The VLDB Journal
DOI 10.1007/s00778-010-0198-2

REGULAR PAPER

Mining learning-dependency between knowledge units from text

Jun Liu · Lu Jiang · Zhaohui Wu · Qinghua Zheng · Yanan Qian

Received: 16 June 2009 / Revised: 7 June 2010 / Accepted: 14 June 2010

Abstract Identifying learning-dependency among the knowledge units (KU) is a preliminary requirement of navigation learning. Methods based on link mining lack the ability of discovering such dependencies among knowledge units that are arranged in a linear way in the text. In this paper, we propose a method of mining the learning-dependencies among the KU from text document. This method is based on two features that we found and studied from the KU and the learning-dependencies among them. They are the distributional asymmetry of the domain terms and the local nature of the learning-dependency, respectively. Our method consists of three stages, (1) Build document association relationship by calculating the distributional asymmetry of the domain terms. (2) Generate the candidate KU-pairs by measuring the locality of the dependencies. (3) Use classification algorithm to identify the learning-dependency between KU-pairs. Our experimental results show that our method extracts the learning-dependency efficiently and reduces the computational complexity.

Keywords Knowledge unit · Learning-dependency · Text · Locality

J. Liu (✉) · L. Jiang · Z. Wu · Q. Zheng · Y. Qian
Department of Computer Science and Technology,
Xi'an Jiaotong University, 710049 Xi'an,
People's Republic of China
e-mail: liukeen@mail.xjtu.edu.cn

L. Jiang
e-mail: roadjiang@yahoo.com

Z. Wu
e-mail: wzh@stu.xjtu.edu.cn

Q. Zheng
e-mail: qhzheng@mail.xjtu.edu.cn

Y. Qian
e-mail: yaya213@gmail.com

J. Liu · L. Jiang · Z. Wu · Q. Zheng · Y. Qian
MOE KLINNS Lab and SKLMS Lab,
Xi'an Jiaotong University, 710049 Xi'an,
People's Republic of China

1 Introduction

Learning is an incremental process that depends on prior knowledge [1]. So learning can be considered a hierarchical process because understanding a new knowledge unit (KU) often relies on the understanding of other existing knowledge units [1,2]. The KU is defined as the smallest integral knowledge object in a given domain, such as definition, theorem, rule, or algorithm [3]. For instance, in "Geometry", there exists a learning-dependency from the KU "*Definition of triangle*" to the KU "*Theorem of angle sum of triangle*", which indicates that a person should learn "*Definition of triangle*" before learning the "*Theorem of angle sum of triangle*".

By visualizing the learning-dependencies among the KU, we can provide learners with a navigation learning path. This navigation learning not only helps learners establish their hierarchical knowledge structure, but also effectively lets them avoid the disorientation problem in learning [4]. So the cognitive load of individuals can be reduced [5]. However, the KU are often embedded in a linear way in a variety of semi-structured or unstructured texts such as TXT, DOC and HTML. These documents usually do not provide a well-established learning-dependency map of the KU. Manually annotating the potential learning-dependencies is not a good solution to this problem, because it is very time-consuming, and requires that the annotators be domain experts.

Published online: 07 July 2010

This paper proposes a method of mining the learning-dependency among the KU. Our method is based on two features of the learning-dependency that we found in our training corpus: the locality of the learning-dependency and the distributional asymmetry of the domain terms. Specifically, the method consists of three steps: first, the text documents are clustered; second, candidate KU pairs, which are the pairs that may potentially have learning-dependencies, are generated among those documents in the same cluster, and finally, the candidate KU pairs are classified by a binary classifier to recognize learning-dependencies. According to the experimental evaluations, the method can efficiently recognize the learning-dependencies among KU and overcome the quadratic problem of link mining.

The rest of the paper is organized as follows. Section 2 discusses related work. Section 3 defines the learning-dependency mining problem. Section 4 studies the two features of the learning-dependency among the KU. Section 5 describes the learning-dependency mining method. Experimental results are discussed in Sect. 6. Section 7 presents the conclusions and our future research.

2 Related work

To the best of our knowledge, there has been no previous work on mining the relationship among KU. Many researchers have been engaged in mining other types of relations in text. For example, Ontology Learning or KAT (Knowledge Acquisition from Text) aims to extract concepts of a specific domain and taxonomic relationship between these concepts from text [6,7], while the goal of RDC (Relation Detection and Characterization) is to identify relationship between named entities [8]. Usually, there are three types of methods for relationship mining, including template-based, clustering-based and classification-based methods.

Template-based methods attempt to discover the relationship between entities by studying frequently used language patterns. For instance, Timothy and Patrick recognized five types of semantic relationships between verbs using non-ambiguous language templates [9]. Their method works well for some types of semantic relationship. However, it is very difficult to define templates in their algorithm, which also has the limitation of domain dependency and poor scalability.

Clustering-based methods, such as [10], try to cluster entities based on their semantic similarity distances before recognizing the relationship of the entities within the same cluster. The computational cost is reduced substantially in this way, compared to the template-based methods. But these methods simply manage to obtain the anonymous relations, because the semantic similarity distance cannot reflect the types of the relationship.

Classification-based methods employ grammar rules and statistical models in their framework. The classifier is built to identify the semantic relationship. For example, Fleischman and Hovy recognized hyponymy relationship by using a decision tree, in which each relationship pair is represented in terms of its templates and adjacent features [11]. Zhou et al. [12] proposed an entity relationship recognition algorithm based on Support Vector Machine.

Previous work enlightens us on mining learning-dependency between KU. However, due to the nature of this problem, we still face two major issues:

1. KU expressed in natural language are ambiguous or ill-formed [13]. Moreover, the structures of KU are far more complex than those of the concepts and named entities, resulting in difficulties in feature extraction and representation.
2. Unlike the relationships between concepts or between named entities, the learning-dependencies among the KU have the characteristic of long distance dependency, indicating that related KU are usually distributed in different paragraphs of a document, or even in different documents. Consequently, mining the relationship among the KU requires detecting the KU-pairs in a larger context. This leads to the quadratic problem of link mining [14], in which the complexity of mining the KU relationship is proportional to the square of the number of the KU.

Because of these characteristics, the previous work on concept and named entities relation mining can hardly be applied to mine the learning-dependencies. To resolve this dilemma, we propose a novel method that utilizes the features of the learning-dependency.

3 Problem formulation

We now formalize the problem of KU learning-dependency mining.

Given a text document set $T = \{t_k\}(1 \leq k \leq m)$ and a KU set $U = \{u_i\}(1 \leq i \leq n)$ extracted from T as the input, the learning-dependency mining process will output a learning-dependency set $A \subseteq U \times U$.

