2010 IEEE/ACM International Conference on Green Computing and Communications & 2010 IEEE/ACM International Conference on Cyber, Physical and Social Computing

A Semantic Bayesian Network for Web Mashup Network Construction

Chunying Zhou[1], Huajun Chen**, Zhipeng Peng[1]
College of Computer Science
Zhejiang University
Hangzhou, China
{cyzhou, huajunsir}@zju.edu.cn,
phang.aaron@gmail.com

Yuan Ni[2], Guotong Xie[2],
IBM Research - China,
IBM
Beijing, China
{niyuan, xieguot}@cn.ibm.com

***Abstract*—With a mashup network in which a link indicates that two applications are mashupable, building a mashup can be simplified into network navigation. This paper presents an approach that constructs a Web mashup network by learning a semantic Bayesian network using a semi-supervised learning method. An RDF model is defined to describe attributes and activities of applications. To process all information sources on the Semantic Web, a semantic Bayesian network (sBN) is proposed where a semantic subgraph template defined using a SPARQL query is used to describe the information about the graph structure. The sBN offers more powerful abilities to process the information sources on Semantic Web, especially the graph structure. To improve the learning performance, a semi-supervised learning method that makes use of both labeled and unlabeled data is proposed. We ran the approach on a data set containing 100 applications collected from the website Programmableweb.com and 3077 links checked manually. The results show that the approach outperforms the PRL and the rule-based methods, and the semi-supervised learning method achieved big improvements in recall and $F_{0.5}$, compared with the direct learning method.**

Keywords:Semantic Web, mashup network, probabilistic learning

I. INTRODUCTION

Originally used to describe the mixing together of musical tracks [1], mashup now refers to websites weaving data from different sources into a new service. Its popularity appealed companies to develop their own mashup tools, such as Yahoo!Pipes [2], Mashup Center [3], and Mash Maker [4].

Inspired by the social network, an approach that builds a mashup network to facilitate building a mashup into network navigation was proposed by Bin, L. et. al [5]. However, it encounters a problem of predicting mashupable applications. In this paper, we propose an approach that combines the Semantic Web [6] and the statistic learning [7] to do the task of predicting mashupable applications. This approach addresses three problems: 1) to describe attributes, activities of applications precisely; 2) given attributes and activities of applications, to predict mashupable applications; 3) using a small amount of training data, to improve the performance.

In this approach, an RDF model is defined to describe attributes and activities of applications. Microformats [8] is used to define semantics of data that applications consume and produce. There are three information sources on the Semantic Web: descriptive attributes, relationships and the semantic graph structure-based attribute that describes the existence of a subgraph with a specified structure. To predict mashupable applications, not only descriptive attributes and relationships, but also the third information source – the graph structure-based attribute affects the prediction result. For example, whether there is another application B in the same category with A and has a link to C, probably affects the existence of a link between A and C. This factor is actually a graph structure-based attribute.

To handle all the information sources, we propose a semantic Bayesian network (sBN) that extends Bayesian networks [9] with abilities of processing relationships and semantic graph structure-based attributes. It consists of the qualitative and quantitative components. The qualitative component is actually a dependency graph composed of a set of attribute nodes (node for short) and dependency relationships amongst the nodes. The dependencies specify each node with a set of parent nodes. By learning from the training data, for each node, a conditional probability distribution (CPD) over its parent nodes is obtained. The quantitative component is the CPDs of all nodes. Given these two components, the joint probability of any combination of the values of all nodes can be computed as a product of their CPDs. Then, the obtained joint probability can be used to make the link prediction. A unique feature of an sBN is that not only descriptive attributes and relationships, but also the semantic graph structure-based attribute can be incorporated and processed. Particularly, a semantic subgraph template that can be defined using a SPARQL [10] query is proposed to describe a semantic graph structure-based attribute.

The existing mashups on the current Web are not all links between mashupable applications. It can't be the training data for learning methods. Also, it is impossible to construct links between all mashupable applications manually. To solve the performance loss caused by lacking of training data, we propose a semi-supervised learning method that makes use of both the labeled data and the unlabeled data.

Two series of experiments were set up to do performance evaluation. First series are to do comparisons between the rule-based, the probabilistic relational learning methods and the presented approach. The results show that our approach achieved the best precision, recall and $F_{0.5}$. The second series are to do the comparison between the semi-supervised learning and the direct learning methods. And the results

978-0-7695-4331-4/10 $26.00 © 2010 IEEE
DOI 10.1109/GreenCom-CPSCom.2010.88

show that the semi-supervised learning method made big improvements in recall and $F_{0.5}$.

II. Related Work

In the context of mashup, many research contributions have been reported. Ohad, G. *et al.* [11] presented an approach of auto-completion for mashups in which 'glue paths' are recommended based on their similarities compared with the given applications. The approach only considered the existing mashups, users are always deprived of creativity. A new idea of building a mashup network to facilitate mashup building was proposed by Bin, L. *et al.* [5]. A rule-based method was used to predict mashupable applications. But, by studying the existing mashups carefully, we found out that the existence of a link between applications does not only depend on this factor. In addition, it's difficult to predict mashupable applications by just using deterministic rules.

Link prediction is to predict the existence of a link between two entities, taking attributes of the entities and other observed links as the known information. Approaches have been proposed to predict friendships; to predict the participation of actors in events using email, telephone calls information [12, 13], predicting semantic relationships such as 'advisor-of' using web page links and content [14], and mining link structure of web pages [15]. Particularly, many contributions related to the collaboration network have been reported [16-18] by applying graph mining, web mining, text mining and machine learning to do the link prediction.

