA-1, MNSA-2, and MNSA-3 under the same expected coverage, and the same training sets, it is noticeable that the number of mature detectors required to satisfy the coverage specification of 99% differs between the three algorithms. As shown in Fig. 7a, MNSA-1 needs almost 100 detectors to satisfy the coverage specification, as shown in b, MNSA-2 needs almost twice the number of mature detectors required by MNSA-1 to satisfy the coverage specification, and as shown in Fig. 7c, MNSA-3 needs almost 300 mature detectors to achieve the expected coverage of 99%. This is due to the increase in the self-radius r_s in which MNSA-1 has a self-radius r_s of 0.01, MNSA-2 has a self-radius r_s of 0.03, and MNSA-3 has a self-radius r_s of 0.05. Therefore, the higher the self-radius r_s, the higher the number of detectors required to cover the entire self-space.

Figure 8 shows the comparisons of the DR and FAR for MNSA-1, MNSA-2, and MNSA-3 under the same expected coverage, and same training sets, it is noticeable from the figures that the DR and the FAR of the three algorithms slightly differ when the expected coverage is greater than 50%. There is a significant increase in the DR of greater than 90% for MNSA-1 due to its smaller self-radius r_s and the increase in the estimated coverage, while the DR for MNSA-2 and MNSA-3 is lower than 90% due to the larger self-radius r_s. However, the FAR slightly increases as the DR increases due to coverage of self-data located at the edge of the self-space. Though it is not ideal for the FAR to slightly increase due to the increase in the DR, it is still

Table 3 The comparisons of detection rates and false alarm rates in terms of self-radius

Algorithm	Self-radius r_s = 0.01		Self-radius r_s = 0.03		Self-radius r_s = 0.05	
	DR %	FAR %	DR %	FAR %	DR %	FAR %
MNSA-1	0.92	0.26	0.88	0.25	0.79	0.10
MNSA-2	0.84	0.25	0.81	0.22	0.72	0.07
MNSA-3	0.81	0.22	0.77	0.18	0.69	0.03

Table 4 The comparisons of detection rates and false alarm rates in terms of self-radius

Algorithm	No. of D. D_{max} = 100		No. of D. D_{max} = 200		No. of D. D_{max} = 300	
	DR %	FAR %	DR %	FAR %	DR %	FAR %
MNSA-1	0.92	0.26	0.93	0.52	0.93	0.53
MNSA-2	0.84	0.25	0.85	0.43	0.85	0.46
MNSA-3	0.81	0.22	0.82	0.40	0.82	0.43

Table 5 The comparisons of detection rates and false alarm rates in terms of self-radius

Algorithm	Estimated coverage E_c = 80%		Estimated coverage E_c = 90%		Estimated coverage E_c = 99%	
	DR %	FAR %	DR %	FAR %	DR %	FAR %
MNSA-1	0.78	0.43	0.86	0.41	0.92	0.26
MNSA-2	0.72	0.40	0.75	0.39	0.84	0.25
MNSA-3	0.69	0.40	0.72	0.38	0.81	0.22