Each $u_i \in U$ can be further represented as a triplet of (*name*, *type*, *content*), where *name* denotes the name of u_i, such as "*definition of subnet mask*"; *type* denotes the semantic type of u_i, such as definition, property or method; and *content* represents the text content of u_i. A pair of $(u_i, u_j) \in A$ if and only if u_i is an immediate precursor of u_j.

Some examples for KU and their learning-dependencies are shown in Table 1. The column "precursor" denotes the ID of a knowledge unit's precursor. For example, from the

Table 1 Examples for knowledge unit and learning-dependence

ID	Name	Type	Content	Precursor
1	Definition of interior angle	Definition	An angle located within a closed figure	–
2	Definition of triangle	Definition	A closed figure with exactly three sides	–
3	Definition of exterior angle	Definition	An exterior angle is the angle formed when one side is extended beyond its adjacent sides	–
4	Theorem of angles sum of triangle	Theorem	The sum of the measures of the interior angles of a triangle is 180°	1, 2
5	Theorem of triangle exterior angle	Theorem	The measure of each exterior angle of a triangle equals the sum of the measures of its two remote interior angles	1, 2, 3, 4

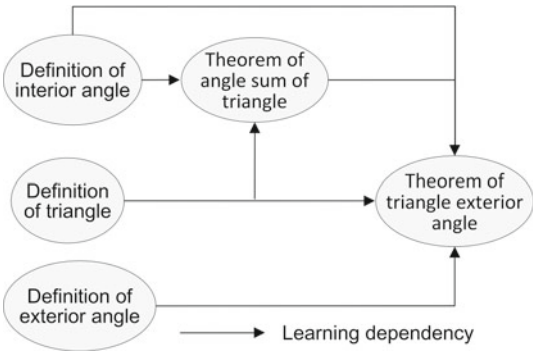

Fig. 1 Knowledge units and their learning-dependencies of the given example

table, we can learn that "*Definition of interior angle*" is a precursor of "*Theorem of angles sum of triangle*" and "*Theorem of triangle exterior angle*". The knowledge units and their learning-dependencies in these examples together form a structure of the knowledge shown as Fig. 1.

4 Feature analysis of the learning-dependency between knowledge units

Through the observation and analysis of the learning-dependency in our sample data, we found some helpful features for mining the learning-dependency. First, for almost every pair of "associated" documents between which there are learning-dependencies, the distribution of domain term frequencies is asymmetrical between the documents. Second, two KU that have a learning-dependency are often close to each other in distance. By quantitative analysis, we also found that the empirical probability of a learning-dependency between two KU decreases exponentially in speed as distance between the KU increases. These two features will be studied in the following two subsections respectively.

4.1 Distributional asymmetry of the domain terms

Usually, a KU is named as the "*definition/case/property of A*", where "*A*" is a term in the given domain. Here, we call "*A*" the core term of the KU. Intuitively, if u_i is a precursor of u_j, it is more likely that u_i's core term appears in u_j than u_j's core term appears in u_i since u_i's core is likely to be cited or refined in u_j. Table 1 shows some good examples. Taking the learning-dependency pair ('*Definition of interior angle*', '*Theorem of angles sum of triangle*') as an example, we can see the core term "*interior angle*" appears in the content of '*angles sum of triangle*', but "*Triangle Interior Angles Sum*" is not present in the content of '*Definition of interior angle*'.

Let C_i be the set of core terms of the KU in a document t_i. We define the average term frequency of C_i in the document t_j as $f(C_i, t_j) = \sum_{c_k \in C_i} tf(c_k.t_j)/|C_i|$, where c_k represents the kth term in C_i, and $tf(c_k, t_j)$ denotes the term frequency of c_k in t_j.

To study our sample data on computer networks, we manually identified pairs of "associated" documents. That is, we identified document pairs (t_i, t_j) for which t_i is clearly a precursor of t_j. We then looked at statistical properties of the associated pairs and found the asymmetric distribution of the core terms: if KU in t_i are precursors of KU in t_j, then $f(C_j, t_i) < f(C_i, t_j)$. The result is shown in Fig. 2 in which the x-axis indicates the text document's ID, and the y-axis represents the value of $F(i) = f(C_{i+1}, t_i)/f(C_i, t_{i+1})$. In the figure, each point corresponds to an F value of an associated document pair, e.g. point i corresponds to the F value of the associated document pair t_i and t_{i+1}. By observing the figure, we can see that the majority of points are located below a threshold line. E.g. for a large majority of associated documents t_i and t_j, with t_i is a precursor of t_j, we have $f(C_j, t_i)/f(C_i, t_j) < F_0$ for some threshold F_0 (with $F_0 < 1$). In the figure, we choose F_0 to be such that 90% of the points fall below the threshold. Indeed, we can use F_0 to discriminate between associated and non-associated documents. In more detail, note that, if t_j is

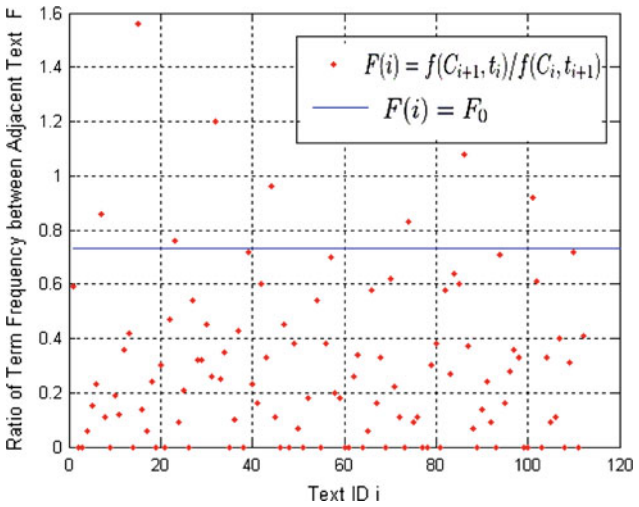

Fig. 2 Ratio of term frequency

a precursor of t_i, then typically $f(C_j, t_i)/f(C_i, t_j) > 1/F_0$. It follows that if $f(C_j, t_i)/f(C_i, t_j) \in [F_0, 1/F_0]$ for a pair of documents t_i and t_j, then we would expect these documents to be non-associated; otherwise, we would expect the documents to be associated.