To make probabilistic models applicable on the relational model for link prediction, Lise, G. et al. [19] proposed a probabilistic relational model (PRM) that extends Bayesian networks with abilities of processing links. Benjamin, T. et al. proposed a relational Markov network model [20] composed of a set of relational clique templates describing network topology-based attributes. A clique template requires the described topology to be 'strong connected', where every pair of nodes must be connected. And, Christoph, K. et al. proposed an approach called SPARQL-ML [21] that added data mining support to SPARQL via the PRL methods.

On Semantic Web, besides attributes and relationships, the graph structure is also another information source. For example, whether there is another application B in the same category with A and has a link to C, also affect the existence of a link between A and C. To describe and incorporate this unprocessed information source, in our approach, a semantic subgraph template is proposed and defined to describe a graph structure-based attribute. It is able to define two types of patterns and combine them together. One type specifies the link structure of subgraphs, and the other type specifies constraints on the values of descriptive properties of the nodes. It is more complex and flexible than a graph clique.

III. An RDF Model for Web Mashup

An RDF model is defined to describe the attributes and relationships of applications. Figure 1 displays this RDF that is described in details in Table.1. An RDF graph of two applications connected by a relationship 'mash:compatible' is shown in Figure 2. This RDF is more expressive by using Microformats that can provide accurate specifications of data semantics. For example, input of 'GoogleMap' are defined as 'XML:String', and so is the output of 'GoogleWebSearch'. Using Microformats, these data are separately defined as 'geo.longitude' and 'geo.latitude', and 'object.string' to differentiate the semantic differences.

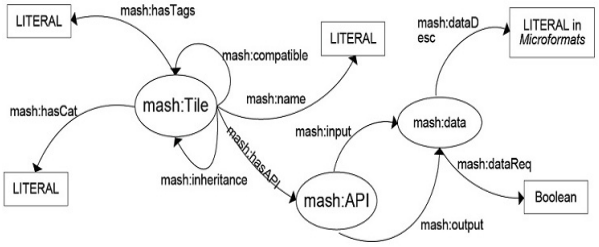


Figure 1. An RDF model defines attributes and activities of applications.

TABLE I. Classes and properties of the RDF model.

RDF literals	Description
mash:Tile	an RDF class describes applications
mash:API	an RDF class describes an API of an application
mash:data	an RDF class describes semantics of input and output data of an API
mash:name	describes the name of applications.
mash:hasTags	describes tags of an application
mash:hasCat	defines the category of an application
mash: compatible	A relationship connects two mashupable applications
mash: inheritance	A relationship connects applications (A, B) that indicats A inherits from B
mash:input mash:output	relationships connect mash:API with mash:data
mash:dataDesc	describes the data semantics using the Microformats
mash:dataReq	defines whether one 'mash:data' is required or not

A new relationship 'mash:inheritance' between instances of 'mash:Tile' is defined in Definition 2.1. Application A is connected to application B by this relationship if and only if all APIs of A are overrode by APIs of B. Definition 2.2 defines 'API override' representing that an API overrides all functionalities of another API.

Definition 2.1 [Relationship 'mash:inheristance'] Given an application A with a set of APIs $S_A = \{A_{a_1}, A_{a_2}, ..., A_{a_n}\}$, and B with APIs $S_B = \{B_{a_1}, B_{a_2}, ..., B_{a_n}\}$, a relationship 'mash:inheritance' between A and B exists if and only if for $\forall B_{a_i} \in S_B$, $\exists A_{a_i} \in S_A$ that overrides B_{a_i}.

Definition 2.2 [API override] Given an instance A_1 of class 'mashup:API' has a set of input $S_{input(A)}$ whose values of 'mash:dataReq' are 'true' and a set of output $S_{output(A)}$, and another instance B_1 of 'mashup:API' containing a set of input parameters $S_{input(B)}$ whose values of 'mash:dataReq' are 'true' and output parameters $S_{output(B)}$. Application B_1 overrides A_1 if and only if $S_{input(A)} = S_{input(B)}$ and $S_{output(A)} \subseteq S_{output(B)}$.

As shown in Figure 3, 'GoogleWebSearch' (A) has one API API_A that requires one input data and produces three

output data. 'LiveNews' (B) also has one API (API_B). It requires one input data that can override the input data that A consumes and produces 6 output data that can override all three output data that A produces. API_B overrides API_A and B is connected to A by the relationship 'mash:inheritance'.

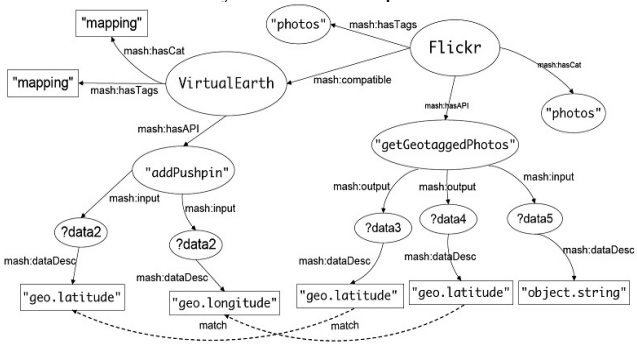


Figure 2. Two applications connected by 'mash:compatible'.

