Probabilistic Topic Models for Web Services Clustering and Discovery

Mustapha Aznag[1], Mohamed Quafafou[1], El Mehdi Rochd[1] and Zahi Jarir[2]

[1] Aix-Marseille University, LSIS UMR 7296, France
{mustapha.aznag,mohamed.quafafou,el-mehdi.rochd}@univ-amu.fr
[2] University of Cadi Ayyad, LISI Laboratory, FSSM, Morocco
jarir@uca.ma

Abstract. In Information Retrieval the Probabilistic Topic Models were originally developed and utilized for topic extraction and document modeling. In this paper, we explore several probabilistic topic models: Probabilistic Latent Semantic Analysis (PLSA), Latent Dirichlet Allocation (LDA) and Correlated Topic Model (CTM) to extract latent factors from web service descriptions. These extracted latent factors are then used to group the services into clusters. In our approach, topic models are used as efficient dimension reduction techniques, which are able to capture semantic relationships between word-topic and topic-service interpreted in terms of probability distributions. To address the limitation of keywords-based queries, we represent web service description as a vector space and we introduce a new approach for discovering web services using latent factors. In our experiment, we compared the accuracy of the three probabilistic clustering algorithms (PLSA, LDA and CTM) with that of a classical clustering algorithm. We evaluated also our service discovery approach by calculating the precision (P@n) and normalized discounted cumulative gain (NDCGn). The results show that both approaches based on CTM and LDA perform better than other search methods.

Keywords: Web service, Data Representation, Clustering, Discovery, Machine Learning, Topic Models

1 Introduction

The Service Oriented Architecture (SOA) is a model currently used to provide services on the internet. The SOA follows the find-bind-execute paradigm in which service providers register their services in public or private registries, which clients use to locate web services. SOA services have self-describing interfaces in platform-independent XML documents. Web Services Description Language (WSDL) is the standard language used to describe services. Web services communicate with messages formally defined via XML Schema. Different tasks like matching, ranking, discovery and composition have been intensively studied to improve the general web services management process. Thus, the web services community has proposed different approaches and methods to deal with these

tasks. Empirical evaluations are generally proposed considering different simulation scenarios. Nowadays, we are moving from web of data to web of services as the number of UDDI Business Registries (URBs) is increasing. Moreover, the number of hosts that offer available web services is also increasing significantly. Consequently, discovering services which can match with the user query is becoming a challenging and an important task. The keyword-based discovery mechanism supported by the most existing services search engines suffers from some key problems: (1) User finds difficulties to select a desired service which satisfies his requirements as the number of retrieved services is huge. (2) Keywords are insufficient in expressing semantic concepts. This is due to the fact that the functional requirements (keywords) are often described by natural language. To enrich web service description, several Semantic Web methods and tools are developed, for instance, the authors of [9, 19, 1] use ontology to annotate the elements in web services. Nevertheless, the creation and maintenance of ontologies may be difficult and involve a huge amount of human effort [2, 12].

To address the limitation of keywords-based queries, we represent web service description as a vector and introduce a new approach for discovering web services using a semantic clustering approach. Service Clustering aims to group together services which are similar to each other. Our clustering approach is based on probabilistic topic models. By organizing the web service data into clusters, services become easier and therefore faster to be discovered and recommended [17].

Probabilistic topic models are a way to deal with large volumes of data by discovering their hidden thematic structure. Their added value is that they can treat the textual data that have not been manually categorized by humans. The concept of "topic" consists on discovering clusters of textual data on similar subjects. These clusters are obtained by calculating the occurrences of words emerging together frequently in different independent texts. Formally, probabilistic topic models use their hidden variables to discover the latent semantic structure in large textual data.

In this paper we investigate using probabilistic machine-learning methods to extract latent factors $z_f \in Z = \{z_1, z_2, ..., z_k\}$ from service descriptions. We will explore several probabilistic topic models : PLSA (Probabilistic latent semantic analysis), LDA (Latent Dirichlet Allocation) and CTM (Correlated Topic Model) and use them to analyze search in repository of web services and define which achieves the best results. By describing the services in terms of latent factors, the dimensionality of the system is reduced considerably. The latent factors can then also be used to efficiently cluster services in a repository. In our experiments, we consider that web services are mixtures of hidden topics, where a topic defines a probability distribution over words.

The rest of this paper is organized as follows. Section 2 provides an overview of related work. In Section 3 we describe in detail our service clustering and discovery approach. Section 4 describes the experimental evaluation. Finally, the conclusion and future work can be found in Section 5.

2 Related work

In this section, we briefly discuss some of research works related to discovering Web services. Although various approaches can be used to locate and discover Web services on the web, we have focused our research on the service discovery problem using a clustering method. The clustering methodology is a technique that transforms a complex problem into a series of simpler ones, which can be handled more easily. Specifically, this technique 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.