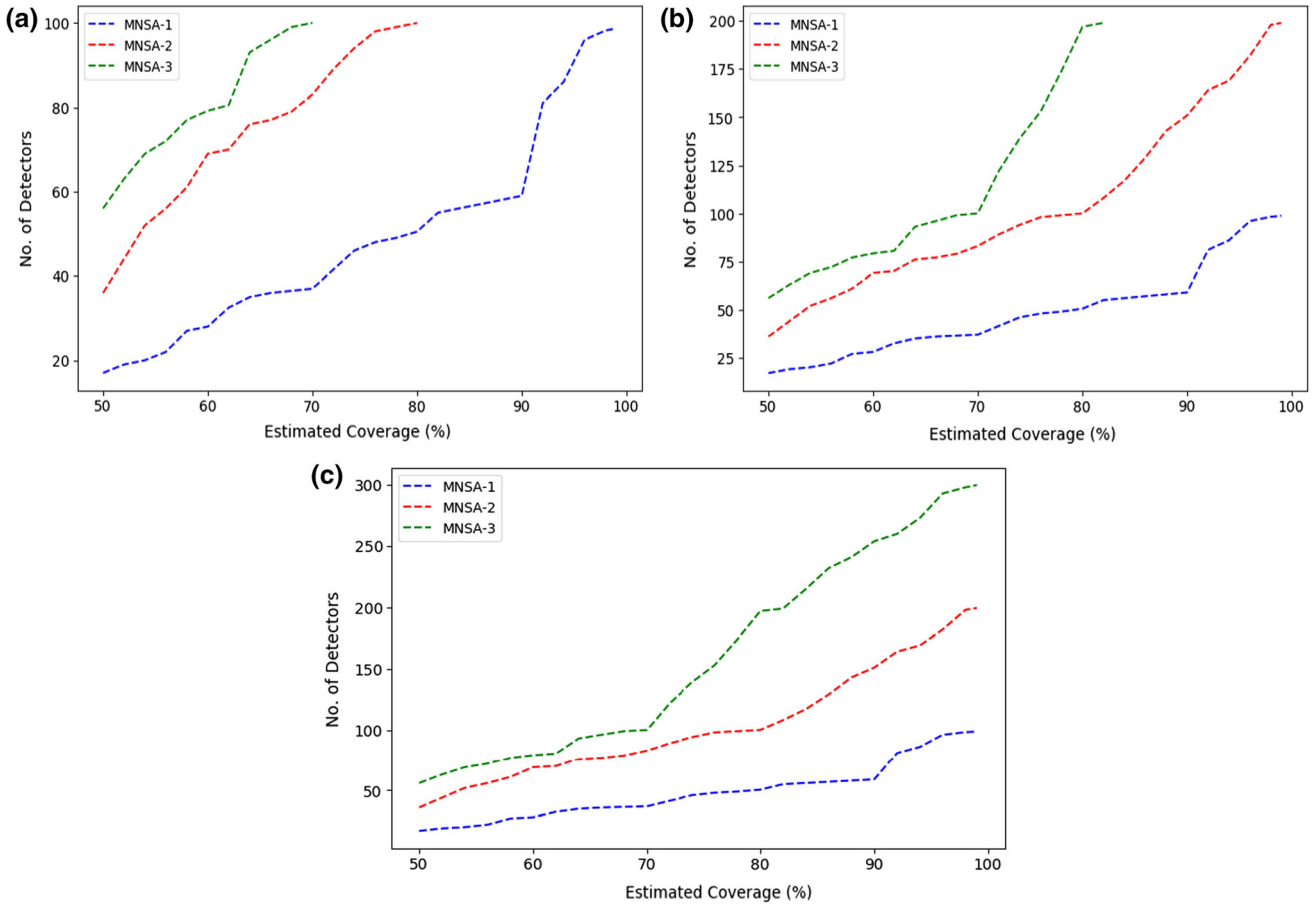

Fig. 7 Comparisons of the maximum number of detectors for MNSA-1, MNSA-2, MNSA-3

more suitable than other cases where the DR is significantly low and FAR is slightly low.

5.5 Evaluation of the matchmaking

The proposed matchmaking algorithm involves the use of the service relevance learning model, a collection of MWS,

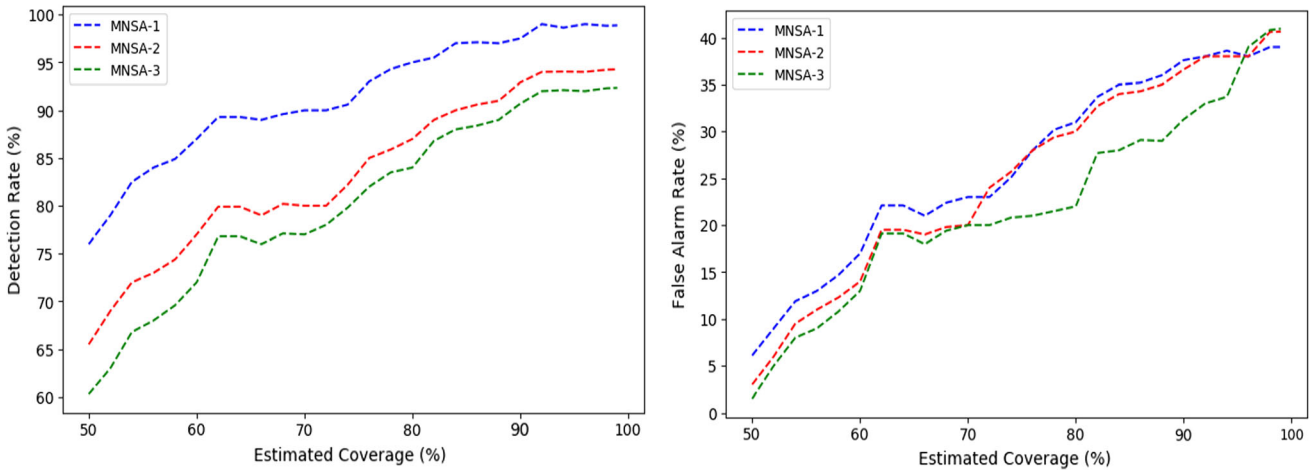

Fig. 8 Comparisons of the DR and FAR for MNSA-1, MNSA-2, MNSA-3

and a set of MWS requests. To measure the effectiveness of the proposed approach, the value of two important parameters (τ and N) needs to be ascertained to ensure the most relevant services are discovered (Top-N). The threshold τ is used to determine the need for generating new detectors in a situation where a given query does not yield a relevant result, while the number of relevant services to be returned for a given request is determined by N. This is because if N is set too high, it can induce overfitting and hence decrease the matchmaking generalization. Conversely, if N is set too low, it can induce underfitting, hence difficult to describe the interactions between offer and request of MWS. Figure 8 shows the matchmaking performance under different N in terms of precision, recall, F-1, and MAP for the ProgrammableWeb dataset.

The threshold value is set between 0.1 and 1 because the increase in the value of the threshold from 0.1 to 0.5, leads to an increase in precision and recall. On the contrary, the increase in the value of the threshold from 0.5 to 0.9, leads to the continued decrease of the precision and recall decrease. Although the higher threshold signifies the discovery of MWS closer to the exact matching, the possibility of having exact matching is sometimes unlikely. Likewise, the lower value of the threshold signifies the discovery of MWS closer to the fail matching, the possibility of having failed matching is more likely than not. Therefore, the best precision and recall are attained when the value of the threshold reaches 0.5. So, the value of the threshold is settled at 0.5.

Moreover, when the value of N is set at 5, the variation of the precision and recall values follows a nonlinear pattern as the value of the threshold increases from 0.1 to 1.0. This is because the small value of the threshold is not significant enough to determine the relevance of MWS for a given request, which results in a lower value of precision and recall at the beginning. As for when N is set at 10, 15, 20, 25, 30, 35, 40, 45, and 50, there is an accelerated improvement in the value of precision and MAP which steadied at 15 and moderate slows down. This is because when N is set at 15, it is quite sufficient to contain the most relevant MWS as well as eliminate less relevant MWS compared to when N is set at a higher number (Fig. 9).

