We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

4,800
Open access books available

122,000
International authors and editors

135M
Downloads

Our authors are among the

154
Countries delivered to

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities

CLARIVATE ANALYTICS BOOK CITATION INDEX INDEXED

WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com

19

Concept Mining and Inner Relationship Discovery from Text

Jiayu Zhou, Shi Wang
School of Computing, Informatics and Decision Systems Engineering, Arizona State University
USA
Institute of Computing Technology, Chinese Academy of Sciences
China

1. Introduction

From the cognitive point of view, knowing concepts is a fundamental ability when human being understands the world. Most concepts can be lexicalized via words in a natural language and are called Lexical Concepts. Currently, there is much interest in knowledge acquisition from text automatically and in which concept extraction, verification, and relationship discovery are the crucial parts (Cao et al., 2002). There are a large range of other applications which can also be benefit from concept acquisition including information retrieval, text classification, and Web searching, etc. (Ramirez & Mattmann, 2004; Zhang et al., 2004; Acquemin & Bourigault, 2000)

Most related efforts in concept mining are centralized in term recognition. The common used approaches are mainly based on linguistic rules (Chen et al., 2003), statistics (Zheng & Lu, 2005; Agirre et al., 2004) or a combination of both (Du et al., 2005; Velardi et al., 2001). In our research, we realize that concepts are not just terms. Terms are domain-specific while concepts are general-purpose. Furthermore, terms are just restricted to several kinds of concepts such as named entities. So even we can benefit a lot from term recognition we cannot use it to learn concepts directly.

Other relevant works in concept mining are focused on concepts extraction from documents. Gelfand has developed a method based on the Semantic Relation Graph to extract concepts from a whole document (Gelfand et al., 1998). Nakata has described a method to index important concepts described in a collection of documents belonging to a group for sharing them (Nakata et al., 1998). A major difference between their works and ours is that we want to learn huge amount of concepts from a large-scale raw corpus efficiently rather than from one or several documents. So the analysis of documents will lead to a very higher time complexity and does not work for our purpose.

There are many types relationships between lexical concepts such as antonymy, meronomy and hyponymy, among which the study of hyponymy relationship has attracted many effort of research because of its wide use. There are three mainstream approaches—the *Symbolic* approach, the *Statistical* approach and the *Hierarchical* approach—to discovery general

hyponymy relations automatically or semi automatically (Du & Li, 2006). The Symbolic approach, depending on lexicon-syntactic patterns, is currently the most popular technique (Hearst, 1992; Liu et al., 2005; Liu et al., 2006; Ando et al., 2003). Hearst (Hearst, 1992) was one of the early researchers to extract hyponymy relations from Grolier's Encyclopedia by matching 4 given lexicon-syntactic patterns, and more importantly, she discussed about extracting lexicon-syntactic patterns by existing hyponymy relations. Liu (Liu et al., 2005; Liu et al., 2006) used the "isa" pattern to extract Chinese hyponymy relations from unstructured Web corpus, and have been proven to have a promising performance. Zhang (Zhang et al., 2007) proposed a method to automatically extract hyponymy from Chinese domain-specific free text by three symbolic learning methods. The statistical approach usually adopts clustering and associative rules. Zelenko et al. (Zelenko et al., 2003) introduced an application of kernel methods to extract two certain kinds of hyponymy relations with promising results, combining Support Vector Machine and Voted Perception learning algorithms. The hierarchical approach is trying to build a hierarchical structure of hyponymy relations. Caraballo (Caraballo, 1999) built a hypernymy hierarchy of nouns via a bottom-up hierarchical clustering technique, which was akin to manually constructed hierarchy in WordNet.

In this paper, we use both linguistic rules and statistical features to learn lexical concepts from raw texts. Firstly, we extract a mass of concept candidates from text using lexico-patterns, and confirm a part of them to be concepts according to their matched patterns. For the other candidates we induce an *Inner-Constructive Model* (CICM) of words which reveal the rules when several words construct concepts through four aspects: (1) parts of speech, (2) syllables, (3) senses, and (4) attributes. Once the large scale concept set is built based on the CICM model, we developed a framework to discover inner relationships within concepts. A lexical hyponym acquisition is proposed based on this framework.

2. Extracting Concepts from Text using Lexico-Patterns

In this research, our goal is to extract huge amount of domain-independent concept candidates. A possible solution is to process the text by Chinese NLU systems firstly and then identity some certain components of a sentence to be concepts. But this method is limited dut to the poor performance of the existing Chinese NLU systems, which still against many challenge at present for Chinese (Zhang & Hao, 2005). So we choose another solution based on lexico-patterns.

2.1 The Lexico-Patterns of Lexical Concepts

Enlightened Hearst's work (Hearst, 1992), we adopt lexico-patterns to learning lexical concepts from texts. But first design a lot of lexico-patterns patterns manually, some of which are shown in Table

ID	Lexical Patterns
1	<?C1><是><一 | ><个 | **种** | ><?C2>
2	<?C1><、><?C2><或者 | 或是 | 以及 | 或 | 等 | 及 | 和 | 与><其他 | 其它 | 其余>
3	<?C1><、><?C2><等等 | 等><?C3>
4	<?C1><如 | 象 | 像><?C2><或者 | 或是 | 或 | 及 | 和 | 与 | 、><?C3>
5	<?C1><、><?C2><是 | **为**><?C3>
6	<?C1><、><?C2><各 | **每** | 之 | **这**><**种** | **类** | 些 | **样** | 流><?C3>
7	<?C1><或者 | 或是 | 或 | 等 | 及 | 和 | 与><其他 | 其它 | 其余><?C2>
8	<?C1><或者 | 或是 | 或 | 及 | 和 | 与><?C2><等等 | 等><?C3>
9	<?C1><中 | 里 | 内 | ><含 | 含有 | 包含 | 包括><?C2>
10	<?C1>由<?C2><**组**成 | 构成>