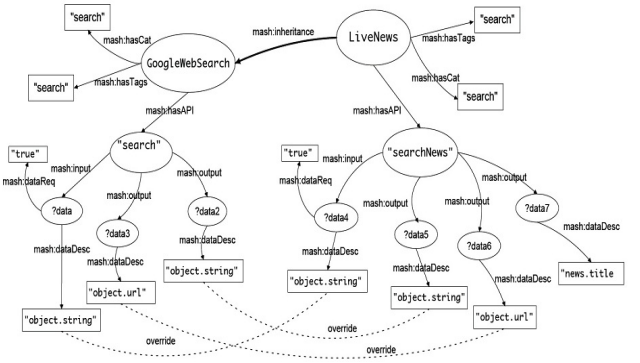


Figure 3. Applications 'LiveNews' and 'GoogleWebSearch'are connected by a relationship 'mash:inheritance'.

IV. Learning a Semantic Bayesian Network for Link Prediction

A. A Semantic Bayesian Network Model

A semantic Bayesian network (sBN) extends Bayesian networks on Semantic Web with extensions to incorporate relationships and semantic graph structure-based attributes. It specifies a template for probability distributions of attributes over the entire semantic graph. It consists of the qualitative and quantitative components. The qualitative component is a dependency graph G that consists of a set of nodes and dependencies amongst nodes. A dependency specifies a node with a parent node that declares that its value depends directly on values of its parent nodes. Definition 3.1 defines the dependency graph that is required to be acyclic where no node's value depends directly or indirectly on itself.

Definition 3.1 In an *sBN*, a dependency graph is composed of a set of nodes S_{attr} and a set of probabilistic dependencies amongst them. $\forall A_x \in S_{attr}$, it is associated with a set of parent nodes $Parents(A_x)$ through dependency relationships.

The quantitative component specifies a parameterization for the qualitative component, which is a set of conditional probability distributions (CPDs) defined in Definition 3.2 associated with the nodes of the dependency graph. For each node, a local probability distribution over all combinations of values of its parent nodes can be defined.

Definition 3.2 [CPD] $\forall A_x \in S_{attr}$, a set of parent nodes $Parents(A_x)$ is assigned with it. Given a set of instantiations of the same RDF model, its CPD is defined as:

$$P_{CPD}(A_x) = P(A_x \mid comb(Parents(A_x))),$$

where $comb(Parents(A_x))$ is the set of values of its parents.

B. A Semantic Relation in a Semantic Bayesian Network

A descriptive attribute is directly mapped to a descriptive property of the RDF, and it can be processed naturally as a usual node in a Bayesian network. But, an RDF model contains not only descriptive properties but also associated properties (relationships). A relationship connecting RDF classes is more complex and impossible to be processed directly in a Bayesian network like a descriptive property.

In the sBN, we propose an extension to the Bayesian network to incorporate a relationship. The existence of a link is defined as a node in the dependency graph, enabling the sBN with abilities of incorporating relationships. For example, to predict whether two applications A and C are connected by a relationship R, the existence of R is defined as a node R.Exists in the dependency graph. Problems caused by the structured nature of the RDF model are solved.

C. A Semantic Graph Structure-based Attribute

Up to now, both descriptive attributes and relationships can be incorporated in the sBN. However, there is another information source on Semantic Web that should also be incorporated. That is the semantic graph structure-based attribute that describes an attribute on the graph level. It is always a graph structure consisting of a set of descriptive properties and relationships. It cannot be processed in neither ways of processing a descriptive property nor a relationship.

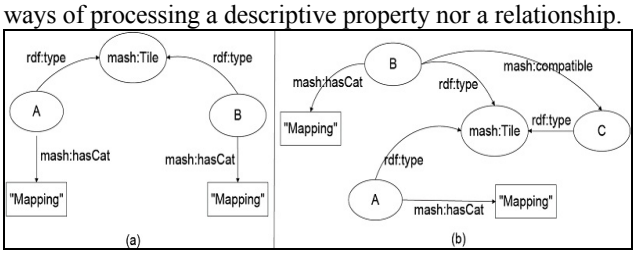


Figure 4. Two examples of semantic graph structure-based properties, (a) describes an attribute of 'Type-A' and (b) shows an attribute of 'Type-B'.

The semantic graph structure-based attribute is classified into two types. One type of the semantic graph structure-based attribute is called 'Type-A'. It describes whether values of a descriptive property of two instances are the same or not. Its value is of the type 'Boolean', either 'true' or 'false'. Figure 4 (a) describes an example of the type 'Type-A'. This attribute describes whether values of 'mash:hasCat' of two applications are the same or not. The other type is called 'Type-B' that is much more complex. It involves more than two instance nodes and both descriptive properties and

relationships are probably be involved. Figure 4 (b) shows an example of the 'Type-B', having three nodes, two values of 'mash:hasCat' and a relationship 'mash:compatible'. This attribute defines that, for two applications A and C, whether there is another application B, in the same category with A and connected to C by 'mash:compatible'.

A semantic subgraph template is defined to describe a semantic graph structure-based attribute. It specifies a set of subgraphs that comply with predefined common patterns. SPARQL is a standard RDF query language that can be used to define a semantic subgraph template by specifying what kinds of subgraphs belong to it. The definition consists of: the 'SELECT' and 'WHERE' components.