5.6 Comparative evaluation with baseline approaches

Several state-of-the-art MWS discovery approaches are chosen as baseline approaches for comparison with the proposed self-adaptive mobile web service discovery approach. Considering that the proposed approach incorporates different attributes of the MWS profile, these selected baseline approaches cover the four types of matchmaking (logic-based, nonlogic based, adaptive, and hybrid), to make a comprehensive comparison while demonstrating the performance of the proposed approach. It should be remembered that, in GitHub [56], most of the source codes are readily and freely accessible, while others are accessible on request for any of these baseline approaches. So, for a reasonable comparison, the actual codes given by the original authors and the settings given in the corresponding articles were explicitly used. The selected baseline approaches for MWS discovery are illustrated as follows:

- *GSD* [10]: Goal-based service discovery (GSD) is an approach that retrieves services by matching goals of services with those contained in an expanded/refine query using keyword-based and topic model-based, NLP-based method is used to extract service goals from services' textual descriptions, and clusters are created based on the semantic similarities between the extracted service goals.

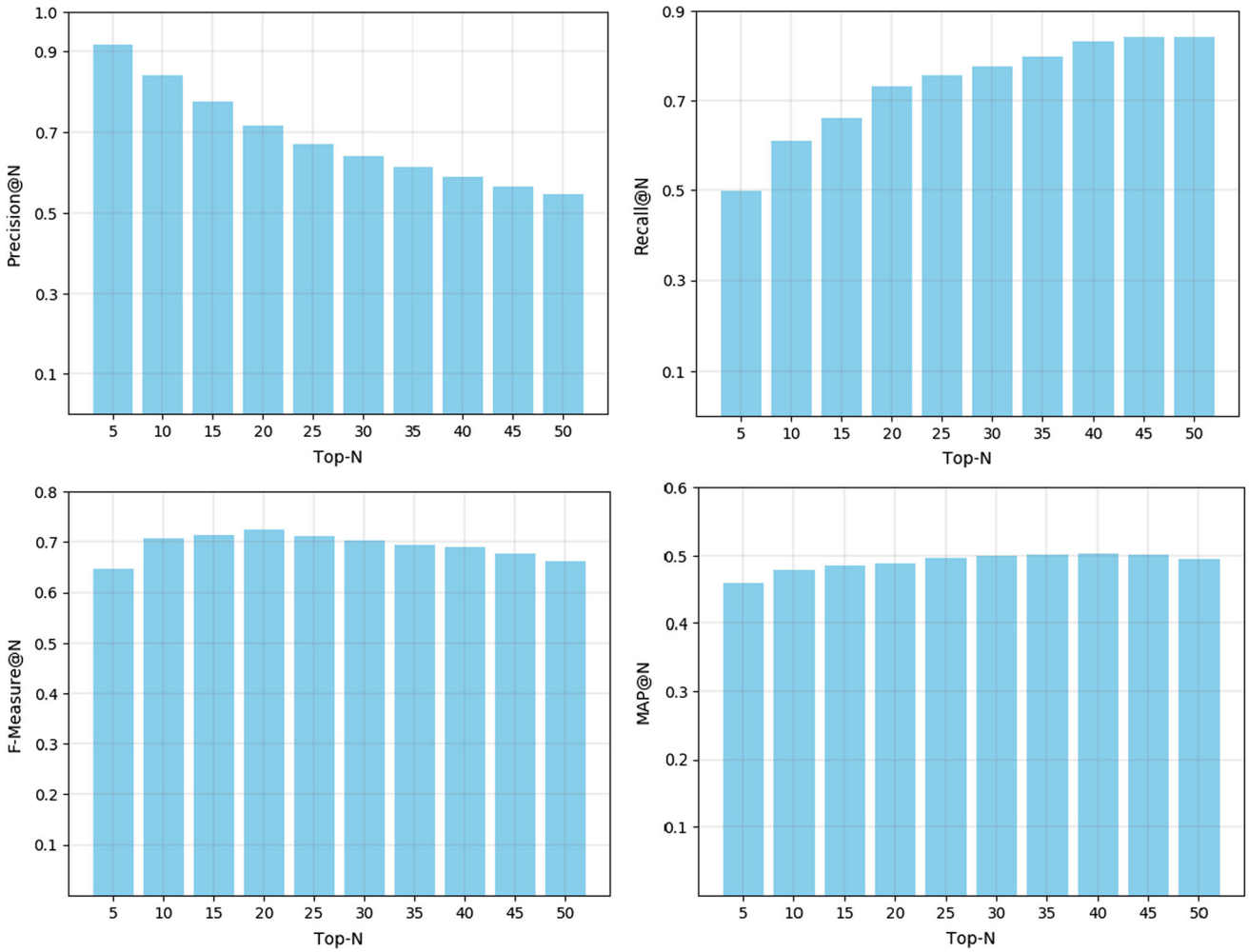

Fig. 9 Matchmaking performance under different N

- *DHCF* [9]: Deep Hybrid Collaborative Filtering (DHCF) is an approach that uses a deep neural network to capture the nonlinear relationship between services. The pointwise loss function is used to transform the discovery task into a regression or classification problem. Collaborative filtering and textual content are employed to improve relevant service discovery.
- *GLDA* [11, 59]: This is an adaptation of probabilistic latent semantic analysis (pLSA) in which the service attributes are modeled as a heterogeneous information network (HIN) and LDA is used to capture the underlying semantics of services and queries. The service discovery result is generated after the integration.
- *RTM-FM* [37]: The Relational Topic Model (RTM) and Factorization Machines (FMs) is an approach that models the relationship between topics of MWS and uses factorization machines to predict MWS for a target request. This approach is also capable of evaluating parameters under low sparsity like support vector machines (SVMs).
- *KWSD* [10]: Keyword-based service discovery is an approach that discovers services based on the TF-IDF. The cosine similarities between services and the requests are calculated after which service discovery result is generated based on the similarities.
- *NCF* [36]: Neural Collaborative Filtering (NCF) is an approach that uses a neural architecture to learn an arbitrary function from web service data and capture the nonlinear relationship between services and then generate service discovery results based on the given request.