Table 1. The Lexico-Patterns for Extracting Concepts from Text

Here is an example to show how to extract concepts from text using lexico-patterns:

Example 1. Lexico-Pattern_No_1{
Pattern: < ?C1> <是><一 | > < 个 | **种**> < ?C2>
Restrict Rules:
not_contain(<?C2>,<! **标**
点>)^lengh_greater_than(<?C1>,1)^lengh_greater_than(<?C2>,1)^
lengh_less_than(<?C1>,80)^lengh_less_than(<?C2>,70)^not_end_with(<?C1>,<**这** | 那>)^
not_end_with(<?C2>,<的|而已|例子|**罢**了>)^
not_begin_with(<?C2>,<**这** | 的 | 它 | 他 | 我 | 那 | 你 | 但>)^
not_contain(<?C2>,<**这些** | **那些** | **他们** | **她们** | **你们** | **我们** | **他** | **它** | **她** | **你** | **谁**>)}

Sample sentences and the concepts extracted:

(1) 地球是一个行星，地球会爆炸**吗**？(The earth is a planet, will it blast?) →<?C1>=地球(The earch); <?C2>=行星(a planet)
(2) 很久很久以前地球是一个充**满**生机的星球.(Long long ago the Earth is a planet full of vitality.) →<?C1>=很久很久以前地球(Long long ago the Earth); <?C2>=充**满**生机的星球(a planet full of vitaligy)

How to devise good patterns to get as much concepts as possible? We summarized the following criteria through experiments:
(1) *High accuracy criterion*. Concepts distributing in sentences meet linguistics rules, so each pattern should reflect at least one of these rules properly. We believe that we should know linguistics well firstly if we want create to good patterns.
(2) *High coverage criterion*. We want to get as much concepts as possible. Classifying all

concepts into three groups by their characteristics, (i.e. concepts which describe physical objects, concepts and the concepts which describe time) is a good methodology for designing good patterns to get more concepts.

2.2 Confirming Concepts using Lexico-Patterns

Obviously, not all the chunks we got in section 2.1 are concepts, such as <?C1>=很久很久以前地球(Long long ago the Earth) in Example 1 above. In order to identify concepts from the candidates, we introduce a hypothesis, called Hypothesis 1.

Hypothesis 1. A chunk ck extracted using lexico-patterns in section 2.1 is a concept if (1) {ck} has been matched by sufficient lexico-patterns, or (2) {ck} has been matched sufficient times.

To testify our hypothesis, we randomly draw 10,000 concept candidates from all the chunks and verify them manually. The association between the possibility of a chunk to be a concept and its matched patterns is shown as Fig. 1:

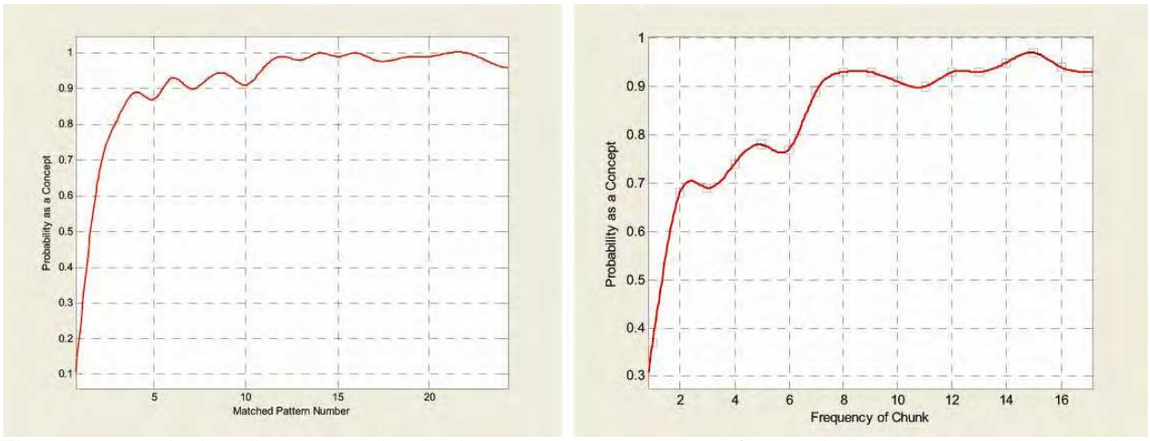


Fig. 1. Association between the lexico-patterns number / the times matched by all the patterns of chunks and their possibility of being concepts}

The left chart indicates our hypothesis that the chunks which matched more patterns are more likely to be concepts and the right chart shows that the frequency of the chunk does work well to tell concepts from candidate chunks too. In our experiments, we take the number of patterns matchings to be 5 and threshold of matching frequency as 14, and single out about 1.22% concepts from all the candidate chunks with a precision rate of 98.5%. While we are satisfied with the accuracy, the recall rate is rather low. So in the next step, we develop CICMs to recognize more concepts from chunks.

3. Learning Concepts using CICM

The CICM is founded on an instinctive hypothesis:
Hypothesis 2. Most lexical concepts obey certain inner constructive rules.
That means, when some words form a concept, each word must play a certain role and has certain features. We develop the hypothesis enlightened mainly from the knowledge of

linguistics (Lu et al., 1992) and the cognitive process of human beings creating lexical concepts (Laurance & Margolis, 1999). Some examples will be given to illuminate the Hypothesis 2 after present the definition of CICM.

3.1 Definition of CICM

According to Hypothesis 2, we can tell whether an unknown chunk is a concept or not by checking whether each word in it whether obeys the CICM. The problems are how to materialize these rules and how to get them. The POS models can reveal these rules using the parts of speech of words but is not precise enough and has many defections (Yu, 2006). To get better performance we probe into the structure of concepts more deeply and find that besides POS, we must ensure each word's more definite role through at least other three aspects.