The 'SELECT' statement specifies the query results. In an sBN, there is an assumption that attributes whose values can be enumerated can be processed. Thus, 'SELECT' needs to be converted into a logical predicate. The 'WHERE' is the key component of the definition. It defines query conditions that are common patterns which subgraphs of this template must comply with. Each condition is an RDF triple defined in the Definition 3.3. Definition 3.4 gives a formal definition of a semantic subgraph template.

Definition 3.3 An RDF triple contains three components:

- Subject is an URI reference or a blank node;
- Predicate is an URI reference that is the property connecting Subject and Object;
- Object is an URI reference, a literal or a blank node.

Conventional written order is: Subject, Predicate and Object. Further information can be found in W3C document of RDF.

Definition 3.4 An RDF model R_i has a set of classes $S_{class} = \{c_1, c_2, ..., c_n\}$, in which each class c_i is associated with a set of descriptive properties denoted as $Pro(c_i)$, and classes may be connected by relationships denoted as $R(S_{class})$. Given R_i, a semantic subgraph template T = (S, W) where S is the 'SELECT' and W is the 'WHERE' is defined:

- 'SELECT' : $S = \{ p_1, p_2, ..., p_n \}$ where $p_i \in \bigcup_{c_i \in S_{class}} \Pr o(c_i)$ returns values of the properties defined in R_i.
- 'WHERE' : W consists of a set of RDF triples $S_{triple} = \{ tr_1, tr_2, ..., tr_n \}$, where $tr_i \in S_{triple}$ defines a condition that subgraphs must satisfy.

An example definition of a semantic subgraph template is shown in Figure 5. This template defines that for two applications A and C, there is another application B that is in the same category with A and is connected to C by 'mashup:compatible'. Detailed explanations of 'SELECT' and 'WHERE' components are listed as follows:

- ✓ 'SELECT' : A variable represents the existence of application *B*. If the result is null, its value is *'false'*; otherwise, its value is *'true'*;
- ✓ 'WHERE' : 7 RDF triples are defined:
  - ✧ 1-3th triples: '?mashletA', '?mashletB', '?mashletC' are instances of *'mash:Tile'*;
  - ✧ 4th triple: the variable '?cat' represents the value of '*mash:hasCat*' of '?mashletA';
  - ✧ 5th triple: '?mashletB' has a value (the variable ?cat2) of property 'mash:hasCat';
  - ✧ 6th triple: '?mashletB' is connected to '?mashletC' by '*mash:compatible*';
  - ✧ 7th triple:?cat and ?cat2 have a same value;

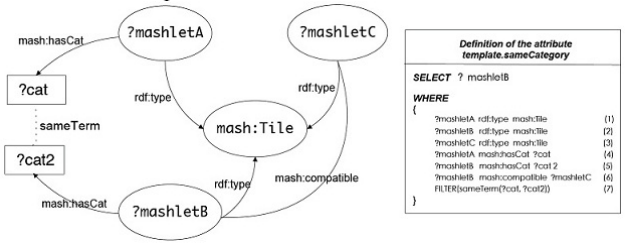


Figure 5. An example semantic subgraph template definition.

The extension of a semantic subgraph template enables an sBN with more powerful descriptiveness of attributes on the RDF graph. An sBN can process all information sources: descriptive properties, relationships and the graph structure.

D. Learning a sBN for Link Prediction

To compute the joint probability of an sBN, there is a prerequisite requiring its dependency graph to be acyclic. Another assumption 'conditional independence' declares that a node's value only depends on the values of its parent nodes. As defined in Definition 3.5, the joint probability of an sBN is computed as a product of nodes' conditional probabilities over their parent nodes. The computation of the conditional probability of the link's existence is simplified as Lemma 3.1.

Definition 3.5 [Joint Probability Computation] For a set of instantiations of an RDF model S_I, given an *sBN* with a dependency graph of a set of nodes $S_{attr} = \{A_1, A_2, ..., A_n\}$ where each node A_x is associated with a set of parent nodes *Parents*(A_x), the joint probability of $C = \{a_1, a_2, ..., a_n\}$ is computed as a product of nodes' conditional probabilities:

$P(a_1, a_2, ..., a_n) =$

$\prod_{A_i \in S_{attr}} P(A_i = a_i \mid Parents(A_i) = \{a_j, ..., a_m\})$, where

$\{a_j, ..., a_m\}$ are values of parents of A_i.

Lemma 3.1 [Conditional Probability Computation]

- ✓ The conditional probability P(R.Exists='true'| $A_1 = a_1, ..., A_n = a_n$) denoted as $\Pr_{true} = \frac{P(R.Exists = 'true', S_{attr} = s_{attr})}{P(S_{attr} = s_{attr})}$
- ✓ The conditional probability P(R.Exists='false'| $A_1 = a_1, ..., A_n = a_n$) denoted as $\Pr_{false} = \frac{P(R.Exists = 'false', S_{attr} = s_{attr})}{P(S_{attr} = s_{attr})}$
- ✓ Normalization to 1.

$$\Pr_{true} = \frac{\Pr_{true}}{\Pr_{true} + \Pr_{false}} =$$

$$\frac{P(R.Exists='true',S_{attr}=s_{attr})}{P(R.Exists='true',S_{attr}=s_{attr})+P(R.Exists='false',S_{attr}=s_{attr})}$$

E. Use Case of Learning a Semantic Bayesian Network for Link Prediction in Domain of Mashup

The process of learning a semantic Bayesian network for link prediction in domain of mashup consists of four steps:

1) A dependency graph is defined and shown in Figure 6, the existence of a link interacts with other 6 nodes;
2) The dependency graph specifying each node with a set of parents, compute the quantitative component by learning a conditional probability distribution for each node from the training data;
3) Compute the joint probability - the product of all nodes' CPDs and then produces a learned *sBN*;
4) Using the learned *sBN*, predict existences of links.