In [1], the authors proposed an architecture for Web services filtering and clustering. The service filtering mechanism is based on user and application profiles that are described using OWL-S (Web Ontology Language for Services). The objectives of this matchmaking process are to save execution time and to improve the refinement of the stored data. Another similar approach [15] concentrates on Web service discovery with OWL-S and clustering technology. Nevertheless, the creation and maintenance of ontologies may be difficult and involve a huge amount of human effort [2, 12].

Generally, every web service associates with a WSDL document that contains the description of the service. A lot of research efforts have been devoted in utilizing WSDL documents [8, 2, 12, 13, 7, 14, 17]. Dong et al. [8] proposed the Web services search engine Woogle that is capable of providing Web services similarity search. However, their engine does not adequately consider data types, which usually reveal important information about the functionalities of Web services [11]. Liu and Wong [13] apply text mining techniques to extract features such as service content, context, host name, and service name, from Web service description files in order to cluster Web services. They proposed an integrated feature mining and clustering approach for Web services as a predecessor to discovery, hoping to help in building a search engine to crawl and cluster non-semantic Web services. Elgazzar et al. [7] proposed a similar approach which clusters WSDL documents to improve the non-semantic web service discovery. They take the elements in WSDL documents as their feature, and cluster web services into functionality based clusters. The clustering results can be used to improve the quality of web service search results.

Some researchers use the proximity measures to cluster web services. Measuring the proximity between a service and other services in a dataset is the basic step of most clustering techniques [15, 17]. If two vectors are closed to each other in vector space, then they have similar service descriptions or functional attributes depending on characteristics used for constructing the model. Various techniques exist to measure the proximity of two vectors. Nayak et al. [15] proposed a method to improve the Web service discovery process using the Jaccard coefficient to calculate the similarity between Web services. Multidimensional Angle is an efficient measure of the proximity of two vectors. It is used in various clustering approaches [17]. This proximity measure applies cosine of the

angle between two vectors. It reaches from the origin rather than the distance between the absolute position of the two points in vector space.

Ma et al. [14] proposed an approach similar to the previously discussed approaches [8, 1, 15] where the keywords are used first to retrieve Web services, and then to extract semantic concepts from the natural language descriptions in Web services. Ma et al. presented a service discovery mechanism called CPLSA which uses Probabilistic Latent Semantic Analysis (PLSA) to extract latent factors from WSDL service descriptions after the search is narrowed down to a small cluster using a K-Means algorithm. The PLSA model represents a significant step towards probabilistic modelling of text, it is incomplete in that it provides no probabilistic model at the level of documents [3]. The Latent Dirichlet Allocation (LDA) [3] is an attempt to improve the PLSA by introducing a Dirichlet prior on document-topic distribution.

Cassar et al. [5, 6] investigated the use of probabilistic machine-learning techniques (PLSA and LDA) to extract latent factors from semantically enriched service descriptions. These latent factors provide a model which represents any type of service's descriptions in a vector form. In their approach, the authors assumed all service descriptions were written in the OWL-S. In [5], Cassar et al. showed how latent factors extracted from service descriptions can be used directly to cluster services in a repository; obtaining a more efficient clustering strategy than the one obtained by a K-Means algorithm. The results obtained from comparing the two methods (PLSA and LDA) showed that the LDA model provides a scalable and interoperable solution for automated service discovery in large service repositories. The LDA model assumes that the words of each document arise from a mixture of topics, each of which is a distribution over the vocabulary. A limitation of LDA is the inability to model topic correlation [4]. This limitation stems from the use of the Dirichlet distribution to model the variability among the topic proportions.

The Correlated Topic Model (CTM) has been developed to address the limitation of LDA [4]. In CTM, topic proportions exhibit correlation via the logistic normal distribution. One key difference between LDA and CTM is the independence assumption between topics in LDA, due to the Dirichlet prior on the distribution of topics (under a Dirichlet prior, the components of the distribution are independent whereas the logistic normal models correlation between the components through the covariance matrix of the normal distribution). However, in the CTM model, a topic may be consistent with the presence of other topics. In this paper, we exploit the advantages of CTM to propose an approach for web service discovery and use a novel semantic clustering algorithm to cluster web services. In our approach, we utilized CTM to capture the semantics hidden behind the words in a query, and the descriptions of the services. Then, we extracted latent factors from web service descriptions. The latent factors can then be used to efficiently cluster services in a repository.

3 Web Service Clustering and Discovery Approach

In this section, we will first describe the necessary pre-processing of WSDL document to construct a web service representation. We then discuss the probabilistic machine-learning techniques used to generate the latent factors. Finally we explain how these latent factors are used to provide an efficient clustering and discovery mechanism.

