Efficiently Finding Web Services Using a Clustering Semantic Approach

Jiangang Ma, Yanchun Zhang
School of Computer Science & Mathematics,
Victoria University
Australia
{ma,yzhang}@csm.vu.edu.au

Jing He
Chinese Academy of Sciences, Research Centre on
Data Technology and Knowledge Economy
P.R.China
hejing@amss.ac.cn

ABSTRACT
Efficiently finding Web services on the Web is a challenging issue in service-oriented computing. Currently, UDDI is a standard for publishing and discovery of Web services, and UDDI registries also provide keyword searches for Web services. However, the search functionality is very simple and fails to account for relationships between Web services. Firstly, users are overwhelmed by the huge number of irrelevant returned services. Secondly, the intentions of users and the semantics in Web services are ignored. Inspired by the success of partitioning approach used in the database design, we used a novel clustering semantic algorithm to eliminate irrelevant services with respect to a query. Then we utilized Probabilistic Latent Semantic Analysis (PLSA), a machine learning method, to capture the semantics hidden behind the words in a query, and the descriptions in the services, so that service matching can be carried out at the concept level. This paper reports upon the preliminary experimental evaluation that shows improvements over recall and precision.

Categories and Subject Descriptors
H.4 [Informaiton Systems Application]: Miscellaneous

General Terms
Algorithms, Design

Keywords
Web service, Web services Matching, Machine Learning

1. INTRODUCTION
Web services have emerged as one of distributed computing technologies and sparked a new round of interest from industrial and research communities. As Web services adopt open standard interfaces and protocols, they are likely used as basic software building blocks in service-oriented applications, which are expected to play important role in a variety of application domains such as business application integration, business-to-business (B2B) and business information management. Meanwhile, inspired by the promise of applications presented by Web services, the research community has identified two major areas of interests: Web service discovery and Web service composition. In this paper, we address the issue of efficiently finding Web services on the Web.

CSSSIA 2008, April 22, Beijing, China

Web service discovery is normally defined as a matching process in which available services' capabilities can satisfy a service requester's requirements. The capability of a Web service is often implicitly indicated through a service's name, a method's name and some descriptions included in the service. And this capability can be described as an abstract interface by using standard Web services Description Language (WSDL). With the help of the standard descriptions of Web services, various approaches can be used to find services on the Web, such as using Web search engines [29, 30], service portals [24] and service registries like Universal Description, Discovery and Integration (UDDI) [20], etc. For example, UDDI allows syntactically keyword-based search and category-based browsing Web services. Thus, a service requester can utilize the keywords through the *Inquiry API* in UDDI for retrieving services via submitting the instruction such as *find_service ()*.

The keyword-based discovering mechanism supported by UDDI and most existing service search engines [29, 30], however, suffer from some key problems. Firstly, it is difficult for a user to obtain the desired services because the number of the retrieved services with respect to the keywords may be huge. One of the possible solutions to this problem is to compress data for reducing the size of services returned to service requesters. However, conventional techniques such as Singular Value Decomposition (SVD) [3, 4] and Support Vector Machines (SVM) may not be suitable for dealing with a large document collection due to the high cost of computing and storage of SVD.

Secondly, keywords are insufficient in expressing semantic concepts. This is partially due to the fact that keywords are often described by natural language, being much richer in terms of diversity. For example, syntactically different words may have similar semantics (synonyms) which results in low recall. In addition, semantically different concepts could possess identical representation (homonyms), leading to low precision. As a result, the retrieved services might be totally irrelevant to the need of their consumers. More recently, this issue sparked a new research into the Semantic Web where some research [9, 15, 17] uses ontology to annotate the elements in Web services. Nevertheless, integrating different ontologies may be difficult while the creation and maintenance of ontologies may involve a huge amount of human effort [12, 2, 11].

In order to address these problems, we present a novel approach for efficiently finding Web services on the Web. Given a query, we first filter out those Web services whose contents are not compatible with a user's query via a clustering algorithm to acquire an initial working dataset. As a next step, Probabilistic Latent Semantic Analysis approach (PLSA) [10] is applied to the working dataset, which is further clustered into a finite

number of semantically related groups. In this phase, we use a Probabilistic Latent Semantic Analysis approach (PLSA) to capture semantic concepts hidden behind the words in a query and the advertisements in services, so that services matching is expected to be implemented at an advanced concept level. We call our approach CPLSA. Broadly speaking, our method combines syntactic analysis with a clustering semantic approach which is based on the current dominating mechanisms of discovering and describing Web services with UDDI and WSDL.