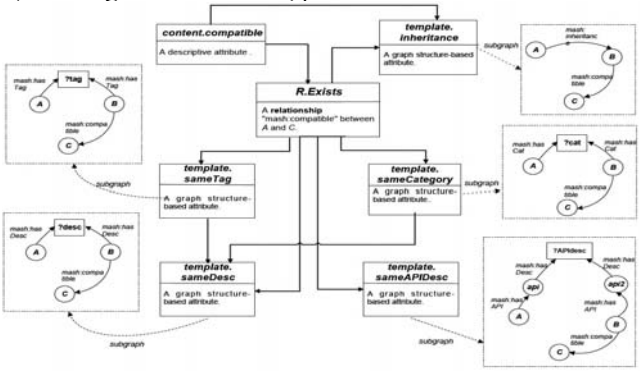


Figure 6. A dependency graph of an sBN for link prediction.

The dependency graph shown in Figure 6 contains 7 nodes and specifies each node with a set of parent nodes.

- '*content.compatible*' ($link_1$): A descriptive attribute describes whether output that an application produces satisfies requirements of input that another application consumes or not. It has no parents
- '*R.Exists*' ($link_2$): the existence of 'mash:compatible' between two applications. Parents($link_2$) = { $link_1$ }
- '*template.inheritance*' ($link_3$): An attribute of 'Type-B' describes that for applications A, C, whether there is B linked to A by 'mash:inheritance' and to C by 'mash:compatible'. Parents($link_3$) = { $link_1$, $link_2$ }
- '*template.sameCategory*' ($link_4$): An attribute of 'Type-B' describes that for A, C, whether there is B having a same 'mash:hasCat' value with A and linked to C by 'mash:compatible'. Parent($link_4$) = { $link_2$ }
- '*template.sameTag*' ($link_5$): An attribute of 'Type-B' describes for A, C, whether B exists that has the same a 'mash:hasTag' value with A and is linked to C by 'mash:compatible'. Parent($link_5$) = { $link_2$ }
- '*template.sameDesc*' ($link_6$): An attribute of 'Type-B' describing for A, C, whether there is B having a same 'mash:hasDesc' value with A and linked to C by 'mash:compatible'. Parent($link_6$)={ $link_2$, $link_4$, $link_5$ }
- 'template.sameAPIDesc' ($link_7$): An attribute of 'Type-B' describing that for A, C, whether there is B having an API having a same 'mash:hasDesc' value with one of APIs of A and also connected to C by 'mash:compatible'. Parent($link_7$) = { $link_2$ }.

V. A Semi-supervised Learning Method

Herein, a semi-supervised learning method is proposed to improve performance. Beside the labeled data, it also makes use of the unlabeled data. This method is composed of a group of learning iterations. In each iteration, the training data is made up of two parts: the training data and the prediction results of its previous iteration. With iteration proceeds, the training data gets increased gradually. Then, the learned sBN is used to make link prediction on the unlabeled data. We define a posterior probability over the unlabeled data in the Definition 4.1. The iteration does not end up until this probability of the iteration no longer gets better. The pseudo code of this method is given in Figure 7.

[pseudo code of the Semi-supervised Learning Method]

For an sBN_t and the probability $para_F$, the code is:

Input:

The training data: applications set S_A , links set S_L .

The links set $S_{predicted}$ that are to be predicted.

Code:

```
1. CONSTRUCT( S_A , S_L , S_predicted );
2. INITIALIZATION-EMPTY( L_exists , L_not exists );
3. INITIALIZATION(previous_Pro);
4. while(true){
5.   sBN_t =Training(Training_data, sBN);
6.   Clear( L_exists , L_not exists , para_F );
7.   for(each link_i ∈ S_predicted ){
8.     Predict( sBN_t , link_i );
9.     Add( A_i and A_j , S_A );
10.    Add( link_i , S_L );
11.    if( link_i Exists){
12.      Add( link_i , L_exists );
13.    }else
14.      Add( link_i , L_not exists );
15.  para_F = EVALUATION( L_exists , L_not exists );
16.  if( para_F <=previous_Pro){
17.      break; //end up the iteration;
18.  }else{
19.    SUBSTITE(previous_Pro, para_F );
20.    Add-PositiveResult-TrainingData ( L_exists , L_not exists );
21.  }
22. }
```

Figure 7. Pseudo code of semi-supervised learning method.

Definition 4.1 [posterior probability] Given the labeled data $D_{labeled}$ and the unlabeled data $D_{unlabeled}$, a posterior probability over $D_{unlabeled}$ is defined as follows:

$post\Pr o(predict(R.Exists) : R \in D_{unlabeled} \mid D_{labeled}) =$

$\prod_{R \in D_{unlabeled}} predict(R.Exists \mid D_{labeled})$, and predict(R.*Exists*) is P(R.Exists='true'| $D_{labeled}$), computed by learning *sBN*.