We draw Fig. 3 to study the distributional asymmetry of the domain terms in both the associated documents and the non-associated documents. We obtain two collections of documents from two textbooks on "Computer Networks" and "Computer Organization and Architecture", respectively. Each document corresponds to a chapter in the textbook. In each figure, the x-axis and y-axis represent the document ID (chapter number) in the textbook. As the textbooks are systematical, it is reasonable to assume that the documents are associated in their natural sequence. The brightness of a block (x, y) represents its F value, i.e. the $f(C_y, t_x)/f(C_x, t_y)$ (the values were normalized to ensure they lie within the range of 0–3). The bright blocks sit in the lower right section of $(y \leq x)$, which suggests that the F value is capable of indicating the natural order of the documents. With a threshold of $1/F_0$, say 1.5, and let document y be the precursor of document x ($y < x$ for short), if the F value of block (x, y) is larger than the threshold, we can obtain the following correct sequence: in the left figure $1 < 2, 2 < 3, 1 < 4, 2 < 4, 3 < 4, 2 < 5, 3 < 5, 1 < 6, 4 < 6, 5 < 6, 1 < 7, 2 < 7, 3 < 7, 4 < 7, 5 < 7, 5 < 8, 6 < 8, 2 < 9, 3 < 9, 4 < 9, 5 < 9, 7 < 9, 8 < 9$; in the right figure $1 < 2, 3 < 5, 1 < 6, 3 < 6, 4 < 6, 5 < 6, 2 < 7, 6 < 7, 1 < 8, 2 < 8, 3 < 8, 4 < 8, 6 < 8$.

4.2 Locality of learning-dependency

Using the KU extraction method we proposed in [15], we acquired 4,318 KU from 336 text documents in the "Computer Networks" domain and "Advanced Mathematics" domain. Among these KU, 5,404 learning-dependencies were labeled manually. Based on the labeled sets, we counted the learning-dependencies under different distances in the two domains respectively. Figure 4 shows the distribution of the number of the learning-dependencies under different distances. The x-axis d denotes the distance of two KU that have learning-dependency. The y-axis s denotes the ratio of learning-dependencies under different distances.

The distance of a learning-dependency pair (u_i, u_j) can be calculated by the following rules:

(1) If u_i and u_j are in the same document, their distance d_{ij} is denoted as $d_{ij} = |j' - i'|$, where i' and j' are the ordinal IDs of u_i and u_j in the document respectively.

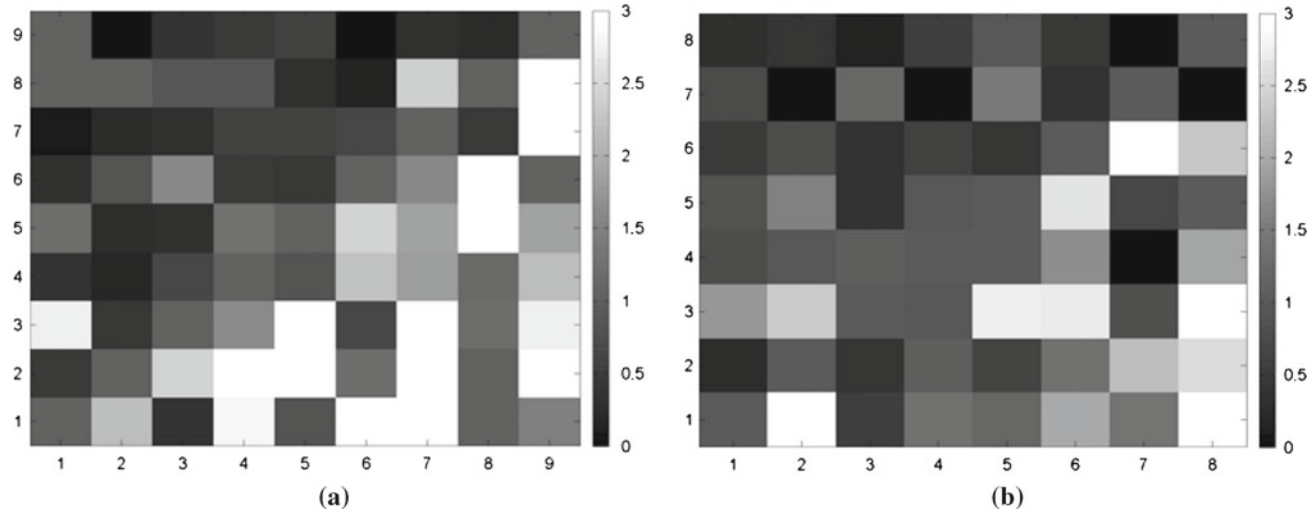

Fig. 3 F value distribution in documents of textbooks. **a** Computer Networks. **b** Computer Organization and Architecture

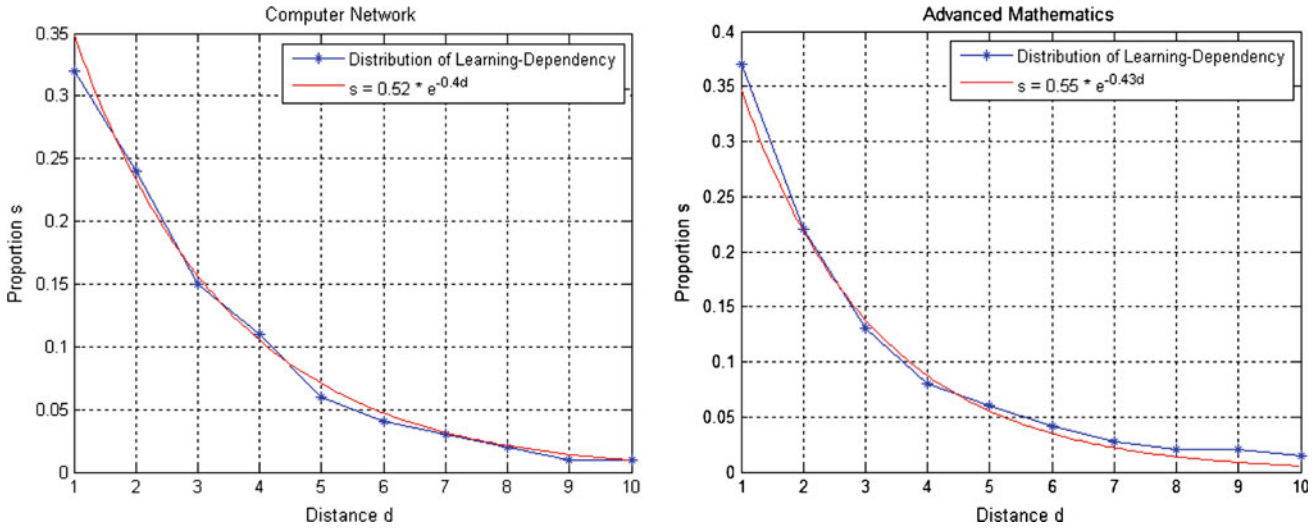

Fig. 4 Distribution of learning-dependency under various distances

(2) If u_i and u_j are in two associated document t_a and t_b, their distance d_{ij} is denoted as $d_{ij} = j' - i' + n_a$, where i' and j' are the ordinal IDs of u_i in t_a, and u_j in t_b, respectively. n_a is the number of KU in t_a.