Definition 1. The word model W=< PS, SY, SE, AT> of a word w is a 4-tuple where (1) PS is all the parts of speech of w; (2) SY is the number of w's syllable; (3) SE is the senses of w in HowNet; and (4) AT is the attributes of w.

The *word models* are integrated information entities to model words. The reason of choosing these four elements listed above will be clarified when we construct CICMs.

Definition 2. Given a concept cpt=w_1… w_{i-1} w_i w_{i+1} …w_n with n words, the C-Vector of the word w_i towards cpt is a n-tuple:

$$\text{C-Vector}(w_i)=< i, W_1,\ldots ,W_{i-1} ,W_{i+1} ,\ldots,W_n > \quad (1)$$

The *C-Vector* of a word stands for one constructive rule when it forms concepts by linking other words and *i* is its position in the concept. A word can have same *C-Vectors* towards many different concepts. The *C-Vector* is the basis of CICM.

Definition 3. The Concept Inner-Constructive Models (CICMs) of a word w is a bag of C-Vectors, in which each C-Vector is produced by a set of concepts contain w.

Essentially, CICMs of words represent the constructive rules when they construct concepts. In the four elements of word models, PS and SY embody the syntactical information which have significant roles when conforming concepts in Chinese (Lu et al., 1992) and are universal for all types of words. SE and AT reveal the semantic information of words and are also indispensably. HowNet is an elaborate semantic lexicon attracted many attentions in many related works (Dong & Dong, 2006). But there are still some words which are missing in it so we need to introduce attributes as a supplement. Attributes can tell the semantic differences at the quantative level or qualitative level between concepts. Tian has developed a practicable approach to acquire attributes from large-scale corpora (Tian, 2007).

ID	C-Vectors	Sample Concepts
1	< 1, W(管理) >	生**产** 管理
2	< 1, W(**许**可**证**) >	生**产** **许**可**证**
3	< 1,W(**实习**),W(**报**告)>	生**产** **实习** **报**告
4	< 2,W(食品)>	食品 生**产**
5	< 2,W(分布式)>	分布式 生**产**
6	< 2,W(国民),W(**总值**)>	国民 **生产** **总值**
7	< 2,W(新疆),W(建**设**),W(兵**团**)>	新疆 生**产** 建**设** 兵**团**
8	< 3,W(广**东**省),W(春耕)>	广**东**省 春耕 生**产**
9	< 3,W(国家),W(安全),W(**监**督),W(管理局)>	国家 安全 生**产** **监**督 管理局
…	…	…

Table 2. CICM of "生**产**"

Note that we omit the details of each word vector for simplicity. Taking ``国民 生**产** **总值**'' for example, the full *C-Vector* is:

< 2,

<{n},2,{属性**值**,**归**属,国,人,国家},{有**组**成,有数量}>,

<{n},2,{数量,多少,**实**体},{有**值**域,是抽象概念}>>

3.2 Learning CICMs

Using CICMs as the inner constructive rules of concepts, our next problem is how to get these models. We use the confirmed concepts obtained in section 2.2 as a training set and learn CICMs hidden in them automatically. It is an instance learning process and the following procedure is implemented for this task:

Algorithm

CICMs Instance Learning Algorithm:

(1) Initializing the resources including (1.1) A words dictionary in which each one has fully parts of speech; (1.2) The HowNet dictionary; and (1.3) An attributes base of words (Tian, 2007).

(2) Constructing a model set MSet to accommodate all the words' models which is empty initially.

(3) For each concept cpt in the training set, segment it and create each word's C-Vector(w_i). Subsequently, if C-Vector(w_i) $\in$ MSet(w_i), then just accumulate the frequency; otherwise add C-Vector(w_i) to MSet(w_i).

(4) Removing the C-Vectors which have low frequency for each word's MSet.

Based on experiments, we choose 10% as the threshold of the number of the concepts containing the word in the training set. We exclude the vectors which have low frequency, that is, if a C-Vector for a word is supported by just a few concepts, we look at it as an exception.

4. Clustering Words for More Efficient Analogy

Essentially, CICMs are models of instance analogy. We want to learn new concepts by ``recalling'' the old ones just as human beings. For example, we can build CICMs for the word ``*生产(produce, production)*'' like Table 2 and then identify that ``*药品 生产(pharmaceutical production)*'' is also a concept, because the latter has the same constructive rule as ``*食品 生产(food production)*''.

But unluckily, even we know ``*药品 生产*'' is a concept, our system still cannot tell whether ``*药品 制造(pharmaceutical manufacture)*'' is also a concept for there are no CICMs for the word ``*制造*''. The reason for this is that the system still cannot make use of word similarity. Therefore, we need to cluster words based on the similarity of CICMs and then learn more new concepts.

4.1 Similarity Measurement of Words

The similarity measurement of CICMs is the basis of clustering words in our task. Our measurement is founded on the intuitive distribute hypothesis that:

Hypothesis 3 In concepts, similar words have similar CICMs.