The procedure of our semi-supervised learning method for mashup network construction consists of three steps. The initial training data (a network N_{init}) and testing data are shown in Figure 8 (a). N_{init} consists of 20 applications and links amongst them, constructed manually. The rest 80 applications (denoted as $S_{predicted}$) need to be added into N_{init}.

Given N_{init}, links $L_{predicted}$ that need to be predicted is defined in Definition 4.2. As shown in Figure 8 (b), the method is composed of a series of learning iterations. In each iteration, the sBN is first learned on the training data. For 1st iteration, the training data is the network N_{init} constructed manually at the start. For other iteration, the training data is composed of: the training data and prediction results of its previous iteration.

For all iteration, links that need to be predicted are the same. The trained sBN is thereby applied on $L_{predicted}$ to predict the existence of potential links. Finally, the iteration process ends up when the posterior probability of the learning iteration no longer gets increased. A mashup network in Figure 8 (c) is constructed of 100 applications.

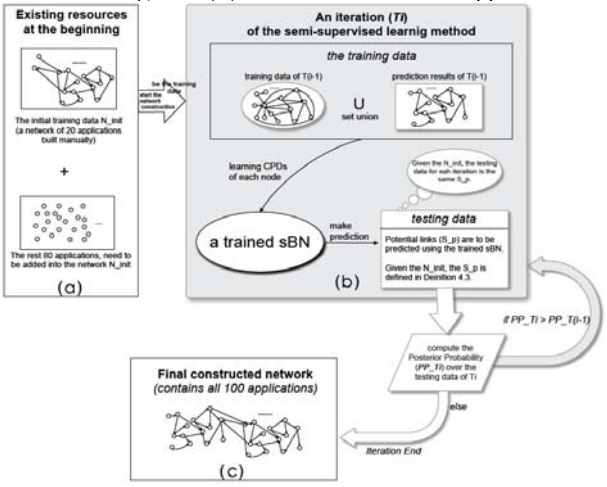


Figure 8. The semi-supervised learning method for the *sBN*.

Definition 4.2 [Predicted Link Set] Given a network N_{init} that consists of 20 applications S_{init} and links amongst them, and the rest $S_{predicted}$, links $L_{predicted}$ to be predicted is:

$$L_{predicted} = \bigcup_{B_i \in S_{predicted}}^{A_i \in S_{init}} ((A_i, B_i) \cup (B_i, A_i)) \cup \bigcup_{B_i \in S_{predicted}}^{A_i \in S_{predicted}} (A_i, B_i),$$

where (A_i, B_i) represents the link from A_i to B_i.

VI. EXPERIMENTS

A. Experiment Setup and Objectives

We collected a set of 100 applications from the website (http://www.programmableweb.com) and converted them into the RDF model. By manually verifying mashupable applications, a network of 100 nodes and 3077 links was built up. It is divided into the training and testing data. Precision, recall and $F_{0.5}$ are used to evaluate its performance.

To evaluate the presented approach, the probabilistic relational learning (PRL) [19] and the rule-based method [5] were used as the competitors. The approach was first compared with the PRL method. Different from the PRL method, besides descriptive attributes and relationships, a semantic Bayesian network is extended to enable processing the semantic graph structure-based attribute. Experiments are set up to evaluate whether the approach works better than the PRL method. Other than probabilistic learning methods, the rule-based method uses logical reasoning to predict links.

- The PRL method: It can process descriptive attributes and relationships using the semi-supervised learning method. Its dependency graph (DG_{prl}) only contains two nodes *'R.Exists'* and *'content.compatible'* and a dependency from *'R.Exists'* to *'content.compatible'*. It is part of the sBN's dependency graph of Figure 6.
- The rule-based method: Specific to our mashup case, one deterministic rule is defined to predict links. It is that if output that application A produces satisfies the requirements of inputs that application B consumes, the link from A to B exits, otherwise does not exist.

To evaluate whether the semi-supervised learning method really improves the performance, comparing with the direct learning method, 20 randomly selected applications were used as the training data. The left 80 applications and links amongst them were used as the testing data. To avoid the impact of using one specific set of 20 applications, our approach and the PRL method were both executed 10 times using ten pairs of training and testing data. The mean of precision, recall and F_2 were got. The rule-based method was also executed 10 times using the same testing data.

The whole experiment data $D_{\exp} = App_{\exp} + Links_{\exp}$, and $App_{\exp} = \{a_1, a_2, ..., a_n\}$, $Links_{\exp} = \{l_1, l_2, ..., l_n\}$. The preparation work for the evaluation includes:

- ***The training data***: The three approaches have to be executed 10 times to get the mean performance. Each time, 20 applications $TD_i = \{T_{i1}, T_{i2}, ..., T_{i20}\}$ were randomly selected from total 100 applications. 10 different training data $D_{training}$ were built up.
- ***The testing data***: Corresponding to each training data TD_i, the testing data $Test_i$ is composed of the rest 80 applications not in TD_i and links amongst them. 10 testing data D_{test} were built up.

To make comparisons between our approach and other two methods, three experiment plans are defined as follows:

- ***Plan 1***: Using the training data $TD_i \in D_{training}$ and the testing data $Test_i \in D_{test}$, an *sBN* whose dependency graph is shown in Figure 6 is learned on the training data and used to predict links on the testing data. The semi-supervised learning method was also used. With 10 pairs of training and testing data, the approach was executed 10 times to get mean performance.
- ***Plan 2***: In a like manner, the PRL method was also executed 10 times, each of which uses TD_i and $Test_i$, with using the semi-supervised learning method. The mean performance was obtained.
- ***Plan 3***: The rule-based method was executed 10 times on $Test_i \in D_{test}$ to get mean performance.