Let s_d denote the proportion of learning-dependencies of distance d. By curve fitting, we find that s_d is inversely proportional to the exponential function of d, that is, $s_d \propto e^{-\beta d}$. Here, $\beta > 0$ is the distribution coefficient of the learning-dependency pairs. The smaller β is, the more learning-dependencies will be distributed between closer KU, and the more sharply the curve will fall as the distance d getting longer. The distribution coefficient is domain dependent. In Fig. 4, the coefficients of "Computer Network" and "Advanced Mathematics" are 0.4 and 0.43, respectively.

We name this feature the locality of learning-dependency, since almost all learning-dependencies are located in the KU from the same document, or from the documents with similar topics. In our experimental data sets on computer science, the largest d is 50. Thus, when mining the learning-dependency, we can set the longest acceptable distance between two elements in a candidate KU-pair. If there are a total of n KU, we set the largest d to be k. Usually, k is much smaller than n, so the candidate pairs of learning-dependency can be reduced from $O(n^2)$ to $O(n)$, and it will significantly help improve the mining efficiency.

5 Mining process

According to the locality of the learning-dependency studied in Sect. 4.2, the learning-dependency mainly exists among the KU from the same text document, or from the documents of similar topics. Hence, we only need to find the learning-dependencies in these documents, namely associated documents. The mining process is shown in Fig. 5, which is divided into three phases: first, find the text documents having similar topics through text clustering and the distributional asymmetry of the term frequency, i.e., text association mining; second, generate the candidate KU-pairs according to the locality of the learning-dependency in these documents; finally, build a binary classifier to recognize the learning-dependency based on the features of term frequency, distance and semantic type of candidate KU-pairs.

5.1 Text association mining

The input of our mining method is a text document set T of a specific domain, which lacks explicit association. Thus, text association mining aims at finding the documents of similar topics, and then ranks them in pairs in accordance with learning-dependencies of KU in them.

Let R be the set of text document associations from T, $(t_i, t_j) \in R$ represents a pair of associated documents of t_i and t_j. R is initialized as an empty set ϕ. First, each text document in T is converted to a term vector based on VSM (Vector Space Model), and the distance of two documents is computed by Euclidean distance; and then the hierarchical clustering algorithm AGNES is applied [16]. The clustering process deals with three cases:

(1) Two documents t_i and t_j are put into one cluster;
(2) A document t_j is put into the cluster S (assume t_i in S is closest to t_j);
(3) Cluster S and cluster merge S' into a new cluster (assume t_i in S and t_j in S' is the closest document pair in S and S').

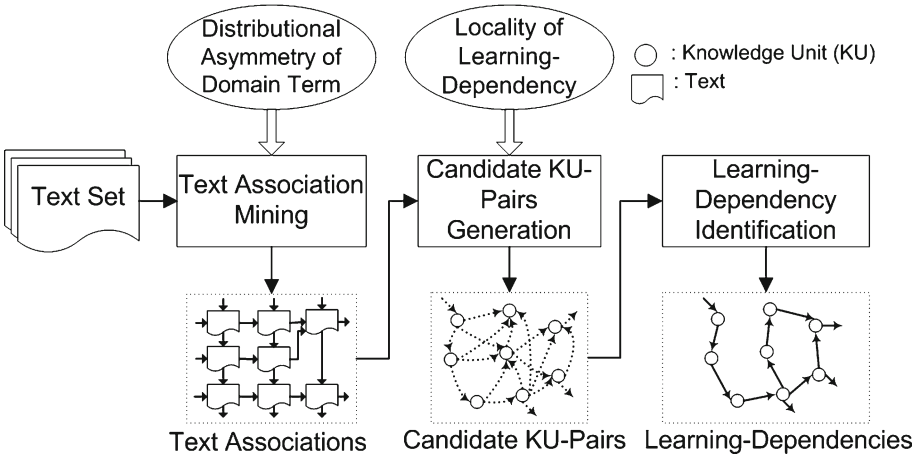

Fig. 5 Mining learning-dependency between knowledge units

For each pair (t_i, t_j) that falls into any of the cases, we need to set a proper threshold $F_0 (F_0 < 1)$, and establish the association of documents according to the feature of distributional asymmetry of the terms.

If $f(C_j, t_i)/f(C_i, t_j) < F_0$, $R = R \cup \{(t_i, t_j)\}$;
If $f(C_j, t_i)/f(C_i, t_j) > 1/F_0$, $R = R \cup \{(t_j, t_i)\}$;
If $f(C_j, t_i)/f(C_i, t_j) \in [F_0, 1/F_0]$, t_i and t_j are not associated with each other.

Once the clustering is finished, the directed graph $< T, R >$ is also generated.

5.2 Candidate KU-pairs generation

According to the locality nature of the learning-dependency, the learning-dependency can be mined in the associated text documents or in a group of text clusters. For each node t_i in the graph $\langle T, R \rangle$ that we have produced in Sect. 5.1, the corresponding text cluster can be defined as $T_i = \{t_i\} \cup \{t_j | (t_i, t_j) \in R\}$. For each KU, we only need to find whether it has the learning-dependency with other KU in the text cluster. We don't need to scan through all the KU in the whole document set.

The next step is to find the candidate KU-pairs in the text cluster. A candidate KU-pair is defined as two KU that may have a learning-dependency. The candidate KU-pair set in text cluster T_i can be divided into two subsets:

(1) The candidate KU-pair set in t_i, denoted by A_i. Let U_i be the KU set of t_i, then $A_i = \{(u_{ix}, u_{iy}) | u_{ix}, u_{iy} \in U_i \wedge x < y \wedge r(u_{ix}, u_{iy})\}$, where x and y are ID of u_{ix} and u_{iy} in t_i, and $r(u_{ix}, u_{iy})$ represents other optional rules that u_{ix} and u_{iy} must obey.
(2) Set A'_i where a KU-pair is partnered by a knowledge unit in t_i and a KU in another text document. Let U'_i be the KU set of other text documents from $T_i - \{t_i\}$, then $A'_i = \{(u_{ix}, u_{iy}) | u_{ix} \in U_i \wedge u_{iy} \in U'_i \wedge r(u_{ix}, u_{iy})\}$, where x and y are ID of u_{ix} and u_{iy} respectively, and $r(u_{ix}, u_{iy})$ represents other optional rules that u_{ix} and u_{iy} must obey.