3.1 Web Service Representation

Generally, every web service has a WSDL (Web Service Description Language) document that contains the description of the service. The WSDL document is an XML-based language, designed according to standards specified by the W3C, that provides a model for describing web services. It describes one or more services as collections of network endpoints, or ports. It provides the specifications necessary to use the web service by describing the communication protocol, the message format required to communicate with the service, the operations that the client can invoke and the service location. Two versions of WSDL recommendation exist: the 1.1[3] version, which is used in almost all existing systems, and the 2.0[4] version which is intended to replace 1.1. These two versions are functionally quite similar but have substantial differences in XML structure.

To manage efficiently web service descriptions, we extract all features that describe a web service from the WSDL document. We recognize both WSDL versions (1.1 and 2.0). During this process, we proceed in two steps. The first step consists of checking availability of web service and validating the content of WSDL document. The second step is to get the WSDL document and read it directly from the WSDL URI to extract all information of the document.

Before representing web services as TF-IDF (Text Frequency and Inverse Frequency) [18] vectors, we need some preprocessing. There are commonly several steps:

- *Features extraction* extracts all features that describe a web service from the WSDL document, such as service name and documentation, messages, types and operations.
- *Tokenization:* Some terms are composed by several words, which is a combination of simple terms (*e.g., get_ComedyFilm_MaxPrice_Quality*). We use therefore regular expression to extract these simple terms (*e.g., get, Comedy, Film, Max, Price, Quality*).
- *Tag and stop words removal*: This step removes all HTML tags, CSS components, symbols (punctuation, etc.) and stop words, such as 'a', 'what', etc. The Standford POS Tagger[5] is then used to eliminate all the tags and stop words and only words tagged as nouns, verbs and adjectives are retained. We also remove the WSDL specific stopwords, such as *host, url, http, ftp, soap, type, binding, endpoint, get, set, request, response*, etc.

[3] `http://www.w3.org/TR/wsdl`
[4] `http://www.w3.org/TR/wsdl20/`
[5] `http://nlp.stanford.edu/software/tagger.shtml`

- *Word stemming*: We need to stem the words to their origins, which means that we only consider the root form of words. In this step we use the Porter Stemmer [16] to remove words which have the same stem. Words with the same stem will usually have the same meaning. For example, 'computer', 'computing' and 'compute' have the stem 'comput'. The Stemming process is more effective to identify the correlation between web services by representing them using these common stems (root forms).
- *Service Matrix construction*: After identifying all the functional terms, we calculate the frequency of these terms for all web services. We use the Vector Space Model (VSM) technique to represent each web service as a vector of these terms. In fact, it converts service description to vector form in order to facilitate the computational analysis of data. In information retrieval, VSM is identified as the most widely used representation for documents and is a very useful method for analyzing service descriptions. The TF-IDF algorithm [18] is used to represent a dataset of WSDL documents and convert it to VSM form. We use this technique, to represent a service description in the form of *Service Matrix*. In the service matrix, each row represents a WSDL service description, each column represents a word from the whole text corpus (vocabulary) and each entry represents the TF-IDF weight of a word appearing in a WSDL document. TF-IDF gives a weight w_{ij} to every term j in a service description i using the equation: $w_{ij} = tf_{ij}.\log(\frac{n}{n_j})$. Where tf_{ij} is the frequency of term j in WSDL document i, n is the total number of WSDL documents in the dataset, and n_j is the number of services that contain term j.

3.2 A Probabilistic Topic Model Approach

In our approach, we apply probabilistic machine-learning techniques; Probabilistic Latent Semantic Analysis (PLSA), Latent Dirichlet Allocation (LDA) and Correlated Topic Model (CTM); to extract latent factors $z_f \in Z = \{z_1, z_2, ..., z_k\}$ from web service descriptions (i.e., *Service Matrix*). We use then the extracted latent-factors to group the services into clusters. In our work, topic models are used as efficient dimension reduction techniques, which are able to capture semantic relationships between *word-topic* and *topic-service* interpreted in terms of probability distributions. In our context, an observed event corresponds to occurrence of a word w in a service description s.

The Probabilistic Latent Semantic Analysis (PLSA) is a generative statistical model for analyzing co-occurrence of data. PLSA is based on the aspect model [10]. Considering observations in the form of co-occurrences (s_i, w_j) of words and services, PLSA models the joint probability of an observed pair $P(s_i, w_j)$ obtained from the probabilistic model is shown as follows [10]:

$$P(s_i, w_j) = \sum_{f=1}^{k} P(z_f)P(s_i|z_f)P(w_j|z_f) \tag{1}$$

We assume that service descriptions and words are conditionally independent given the latent factor. We have implemented the PLSA model using the Pen-

nAspect[6] model which uses maximum likelihood to compute the parameters. The dataset was divided into two equal segments which are then transformed into the specific format required by the PennAspect. We use words extracted from service descriptions and create a PLSA model. Once the latent variables $z_f \in Z = \{z_1, z_2, ..., z_k\}$ are identified, services can be described as a multinomial probability distribution $P(z_f|s_i)$ where s_i is the description of the service i.The representation of a service with these latent variables reflects the likelihood that the service belongs to certain concept groups [14]. To construct a PLSA model, we first consider the joint probability of an observed pair $P(s_i, w_j)$ (Equation 1). The parameters $P(z)$, $P(s|z)$ and $P(w|z)$ can be found using a model fitting technique such as the Expectation Maximization (EM) algorithm [10]. The learned latent variables can be used to cluster web services. If a probability distribution over a specific z_f when given a web service s is high, then the service s can be affected to the cluster z_f.