According to Hypothesis 3, the similarity of two words w_1, w_2 is defined as:

$$\mathrm{sim}(w_1, w_2)=\mathrm{sim}(\mathrm{CICM}(w_1), \mathrm{CICM}(w_2)) \tag{2}$$

The commonly used similarity measure for two sets includes *minimum distance*, *maximum distance*, and *average distance*. Considering that there are still some noises in our training set which would result in some wrong C-Vectors in CICMs, we choose the average distance for it is more stable for noisy data, that is:

$$\begin{aligned}\mathrm{sim}(w1, w2) &= \frac{1}{|CICM(w1)|}\sum_{veci\in CICM(w1)} sim(veci, CICM(w2)) \\ &= \frac{1}{|CICM(w1)||CICM(w2)|}\sum_{veci\in CICM(w1)}\sum_{veci\in CICM(w1)} sim(veci, vecj)\end{aligned} \tag{3}$$

Now the problem is how to calculate the similarity of two C-Vectors of two words now. For two C-Vectors:

$$\text{C-Vector}_i=<i, W_1,\ldots,W_n>,\ \text{C-Vector}_j=<j,W_1,\ldots,W_m> \tag{4}$$

We standardize them to an \emph{N-Vector} that is:

$$\text{C-Vector}_i=<W^i_{-N},...,W^i_N>,\ \text{C-Vector}_j=<W^j_{-N},\ldots,W^j_N> \tag{5}$$

and $W_k=\varnothing$ if there is no word model in position k for both of them. We adopt the cosine

similarity when compare two vectors, that is:

$$sim(vec_i, vec_j) = \cos(\overrightarrow{vec_i}, \overrightarrow{vec_j}) = \frac{\overrightarrow{vec_i} \cdot \overrightarrow{vec_j}}{|\overrightarrow{vec_i}| \times |\overrightarrow{vec_j}|} \tag{6}$$

4.2 Clustering Words based on the Density

Among all the clustering methods using density functions has prominent advantages--anti-noisiness and the capability of finding groups with different Inspired by DENCLUE (Hinneburg & Keim, 1998), we define a influence function of a word w_0 over another word w:

$$f_B^w = f_B(w_0, w) \tag{7}$$

which is a function proportionately to the similarity of w_0 and w, and reveals the influence degree w_0 over w. Commonly used influence functions include *Square Wave Function* and *Gauss Function*. The former is suitable for the data which dissimilar distinctly while the later is more suitable for reflect the smooth influence of w_0. Because a word is related with many other words in different degrees but no simply 1 or 0 in corpus, it is more reasonable to choose Gauss Influence Function:

$$f_{Gauss}^w(w_0) = e^{\frac{-(1-\sin(w0,w1))^2}{2\sigma^2}} \tag{8}$$

We call Equation (8) the *Gauss Mutual Influence* of w, w_0 for $f^{w}_{Gauss}(w_0)=f^{w0}_{Gauss}(w)$. It makes each word linked with many other words to some extent. According to it, we can cluster words into groups. Before giving the definition of a word group, we develop some definitions first for further discussing:

Definition 4. Given a parameter ξ, ξ _region(w_0)={w|fGuess(w,w_0)> ξ} is called ξ _region of w_0. Given a parameter *MinPts*, w_0 is called a *CoreWord* if |ξ _region(w_0)|>MinPts. The minimal ξ which makes w_0 to be a CoreWord is called the CoreDistance of w_0 and be marked as ξ*.

Definition 5. We call w_0 is direct reachable to w' if w_0 is a CoreDistance and w'$\in \xi$ _region(w_0)$ and marked as $d_{reachable(w0,w')}$. For a set of words w_0, w_1, …,w_n=w', if $d_{reachable(wi,wi+1)}$ for all w_i, $1 \le i<n$$, then w_0 is reachable to w', that is, reachable(w_0,w').

Based on the definitions above, a word group can be seen as the maximal words set based on the reachable property. The corresponding clustering algorithm is given below:

(1)Taking $\xi = \xi$* and for all the words w perform the following operation:

$\xi_{cur} = \xi$*;cw_{cur}={w};

while(ξ_{cur} < 1){

$cw_{pre}=cw_{cur}$;

if(| ξ_{cur}_region(w_0) | > MinPts){
Build a word group cv_{cur} which contains all the words in ξ_{cur}_region(w) and takes w as the CoreWord of it.
if($\frac{|cv_{cur} - cv_{pre}|}{|cv_{cur}|} < \alpha$){break;}
}else{
break;
}\\if
$\xi_{pre} = \xi_{cur}$; $\xi_{cur} += \Delta$;
}//while
}$cw=cw_{pre}$;

(2)For each pair of CoreWords w_i, w_j
if(d_reachable(w_i,w_j)
Merge cw_i,cw_j into a new group cw_{i+j} which has two CoreWords cw_i and cw_j
(3)Repeate (2) until no new groups are generated.

Many groups with different density will be generated in (2) for we set value for ξ not a single number but a large range of field. The groups with high density will be created firstly and be covered by the dilute groups. We escape choosing the parameter of ξ by doing this.

4.3 Identifying Concepts using CICMs

Having the learned CICMs and word cluster, identifying method of new concepts is straightforward. Given a chunk, we just create its local L-Vector and judge whether it satisfies one of its or its similar words' C-Vector we have learned.

Definition 6. For a chunk $c_k=w_0 \ldots w_n$, the Local C-Vector for a word w_i in it :
L_Vector(w_i, c_k)=<i,W_0,…,W_{i-1}, W_{i+1},…, W_n>.

Theorem 1. For a chunk $c_k=w_0 \ldots w_n$, for each word w_i in it, there is L_Vector(w_i, c_k) $\in$ CICM(gw_i), then c_k is a concept, where gw_i is the similar word group of w_i.

5. Inner Relationship Discovery

The concept extractor introduced in last chapters makes a large-scale concept set available to be used for discovering inner relationships of the concept. In this study, we have found a special kind of Chinese hyponymy relationship, called lexical hyponymy, which is of great importance in ontology learning. To the best of our knowledge, no existing method can extract these hyponym relations. In this chapter, we will show a semi-automatic lexical hyponymy acquisition approach within a large-scale concept set, integrating symbolic, statistical and hierarchal techniques.

In a large-scale concept set C, if a subset S={<cpt1>,<cpt2>,…,<cptn>} exists, where
<cpt1>=<pref1><suf>,
<cpt2>=<pref2><suf>,
…
<cptn>=<prefn><suf>.