Our key contributions are as follows:

1) A novel finding service approach through the combination of keyword technique and the semantics extracted from the services' descriptions.
2) Description of preliminary experiment to evaluate the effectiveness of our approach, and results show improvements over recall and precision.

The organization of this paper is as follows: First, we briefly discuss some of the research work related to locating Web services. In Section 3, a clustering probabilistic semantic matching approach is discussed. The detailed finding services, probabilistic model and matching algorithms are introduced in section 4. The preliminary experiment evaluation is presented in section 5. Finally, the conclusion and future work can be found in Section 6.

2. RELATED WORK

In this section we briefly discuss some of the research work related to locating Web services.

Although various approaches can be used to locate Web services on the Web, this research is focused on the service discovery problem using a clustering method. The clustering methodology is a technology that transforms a complex problem into a series of simpler ones, which can be handled more easily. Specifically, this technology re-organizes a set of data into different groups based on some standards of similarity. Clustering analysis has been often used in computer science, as in data mining, in information retrieval, and in pattern classification.

More recently, clustering approaches are used for discovering Web services [6, 1, 16]. Dong [6] puts forward a clustering approach to search Web services where the search consisted of two main stages. A service user first types keywords into a service search engine, looking for the corresponding services. Then, based on the initial Web services returned, the approach extracts semantic concepts from the natural language descriptions provided in the Web services. In particular, with the help of the co-occurrence of the terms appearing in the inputs and outputs, in the names of the operations and in the descriptions of Web services, the similarity search approach employs the agglomerative clustering algorithm for clustering these terms to the meaningful concepts. Through combination of the original keywords and the concepts extracted from the descriptions in the services, the similarity of two Web services can be compared at the concept level so that the proposed approach improves the precision and recall.

Arbramowicz [1] proposes an architecture for Web services filtering and clustering. The service filtering is based on the profiles representing users and application information, which are further described through Web Ontology Language for Services (OWL-S). In order to improve the effectiveness of the filtering process , a clustering analysis is applied to the filtering process by comparing services with related the clusters. The objectives of the proposed matchmaking process are to save execution time, and to improve the refinement of the stored data. Another similar approach [16] concentrates on Web service discovery with OWL-S and clustering technology, which consists of three main steps. The OWL-S is first combined with WSDL to represent service semantics before a clustering algorithm is used to group the collections of heterogeneous services together. Finally, a user query is matched against the clusters, in order to return the suitable services.

Other approach [5] focuses on service discovery based on a directory where Web services are clustered into the predefined hierarchical business categories. In this situation, the performance of reasonable service discovery relies on both service providers and service requesters having prior knowledge on the service organization schemes.

Our approach CPLSA has similarities to approaches [6, 1, 16] in that keywords are used to first retrieve Web services, and extract semantic concepts from the natural language descriptions in the Web services. However, our work differs from these works in several ways. Firstly, we eliminate irrelevant service via exploiting a clustering algorithm to diminish the size of services returned; this approach shows some potential applications like over mobile uses. Secondly, based on the characteristics of Web services with a very limited amount of information, we regard the extraction of semantic concepts from service description as a problem of dealing with missing data. Therefore, we utilize Probabilistic Latent Semantic Analysis (PLSA) [10], a machine learning method, to capture the semantic concept hidden behind the words in a query and the advertisements in services.

3. OVERVIEW OF CPLSA APPROACH

Our clustering semantic approach (CPLSA) is dependent on combination of the keyword technique and the semantics extracted from the services' descriptions. The objectives of CPLSA are to diminish the cost of computing a large dataset and to match services at the semantic concept level. To realize these goals, we first eliminate irrelevant Web services with respect to a query by using a modified clustering algorithm. After acquiring an initial service dataset, we use Probabilistic Latent Semantic Analysis to find a common semantic concept between Web services and query so that service matching against the query can be carried out at the concept level.

The CPLSA approach is based on the assumption that the efficiency of finding services can be improved if irrelevant data can be eliminated before the extracting semantics algorithm is implemented. In this paper, the analysis of the proposed approach focuses on the scenario of discovering public Web services on the Web environment, which consists of following main procedures. Given a query, the proposed approach first retrieves a set of samples of Web services from a source of Web services. As the samples returned may include irreverent services with respect to the query, we then filter out those Web services whose contents are not compatible to a user's query via using a clustering algorithm to obtain an initial working dataset. Next Probabilistic Latent Semantic Analysis approach (PLSA) is applied to the working dataset, which is further clustered into a finite number of semantically related groups. This phase focuses on capturing semantic concepts hidden behind the advertisements in services. Finally, the semantic similarity of a query and Web services is measured within the related semantic

cluster. Figure 1 illustrates the outline of the proposed clustering semantic probabilistic approach.