The Latent Dirichlet Allocation (LDA) is a probabilistic topic model, which uses a generative probabilistic model for collections of discrete data [3]. LDA is an attempt to improve the PLSA by introducing a Dirichlet prior on service-topic distribution. As a conjugate prior for multinomial distributions, Dirichlet prior simplifies the problem of statistical inference. The principle of LDA is the same as that of PLSA: mapping high-dimensional count vectors to a lower dimensional representation in latent semantic space. Each word w in a service description s is generated by sampling a topic z from topic distribution, and then sampling a word from topic-word distribution. The probability of the ith word occurring in a given service is given by Equation 2:

$$P(w_i) = \sum_{f=1}^{k} P(w_i|z_i = f)P(z_i = f) \tag{2}$$

Where z_i is a latent factor (or topic) from which the ith word was drawn, $P(z_i = f)$ is the probability of topic f being the topic from which w_i was drawn, and $P(w_i|z_i = f)$ is the probability of having word w_i given the fth topic.

Let $\theta^{(s)} = P(z)$ refer to the multinomial distribution over topics in the service description s and $\phi^{(j)} = P(w|z = j)$ refer to the multinomial distribution over words for the topic j. There are various algorithms available for estimating parameters in the LDA: Variational EM [3] and Gibbs sampling [20]. In this paper, we adopt an approach using Variational EM. See [3] for further details on the calculations. For the LDA training, we used Blei's implementation[7], which is a C implementation of LDA using Variational EM for Parameter Estimation and Inference. The key objective is to find the best set of latent variables that can explain the observed data. This can be made by estimating $\phi^{(j)}$ which provides information about the important words in topics and $\theta^{(s)}$ which provides the weights of those topics in each web service. After training the LDA model, we use the learned latent factors to cluster web services. If a probability distribution $\theta^{(s)}$ over a specific z_f when given a web service s is high, then the service s can be affected to the cluster z_f.

[6] `http://cis.upenn.edu/~ungar/Datamining/software\_dist/PennAspect/`
[7] `http://www.cs.princeton.edu/~blei/lda-c/`

The Correlated Topic Model (CTM) is another probabilistic topic model that enhances the basic LDA [3], by modeling of correlations between topics. One key difference between LDA and CTM is that in LDA, there is an independence assumption between topics due to the Dirichlet prior on the distribution of topics. In fact, under a Dirichlet prior, the components of the distribution are independent whereas the logistic normal used in CTM, models correlation between the components through the covariance matrix of the normal distribution. However, in CTM, a topic may be consistent with the presence of other topics. Assume we have S web services as a text collection, each web service s contains N_s word tokens, T topics and a vocabulary of size W. The Logistic normal is obtained by :

- For each service, draw a K-dimensional vector η_s from a multivariate Gaussian distribution with mean μ and covariance matrix Σ : $\eta_s \sim \mathcal{N}(\mu, \Sigma)$
- We consider the mapping between the mean parameterization and the natural parameterization: $\theta = f(\eta_i) = \frac{\exp \eta}{\sum_i \exp \eta_i}$
- Map η into a simplex so that it sums to 1.

The main problem is to compute the posterior distribution of the latent variables given a web service : $P(\eta, z_{1:N}, w_{1:N})$. Since this quantity is intractable, we use approximate techniques. In this case, we choose variational methods rather than gibbs sampling because of the non-conjugacy between logistic normal and multinomial. The problem is then to bound the log probability of a web service :

$$\begin{array}{rl} \log P(w_{1:N}|\mu, \Sigma, \beta) \geq & E_q[\log P(\eta|\mu, \Sigma)] + \sum_{n=1}^{N} E_q[\log P(z_n|\eta)] \\ + & \sum_{n=1}^{N} E_q[\log P(w_n|z_n, \beta)] + H(q) \end{array} \tag{3}$$

The expectation is taken with respect to a variational distribution of the latent variables :

$$q(\eta, z|\lambda, \nu^2, \phi) = \prod_{i=1}^{K} q(\eta_i|\lambda_i, \nu_i^2) \prod_{n=1}^{N} q(z_n|\phi_n) \tag{4}$$

and $H(q)$ denotes the entropy of that distribution (See [4] for more details).