After the determination of the similarity threshold and the maximum number of N, the five-evaluation metrics discussed in Sect. 5.2 (MAP, NDCG, precision, recall, F-measure) are used to compare the performance of the proposed approach and baseline approaches. Tables 6, 7, and 8, and Fig. 10 show the evaluation result of the

Table 6 Evaluation results of Precision@N

Approaches	Precision@N									
	$N = 5$	$N = 10$	$N = 15$	$N = 20$	$N = 25$	$N = 30$	$N = 35$	$N = 40$	$N = 45$	$N = 50$
GDS	0.8498	0.8060	0.7658	0.7055	0.6475	0.5687	0.5044	0.4274	0.3899	0.3688
DHCP	0.1719	0.1090	0.0801	0.0642	0.0538	0.0461	0.0415	0.0368	0.0337	0.0311
GLDA	0.8002	0.7648	0.7180	0.6784	0.6543	0.6318	0.6064	0.5838	0.5598	0.5400
RTM-FM	0.6896	0.4425	0.3870	0.3352	0.2797	0.2590	0.2405	0.2257	0.2161	0.2094
KWSD	0.1023	0.0730	0.0539	0.0441	0.0370	0.0310	0.0261	0.0234	0.0201	0.0191
NCF	0.1641	0.1400	0.1282	0.1218	0.1182	0.1152	0.1126	0.1109	0.1092	0.1088
SAMM	0.9167	0.8413	0.7745	0.7155	0.6703	0.6414	0.6139	0.5888	0.5648	0.5450

Table 7 Evaluation results of Recall@N

Approaches	Recall@N									
	$N = 5$	$N = 10$	$N = 15$	$N = 20$	$N = 25$	$N = 30$	$N = 35$	$N = 40$	$N = 45$	$N = 50$
GDS	0.4892	0.6007	0.6515	0.7121	0.7374	0.7560	0.7796	0.7915	0.8211	0.8308
DHCP	0.5085	0.6200	0.6708	0.7097	0.7350	0.7536	0.7671	0.7790	0.7908	0.8010
GLDA	0.4207	0.4626	0.5000	0.5264	0.5552	0.5851	0.6161	0.6431	0.6649	0.6787
RTM-FM	0.2524	0.4689	0.5218	0.5747	0.6234	0.6275	0.6302	0.6358	0.6400	0.6455
KWSD	0.3382	0.4676	0.5303	0.5931	0.6234	0.6457	0.6659	0.6841	0.7003	0.7145
NCF	0.3912	0.5008	0.5481	0.5801	0.6020	0.6240	0.6392	0.6476	0.6611	0.6695
SAMM	0.4991	0.6106	0.6614	0.7320	0.7569	0.7755	0.7990	0.8327	0.8417	0.8430

Table 8 Evaluation results of F-measure@N

Approaches	F-measure@N									
	$N = 5$	$N = 10$	$N = 15$	$N = 20$	$N = 25$	$N = 30$	$N = 35$	$N = 40$	$N = 45$	$N = 50$
GDS	0.6210	0.6884	0.7040	0.7088	0.6895	0.6491	0.6125	0.5551	0.4285	0.3794
DHCP	0.2326	0.1713	0.1342	0.1115	0.0946	0.0829	0.0750	0.0681	0.0623	0.0575
GLDA	0.5502	0.5778	0.5942	0.6006	0.5976	0.6090	0.6204	0.6112	0.6050	0.6002
RTM-FM	0.4891	0.3891	0.3636	0.3380	0.3141	0.2897	0.2685	0.2495	0.2402	0.2272
KWSD	0.1571	0.1263	0.0979	0.0821	0.0699	0.0592	0.0502	0.0453	0.0391	0.0372
NCF	0.2302	0.2220	0.2165	0.2132	0.2111	0.2093	0.2082	0.2068	0.2062	0.2055
SAMM	0.6463	0.7076	0.7135	0.7237	0.7110	0.7021	0.6943	0.6898	0.6760	0.6620

different approaches. Figure 10a depicts precision comparison, Fig. 10b depicts recall comparison, Fig. 10c depicts F-measure comparison, Fig. 10d depicts MAP comparison, Fig. 10e depicts NDCG comparison. Several important observations are attained from the comparison which is discussed in the following section.

5.7 Results and discussion

As can be seen, the proposed approach exhibits significant improvements over the competing approaches at certain points within certain parameters, more specifically, the number of N. As illustrated in Fig. 10a, with the gradual increase in the number of N, the precision decreases slightly in SAMM, GDS, and GLDA. This is because the larger the size of N, the more services will be matched and