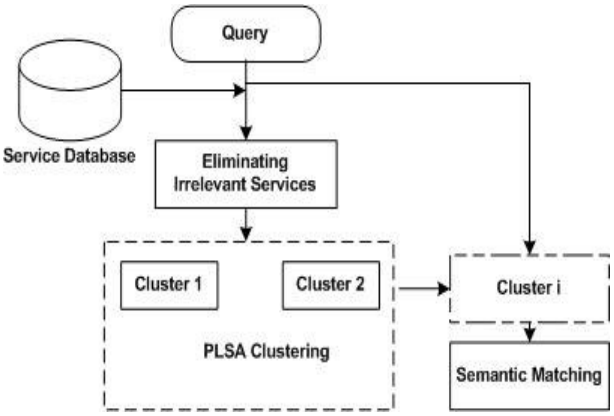

Figure 1. Outline of the matching approach

4. CPLSA: CLUSTERING PROBABILISTIC SEMANTIC APPRAOCH

In this section we propose our clustering probabilistic semantic approach (CPLSA) for efficiently finding Web services. As we noted earlier, the samples returned may include irreverent services with respect to a query, so we first filter out those Web services whose contents are not compatible to a user's query to form a working dataset. Then we apply PLSA to the working dataset for further clustering the dataset into a finite number of semantically related groups.

4.1 Eliminating Irrelevant Services from Service Collection

We first retrieve a set of samples of Web services from a source Web services. Given a query *q*, a source of services would return a set of services based on some kind of similarity. To calculate the similarity, we use the Vector Space Model (VSM) to represent Web services as points in syntactic space. Based on VSM, we can measure the similarity between a query *q* and a service *s* in the samples by computing the cosine of the angle between query vector *q* and service vector *s* as:

$$Sim(q,s) = \frac{|q \bullet s|}{\|q\|^2 \bullet \|s\|^2} \quad (1)$$

Using the above similarity computation, we can acquire an initial set of samples of services through selecting a predefined threshold.

Considering the possibility that the initial set of services may contain the services whose contents are not compatible with a user's query, we eliminate them accordingly from the sample set to improve the efficiency of service discovery, and also to reduce the cost of computation. Intuitively, these irrelevant data may have some negative impact on efficiently finding Web services; for one thing, the data may diminish the accuracy of the learning algorithms; for the other, they would increase the computational load. Therefore, as the first step towards efficiently locating Web services, these irrelevant services should be eliminated before the clustering semantic algorithm is implemented.

Several ways can be used to remove unrelated data from a dataset. One of the possible solutions is based on the feature selection, as indicated in [14]. This approach first sets a numerical threshold, and then computes the number of times a data object appears in a collection. If the number of times an object appearing in a collection is less than the predetermined threshold, the object is regarded as unrelated data and should be removed.

We use a different approach to eliminate the unrelated services from the dataset. The method consists of two main steps. Given a query, the initial sample set of services retrieved is first divided into different groups by using a clustering algorithm, each group gathering related services and including a cluster centre. On the next step, the distance between a data object and each centre of each cluster is computed. If the distance between a data object and every cluster's centre is higher than a predefined threshold *u*, the object is regarded to be irrelevant to query, and should be eliminated. We first formulate the problem of irrelevant service elimination as follows.

Definition 1. *Given w returned services* $S = \{s_1, s_2, ..., s_w\}$ *with respect to a query, cluster S to k groups* $C = \{c_1, c_2, ...c_k\}$ *and remove service* s_i *such that*

$$\|s_i - c_j\| \geq \varepsilon \quad , \; k < w \; , \quad j \in (1,2,...,k) \quad (2)$$

Where ε *is a predefined threshold and* c_j *is the centre of a cluster.*

◊

Specifically, a k-means algorithm is used to clean the initial sample set of services retrieved. With k-means algorithm, a service set *S* is divided into k clusters c_j, each including a centre denoted as:

$$cm_j = \frac{1}{|c_j|} \sum_{a_i \in c_j} a_i \quad (3)$$

Where $|c_j|$ is the number of data points in cluster c_j

Based on the Euclidean distance measure, the distance between a data point a_i and a cluster centre cm_j can be represented as:

$$dis(a_i, cm_j) = \|a_i - cm_j\|_2 = \sqrt{\sum_{d=1}^{n} (a_{i,d} - cm_{j,d})^2} \quad (4)$$

, and the following objective function is used to represent quality of a cluster:

$$O = \sum_{j=1}^{k} \sum_{a_i \in c_j} dis(a_i, cm_j)^2 \quad (5)$$

K-means algorithm continues until the objective function reaches minimum.