Given a model parameters $\{\beta_{1:K}, \mu, \Sigma\}$ and a web service $w_{1:N}$, the variational inference algorithm optimizes the lower bound (Equation 3)) with respect to the variational parameters using the variational EM algorithm. In the E-step, we maximize the bound with respect to the variational parameters by performing variational inference for each web service. In the M-step, we maximize the bound with respect to the model parameters. The E-step and M-step are repeated until convergence. For the CTM training, we used the Blei's implementation[8], which is a C implementation of Correlated Topic Model using Variational EM for Parameter Estimation and Inference. We estimate the *topic-service* distribution by computing: $\theta = \frac{\exp(\eta)}{\sum_i \exp(\eta_i)}$. Where $\exp(\eta_i) = \exp(\lambda_i + \frac{\nu_i^2}{2})$ and the variational parameters $\{\lambda_i, \nu_i^2\}$ are respectively the mean and the variance of the normal distribution. Then, we estimate the *topic-word* distribution ϕ by calculating the

[8] `http://www.cs.princeton.edu/~blei/ctm-c/index.html`

exponential of the log probabilities of words for each topic. As already mentioned, the learned latent factors can be used to cluster web services. Thus, if a probability distribution θ over a specific z_f when given a web service s is high, then the service s can be affected to the cluster z_f.

The three topic models were trained using different number of classes (e.g 5 to 100) to compare the results (See Section 4).

The key idea of our approach is to cluster the services into a group of learned latent variables, which can be achieved by computing the probability $P(latent\_variable|service)$ for each latent variable. The rationale for this is that the dimensionality of the model is reduced as every web service can be described in terms of a small number of latent factors (topics) rather than a large number of concepts. With the maximum value of the computation used for the cluster for a service, we can categorize services into their corresponding group. Consequently, searching for a service inside a cluster can be performed by searching for matching topics rather than matching the text describing the web service to a set of keywords extracted from the user query.

Based on the clustered service groups, a set of matched services can be returned by comparing the similarity between the query and the related topic, rather than computing the similarity between query and each service in the dataset. If the retrieved services are not compatible with user's query, the second best cluster would be chosen and the computing proceeds to the next iteration.

Service discovery aims to find web services with user required functionalities. The service discovery process assumes that services with similar functionalities should be discovered. In our work, we propose to use the probabilistic topic model to discover the web services that match with the user query. Let $Q = \{w_1, w_2, \ldots, w_n\}$ be a user query that contains a set of words w_i produced by a user. In our approach, we propose to use the generated probabilities θ and ϕ as the base criteria for computing the similarity between a service description and a user query. For this, we model information retrieval as a probabilistic query to the topic model. We note this as $P(Q|s_i)$ where Q is the set of words contained in the query. Thus, using the assumptions of the topic model, $P(Q|s_i)$ can be calculated by equation 5.

$$P(Q|s_i) = \prod_{w_k \in Q} P(w_k|s_i) = \prod_{w_k \in Q} \sum_{z=1}^{T} P(w_k|z_f) P(z_f|s_i) \tag{5}$$

The most relevant services are the ones that maximize the conditional probability of the query $P(Q|s_i)$. Consequently, relevant services are ranked in order of their similarity score to the query. Thus, we obtain automatically an efficient ranking of the services retrieved.

4 Evaluation

Our experiments are performed out based on real-world web services obtained from [21]. The WSDL corpus consists of over 1051 web services from 8 different

application domains. Each web service belongs to one out of eight service domains named as: Communication, Education, Economy, Food, Travel, Medical and Military. Table 1 lists the number of services from each domain.

Before applying the proposed service clustering and discovery, we deal the WSDL corpus. The objective of this pre-processing is to identify the functional terms of services, which describe the semantics of their functionalities. WSDL corpus processing consists of several steps: *Features extraction*, *Tokenization:*, *Tag and stop words removal*, *Word stemming* and *Service Matrix construction* (See Section 3.1).

Domain	**Services**	**Domain**	**Services**
Communication	59	Geography	60
Economy	354	Medical	72
Education	264	Travel	161
Food	41	Military	40

Table 1: Domains of Web services

4.1 Web Service Clustering Evaluation

In order to evaluate the effectiveness of the clustering technique, we use two different measures: *entropy* and *purity* [23, 22]. Suppose q classes represent the partitioned web services (service domains), k clusters produced by our clustering approach and n the total number of services.