To sum up, the candidate KU-pair set of a text cluster T_i is $A_i \cup A'_i$. Therefore, the candidate KU-pair set of the whole text document set can be presented as $A_{cand} = \bigcup_{t_i \in T} (A_i \cup A'_i)$.

Assume the number of KU in the text document t_i is n_i, and the number of KU in a text document has an upper limit of $n_{\max}$. It can be deduced that $|A_{cand}| \leq n_{\max} \sum_{t_i \in T} n_i$, where $\sum_{t_i \in T} n_i$ indicates the total number of KU in the whole document set. If we do not make use of the locality feature of the learning-dependency to filter out the un-qualified KU-pairs, the number of candidate KU-pairs will reach $\sum_{t_i \in T} n_i \cdot \left(\sum_{t_i \in T} n_i - 1\right)$. Usually, $n_{\max}$ is far smaller than the total number of KU; as a result, our method will reduce the time complexity of the learning-dependency mining from quadratic to linear in terms of the number of knowledge units. Furthermore, we specify $r(u_{ix}, u_{iy})$ in both A_i and A'_i, which is true if u_{ix} and u_{iy} have a common term, otherwise false. Our experiments show this rule can rule out 60–70% of unqualified learning-dependency pairs, but result in an insignificant loss to the recall of learning-dependency pairs.

5.3 Feature selection for learning-dependency recognition

The remaining issue now is to recognize the learning-dependencies of the given candidate pairs. Up to this point, we have analyzed the features of the learning-dependencies and introduced a binary classification algorithm to recognize the potential learning-dependencies. According to experimental observation and analysis, there are three useful features, namely term frequency, distance and semantic type. For the convenience of discussion, the candidate pair of learning-dependency is denoted as (u_f, u_b) where u_f and are u_b

two KU. We use the following example to illustrate these features.

u_f (Definition of IP Address): *an Internet Protocol (IP) address is a numerical identification and logical address that is assigned to devices participating in a computer network utilizing the Internet Protocol for communication between its nodes*.

u_b (Definition of Subnet Mask): *Subnet mask is the sequence of leading bits of an IP address that precede the portion of the address used as host identifier in IPv4 networks*.

According to the core term definition in Sect. 4.2, the core term of u_f is "*Internet Protocol*" or "*IP*", and the core term of u_b is "*subnet mask*".

(1) Term frequency feature

The term frequency feature F_{fb} represents the asymmetry of term distribution; the feature can be computed from Eq. (1).

$$F_{fb} = \frac{F_f}{F_f + F_b} \tag{1}$$

where F_f denotes the frequency of the core terms of u_f that appear in the content of u_b. Similarly, F_b denotes the frequency of the core terms of u_b that appear in the content of u_f. In the above example, $F_{fb} = 1$ and $F_{bf} = 0$. Given a candidate pair (u_f, u_b), obviously $F_{fb} \in [0, 1]$ (Here we suppose $F_{fb} = 0$ if $F_f = 0$ and $F_b = 0$). The greater the F_{fb} is, the more likely that has (u_f, u_b) learning-dependency.

Different terms, e.g., *IP* and *Internet Protocol*, may refer to the same concept. In practice, a core term should first be looked up in a thesaurus to find out its synonyms, and two core terms are matched if and only if they share at least a common synonym. The quality of the thesaurus may affect the performance of our method. Nevertheless, the impact seems to be less obvious as the synonyms in our data sets are rare. In this work, we establish the thesaurus with the following two steps: first, we had annotators manually annotating all core terms in our data sets, which were later inspected by different annotators for a cross check; and then, we semi-automatically retrieved all synonyms of the annotated terms from WordNet [17].

(2) Distance feature

The distance feature D_{fb} represents the distance between u_f and u_b. According to the locality of the learning-dependency, this feature can be defined as:

$$D_{fb} = e^{-\beta d_{fb}} \tag{2}$$

where d_{fb} is the distance between u_f and u_b. D_{fb} indicates that the possibility of the learning-dependency existing in (u_f, u_b) decays exponentially as d_{fb} grows. The value of β can be estimated from the training set. However, our sensitivity analysis shows that changing β in the range of [0.3, 1.5] has little impact on our learning-dependency mining results.

(3) Semantic type feature

The KU have their semantic types, which may be classified into eight types: definition, property, instance, case, method, classification, distinction and evolution [15]. The semantic type of a knowledge unit usually indicates what kind of information it intends to deliver. According to our statistical analysis of the annotated learning-dependencies corpus on computer science, we found that:

1. $$KP_{\max} = \begin{Bmatrix} (definition, attribute), \\ (definition, classification) \\ (definition, method), (definition, case), \\ (instance, method) \end{Bmatrix}$$

   is a set containing the five most frequently used patterns of learning-dependency. These five patterns occupy 15.9, 12.5, 12.5, 10.5 and 9.1% of the total corpus, respectively.

2. $$KP_{\min} = \begin{Bmatrix} (distinction, instance), \\ (case, definition) \\ (case, classification), (instance, attribute), \\ (distinction, method) \end{Bmatrix}$$

   is the set containing the five least frequently used patterns of learning-dependency. These five patterns occupy 2.1, 1.3, 1.0, 0.3 and 0.1% of the total corpus, respectively. Frequently used patterns may vary in terms of domains. For a new domain, we can retrieve these patterns by sampling in training set.

These pattern frequencies show that the learning-dependency tends to exist in the pairs that have certain combinations of types. We use KP_{fb} to measure this feature:

$$KP_{fb} = \begin{cases} 1 & (u_f, u_b) \in KP_{\max} \\ -1 & (u_f, u_b) \in KP_{\min} \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

In the previous example, since the semantic types are both *definition*, so it has a $KP_{fb} = 0$.

6 Experimental evaluation

In order to validate the efficiency of our mining method, we conducted experiments on a set of documents that cover four courses: "Computer Network", "Computer Organization and Architecture", "Database System and Application" and "Geometry". To the best of our knowledge, there is no public data set for mining KU relationship. We created the experimental data set through a two-stage process. First, we generated KU in the given document set by using the method of [15] and manually refined the extracted results. Then, we manually annotated the learning-dependencies among the extracted KU.