The algorithm of services elimination based on k-means is shown in Figure 2.

Algorithm 1: EliminatingIrrelevantService

```
1.  ServiceElimination (S, k, μ ) {
2.  Input: services_corpus(S); k: number_of_cluster
3.  μ : similarity _ threshold
4.  Output: a set of clean services MC;
5.  MC ← φ
6.  Assigning initial values to means cm1, cm2,..., cmk
7.  Begin
8.    for service s ∈ S do
9.       Finding s cluster centre cmi , assign s into the cluster
10.      Computer new centre
11.   end for
12.      /* service elimination */
13.   for each cluster ci ∈ C = {c1, c2,..., ck} do
14.       for each service s ∈ ci do
15.           if distance (s, ci • center) ≤ μ then do
16.               MC = MC ∪ s
17.           end if
18.       end for
19.   end for
20.   Return MC
21. End
```

Figure 2. Algorithm for eliminating irrelevant services

4.2 Constructing Service Transaction Matrix

As described in the previous section, an initial working dataset is obtained by eliminating irrelevant services from the sample set of services retrieved with a k-means algorithm. In this section, we further consider the relationship amongst services and construct a service matrix to be used as the input for our cluster-based algorithm introduced in the next section.

In traditional distributed databases, the relationship between the words in a dataset and service documents can be represented as a transaction matrix, where each column corresponds to a Web service document; each row represents a word (transaction). Meanwhile, the entries in the transaction matrix represent the frequency of occurrence of a word appearing in a service document. A service transaction matrix is shown in Table 1.

Table 1: An Example of Service Transaction Matrix

	Service1	Service2	Service 3	service 4
Transaction1	2	4	1	5
Transaction2	0	1	2	2
Transaction3	2	0	0	2
Transaction4	3	2	2	1

To construct such a matrix, we exploit the Vector Space Model (VSM) to describe each service document in the working dataset. In VSM, each document is described by bag of words or terms, which means the frequency of the words in a document is considered while the positional relationship between terms is ignored. In addition, all terms in a data collection form a vocabulary, which spans a high dimensional feature space in which documents are represented as a set of points. According to VSM, each document can be represented a vector as

$$\vec{a}_i = (w_{1,i}, w_{2,i}, ..., w_{v,i})$$

Where v is the size of the vocabulary

The values of entries $w_{i,j}$ in the document vector $\vec{a}_i$ can be determined through different schemes such as with counting the time of co-occurrence of a word appearing in a document.

In our case, the term-frequency and inverse-document-frequency (TF-IDF) [18] are used to denote the entries $w_{i,j}$ in a document vector $\vec{a}_i$. The weight w_{ij} is defined as the TF-IDF weight of the word j in documents i , denoted as following:

$$w_{ij} = tf_{ij} \bullet \log(\frac{n}{n_i}), \quad (6)$$

Where $tf_{ij} = \frac{n_{ij}}{|a_i|}$ denotes word frequency, that is, the number of times word j appears in service i, and n_i is the number of services that contain word j.

Using formula 1, we can denote the similarity between two documents by computing the cosine of the angle between the two document vectors.

Based on the above description, the service documents in the working dataset can be represented as a matrix:

$$A = [a_1, a_2, ..., a_n] \in R^{m \times n} \quad with \quad a_i \in R^m$$

In the above descriptions of service elimination and service matrix construction, the focus is put on analysing the syntactical correlation between the query and services at a basic level, aiming to improve the performance of service discovery. In the following sections, we shift our attention to the analysis of semantic concept.

4.3 Finding Services Based on PLSA

In this section, we discuss services matching at advanced concept level. We extend our previously reported work of discovering Web services based on PLSA [13, 23] with the introduction of a new algorithm after briefly introducing the basic principle of PLSA.