- *Entropy*: The entropy measures how the various semantic classes are distributed within each group (cluster). Given a particular cluster C_j of size n_j, the *entropy* of this cluster is defined to be:

$$E(C_j) = -\frac{1}{log(q)} \sum_{i=1}^{q} \frac{n_j^i}{n_j} log(\frac{n_j^i}{n_j}) \tag{6}$$

  Where q is the number of domains in the dataset, and n_j^i is the number of services of the ith domain that where assigned to the jth cluster. The averaged entropy of the clustering solution is defined to be the weighted sum of the individual cluster entropies (Equation 7). In general, smaller entropy values indicate better clustering solutions.

$$Entropy = \sum_{j=1}^{k} \frac{n_j}{n} E(C_j) \tag{7}$$

- *Purity*: The purity measure evaluates the coherence of a cluster. It is the degree to which a cluster contains services from a single domain. The purity of C_j is formally defined as:

$$P(C_j) = \frac{1}{n_j} max_i(n_j^i) \tag{8}$$

Where $max_i(n_j^i)$ is the number of services that are from the dominant domain in cluster C_j and n_j^i represents the number of services from cluster C_j assigned to domain i.
The purity gives the fraction of the overall cluster size that the largest domain of services assigned to that cluster. For a clustering solution, the overall purity is then again the weighted sum of the individual cluster purities (Equation 9). In general, larger purity values indicate better clustering solutions.

$$Purity = \sum_{i=1}^{k} \frac{n_i}{n} P(C_i) \tag{9}$$

In our experiment, we compared the accuracy of three probabilistic clustering algorithms (PLSA, LDA and CTM) to that of a classical clustering algorithm (K-means). The eight service domains described previously (Table 1), are used as the base classes to evaluate *Purity* and *Entropy* of clusters. Thus, we generate k clusters using each algorithm starting with 5 clusters and increasing in steps of 5 up to 100 clusters. The results of Entropy and Purity for clustering solutions are shown respectively in Figure 1(a) and 1(b). The results show that the clustering method based on the CTM performs significantly than others algorithms. We also note that LDA performs better than PLSA and K-means. The K-means is a simple algorithm and does not an always converge in an optimal way. It depends on the random factor of where the initial cluster centroids are generated. As can be seen from Figure 1, CTM and LDA perform better than PLSA and K-means for a large number of clusters. This makes them ideal solutions for web services clustering in large dataset. The Correlated Topic Model allows each service to exhibit multiple topics with different proportions. Thus, it can capture the heterogeneity in grouped data that exhibit multiple latent factors.

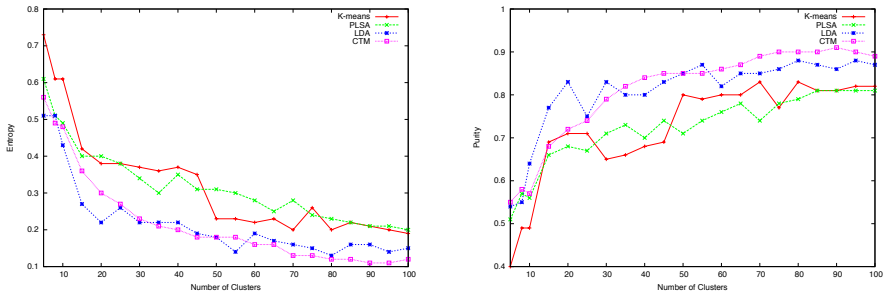

Fig. 1: (a) Entropy of clusters for the proposed clustering solutions. (b) Purity of clusters for the proposed clustering solutions.

4.2 Web Service Discovery Evaluation

We also evaluated the effectiveness of web service discovery based on the three probabilistic topic models (labeled *PLSA*, *LDA* and *CTM*). The probabilistic methods are compared with a text-matching approach (labeled *Text-Search*). For this experiment, we use the services description collected from the WSDL corpus. As described previously, the services are divided into eight domains and some queries templates are provided together with a relevant response set for each query. The relevance sets for each query consists of a set of relevant service and each service s has a graded relevance value $relevance(s) \in \{1, 2, 3\}$ where 3 denotes *high relevance* to the query and 1 denotes a *low relevance*.

In order to evaluate the accuracy of our approach, we compute two standard measures used in *Information Retrieval*: *Precision at n* (*Precision@n*) and *Normalised Discounted Cumulative Gain* ($NDCG_n$). These evaluation techniques are used to measure the accuracy of a search and matchmaking mechanism.

- *Precision@n*: In our context, $Precision@n$ is a measure of the precision of the service discovery system taking into account the first n retrieved services. Therefore, $Precision@n$ reflects the number of services which are relevant to the user query. The precision@n for a list of retrieved services is given by Equation 10:

$$Precision@n = \frac{|RelevantServices \cap RetrievedServices|}{|RetrievedServices|} \tag{10}$$

  Where the list of relevant services to a given query is defined in the test collection. For this evaluation, we have considered only the services with a graded relevance value of 3 and 2.
- *Normalised Discounted Cumulative Gain*: $NDCG_n$ uses a graded relevance scale of each retrieved service from the result set to evaluate the gain, or usefulness, of a service based on its position in the result list. This measure is particularly useful in Information Retrieval for evaluating ranking results. The $NDCG_n$ for n retrieved services is given by Equation 11.

$$NDCG_n = \frac{DCG_n}{IDCG_n} \tag{11}$$

  Where DCG_n is the Discounted Cumulative Gain and $IDCG_n$ is the Ideal Discounted Cumulative Gain. The $IDCG_n$ is found by calculating the DCG_n of the first n returned services. The DCG_n is given by Equation 12.

$$DCG_n = \sum_{i=1}^{n} \frac{2^{relevance(i)} - 1}{log_2(1+i)} \tag{12}$$

  Where n is the number of services retrieved and $relevance(s)$ is the graded relevance of the service in the ith position in the ranked list.