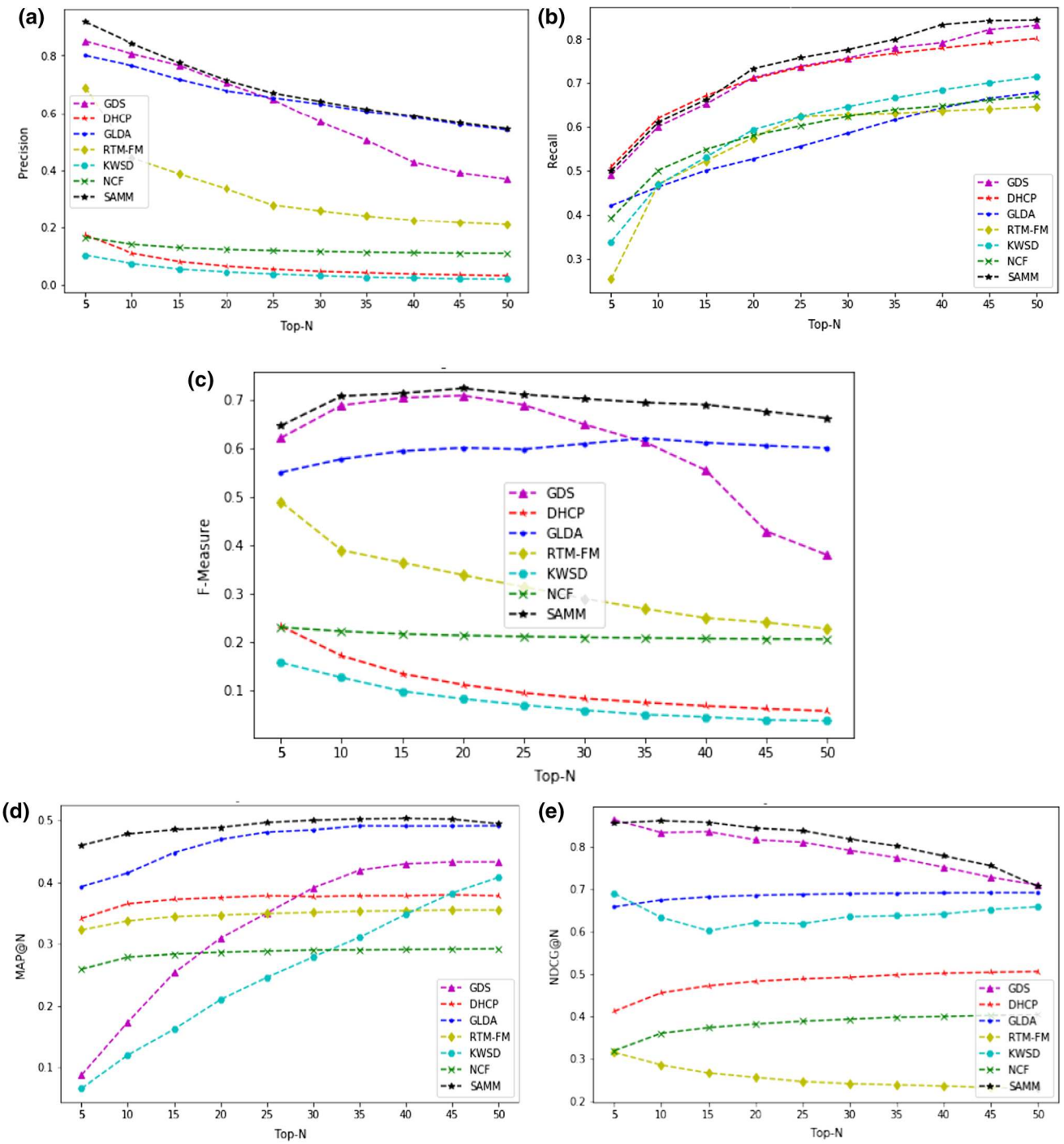

Fig. 10 The performance comparison of different approaches

retrieved. Other approaches such as MTR-FM, DHCP, KWSD, and NCF show stability as the number of N increases. However, recall and the F-measure of all the approaches show a gradual increase as the number of N increases as expected considering that N is used to limit the number of MWS to be returned for every request. Given that the recall ensures that no relevant MWS is missing, it is still outfitted with a top-N bound due to the objective of SAMM which is to return the most relevant MWS.

Compared with the latest baseline approaches, SAMM performs slightly better in evaluation metrics such as recall (0.8430) and performs significantly better in precision (0.9167), MAP (0.5028), and NDCG (0.8613). This is because the high sparsity of the interaction matrix in MTR-FM hinders the accurate classification of MWS. While the

adoption of neural network strategy in NCF and DHCF leads to significant improvement in the performance by capturing nonlinear relations between MWS, these approaches fall slightly short, given that SAMM does not have the partial score issues that are widely attributed to NCF and DHCF.

The evaluation shows that GSD, a goal-based approach that measures the similarities between the goals of MWS and MWS, requests that can be expanded with alternative semantics achieves better precision (0.8498) than GLDA (0.8002), NCF (0.1641), MTR-FM (0.6896), KWSD (0.1023), and DHCF (0.1719). The results indicate that the MWS goals can provide significant support for MWS discovery if extracted appropriately, which also validates the decision to integrate MWS goals at the base of SAMM. Figure 10a depicts a precision comparison in which KWSD performs lower (0.1023) than all the other approaches, given that it is based on TF-IDF after extensive refinement that leads to a high number of false positive (FP/F + ve) and the false negative (FN/F-ve) in matchmaking.

Another significant observation is that the performance of SAMM based on few numbers of N is better than the LDA-based approach but not as consistent. One possible explanation is the use of context information to enrich the semantics of both MWS and well as user requests, thereby boosting the MWS discovery performance of SAMM. Moreover, the LDA-based approach is less effective as it can only capture fewer semantically consistent topics without the aid of word embedding or other query expansion strategies. The use of heterogeneous information network (HIM) together with implicit feedbacks makes consistency possible across an extended number of N.

Given that the precision is slightly higher than the recall when Top-N is low and vice versa, the harmonic means is obtained to determine the effectiveness of SAMM and the baseline approaches. It can be observed that though SAMM has high precision (0.9167) but low recall (0.8327), it is still very accurate, but it misses an insignificant number of MWS that are difficult to classify. This is why the stability of F-measure as the Top-N increases. The MAP and NDCG values of the top 50 discovered MWS are measured under the specified parameter settings, and the results show reassuring improvement in the case of SAMM. Moreover, compared with the latest approaches, SAMM records different levels of improvement at different values of N, and the best improvements recorded are 6.69%, 5.37%, 8.18%, 6.32%, and 2.76%, on Precision@5, Recall@40, F1@35, MAP@10, and NDCG@10, respectively.

After a thoughtful analysis, it is safe to say that the best performance achieved by SAMM is mainly because of the service relevance learned using M-NSA which contributes to the separation of relevant MWS from a collection of MWS. The effectiveness of SAMM is owing to the additional consideration of variable-sized detector and the elimination of other unnecessary control parameters such as maximum age the detector (t) in the service relevance learning process contributes greatly to the higher performance of SAMM as well as the way it leverages more comprehensive information that covers the entire discovery space (service requester, service provider, and the environment) to achieve better performances.