Our probabilistic semantic approach is based on the PLSA model that is called *aspect model* [10]. PLSA utilizes a Bayesian network to model an observed event of two random objects with a set of probabilistic distributions. In the text context, an observed event corresponds to occurrence of a word *w* occurring in a document. The model indirectly associates keywords to its corresponding documents by introducing an intermediate layer called hidden factor variable $Z = \{z_1, z_2, ..., z_k\}$. Based on the assumption that a document and a word are conditionally independent when the latent concept is given, the joint probability of an observed pair (d_i, w_j) obtained from the probabilistic model is shown as following:

$$P(d_i, w_j) = P(d_i)P(w_j \mid d_i), \quad (7)$$

Where

$$P(w_j \mid d_i) = \sum_{f=1}^{k} P(z_f \mid d_i)P(w_j \mid z_f) \quad (8)$$

From formula 8, the aspect model expresses dimensionality reduction by mapping a high dimensional term document matrix

into the lower dimensional one (k dimension) in latent semantic space.

The learned latent variables can be used to cluster Web services. As already mentioned, a Web service can be described as a multinomial probability distribution $P(z_f \mid d)$ over the latent variables $z_f \in Z = z_1, z_2, ..., z_k$. This representation of a service with these factors reflects the likelihood that the service belongs to certain concept groups. If a probability distribution over a specific factor z_f when given a Web service d_i is high, the Web service d_i can be clustered to the aspect z_f. This fact indicates that the PLSA model can function as a soft clustering approach that maps the observed object corresponding to natural concepts. In other words, if the objective of a service user, represented by a query, is closely associated to some Web services, the query and the services are expected to be mapped to some given factors with higher probability, compared to others with lower probability. In order to locate memberships that are associated with the latent factors, we can compute mixing coefficients:

Thus, for each hidden factor, we can compute $P(z_f \mid d_i)$ and get a maximised value for a specific Web service $d_i : Z_{\max}(d_i)$ that can be used as the class label for this service. In this way, all Web service documents are clustered to different categories in which all Web services indicate similar types.

The key to our approach is to cluster the services into a group of learned latent variables, which can be achieved by computing probability $P(latent - \text{var}iable \mid service)$ for each latent variable using formula 9. The rationale for this is that in the dimension-reduced semantic space, each Web service can be represented as a mixture of latent variables and the services with similar semantic concepts are projected to be close to each other. With the maximum value of the computation used for the class label for a service, we can categorize services into their corresponding group.

$$P(z \mid d_{new}) = \frac{P(d_{new} \mid z)P(z)}{P(d_{new})} = \frac{P(d_{new} \mid z)P(z)}{\sum_{z_f \in Z} P(d_{new} \mid z_f)} \quad (9)$$

The outline of clustering services based-on PLSA is shown as following:

Input: service matrix
Output: k service communities
Step1: choosing a service, compute probabilistic with respect to each hidden variable using formula (9)
Step2: find the maximum value of the probability for the Service
Step3: put the service to its corresponding to group and select next service

As a query may be outside the model, we use Expectation Maximization [10] algorithm to fold the query in the model. Finally, we use the following formula for computing the similarity.

$$sim_{PLSA}(d_i, q) = \frac{\sum_{z_f \in Z} P(z_f \mid q)P(z_f \mid d_i)}{\sqrt{\sum_{z_f \in Z} P(z_f \mid q)^2} \sqrt{\sum_{z_f \in Z} P(z_f \mid d_i)^2}} \quad (10)$$

Algorithm 2: SimilarityMatching

```
1.  SimilarityMatching (SM, q, μ_s ) {
2.  Input: services matrix(SM); q: query; μ_s : similarity _threshold
3.  Output: Matched Services, MC;
4.  MC ← φ
5.  Begin
6.  SC ← CategorizingServices(SM, K)
7.    /* add new query to model */
8.   P (z | q) ← fold _in _query()
9.   sc_m ← find_ matched_ category for query
10. for  each service sm_i in MatchedCategory sc_m do
11.        S_sm_i _QoS = calculate _S_sm_i _QoS
12.        /*compute similarity using formula (10)*/
13.        sm_i _score = calculate _sim_PLSA(sm_i, q)
14.        sm_i _FinalScore = S_sm_i_QoS + sm_i _score
15.        if sm_i _FinalScore > μ_s then do
16.            MC ← MC.append ( sm_i )
17.        end if
18.   end for
19.   return MC
20.  End
```

Figure 3. Algorithm for semantic similarity matching

4.4 Evaluating Quality of Cluster

In this research, entropy and purity are used to evaluate a cluster's quality. Suppose m classes represent partitioned services (service categories) and k clusters produced by our clustering algorithm, then the following definitions [14] apply:

For a cluster c_j, its entropy is defined as:

$$E(c_j) = -\sum_{i=1}^{m} \frac{n_j^i}{n_j} \bullet \log\left(\frac{n_j^i}{n_j}\right), \quad (11)$$

Where $n_j = |c_j|$, representing the size of cluster c_j, and n_j^i indicates the number of services in cluster c_j that belongs to class *i*.