The <suf> here denotes a common suffix of all concepts. We may think the <suf> *could* be a hypernymy concept and following relations *may* exist:
HISA(<cpt1>, <suf>), HISA(<cpt2>,<suf>),…, HISA(<cptn>,<suf>)
For instance, given S={炭疽活菌苗，冻干鼠疫活菌苗，结核活菌苗，自身菌苗，外毒素菌苗}, we can segment the concepts as follows,

<自身菌苗>=<自身><菌苗>,
<外毒素菌苗>=<外毒素><菌苗>,
<结核活菌苗>=<结核活><菌苗>,
<炭疽活菌苗>=<炭疽活><菌苗>,
<冻干鼠疫活菌苗>=<冻干鼠疫活><菌苗>,

where the corresponding hypernymy concept suffix <suf> is <菌苗> and all HISA relations come into existence. However, if we consider the suffix chunk <苗> to be <suf> instead of <菌苗> (i.e. we segment the concept <外毒素菌苗>:=<外毒素菌><苗>), all HISA relations do not exist. Moreover, the suffix <苗> can not even be considered as a concept. We notice that a subset S'={结核活菌苗，炭疽活菌苗，冻干鼠疫活菌苗} of S contains a longer common hyponymy <活菌苗>, lexical hyponymy relations HISA(结核活菌苗，活菌苗), HISA(炭疽活菌苗，活菌苗) and HISA(冻干鼠疫活菌苗，活菌苗).

We will investigate into such common suffix in a concept set and mine lexical hyponymy taking advantage of the common suffix features. There is a limitation in this approach: the size of the concept set should be *very large* in order to find such common chunks. In an extreme case, we can extract nothing if there is only one concept in the concept set, even if the only concept in the set contains rich lexical hyponymy relations. However, there is no definition how large can be thought to be very large and we will analysis this factor in the experiment section.

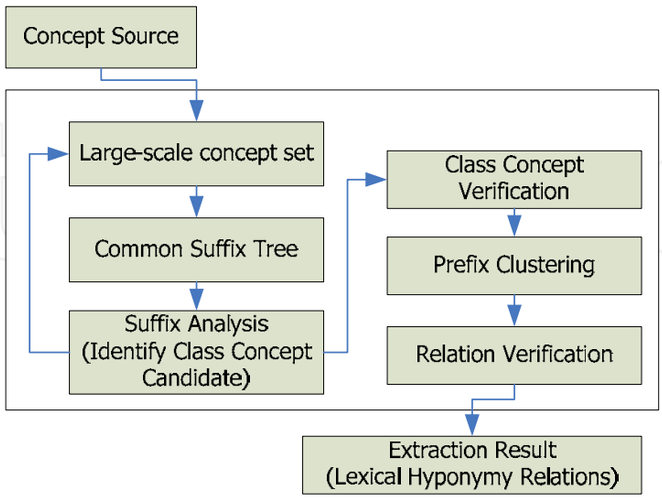


Fig. 2. Lexical hyponymy acquisition framework

Figure 2 describes our framework of lexical hyponymy acquisition. We use a Google-based

statistical acquisition model [16] to extract concepts from web corpus, which results in a large-scale concept set and then clustered them into a common suffix tree according to suffixes of concepts. The suffix analysis module uses a set of statistical-based rules to analyze suffix nodes. Class concept candidates, which are concepts, are identified by our Google-base verification module and used to enlarge the original concept set. A class concept verification process was taken to verify class concept candidates. Human judgment-based relation verification is taken after a prefix clustering process dedicating to reduce the verification cost is done. Finally we got extracted hyponymy relations from the common suffix tree with a hierarchical structure.

6. Common Suffix Tree Clustering

To find and analyze the common suffix, we propose a data structure called *common suffix tree (CST)*, inspired by suffix tree clustering (Cusfield, 1997).

Definition 7. A common suffix tree containing *m* concepts is a tree with exactly m leaves. Each inner node, other than leaf, has more than two children, and contains a single Chinese gram. Each leaf indicates a concept with a *longest shared suffix* that equals the string leading from the leaf to root. Along with the path, the string from each inner node to root is a shared suffix of the concept indicated by leaves it can reach.

With CST, not only are we able to find what is the longest shared suffix, we can also find which concepts share a certain common suffix. Following CST clustering algorithm will help us construct a CST in liner time complexity:

```
CST Clustering Algorithm:
Use the suffix-based clustering, and compute big 1-gram concept clusters.
Until(convergence) {
      From each n-gram cluster, iterate the algorithm to get finer, hierarchy n+1
      gram clusters.
}
```

The convergence condition of algorithm above is when the all clusters leave one leaf. For instance, in a given concept-set S={北京第六中学，南京第十六中学，天津第二十六中学，**经济**学，生物学，好学，同学，木**鱼**，黄花**鱼**，**烧**黄花**鱼**，**鲤鱼**}, The CST algorithm can be described as following steps:

1) Using the suffix-based clustering, we get big 1-gram clusters ({*} represents the least common suffix):

[北京第六中,南京第十六中,天津第二十六中,**经济**,生物,好,同]{学}

[木，黄花，**烧**黄花，**鲤**]{**鱼**},

2) From each 1-gram cluster, we iterate the algorithm to get finer, hierarchical clusters until convergence:

[[[北京第, [南京第,天津第二]{十}]{六}]{中},**经济**,生物,好,同]{学}

[木, [#，**烧**]{黄花}，**鲤**]{**鱼**},

where # represents an empty entry.
Figure 3 visualized the CST structure of {学}-cluster. The rest parts of our framework are built on the computing and analysis on suffixes of CST.