The annotating work was conducted as follows: We hired 24 undergraduate students in their junior year or senior year from the computer science department. They were asked to log on to a Web-based annotating system, where they were given the annotation assignments. They were asked to annotate the learning-dependency between the KU by using their own knowledge background and additional textbook resources. The work lasted 6 months. We created the experimental data set that covers the four courses after we double checked the students' work. Sometimes two annotators had different opinions on a specific KU. To minimize the errors caused by the annotators' incorrect judgments, we established a set of rules to guide our work.

Rule 1. If KU A contains a term$T$, then T's definition has a learning-dependency with A.

For example, KU A *(Inscribed angle theorem, theorem, an inscribed angle is half of the central angle that subtends the same arc)* contains the terms *inscribed angle, central angle* and *arc*, so definitions of these terms have learning-dependency
with A.

Rule 2. If KU A is a theorem and its proof needs a reference to theorem B, then B has a learning-dependency with A.

For example, proof of KU A (*Interior angles' sum of triangle theorem, theorem, the sum of the interior angles of any triangle is* 180°) refers to (*Parallel lines' theorem 1, theorem, if a transversal intersects two parallel lines, then the alternate interior angles are congruent*) and (*Parallel lines' theorem 2, theorem, if a transversal intersects two parallel lines, then the corresponding angles are congruent*). Hence, both of the two KUs have learning-dependency with A.

Rule 3. If KU A is a method and its sub procedure is another method B, then B has a learning-dependency with A.

For example, suppose KU A describes the method of creating an inscribed circle of a triangle by ruler-and-compass. It depends on two sub procedures constructing a perpendicular to a line and an angle bisector. Hence, the two KUs have learning-dependency with A.

The number of KU and their relations in the experimental data sets are shown in Table 2.

Table 2 Experimental data sets

ID	Course name	#KUs	#learning-dependencies
1	Computer network	889	758
2	Computer organization and architecture	743	839
3	Database system and application	1,398	1,176
4	Geometry	427	1,325

Table 3 Number of candidate pairs and training instances

ID	#Possible pairs	#Candidate pairs	Retention ratio	#Training samples	
				−	+
1	49,506	1,858	91.9	1,828	620
2	28,392	4,678	94.3	1,477	680
3	195,806	3,219	96.8	2,524	890
4	12,882	2,313	95.2	1,454	704

In our experiments, we selected the approximate first 70% of KU to build the training set and the remaining 30% to build the test set. If the number of KU to be tested is n, the number of all possible pairs (***# possible pairs***) that should be checked for learning-dependencies could reach $n \times (n-1)$. However, our method only requires checking the learning-dependencies in the candidate pairs that are generated by the algorithm in Sect. 5.2.

As shown in Table 3, # ***candidate pairs*** denotes the number of pairs to be checked for learning-dependency. The ***retention ratio*** is the percentage of learning-dependencies preserved in the candidate set. It can be seen that ***#candidate pairs*** is far less than the number of all possible pairs (***#possible pairs***), and ***#candidate pairs*** seems to be proportional to the number of KU. The ***#training samples*** denotes the number of preorder candidate pairs in the training set, in which '+' denotes the number of positive instances, i.e. pairs who have learning-dependencies, and '−' denotes the number of negative instances. Usually, there are much fewer positive instances than negative instances in the corpus, so we duplicated some of positive instances and restricted the number of negative instances to balance the training set.

Four binary classifiers, including Naïve Bayes (NB), Decision Tree (DT), SVM and Multilayer Perceptron (MLP), were employed in the experiments. In the experiments, if the recognized relationship label matched the annotated relationship label, we considered it a correct case. We used *precision*, *recall*, and F_1-score to evaluate the effectiveness of the learning-dependency recognition results. The experimental results are presented in Table 4.

As shown in Table 4, both the precision and recall of negative pairs are above 95%. This is because of two reasons. First,

Table 4 Recognition results

Classifier	Criteria	ID = 1		ID = 2		ID = 3	
		−	+	−	+	−	+
SVM	Precision	99.3	93.3	99.4	61.6	99.1	73.3
	Recall	99.5	89.6	97.2	88.3	95.6	93.5
	F_1-score	99.4	***91.4***	98.3	***72.6***	97.3	***82.2***
DT (C4.5)	Precision	97.6	70.3	99.8	52.4	96.2	81.8
	Recall	98.0	66.4	95.5	95.7	98.0	69.8
	F_1-score	97.8	68.3	97.6	67.7	97.1	75.3
NB	Precision	99.4	60.8	99.7	48.8	97.2	75.5
	Recall	95.7	92.0	94.8	94.8	96.7	77.9
	F_1-score	97.5	73.2	97.2	64.4	96.9	76.7
MLP	Precision	99.5	56.3	99.7	53.3	99.3	70.8
	Recall	94.7	93.6	95.7	95.2	95.0	94.6
	F_1-score	97.1	70.3	97.7	68.3	97.1	81.0

The best F_1-scores are in bold and italic

the existence of the learning-dependency is sparse among the KU, so if we randomly select a pair, it is very likely to be a negative pair. Second, the proposed features successfully capture the characteristics of negative pairs. The precision and recall of the positive pair varies by different classifiers. It turns out that NB performs the worst. This seems to be caused by the fact that the independence assumption of features is not satisfied, and some information is lost during the feature normalization. SVM beats others, especially on small-scale sample sets. Its average $F_1\text{-}score$ for positive instances ranges from 72.6 to 91.4%. Moreover, the execution time of SVM, although slightly more than DT and NB, is far less than that of MLP. So, we recommend SVM binary classifier in solving this learning-dependency mining problem. The parameter settings for the above classifiers are presented in Table 5. In SVM, C and $epsilon$ are two standard parameters [18], and e is the exponent value of the Polynomial Kernel. Please refer to [19,20] for the meaning of other parameters in Table 5.