Entropy expresses a cluster's consistence. If the members of a cluster come from different classes, the value of the entropy is high.

The purity of a cluster c_j is defined as:

$$P(c_j) = \frac{1}{n_j} \sum_{j=1}^{k} \max_i \{n_j^i\} \quad (12)$$

Where *i* varies over all classes.
Purity indicates the classification accuracy.

4.5 Summary of Our Approach

Our clustering semantic approach (CPLSA) uses a dynamic algorithm that partitions a service working dataset into smaller pieces. It includes the two main phases: eliminating irrelevant services and matching services at semantic concept level. The irrelevant services are first removed from the initial samples of services to form a working dataset. Note that at this stage, no semantic similarity is involved because the main objectives are to reduce the initial size of service collection and also to diminish the cost of calculating a large data set. Once the irrelevant services are eliminated, a Probabilistic Latent Semantic Analysis approach is applied to the working dataset for capturing semantic concepts. As a result, Web services are clustered into a finite number of semantically related groups. Based on the clustered service groups, a set of matched services can be returned by comparing the similarity between the query and related group, rather than computing the similarity between query and each service in the dataset. If the service results returned are not compatible to the user's query, the second best cluster would be chosen and the computing proceeds to the next iteration. The pseudo code for CPLSA algorithm is given as following.

Algorithm 3: SummaryOfClusteringAlgorithm

1. Retrieving initial samples of services
2. Eliminating irrelevant services to form a working dataset
3. Applying PLSA to the dataset
4. Semantic matching query with services in related clustered group
5. **if** the results match the query **then** goto step 8
6. **else** choosing next cluster goto step 4
7. **end if**
8. **end**

Figure 4. Summary of CPLSA algorithm

5. PRELIMINARY EVALUATION

In this section we present our preliminary experiments to evaluate the effectiveness of our clustering semantic approach CPLSA. We first describe the experimental dataset and the evaluation metric, and then present the experimental results.

5.1 Experimental Dataset

Our preliminary experiments were implemented over the real dataset of Web services whose WSDL files can be accessed via [27]. The collection of services includes 424 Web service descriptions covering the 25 categorises such as Zip code finder and weather information, etc. We selected the dataset of Web services for several reasons. Firstly, up to now, there are no extensive datasets of real services available. Secondly, the Web services in the collection are gathered from real-world service sites like SALCentral and XMethods [24], and artificially classified into different categories so that these Web services provide a basis on which testing and comparison can be implemented based on a variety of situations.

For the experimental comparison, we particularly choose four categories: Business, Communications, Converter and Money.

5.2 Data Processing

The goal of the data processing is to transform raw Web service information into an appropriate data format suitable for model learning. We extracted keywords from service description, names of operation, etc., and applied commonly used approaches for word processing. One of methods for data processing included word stemming and stopwords removing. The former removes common term suffix while the latter eliminates very frequently used words. In our experiment, we used the Porter stemmer to parse the 320 Web services documents. All these processes were expected to improve the performance of matching Web services. An example of service is shown as following:

Converts between different currencies in the Euro zone and the Euro.

After extracting the keywords, we obtain a service collection consisting of 320 services which are divided into two data sets: training data and test data.

5.3 Performance Measure

In order to evaluate the effectiveness of the proposed approach, we use the standard accuracy, precision and recall to measure overall performance. After training the model with PLSA, a set of hidden semantic variables, each of which indicates a service category is given. With the learned hidden semantic features, a query's or a new service's related category by computing the probability using formula 9 was determined. To evaluate the performance of the clustering algorithm, the accuracy is defined as follows:

$$\text{Accuracy} = \frac{number\_of\_identified\_serivces\_j}{number\_of\_service\_in\_Category\_j} \quad (13)$$

The two measurements the standard recall and precision indicate how relevant and appropriated a retrieved service is to a service user's needs. The recall of our approach is defined as

$$\text{Recall} = \frac{B}{A} \quad (14)$$

Where A is the total number of relevant services in the service collection and B is the number of relevant services retrieved.

The precision of the approach is defined as

$$\text{Precision} = \frac{B}{C} \quad (15)$$

Where C is the total number of services retrieved and B is the same numerator in the formula 14.

5.4 Results

In this experiment, we evaluated the performance using our semantic matching service approach (CPLSA). We first trained the probabilistic semantic model through setting the different numbers of latent semantic variables ranging from 2 to 20, in order to observe the performance of the retrieval. As data used in this work is artificially classified into different categories, the results were observed in various service categories. In particular, we calculate and compare the recall and precision by selecting four hidden semantic variables, which represent four service categories: Business, Communications, Converter and Money.