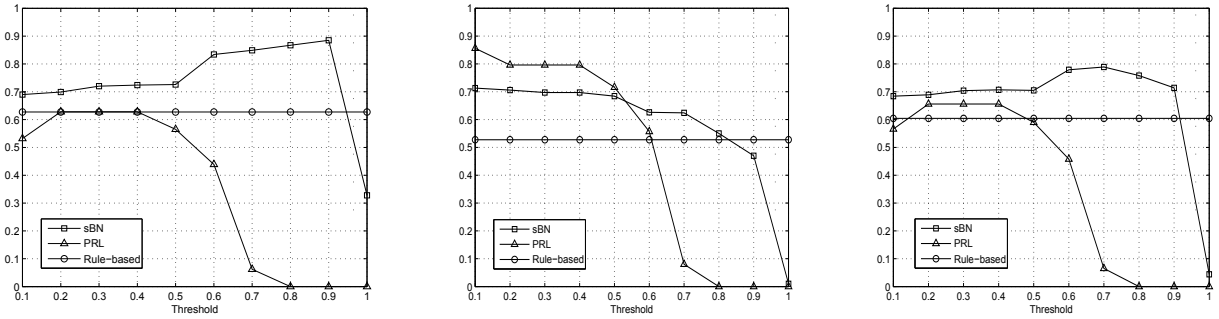
The direct learning method is that an *sBN* is learned directly on the training data and used to do link prediction on the testing data. The experiment plans for the semi-supervised and direct learning methods are defined:

- ***Plan 4*** (for the semi-supervised learning method): The experiment plan is the same to '*Plan 1*'.
- ***Plan 5*** (the direct learning method): For each pair of TD_i and $Test_i$, an *sBN* is learned on TD_i and used to do prediction on $Test_i$ directly to get mean performance.

B. Results and Analysis

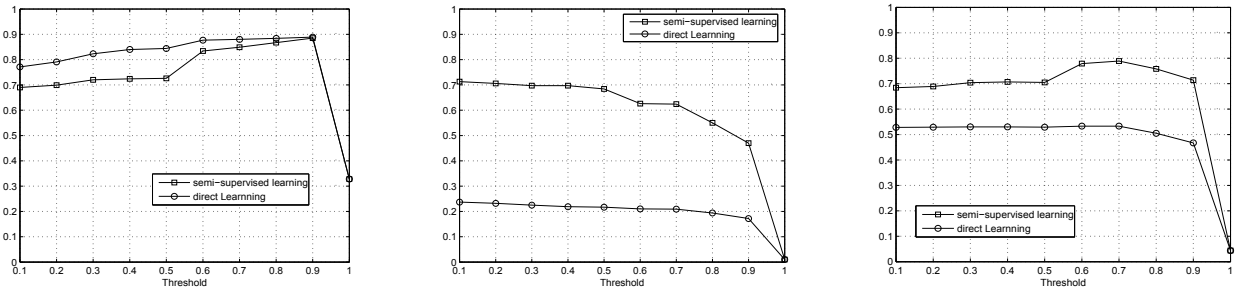
To evaluate the performance of our approach, a series of experiments of *Plan 1, Plan 2* and *Plan 3* were set up and executed to evaluate precision, recall and $F_{0.5}$ of the three methods. The 'threshold' was assigned as {0.1, 0.2,...,1}. Figure 8 shows the mean performance over 10 executions.

As shown in Figure 9 (a, b), precision of our approach got increased as 'threshold' increased from 0.1 to 0.9. While its recall got decreased as 'threshold' increased. As shown in Figure 9 (c), $F_{0.5}$ got increased as 'threshold' increased from 0.1 to 0.7 and the best performance when 'threshold'= 0.7, while from 0.8 to 1.0, $F_{0.5}$ got decreased. The results show that as 'threshold' increased from 0.1 to 0.7, the influence made by precision on $F_{0.5}$ was bigger than that made by recall. As 'threshold' increased from 0.8 to 1.0, the influence of recall's decrease on $F_{0.5}$ was bigger than precision's increase. As shown in Figure 9, precision of the PRL method got increased or no changes as 'threshold' increased from 0.1 to 0.4, while got decreased and finally got to 0 as 'threshold' increased from 0.5 to 1. As 'threshold' increased from 0.1 to 1, recall decreased. And as 'threshold' increased from 0.1 to 0.4, $F_{0.5}$ got increased and got the best when 'threshold'=0.4. While it increased from 0.5 to 1, $F_{0.5}$ got decreased to 0 finally.



(a) Precision (b) Recall (c) $F_{0.5}$

Figure 9. The semi-supervised learning method for the *sBN*.



(a) Precision (b) Recall (c) $F_{0.5}$

Figure 10. The semi-supervised learning method for the sBN.

Precision, recall and $F_{0.5}$ of the rule-based method shown in Figure 9 never changed. The results show that, our approach achieved the best precision and $F_{0.5}$.

To compare the semi-supervised and direct learning methods, Plan 4 and Plan 5 were executed. Figure 10 shows the results, in which the mean precision of the direct learning method was always higher than that of the semi-supervised learning. Because training data for the direct learning is of good quality with no wrong links. The mean recall of our method was much higher than recall of the direct learning method. However, as the improvement of recall was much bigger than the decrease of precision, $F_{0.5}$ of our method was higher than that of the direct learning.