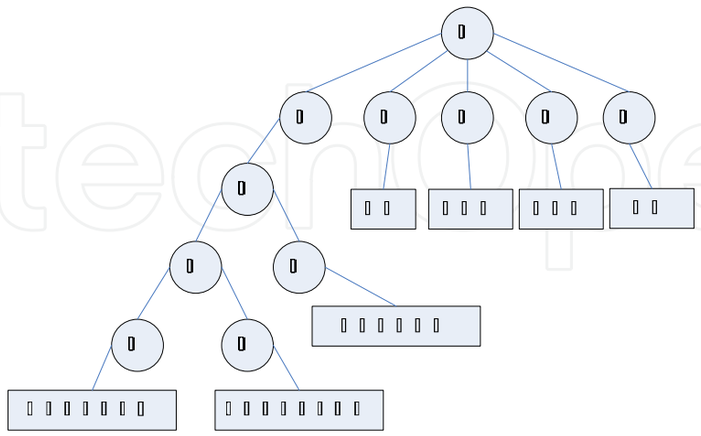


Fig. 3. Common Suffix Tree of {学}-cluster

7. Suffix Analysis

Given the "学" cluster in the example above, the suffix collection **S**={*第十六中学, 十六中学, 六中学, 中学*} may all hypernymy concepts we interested in, without any other information supporting (1-gram suffix causes great ambiguous, therefore we leave it alone in our system).
Some suffix concepts may be extracted by some Chinese word segment systems [20], however, there is no word segment system adopted in our system, because the segment system performs poor in a large scale general-purposed concept set, where many suffixes cannot be correctly segmented and thus lowered the performance of the entire system.
However, some useful statistic features can be obtained in a concept-set to identify class concepts. For a suffix chunk <*ck*> in concept-set, we may have patterns such as *CNT[*<*ck*><*>*]*, *CNT[*<*>*<*ck*>*]*, *CNT[*<*>*<*ck*><*>*]* and etc., where *CNT[*<*pattern*>*]* means the frequency of <*pattern*> in concept set. A list of examples of such patterns was listed in Table 3.

Pattern	Example
(1) *ISCpt[*<*ck*>*]*	<大学>∈S
(2) *CNT[*<*ck*><*>*]*	<大学>学生服**务**部, <大学>校区, …
(3) *CNT[*<*>*<*ck*>*]*	理工<大学>, 科技<大学>, …
(4) *CNT[*<*>*<*ck*><*>*]*	北京<大学>学生会, 中国<大学>**评**估**组**, …

Table 3. Statistical patterns and examples

Pattern (1) is not a real statistic. The pattern, once appears in the given concept set, prove that indicated suffix <*ck*> is a class concept candidate. If the concept set is *large enough* (i.e. for any <*cpt*>, always exists <*cpt*> ∈*S*), this single rule can be used to identify all class concept

candidates. Actually, our concept set can never achieve that large.
The emergence of pattern (2) and (3) is a strong indication of class concept, which usually can be some components of other words. Class concept <*大学(university)*> can be used as *limitation* of other concept, such as <*大学校区(university campus)*>, which indicates a special kind of <*校区(campus)*>. Experiment in following content shows that once the pattern (2) or (3) appears, the empirical probability of <*ck*> to be a concept is very high.

The information embedded in the pattern (4) is richer. We rewrite the *CNT*[<*><*ck*>] as f_{suf}(<*ck*>), called *suffix frequency*. The *i*th suffix of concept <*cpt*>= <x_m>…<x_1> (i.e. <x_i>…<x_1>), where <x_i> a is single gram, is denoted as *Suf*(<*cpt*>, *i*) and *m* is the length of <*cpt*>. The *suffix probability* S_{suf}(<*cpt*>,*n*) is defined as:

$$S_{suf}(<cpt>,n) = p_{suf}(<x_m>\ldots<x_1>, <x_{n+1}><x_n>\ldots<x_1>) \\ = f_{suf}(Suf(<cpt>,n+1)) / f_{suf}(Suf(<cpt>,n)) \tag{9}$$

where $n \leqq Length(<cpt>)-1$. p_{suf}(<ck_1>, <ck_2>) is the joint probability of chunk <ck_1> and <ck_2>. We define that single-gram concepts have no suffix probability.