In addition, CPLSA performance was investigated by comparing our probabilistic semantic approach with a keyword approach.

The first experiment was implemented over the dataset to observe the performance of probability model learning. Table 2 lists the 10 extracted latent aspects and their corresponding categories. An example of the likely words for four hidden semantic concept is shown in Table 3.

Table 2. Aspects and their service categories

Aspect	Service categories
1	Business
2	Web
3	News
4	Money
5	Developers
6	Finder
6	Converter
8	Games
9	Mathematics
10	Country

To see how effective our clustering semantic approach described in the previous section is, we compared the performance of the CPLSA approach with the keyword-based approach. In this experiment, we first fixed the parameter for the experiment, e.g., setting clusters' number to 4. The comparison was made on the top n=10 results retuned by each method, and the experiments were repeated 10 times. The result of the comparison on the accuracy is shown in the Figure 5. As can be seen from this figure, the performance of keyword-based technique only based on the text description in Web services is poor. However, the accuracy of service retrieval is improved when CPLSA approach is introduced.

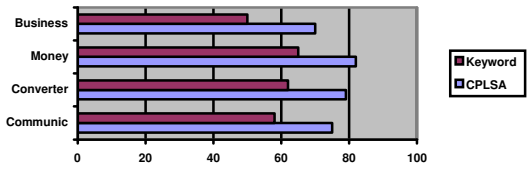

Figure 5. Accuracy of CPLSA and keyword for four categories

The next experiment implemented was to inspect the recall and precision by comparing CPLSA and keyword-based approach. The results are illustrated in Figure 6. From the Figure 6, we can see that CPLSA performed better than keyword-based approach in the selected four categories. For example, for the query containing "conversion", keyword-based approach may retrieve only those service that include the word conversion, but PLSA-based approach can get services including words "conversion" and "translation".

Table 3. examples of the most likely words for 4 hidden concept

	P(word | aspect)	most likely words
aspect 1	0.0835	information
	0.0683	Address
	0.0612	Telephone
	0.0607	Service
	0.0531	source
aspect 2	0.1368	Translate
	0.0879	Convert
	0.0684	State
	0.0586	English
	0.0489	system
aspect 3	0.1093	address
	0.1054	Zip
	0.0828	Place
	0.0753	Name
	0.0702	code
aspect 4	0.2052	Service
	0.1758	Web
	0.0892	Fax
	0.0803	Address
	0.0708	Italian

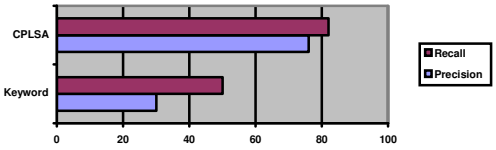

Figure 6. Precision and recall of PLSA and keyword for four categories

6. CONCLUSION AND FUTURE WORK

It is a challenging task to effectively find the desired Web services that conceptually match user's needs. In this paper, we studied two main problems introduced by the keyword-based search approach: lacking semantics and high cost of computation. To overcome the problems, we proposed a Clustering Probabilistic Semantic Approach (CPLSA). Based on the assumption that the efficiency of finding services can be improved if irrelevant data is eliminated, we applied a k-means approach to eliminate irrelevant services. After removing irrelevant services with respect to a query, the PLSA technique is applied to the service dataset so that service matching against the query can be carried out at the concept level. We also performed several experiments to evaluate the effectiveness of the proposed approach. Overall, the results show that our approaches improve over recall and precision.

7. REFERENCE

[1] W. Abramowicz, K. Haniewicz, M. Kaczmarek and D. Zyskowski. Architecture for Web services filtering and clustering. In *Internet and Web Applications and Services, (ICIW '07),*2007.

[2] C. Atkinson, P. Bostan, O. Hummel and D. Stoll. A Practical Approach to Web service Discovery and Retrieval. In *2007 IEEE International Conference on Web services (ICWS 2007),*2007

[3] M. W. Berry, S. A. Pulatova and G. W. Stewart. Computing Sparse Reduced-Rank Approximations to Sparse Matrices. In *ACM Transactions on Mathematical Software, Vol. 31, No. 2, Pages 252–269*, 2005.

[4] J. Baliński and C. Daniłowicz. Re-ranking Method based on Inter-document Distances. In *Journal of the Information Processing and Management.. V. 41 , Issue 4,* 2005.