In comparison with other one-class classification methods such as FM, SVM prior knowledge about the MWS data is not a prerequisite for SAMM and is barely stifle by discrete data; therefore, runtime adaption strategy is applicable. Moreover, the extensive experiment and analysis carried out to curtail the classification error improves the performance of SAMM. This is because on the one hand, too small self-radius and the insufficiency of self-samples to cover sufficient self-space result in a high false-positive error. On the other hand, too-large self-radius and the insufficiency of self-samples to cover nonself space result in a high false-negative error. Therefore, the balance stroked while selecting the self-radius to minimize the classification error.

5.8 Assumptions and limitations

The proposed self-adaptive mobile web service discovery approach based on a modified negative selection algorithm has some shortcomings. Currently, the proposed approach is trained and tested for steady data, and various steps would be taken in the face of the new MWS data depending on the amount of the new MWS. If the amount of the new MWS is minimal, the parameters of the SRL model will be retained; otherwise, the SRL model will be re-trained with different parameters. The estimation of the threshold that enables the self-adaptation depends on balancing the reliability of the algorithm and the effectiveness of the MWS discovery. This calls for further investigation.

The size of the ProgrammableWeb dataset is one of the constraints of this paper. Though it is the largest RESTful web service collection publicly accessible for evaluation, 13,520 MWS is not the desired number (large enough) for accurate training and testing of the model. On the other hand, transfer learning provides an alternative solution. However, it is affected by reproducibility issues, and catastrophic forgetting as the distribution of new testing data changes, the SLR model trained on a task of service relevance learning typically fails to use previously learned information. Another limitation of the proposed approach is that it is based on a negative selection algorithm, which uses a specific matching rule and a specific detector shape. The most widely used matching rule (Euclidean) and detector shape (hyperspherical) adopted for M-NSA depend on real-valued data representation. This limitation

is not restricted to the proposed approach. The binary data representation is an alternative, but its performance is worse in the MWS discovery domain.

The time complexity of the proposed approach is assumed to be less than that of the traditional negative selection algorithms which are (exponential). This is because the selected MNSA-1 selected for SRL has a smaller detector size, as well as the elimination of other unnecessary control parameters such as the maximum age of the detector (t). This decreases the complexity of time and increases the efficiency of producing the detectors. Moreover, the proposed approach aims to adapt the traditional NSA for service relevance learning and the discovery of the most relevant MWS rather than to introduce a new NSA. However, the time complexity analysis of the proposed approach will be looked at in the future to further confirm its effectiveness or otherwise.

6 Conclusion

This paper has presented an improved self-adaptive MWS discovery approach for DME based on M-NSA in which the detector generation stage is transformed into a service relevance learning by modeling the self-space with an appropriate self-radius r_s and a number of NWS and generated a dynamic number of detectors with variable sizes. The detection stage is transformed into self-adaptive matchmaking (SAMM) and fitted with a reinforced learning process to adapt itself at runtime in response to changes of self-space or when the discovery result is not as expected. The proposed algorithm was evaluated using the RESTful description of MWS (ProgrammableWeb dataset) to demonstrate the M-NSA's detector generation process, after appropriate selection and setting of various parameters, a high detection rate, and low false alarm rate is achieved. The experimental results showed the applicability and effectiveness of the proposed improved approaches in MWS discovery. The service relevance learned using M-NSA contributes to the separation of relevant MWS from a collection of MWS, which contributes to the effectiveness of SAMM. Moreover, additional consideration of variable-sized detector and the elimination of other unnecessary control parameters in the service relevance learning process contribute greatly to the higher performance of SAMM as well as the way it leverages more comprehensive information that covers the entire discovery space (service requester, service provider, and the environment) to achieve better performances. Although the proposed approach demonstrates improved performance and can be considered a success in the MWS discovery domain, a thorough investigation as well as a decent empirical study should be conducted for application in other related domains. It is noteworthy that the detector generation is a continuous process considering the high influx of MWS and the changes in DME; therefore, it is essential to explore alternatives to decrease the time and space complexities in the future. Strategies for effective management of the threshold value can be considered for further research, while integrating PSO in the detector generation stage can be an interesting idea for future works in the MWS discovery domain.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Bawazir A, Alhalabi W, Mohamed M, Sarirete A (2018) A formal approach for matching and ranking trustworthy context-dependent services. Appl Soft Comput J 73:306–315. https://doi.org/10.1016/j.asoc.2018.07.062
2. Elgazzar K, Hassanein H, Martin P (2014) Daas: cloud-based mobile web service discovery. Pervasive Mob Comput 13:67–84. https://doi.org/10.1016/j.pmcj.2013.10.015
3. Ruta M, Scioscia F, Di Sciascio E (2015) A mobile matchmaker for resource discovery in the ubiquitous semantic web. In: Proc—2015 IEEE 3rd Int Conf Mob Serv MS 2015 pp 336–343. https://doi.org/10.1109/MobServ.2015.76
4. Sivakumaran M, Iacopino P (2018) The mobile economy. GSMA Intell pp 11–11
5. Bobek S, Nalepa GJ (2017) Uncertain context data management in dynamic mobile environments. Future Gener Comput Syst 66:110–124. https://doi.org/10.1016/j.future.2016.06.007
6. Verma R, Srivastava A (2018) A dynamic web service registry framework for mobile environments. Peer-to-Peer Netw Appl 11:409–430. https://doi.org/10.1007/s12083-016-0540-6
7. Barakat L, Miles S, Luck M (2018) Adaptive composition in dynamic service environments. Future Gener Comput Syst 80:215–228. https://doi.org/10.1016/j.future.2016.12.003
8. Mezni H, Sellami M (2016) AWS-Ont: an ontology for the self-management of service-based systems. In: Proc—2015 IEEE 8th Int Conf Serv Comput Appl SOCA 2015 pp 85–92. https://doi.org/10.1109/SOCA.2015.17
9. Xiong R, Wang J, Zhang N, Ma Y (2018) Deep hybrid collaborative filtering for web service recommendation. Expert Syst Appl 110:191–205. https://doi.org/10.1016/j.eswa.2018.05.039
10. Zhang N, Wang J, Ma Y et al (2018) Web service discovery based on goal-oriented query expansion. J Syst Softw 142:73–91. https://doi.org/10.1016/j.jss.2018.04.046
11. Xie F, Wang J, Xiong R et al (2019) An integrated service recommendation approach for service-based system development. Expert Syst Appl 123:178–194. https://doi.org/10.1016/j.eswa.2019.01.025
12. Vargas-Santiago M, Morales-Rosales L, Pomares-Hernandez S, Drira K (2018) Autonomic web services enhanced by asynchronous checkpointing. IEEE Access 6:5538–5547. https://doi.org/10.1109/ACCESS.2017.2756867