We analyzed the classification results and observed that the misclassification of positive pairs are mainly caused by omissions in the text of KU, i.e. if a term string has been introduced previously, part of this string, or even the whole string, may be absent from the text of the current KU. E.g. a KU may use the "*predication method*" to refer to the term "*transfer*

Table 5 Parameter settings for the classifiers

Classifier	ID = 1	ID = 2	ID = 3	ID = 4
SVM	$C = 1.0$	$C = 1.0$	$C = 1.0$	$C = 1.0$
	PolyKernel	PolyKernel	PolyKernel	PolyKernel
	$e = 1.7$	$e = 2.0$	$e = 1.0$	$e = 1.0$
	$epsilon = 10^{-12}$	$epsilon = 10^{-12}$	$epsilon = 10^{-12}$	$epsilon = 10^{-12}$
DT (C4.5)	confidence factor = 0.25	Confidence factor = 0.25	Confidence factor = 0.25	Confidence factor = 0.25
	Binary split = *false*	Binary split = *false*	Binary split = *false*	Binary split = *false*
NB	–	–	–	–
MLP	Learning rate = 0.25	Learning rate = 0.25	Learning rate = 0.3	Learning rate = 0.25
	Training time = 500	Training time = 500	Training time = 500	Training time = 500

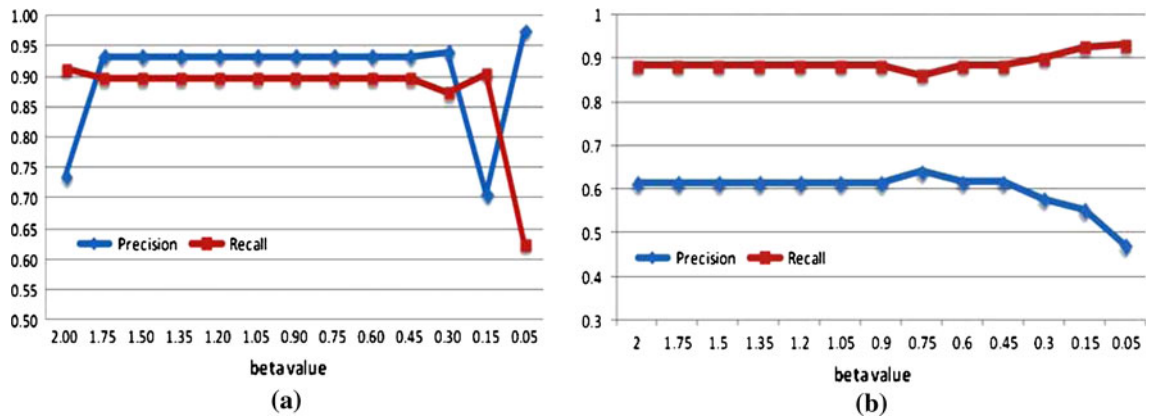

Fig. 6 Impact of β on the precision and recall of the positive instances

Table 6 Performance comparison with baseline

Method	Criteria	ID = 1	ID = 2	ID = 3	ID = 4
Our method	Precision	85.7	58.1	71.0	72.4
	Recall	82.3	83.3	90.5	91.4
	F_1-score	84.0	68.5	79.6	80.8
Baseline	Precision	27.0	28.0	13.0	21.0
	Recall	43.0	20.0	15.0	8.0
	F_1-score	33.1	23.3	13.9	11.6

prediction method" in the previous KU. Consequently, our method can only recognize the explicit and unambiguous learning-dependencies from text corpus.

We also conducted experiments to analyze the impact of the parameter β on the classification result. Figure 6 presents the results where x-axis represents the value of β. It can be seen that the classification results stay stable as β varies within a certain range. The classification results are immune to the changing of β. Based on our experimental observation, we suggest set β to 0.4.

We calculate the overall performance of the method by combining the performance of the candidate pair generation step and the classification step. To validate the effectiveness of our method, we compare it with a baseline method, in which all consecutive KU are classified as true positives, see Table 6. To validate the extensibility of our proposed method, we also tested it in the geometry domain. Compared to the computer science data set, the knowledge in geometry is systematic and the learning-dependency in the geometry data set is much denser. For example, 1,325 learning-dependencies were annotated from 427 knowledge units. In the experiment, we manually annotated the learning-dependencies and trained a SVM classifier on plane geometry. Then, we applied the classifier on solid geometry ("ID = 4" in Table 6).

There is a palpable disparity between the precision and recall of the two methods, and our method outperforms the baseline method. For example, the precision and recall of our method on "Database System and Application" ("ID = 3" in Table 6) is about six times and five times of that of the baseline method, respectively. The experimental results demonstrate that our proposed method

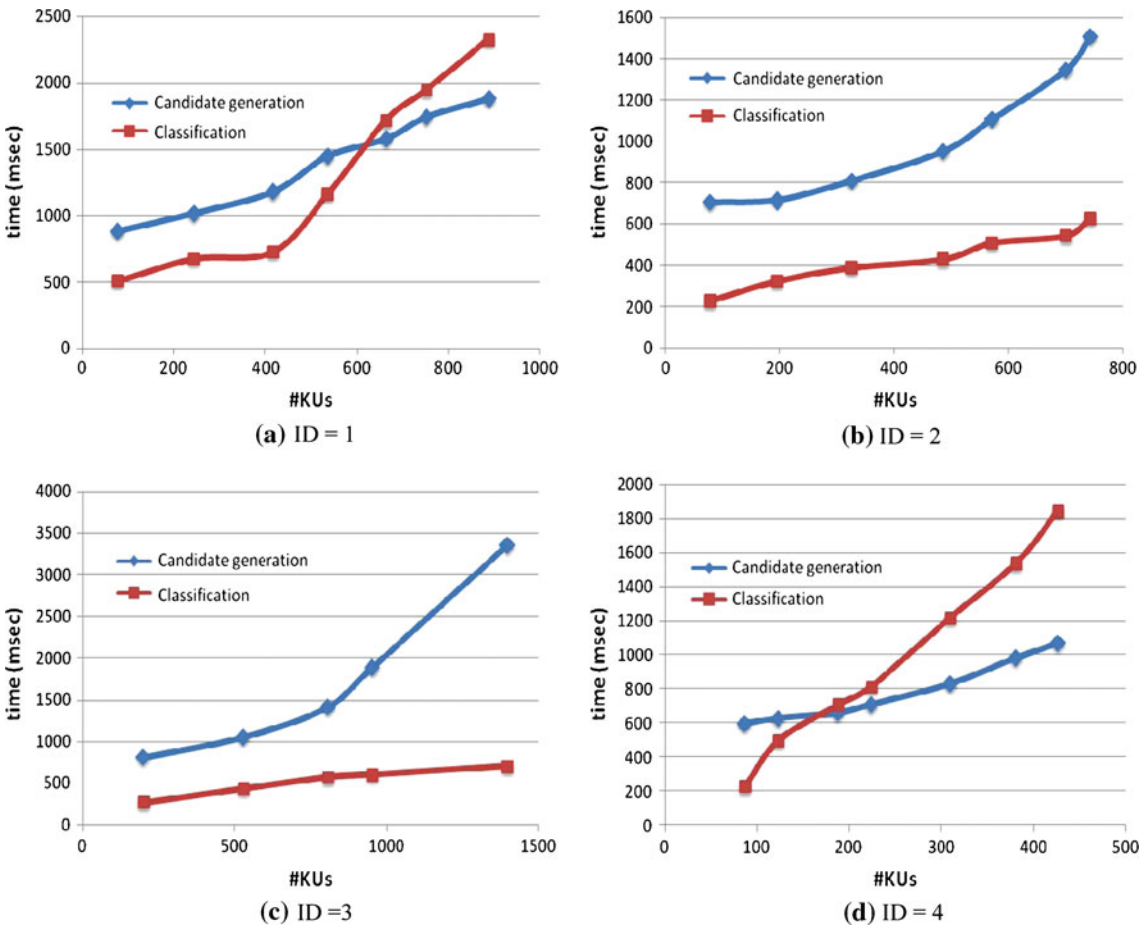

Fig. 7 Running time analysis using varied number of knowledge units. **a** ID = 1, **b** ID = 2, **c** ID = 3, **d** ID = 4

can effectively extract learning-dependencies from the text corpus.