We evaluated our service discovery approach by calculating the $Precision@n$ and $NDCG_n$. In this experiment, we have selected randomly 12 queries from the test collection. The text description is retrieved from the query templates and

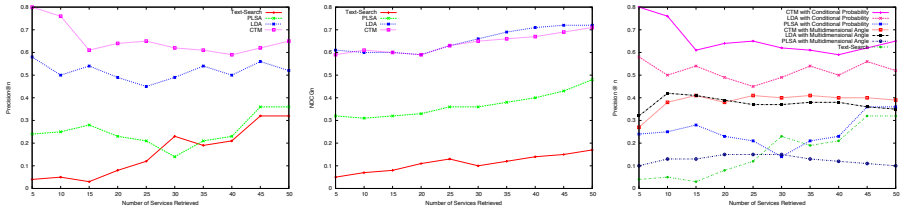

Fig. 2: (a) Comparaison of average Precision@n values over 12 queries. (b) Comparaison of average $NDCG_n$ values over 12 queries. (c) Comparaison of average Precision@n values over 12 queries for all methods using both Conditional Probability and Multidimensional Angle.

used as the query string. We consider that the size of the services to be returned was set to 50. The average $Precision@n$ and $NDCG_n$ are obtained over all 12 queries for CTM, LDA, PLSA and Text-Search. The results are shown in Figure 2(a) and 2(b).

The comparison of $Precision@n$ shows that the CTM and LDA perform better than Text-Search and PLSA. The probabilistic methods based on CTM and LDA used the information captured in the latent factors to match web services based on the conditional probability of the user query. Text-Search and PLSA were unable to find some of the relevant web services that were not directly related to the user's queries through CTM and LDA. The low precision results obtained by probabilistic method based on PLSA are due to limited number of concepts used for training the model. In this context, web service descriptions are similar to short documents. Therefore, the method based on PLSA model is not able to converge to a high precision using these limited concepts.

In Information retrieval, $NDCG_N$ gives higher scores to systems which rank a search result list with higher relevance first and penalizes systems which return services with low relevance. The $NDCG_n$ values for all queries can be averaged to obtain a measure of the average performance of a ranking algorithm. In our experiments, we consider services with graded relevance values from 3 (high relevance) to 1 (low relevance) for this evaluation. $NDCG_n$ values vary from 0 to 1. The results obtained for $NDCG_n$ show that the both CTM and LDA perform better than the other search methods. Thus, the probabilistic methods based on both CTM and LDA give a higher $NDCG_n$ than all other methods for any number of web services retrieved. This reflects the accuracy of the ranking mechanism used by our method. Text-Search and PLSA methods have a low $NDCG_n$ because, as shown in the $Precision@n$ results, both methods are unable to find some of the highly relevant services.

In order to compare the accuracy of our approach with existing approaches, we have implemented the approach proposed by Cassar et al. [6], which uses the proximity measure called *Multidimentional Angle* (also known as *Cosine Similarity*); a measure, which uses the cosine of the angle between two vectors

[17]. In the first time, we represent the user's query as a distribution over topics. Thus, for each topic z_f we calculate the relatedness between query Q and z_f based on $topic - word$ distribution ϕ using Equation 13.

$$P(Q|z_f) = \prod_{w_i \in Q} P(w_i|z_f) \tag{13}$$

Then, we calculate the similarity between the user's query and a web service by computing the Cosine Similarity between a vector containing the query's distribution over topics q and a vector containing the service's distribution over topics p. The multidimensional angle between a vector p and a vector q can be calculated using Equation 14:

$$Cos(p,q) = \frac{p.q}{\parallel p \parallel . \parallel q \parallel} = \frac{\sum_{i=1}^{t} p_i q_i}{\sqrt{\sum_{i=1}^{t} p_i^2 \sum_{i=1}^{t} q_i^2}} \tag{14}$$

where t is the number of topics.

The comparison of average $Precision@n$ (See Figure 2(c)) shows that the probabilistic method CP (i.e. Conditional Probability) performs better than the MA (i.e. Multidimensional Angle) for all the probabilistic topic models. The results show that the CTM and LDA perform better than Text-Search and PLSA.