13. Vinh PC (2016) Concurrency of self-⁎in autonomic systems. Future Gener Comput Syst 56:140–152. https://doi.org/10.1016/j.future.2015.04.017
14. Klusch M, Kapahnke P, Schulte S et al (2016) Semantic web service search: a brief survey. KI—Künstliche Intell 30:139–147. https://doi.org/10.1007/s13218-015-0415-7
15. Ghahramani Z (2015) Probabilistic machine learning and artificial intelligence. Nature 521:452–459. https://doi.org/10.1038/nature14541
16. Muda NA, Muda AK, Huoy CY (2018) Recognizing music features pattern using modified negative selection algorithm for songs genre classification. In: Abraham A, Muhuri PK, Muda AK, Gandhi N (eds) Advances in intelligent systems and computing. Springer International Publishing, Cham, pp 242–251
17. Forrest S, Perelson AS, Allen L, Cherukuri R (1994) Self-nonself discrimination in a computer. In: Proc 1994 IEEE Comput Soc Symp Res Secur Priv pp 202–212. https://doi.org/10.1109/RISP.1994.296580
18. Dong H, Hussain FK, Chang E (2013) Semantic Web Service matchmakers: state of the art and challenges. Concurr Comput Pract Exp 25:961–988. https://doi.org/10.1002/cpe.2886
19. Gmati FE, Chakhar S, Ayadi NY, et al (2018) Efficient versus accurate algorithms for computing a semantic logic-based similarity measure. In: International conference on industrial, engineering and other applications of applied intelligent systems. Springer, Cham, pp 808–820
20. Pal K (2018) Ontology-based web service architecture for retail supply chain management. Procedia Comput Sci 130:985–990. https://doi.org/10.1016/j.procs.2018.04.101
21. Sambasivam G, Amudhavel J, Vengattaraman T, Dhavachelvan P (2018) An QoS based multifaceted matchmaking framework for web services discovery. Future Comput Inform J 3:371–383. https://doi.org/10.1016/j.fcij.2018.10.007
22. Abid A, Messai N, Rouached M, et al (2017) Semantic similarity based web services composition framework. In: Proceedings of the symposium on applied computing—SAC '17. ACM, New York, NY, USA, pp 1319–1325
23. Corbellini A, Godoy D, Mateos C, et al (2017) Mining social web service repositories for social relationships to aid service discovery. In: IEEE international working conference on mining software repositories. IEEE Press, Piscataway, NJ, USA, pp 75–79
24. Stavropoulos TG, Andreadis S, Bassiliades N et al (2016) The tomaco hybrid matching framework for SAWSDL semantic web services. IEEE Trans Serv Comput 9:954–967. https://doi.org/10.1109/TSC.2015.2430328
25. Tripathy AK, Tripathy PK (2018) Fuzzy QoS requirement-aware dynamic service discovery and adaptation. Appl Soft Comput J 68:136–146. https://doi.org/10.1016/j.asoc.2018.03.038
26. Cheng B, Zhao S, Li C, Chen J (2017) A web services discovery approach based on mining underlying interface semantics. IEEE Trans Knowl Data Eng 29:950–962. https://doi.org/10.1109/TKDE.2016.2645769
27. Saadon NA, Mohamad R (2015) Semantic-based discovery framework for web services in mobile computing environment. J Teknol 77:25–38. https://doi.org/10.11113/jt.v77.6183
28. Cheng B, Li C, Zhao S, Chen J (2018) Semantics Mining & Indexing-based Rapid Web Services Discovery Framework. IEEE Trans Serv Comput 14:864–875. https://doi.org/10.1109/TSC.2018.2831678
29. Win NNH, Jianmin B, Gang C, Rehman SU (2019) Self-adaptive qos-aware web service discovery using ontology approach. In: Information Resources Management Association (IRMA) (ed) Web services: concepts, methodologies, tools, and applications. IGI Global, USA, pp 822–841. https://doi.org/10.4018/978-1-5225-7501-6
30. Athanasopoulos D (2017) Self-adaptive service organization for pragmatics-aware service discovery. In: Proceedings—2017 IEEE 14th international conference on services computing, SCC 2017. pp 164–171
31. Nabli H, Cherif S, Djmeaa R Ben, Amor IA Ben (2018) SADICO: self-adaptive approach to the web service composition. In: International conference on intelligent interactive multimedia systems and services. pp 254–267
32. Kafaf DAL, Kim DK (2017) A web service-based approach for developing self-adaptive systems. Comput Electr Eng 63:260–276. https://doi.org/10.1016/j.compeleceng.2017.06.030
33. Moreno GA, Cámara J, Garlan D, Klein M (2018) Uncertainty reduction in self-adaptive systems. In: 2018 IEEE/ACM 13th international symposium on software engineering for adaptive and self-managing systems (SEAMS). IEEE, pp 51–57
34. Paz A, Arboleda H (2016) A model to guide dynamic adaptation planning in self-adaptive systems. Electron Notes Theor Comput Sci 321:67–88
35. Di Nitto E, Ghezzi C, Metzger A et al (2008) A journey to highly dynamic, self-adaptive service-based applications. Autom Softw Eng 15:313–341. https://doi.org/10.1007/s10515-008-0032-x
36. He X, Liao L, Zhang H, et al (2017) Neural collaborative filtering. In: 26th Int World Wide Web Conf WWW 2017 pp 173–182. https://doi.org/10.1145/3038912.3052569
37. Cao B, Liu J, Wen Y et al (2019) QoS-aware service recommendation based on relational topic model and factorization machines for IoT Mashup applications. J Parallel Distrib Comput 132:177–189. https://doi.org/10.1016/j.jpdc.2018.04.002
38. Klusch M, Kapahnke P (2012) The iSeM matchmaker: a flexible approach for adaptive hybrid semantic service selection. J Web Semant 15:1–14. https://doi.org/10.1016/j.websem.2012.07.003
39. Ramdane C, Chikhi S (2017) Negative selection algorithm: recent improvements and its application in intrusion detection system. Int J Comput Acad Res 6:20–30
40. Liu Z, Li TAO, Yang JIN, Yang TAO (2017) An improved negative selection algorithm based on subspace density seeking. IEEE Access 5:12189–12198
41. Mohi-Aldeen SM, Mohamad R, Deris S (2016) Application of negative selection algorithm (NSA) for test data generation of path testing. Appl Soft Comput J 49:1118–1128. https://doi.org/10.1016/j.asoc.2016.09.044
42. Ji Z, Dasgupta D (2004) Real-valued negative selection algorithm with variable-sized detectors. Lect Notes Comput Sc 3102:287–298. https://doi.org/10.1007/978-3-540-24854-5_30
43. Fouladvand S, Osareh A, Shadgar B et al (2017) DENSA: an effective negative selection algorithm with flexible boundaries for self-space and dynamic number of detectors. Eng Appl Artif Intell 62:359–372. https://doi.org/10.1016/j.engappai.2016.08.014
44. Zeng J, Liu X, Li T et al (2009) A self-adaptive negative selection algorithm used for anomaly detection. Prog Nat Sci 19:261–266. https://doi.org/10.1016/j.pnsc.2008.06.008
45. Nanda SJ, Panda G (2014) A survey on nature inspired metaheuristic algorithms for partitional clustering. Swarm Evol Comput 16:1–18. https://doi.org/10.1016/j.swevo.2013.11.003
46. Dasgupta D, Yu S, Nino F (2011) Recent advances in artificial immune systems: Models and applications. Appl Soft Comput J 11:1574–1587. https://doi.org/10.1016/j.asoc.2010.08.024
47. Cui L, Pi D, Chen C (2015) BIORV-NSA: bidirectional inhibition optimization r-variable negative selection algorithm and its application. Appl Soft Comput J 32:544–552. https://doi.org/10.1016/j.asoc.2015.03.031
48. Zhu F, Chen W, Yang H et al (2017) A quick negative selection algorithm for one-class classification in big data era. Math Probl Eng. https://doi.org/10.1155/2017/3956415