[5] I. Constantinescu, W. Binder and B. Faltings. Flexible and efficient matchmaking and ranking in service directories. In *Proceedings of the IEEE International Conference on Web Services(ICWS'05),* 2005.

[6] X. Dong, A. Halevy, J. Madhavan, E. Nemes and J. Zhang. Similarity Search for Web services. In *Proceedings of the 30th VLDB Conference, Toronto, Canada,* 2004.

[7] J. T. Giles, L. Wo and M.W. Berry. GTP(General Text Parser) software for text mining. In *Statistical Data Mining and Knowledge Discovery, H. Bozdogan, ed., CRC Press, Boca Raton, FL, 2003, papers: 455-471*. 2003

[8] J. Garofalakis, Y. Panagis, E. Sakkopoulo and A. Tsakalidis. Web service Discovery Mechanisms: Looking for a Needle in a Haystack? In *International Workshop on We b Engineering, August 10*, 2004.

[9] A. Hess and N. Kushmerick. Learning to Attach Semantic Metadata to Web services. In *2nd International Semantic Web Conference (ISWC2003), Sanibel Island, Florida, USA*, 2003

[10] T. Hofmann. Probabilistic Latent Semantic Analysis. In *Proceedings of the 22nd Annual ACM Conference on Research and Development in Information Retrieval Berkeley, California, pages: 50-57, ACM Press, August*, 1999.

[11] H. Lausen and T. Haselwanter. Finding Web services. In *the 1st European Semantic Technology Conference, Vienna, Austria,2007*

[12] M. Klein and A. Bernstein. Toward High-Precision Service Retrieval. In *IEEE Internet Computing, Volume: 8, No. 1, Jan. – Feb. pages: 30 – 36*, 2004.

[13] J. Ma, J. Cao and Y. Zhang. A Probabilistic Semantic Approach for Discovering Web services. *InThe 16th International World Wide Web Conference (WWW2007). Banff, Alberta, Canada, May 8 -12,* 2007.

[14] B. Mandhani, S. Joshi and K. Kummamuru. A Matrix Density Based Algorithm to Hierarchically Co-Cluster Documents and Words. In *the 12th International World Wide Web Conference (WWW2003). May 20- 24, Budapest, Hungary,* 2003.

[15] M. Paolucci, T. Kawamura, T. Payne and K. Sycara. Semantic Matching of Web services Capabilities. In *Proceedings of the 1st International Semantic Web Conference (ISWC2002).* 2002.

[16] R. Nayak and B. Lee. Web service Discovery with Additional Semantics and Clustering. In *Web Intelligence, IEEE/WIC/ACM International Conference*, 2007

[17] K. Sivashanmugam, K. Verma, A.P and J.A. Miller. Adding Semantics to Web services Standards. In *Proceedings of the International Conference on Web services ICWS'03 , pages: 395-401*, 2003.

[18] G. Salton. Automatic Text Processing— The Transformation, Analysis, and Retrieval of Information by Computer. In *Published by Addison-Wesley Publishing Company*. 1988.

[19] A. Sajjanhar, J. Hou and Y. Zhang. Algorithm for Web services Matching. In *Proceedings of the 6th Asia- Pacific Web Conference, APWeb 2004, Hangzhou, China, April 14-17,* 2004, Lecture Notes in Computer Science 3007 spr9nger 2004.

[20] UDDI Version 2.03 Data Structure Reference UDDI Committee Specification, 19 July 2002,

[21] Y. Wang and E. Stroulia. Semantic Structure Matching for Assessing Web service Similarity. In *the First International Conference on Service Oriented Computing , Trento, Italy, December 15-18,* 2003.

[22] G. Xu, Y. Zhang, J. Ma and X. Zhou. Discovering User Access Pattern Based on Probabilistic Latent Factor Model. In *Proceedings of the 16th Australasian Database Conference – Volume: 39 pages: 27 – 35, Newcastle, Australia,* 2005.

[23] Y. Zhang and J. Ma. Discovering Web services based on Probabilistic Latent Factor Model. In *the Joint Conference of the 9th Asia-Pacific Web Conference and the 8th International Conference on Web-Age Information Management APWeb/WAIM'07 ,Huang Shan , China*, 2007

[24] XMethods. http://www.xmethods.com/

[25] http://www.census.gov/epcd/www/naics.html.

[26] http://www.Webservicelist.com

[27] http://www.andreashess.info/projects/annotator/ws2003.html

[28] Binding point. http://www.bindingpoint.com.

[29] Google: http://www.google.com.

[30] Yahoo: http://www.yahoo.com