To evaluate the running time of the algorithm as the number of inputted KU increases, we conducted the following experiments. The results are illustrated in Fig. 7 in which x-axis represents the number of KU, and y-axis denotes the running time in milliseconds. The experiments were performed on the Intel Core 2 Duo CPU 2.53 GHz with 2 GB main memory and Microsoft Windows XP Professional. It can be seen that the running time of our method grows approximately linearly as the number of KU increases. This evidence in turn substantiates the claim that our method can reduce the time complexity of the learning-dependency mining from quadratic to linear in terms of the number of KU.

7 Conclusion and future work

Learning-dependency mining is a novel and challenging problem. It is the foundation of knowledge organization and navigation learning. Due to the nature of KU, Ontology learning, KAT and RDC can hardly be applied to mine learning-dependencies. To solve the problem, this paper demonstrates two features of learning-dependency: the distributional asymmetry of the domain terms,the locality of the learning-dependency, and then introduces a classification-based learning-dependency mining method. Experimental results show that our method exhibits satisfactory precisions and recalls.

There are two specific directions for our future work. First, we will investigate the effectiveness of our method in the domains other than "computer science" and "geometry". In addition, we plan to extend the method to mining the learning-dependency residing in an online repository—Wikipedia.

Acknowledgement The research was supported in part by the National High-Tech R&D Program of China under Grant No.2008AA01Z131, the National Science Foundation of China under Grant Nos.60825202, 60803079, 60921003, the National Key Technologies R&D Program of China under Grant Nos. 2006BAK11B02, 2006BAJ07B06, the Program for New Century Excellent Talents in University of China under Grant No.NECT-08-0433, the Doctoral Fund of Ministry of Education of China under Grant No. 20090201110060, Cheung Kong Scholar's Program. The authors are grateful to the anonymous reviewers for their comments, which greatly improved the quality of the paper.

References

1. Gordon, J.L.: Creating knowledge maps by exploiting dependent relationships. Knowl. Based Syst. **13**(2–3), 71–79 (2000)
2. Henze, N., Nejdl, W.: Logically characterizing adaptive educational hypermedia systems. In: Proceedings of International Workshop on Adaptive Hypermedia and Adaptive Web-based Systems (AH2003), pp. 15–28. Budapest, Hungary (2003)
3. Wen, Y.K., Xu, G.H.: Knowledge element linking theory. J. China Soc. Sci. Tech. Inf. **22**(6), 665–670 (2003)
4. Jeschke, S., Richter, T.: Individualization and flexibility through computer algebra systems in virtual laboratories. In: The 8th IEEE International Symposium on Multimedia, Proceedings, vol. 1014, pp. 965–970 (2006)
5. Lin, F.R., Hsueh, C.M.: Knowledge map creation and maintenance for virtual communities of practice. Inf. Process. Manag. **42**(2), 551–568 (2006)
6. Navigli, R., Velardi, P., Gangemi, A.: Ontology learning and its application to automated terminology translation. IEEE Intell. Syst. **18**(1), 22–31 (2003)
7. Sanchez, J., Garcia, R., Breis, J., Bejar, R., Compton, P.: An approach for incremental knowledge acquisition from text. Expert. Syst. Appl. **25**(3), 313–330 (2003)
8. Kambhatla, N.: Combining lexical, syntactic, and semantic features with maximum entropy models for extracting relations. In: Proceedings of the ACL 2004 on Interactive poster and demonstration sessions, 2004, pp. 178–181. Barcelona, Spain (2004)
9. Timothy, C., Patrick, P.: VerbOcean: Mining the web for fine-grained semantic verb relations. In: Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP-04), pp. 33–40. Barcelona, Spain, July 25–26 (2004)
10. Faure, D., Poibeau, T.: First experiments of using semantic knowledge learned by ASIUM for information extraction task using INTEX. In: Proceedings of the Workshop on Ontology Learning, 14th European Conference on Artificial Intelligence (ECAI-2000), pp. 7–12. Berlin, Germany (2000)
11. Michael, F., Eduard, H.: Offline strategies for online question answering: Answering questions before they are asked. In: Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL-03), pp. 1–7. Sapporo, Japan, 9–10 July (2003)
12. Zhou, D., Su, J., Zhang, M.: Modeling commonality among related classes in relation extraction. In: Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (COLING-ACL'2006), pp. 121–128. Sydney, Australia, July (2006)
13. Witbrock, M., Baxter, D., Curtis, J., et al.: An interactive dialogue system for knowledge acquisition in CYC. In: Proceedings of the 18th International Joint Conference on Artificial Intelligence, pp. 138–145. Acapulco, Mexico (2003)
14. Petteri, S., Lauri, E., Petteri, H., Kimmo, K., Hannu, T.: Link discovery in graphs derived from biological databases. In: Data Integration in the Life Sciences, Third International Workshop, pp. 35–49. Hinxton, UK (2006)
15. Chang, X., Zheng, Q.H.: Knowledge element extraction for knowledge-based learning resources organization. In: The 6th International Conference on Web-based Learning, pp. 102–113. Edinburgh, United Kingdom (2007)
16. Kaufman, L., Rousseeuw, P.J.: Finding Groups in Data: An Introduction to Cluster Analysis. John Wiley and Sons, London (1990)
17. Miller, G.A., Beckwith, R., Fellbaum, C.D., Gross, D., Miller, K.: WordNet: An online lexical database. Int. J. Lexicogr. **3**(4), 235–244 (1990)
18. Cortes, C., Vapnik, V.: Support vector networks. Mach. Learn. **20**, 273–297 (1995)
19. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufman, San Mateo, Calif (1993)
20. Gardner, M.W., Dorling, S.R.: Artificial neural networks (The multilayer perceptron)—a review of applications in the atmospheric sciences. Atmos. Environ. **32**(14/15), 2627–2636 (1998)