We chose a case that takes a certain pair of the training and testing data and 'threshold'=0.7 as an example process of the semi-supervised learning method. Figure 11 shows variations of precision, recall and $F_{0.5}$ of iterations (total 7 iterations). As the iteration forwarded, recall got increased, while precision decreased.

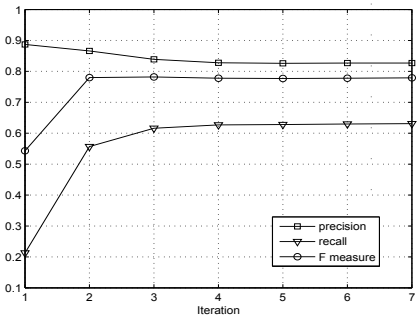


Figure 11. Precision, Recall and $F_{0.5}$ changes of iterations.

VII. CONCLUSION

We propose an approach that learns an sBN using the semi-supervised learning method to do link prediction. An sBN extends Bayesian networks to process relationships and semantic graph structure-based attributes. A semantic subgraph template is defined to describe a semantic graph structure-based attribute. And, a semi-supervised learning method is used to improve the performance. The approach was executed on a data set composed of 100 applications and 3077 links. The results demonstrate that the approach outperformed the PRL and rule-based methods. The results show that the semi-supervised learning method made big improvement in recall and $F_{0.5}$, compared with the direct learning method.

In the future, three directions of work need to be proceeded. One is to continuously leverage probabilistic learning on Semantic Web. More probabilistic models can be enabled working well on Semantic Web. The second one is to learn the dependency structure of a sBN instead of manually defining it. The third one is to do mashup recommendation, taking advantages of the constructed mashup network that can inspire users with creativity to build more creative and interesting mashups.

ACKNOWLEDGEMENTS

This work is funded by NSFC61070156, 2009QNA5025, 2010QNA5044 and the IBM-ZJU Joint research projects.

REFERENCES

[1] Declan, B., Mashups mix data into grobal service. *Nature.* 439, pp. 6-7, 2006.

[2] Yahoo! Pipes. http://pipes.yahoo.com/

[3] Mashup Center. http://www.ibm.com/software/info/mashup-center/

[4] Mash Maker. http://mashmaker.intel.com/

[5] Bin, L., Zhaohui, W., Yuan, N., Guotong, X., Chunying, Z., Huajun, C., sMash: semantic-based mashup navigation for data API network, In Proceedings of World Wide Web 2009 (WWW'09). 2009.

[6] Tim Berners-Lee, James, H. and Ora, L., The Semantic Web, Scientific American Magazine, 2001.

[7] Vapnik, V., The Nature of Statistical Learning Theory. Springer Press. ISBN: 978-0-387-98780-4. 2000.

[8] Microformat specifications, *http://microformats.org*, 2009.

[9] Nir, F., Dan, G. and Moises, G. Bayesian Network Classifiers, Machine Learning, v. 29, n(2-3), 131-163. 1997.

[10] SPARQL, http://www.w3.org/TR/rdf-sparql-query/ , 2008.

[11] Ohad, G., Tova, M., Neoklis, P., Autocompletion for Mashups, in Proceedings of 35th International Conference on Very Large Data Bases (VLDB'09), pp. 538-549, 2009.

[12] J. O'Madadhain, J. Hutchins, and P. Smyth. Prediction, Prediction and ranking algorithms for even-based network data., SIGKDD Explorations, 7(2), 2005.

[13] M. Craven, D. DiPasquo, D. Freitag, A. McCallum, T. Mitchell, K. Nigam, and S. Slattery., Learning to construct knowledge bases from the world wide web., Artificial Intelligence, 118(1\{2):69114\}, 2000.

[14] B. Taskar, M.-F.Wong, P. Abbeel, and D. Koller., Link prediction in relational data.," In Proceedings of Neural Information Processing Systems Conference, 2003.

[15] Kamvar, S.D., Haveliwala, T.H., Manning, C.D., & Golub, G.H. (2003). Exploiting the block structure of the Web for computing PageRank (Technical Report). Stanford, CA: Stanford University.

[16] Barabási, A.L., Jeong, H., Néda, Z., Ravasz, E., Schubert, A., & Vicsek, T. (2002). Evolution of the social network of scientific collaboration. Physica A, 311(3– 4), 590–614.

[17] Ding, Y., Foo, S., Chowdhury, G.. A bibliometric analysis of collaboration in the field of information retrieval. International Information and Library Review, p.367–376. 1999

[18] Grossman, J.W. (2002). The evolution of the mathematical research collaboration graph. Congressus Numerantium, 158, 201–212. 2002.

[19] Lise, G., Nir, F., Daphne, K., Benjamin, T., Learning Probabilistic Models of Relational Structure, In Proceedings of International Conference on Machine Learning (ICML'01), pp. 170-177, 2001.

[20] Benjamin, T., Carlos, G., Daphne, K., Max-Margin Markov Networks, In *Proceedings Neural Information Processing Systems (NIPS'03),* 2003.

[21] Christoph, K., Abraham, B., Jonas, T., Mining Software Repositories with iSPAROL and a Software Evolution Ontology, In *Proceedings of International Workshop on Mining Software Repositories (MSR'07)*, 2007.