5 Conclusion

In this paper, we have used several probabilistic topic models (i.e. PLSA, LDA and CTM) to extract latent factors from web service descriptions. Then, the learned latent factors are used to group services into clusters. Indeed, the categorization of services is often done with human intervention. To overcome this limitation, we propose to vary the number of topics (which can be considered as clusters, with one difference, which is that we can model the observations in a more compressed way than it would be if the model was based on clusters) to automatically obtain the categories to which services belong. The accuracy of the three probabilistic clustering algorithms is compared with a classical clustering algorithm (i.e. K-means). The results show that the clustering method based on both CTM and LDA perform better than PLSA and K-means. In our work, we propose also to use the probabilistic topic models to discover the web services that match with the user query. We evaluated our service discovery approach by calculating the $Precision@n$ and $NDCG_n$. The comparison of $Precision@n$ and $NDCG_n$ show that the CTM and LDA perform better than the other search methods (i.e. Text-Search and PLSA). This reflects the accuracy of the ranking mechanism used by our method. The probabilistic methods based on both CTM and LDA used the information captured in the latent factors to match web services based on the conditional probability of the user query. The obtained results show that the topic models provide a scalable and interoperable solution for automated service discovery in large service repositories. Future work will focus on developing a new probabilistic model based on the latent factors to tag web services automatically.

References

1. Abramowicz, W., Haniewicz, K., Kaczmarek, M. and Zyskowski, D.: Architecture for Web services filtering and clustering. In ICIW'2007.
2. Atkinson, C., Bostan, P., Hummel O. and Stoll, D.: A Practical Approach to Web service Discovery and Retrieval. In ICWS'2007.
3. Blei, D., Ng, A. Y. and Jordan, M. I.: Latent dirichlet allocation. J. Mach. Learn. Res., 3:993-1022, 2003.
4. Blei, D., and Lafferty, John D.: A Correlated Topic model of Science, In AAS 2007. pp. 17-35.
5. Cassar, G., Barnaghi, P. and Moessner, K.: Probabilistic methods for service clustering. In Proceeding of the 4th International Workshop on Semantic Web Service Matchmaking and Resource Retrieval, Organised in conjonction the ISWC'2010.
6. Cassar, G.; Barnaghi, P.; Moessner, K.: A Probabilistic Latent Factor approach to service ranking. In ICCP'2011, pp.103-109.
7. Elgazzar, K., Hassan A., Martin, P.: Clustering WSDL Documents to Bootstrap the Discovery of Web Services. In ICWS'2010, pp. 147-154.
8. Dong, X., Halevy, A., Madhavan, J., Nemes, E., Zhang, J.: Similarity Search for Web Services. In VLDB Conference, Toronto, Canada, pp. 372-383, 2004.
9. Hess, A. and Kushmerick, N.: Learning to Attach Semantic Metadata to Web services. In ISWC'2003, Sanibel Island, Florida, USA, 2003
10. Hofmann, T.: Probabilistic Latent Semantic Analysis. In UAI(1999), pp. 289-296.
11. Kokash, N.: A Comparison of Web Service Interface Similarity Measures. Frontiers in Artificial Intelligence and Applications, Vol. 142, pp.220-231, 2006.
12. Lausen, H. and Haselwanter, T.: Finding Web services. In European Semantic Technology Conference, Vienna, Austria,2007
13. Liu, Wei., Wong, W.: Web service clustering using text mining techniques. In IJAOSE'2009, Vol. 3, No. 1, pp. 6-26.
14. Ma, J., Zhang, Y. and He, .J.: Efficiently finding web services using a clustering semantic approach. In CSSSIA'08, pp 1-8. ACM, New York, NY, USA.
15. Nayak, R. and Lee, B.: Web service Discovery with Additional Semantics and Clustering. In IEEE/WIC/ACM 2007
16. Porter, M. F.: An Algorithm for Suffix Stripping, In: Program 1980, Vol. 14, No. 3, pp. 130-137.
17. Platzer, C., Rosenberg F. and Dustdar, S.: Web service clustering using multidimentional angles as proximity measures. ACM Trans. Internet Technol. 9(3), pp. 1-26 (2009).
18. Salton, G.: Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer. Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA (1989).
19. Sivashanmugam, K., Verma, A.P and Miller, J.A.: Adding Semantics to Web services Standards. In ICWS'2003, pp: 395-401.
20. Steyvers, M. and Griffiths, T.: Probabilistic topic models. In Latent Semantic Analysis: A Road to Meaning, T. Landauer, D. Mcnamara, S. Dennis, and W. Kintsch, Eds. Laurence Erlbaum, 2007.
21. Yu, Q.: Place Semantics into Context: Service Community Discovery from the WSDL Corpus. In ICSOC 2011, LNCS 7084, pp. 188-203.
22. Zhao, Y. and Karypis, G.: Empirical and theoretical comparisons of selected criterion functions for document clustering. In Machine Learning'2004, 55, pp. 311-331.
23. Zhao, Y. and Karypis, G. Evaluation of hierarchical clustering algorithms for document datasets. In CIKM'2002.