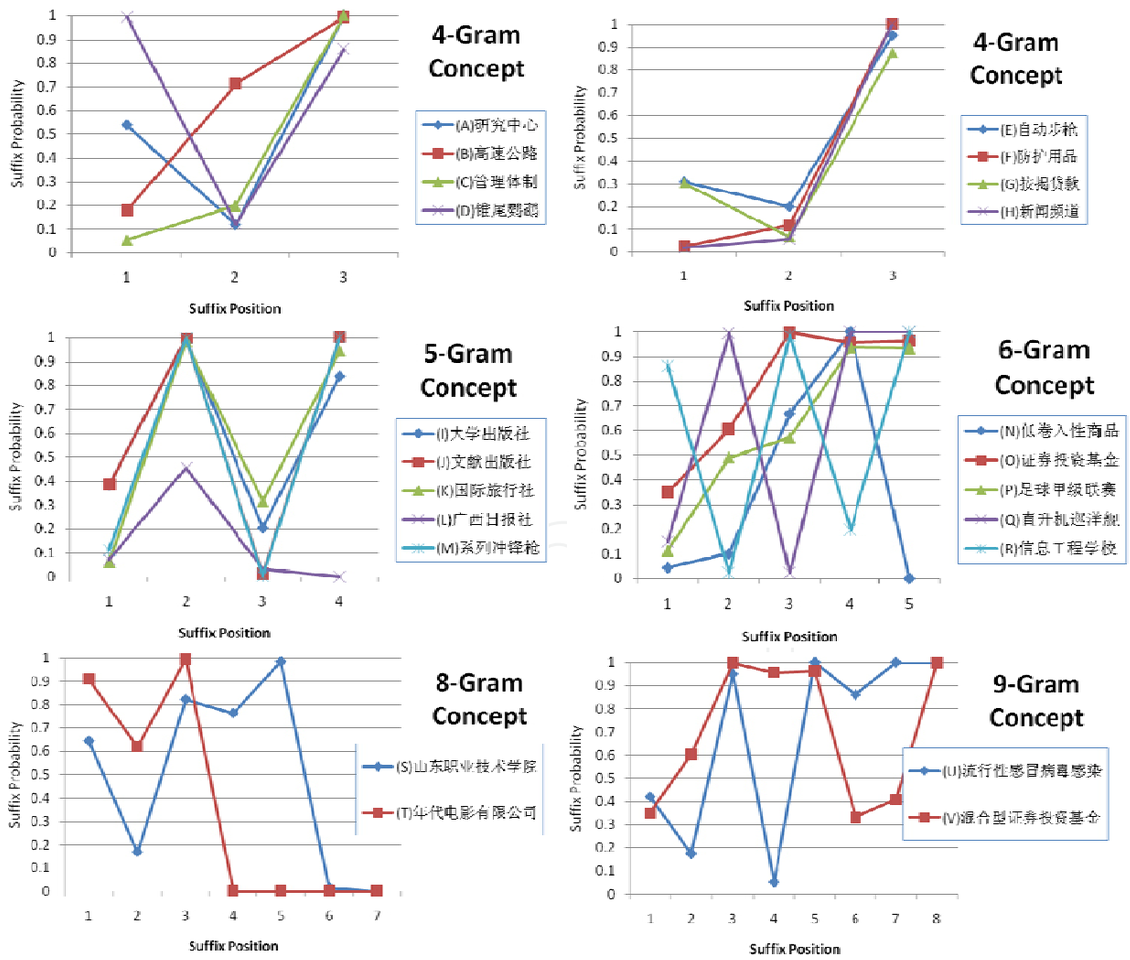


Fig. 4. Case study: suffix probability of 4, 5, 6, 8, 9-gram concepts respectively.

Figure 4 shows some cases of suffix probability whose numbers of grams composed are ranging from 4 to 9. Such cases illustrate how suffix probability changes with varying number of grams.

Figure 4 (V) shows the change of suffix frequency of concept <*cpt*>:= "*混合型证券投资基金*" in a concept set with a size of 800,000. Figure 4 (U) shows the situation when <*cpt*>:="*流行性感冒病毒感染*". For instance,

$$S(\text{"混合型证券投资基金"}, 2)=P(\text{"基金"}, \text{"资基金"})$$
$$=F(\text{"资基金"})/F(\text{"基金"})= 300/498 = 0.60241$$

From the case (S) we observe that *S(<cpt>, 3)=0.99667* is the maximum among *S(<cpt>, 2)*, *S(<cpt>, 3)* and *S(<cpt>, 4)*. At the same time *Suf(<cpt>, 4)* (i.e. 投资基金) is a class concept. Same situation could be found in maximum point *S(<cpt>, 5)* and *S(<cpt>, 8)*, while *Suf(<cpt>, 6)* and *Suf(<cpt>, 8)* are both class concept. In another case when <cpt>="流行性感冒病毒感染", we find the same phenomenon that class concept happened to appear in inflexions, which makes us believe it to be a useful rule. The rule is proved to be very effective in later experiment and is defined as follows:

Definition 8. *(Suffix Probability Inflexion Rule)* In a large-scale concept set, whenever the suffix probability *S(<cpt>,n)* encounters an inflexion, the suffix *Suf(<cpt>,n+1)* =<w_{n+1}><w_n>…<w_1> is considered to be a class concept candidate, which is called *Inflexion Rule*.

The suffix probability inflexion rule is exported from empirical study, and the hidden theoretical support of this rule is based on *mutual information*. The higher the *S(<cpt>,n)*, then the suffix <w_n>…<w_1> and <w_{n+1}><w_n>…<w_1> has higher mutual information, which may lead to a close correlation, the sudden reduce of mutual information means differentiation in linguistic usage.

Based on the discussions above, we summarize three *Suffix Concept Identification (SCI)* Rules:

1. *Pattern ISCpt[<wx>] appears, then <wx> must be a concept.*
2. *Pattern CNT[<wx><*>] or CNT[<*><wx><*>] appears, then <wx> can be a concept.*
3. *Suffix Probability Inflexion Rule.*

The experimental baseline comparisons among three rules are listed in Table 4. We use SCI rules in an 800,000 concept set and 300 test cases and manually extract all the class concept candidates in test cases, denoted by *cm*. Then we use SCI rules to extract class concepts, denoted by *ca*. We adopt following evaluation measurements in baseline experiment:

$$Precision = |\ ca\ \cap\ cm\ |\ /\ |\ ca\ |$$
$$Recall = |\ ca\ \cap\ cm\ |\ /\ |\ cm\ |$$

The average value and standard deviation of precisions and recalls are computed in 5 baseline scheme. Rules based on (1), (2) or the combinations of which have a low recall although with a high precision, as a result of the data sparsity. However, rule (3) holds a high precision and at the same time has a promising recall once combined with the other two rules.

	Precision		Recall	
	Average	Std. Dev	Average	Std. Dev
Rule(1)	100%	0	-	n/a
Rule(2)	95.753%	0.4603	-	n/a
Rule(3)	98.641%	0.1960	65.125%	2.393
Rule(1,2)	96.561%	0.5133	-	n/a
Rule(1,2,3)	98.145%	0.5029	66.469%	2.792

Table 4. SCI Rules Baseline Comparison (- mean the value is lower than 5%).

8. Class Concept Verification

In previous section we mentioned that not every concept could be a class concept. In this section, we proposed a lexicon-syntactic approach to verify class concept by scoring concepts via Googling web corpus.