49. Idris I, Selamat A, Thanh Nguyen N et al (2015) A combined negative selection algorithm-particle swarm optimization for an email spam detection system. Eng Appl Artif Intell 39:33–44. https://doi.org/10.1016/j.engappai.2014.11.001
50. Dong L, Liu S, Zhang H (2016) A boundary-fixed negative selection algorithm with online adaptive learning under small samples for anomaly detection. Eng Appl Artif Intell 50:93–105. https://doi.org/10.1016/j.engappai.2015.12.014
51. Dong L, Liu S, Zhang H (2017) A method of anomaly detection and fault diagnosis with online adaptive learning under small training samples. Pattern Recognit 64:374–385. https://doi.org/10.1016/j.patcog.2016.11.026
52. Zhao X, Wen Z, Li X (2014) QoS-aware web service selection with negative selection algorithm. Knowl Inf Syst 40:349–373. https://doi.org/10.1007/s10115-013-0642-x
53. Zhao X, Li R, Zuo X (2019) Advances on QoS-aware web service selection and composition with nature-inspired computing. CAAI Trans Intell Technol 4:159–174. https://doi.org/10.1049/trit.2019.0018
54. Garba S, Mohamad R, Saadon NA (2020) Search space reduction approach for self-adaptive web service discovery in dynamic mobile environment. In: Saeed F, Mohammed F, Gazem N (eds) Emerging trends in intelligent computing and informatics. Springer International Publishing, Cham, pp 1111–1121
55. Abid A, Khan MT, de Silva CW (2017) Layered and real-valued negative selection algorithm for fault detection. IEEE Syst J. https://doi.org/10.1109/JSYST.2017.2753851
56. Garba S, Mohamad R, Saadon NA (2020) Self-adaptive MWS matchmaker. GitHub Repos
57. Cao B, Frank Liu X, Liu J, Tang M (2017) Domain-aware Mashup service clustering based on LDA topic model from multiple data sources. Inf Softw Technol 90:40–54. https://doi.org/10.1016/j.infsof.2017.05.001
58. Xu Y, Goodacre R (2018) On splitting training and validation set: a comparative study of cross-validation, bootstrap and systematic sampling for estimating the generalization performance of supervised learning. J Anal Test 2:249–262. https://doi.org/10.1007/s41664-018-0068-2
59. Tian G, Zhao S, Wang J et al (2019) Semantic sparse service discovery using word embedding and Gaussian LDA. IEEE Access 7:88231–88242. https://doi.org/10.1109/ACCESS.2019.2926559