Through our investigation, class concepts primarily appear in three kinds of lexicon-syntactic patterns which have different semantic meanings: Class I patterns appear when people are trying to give examples. Class II patterns are used when people construct question sentences. Class III patterns are, on the other hand, commonly used when we give definitions. The generic type of Class II is <Which><*>, where <Which> is some of the interrogatives. The generic type of Class II is <是> <Unit><*>, and here <Unit> is some of the unit quantifiers. Therefore, the pattern II and III includes a number of patterns. All three types of pattern with examples are summarized as shown in Table 5.

Pattern Type	Pattern Examples	Examples
Class I <*Such as*><*>	<ClassCpt>例如	一些*水果例如*香蕉，它如何繁衍后代
	等<ClassCpt>	76%**预**期深圳*等城市*的房价将下跌
Class II <*Which*><*>	什么<ClassCpt>	福威**镖**局在福州府的*什么大街*
	哪些<ClassCpt>	中国*哪些城市*适宜工作?
	那**种**<ClassCpt>	青苹果和**红**苹果*哪**种**苹果*有**营养**
Class III <*是*><*Unit*><*>	是一个<ClassCpt>	法国夏特瑞城*是一个小**镇***
	是一**种**<ClassCpt>	**宪**政*是一**种**文化*
	是一**类**<ClassCpt>	他和你*是一**类**人*

Table 5. Patterns and examples in three classes

Definition 9. Google provides statistical information in web corpus, probability framework based on which has been built by (Zhou et al., 2007; Cilibrasi & Vitanyi, 2007). Given a lexical chunk <ck>, the frequency of this term is defined as number of pages containing such term, denoted by f(<ck>).

Definition 10. For a concept <cpt>, the pattern frequency is defined as f(Pattern(<cpt>)), where Pattern(<cpt>) is applying the concept to a certain pattern. Pattern association is defined as the pattern frequency of the concept dividing its frequency, denoted by p(Pattern(<cpt>)).

$$p(Pattern(< cpt >)) = f(Pattern(< cpt >)) / f(< cpt >) \tag{10}$$

To verify class concepts, pattern associations can be used as attributes to train a classifier by machine learning algorithms. However, according to the linguistic property of the three classes, the pattern associations of a certain concept are likely to associate well with only one pattern in each class. Therefore we only use the patterns that can have the maximum pattern association in each class. We use the liner combination to sum pattern associations of all three classes into a scoring function, which is proved to be more effective than adopting three separate attributes.

Three classes of patterns are assigned with different class weights w_I, w_{II}, w_{III}, which can be used to adjust score according to liner analysis methods. Besides, we take the frequency of concept as a coefficient of the score, which indicates that a concept with a higher frequency is more likely to be a class concept. To sum all effects above, the expression of scoring a concept <cpt> is:

$$Score(< cpt >) = Log(f(< cpt >)) \times \sum_{i \in (I,II,III)} (w_i \times \underset{j \in Class_i}{Max}(p(Pattern_j(< cpt >)))) \tag{11}$$

To obtain a score threshold identifying class concept, we firstly annotate a training set of 3000 concepts, including 1500 class concept and 1500 non-class concept. We then use Google to retrieve pattern associations of training set. So the pattern associations are calculated into a score. And we use a linear analysis method to adjust the class weighs that can maximize the scoring function, and finally we get a score threshold. Concepts that exceed the given threshold are classified as class concept and vice versa. In our experiment, the class concept classifier we built is proved to achieve a remarkable high accuracy at 95.52%.

9. Prefix Clustering

Due to the property of lexical hyponymy relations, they hardly appear in other sources such as text corpus and web corpus, which makes human judgment a compulsory step in the relation verification process. In a large-scale concept set, the number of lexical hyponymy relations is huge, and thus it becomes a misery if we need to manually verify each relation.

In a concept sub-set S={<京津塘高速公路>, <长株潭高速公路>, <京石高速公路>, <京承高速公路>, <信息高速公路>} with the suffix Suf={<*>, 4} and Suf={<*>, 2}, where <*> denotes the wildcard of concepts, but the hyponymy relation within term <信息高速公路(Information High-Way)> is different from others. Since the concept is a kind of metaphor, there is not a real lexical hyponymy relation. If we can cluster the relations into meaningful groups, such as, metaphor group and non-metaphor group, it is possible for us to verify parts of the relation group instead of all relations.

We notice that a prefix <pref> of a concept <cpt>=<pref><suf> is typically a term that forms parts of other concepts in our concept set. Given a <pref>, H(<pref>) denotes all chunks that

appears before <pref> in other concepts and T(<pref>) denotes all chunks that appears after <pref> in other concepts. The two statistical information, that provided by concept set context, can be used to define the similarity of two prefixes.

Definition 11. Prefix Similarity is a quantity for measuring the similarity of two prefixes within a concept-set context. It is the average of Crossover Coefficients of Head Similarity and Tail Similarity.

$$Sim(<x>,<y>) = \left(\frac{|H(<x>) \cap H(<y>)|}{\min(|H(<x>)|,|H(<y>)|)} + \frac{|T(<x>) \cap T(<y>)|}{\min(|T(<x>)|,|T(<y>)|)}\right) \quad (12)$$

K-cluster technique, which is the simplest unsupervised learning algorithm, enables us to cluster data according to a given number of clusters k (MacQueen, 1967). With the ease to control cluster number, we can then flexibly choose a specific grain to cluster our relations. We perform a k-cluster algorithm on concept set using prefix similarity. In the case above, there are 1210 concepts containing “信息” in our 800,000 concept set. Other prefix terms rarely appear and share some terms such as <*><收费站>. Given k=2, the prefix <信息> will be placed in a separate group through clustering, while the rest four prefixes are grouped into one cluster. Hence, we only need to judge two hyponymy relations respectively from each cluster. From the empirical study, the best k-value is a median proportion of the size of the target concept sub-set.

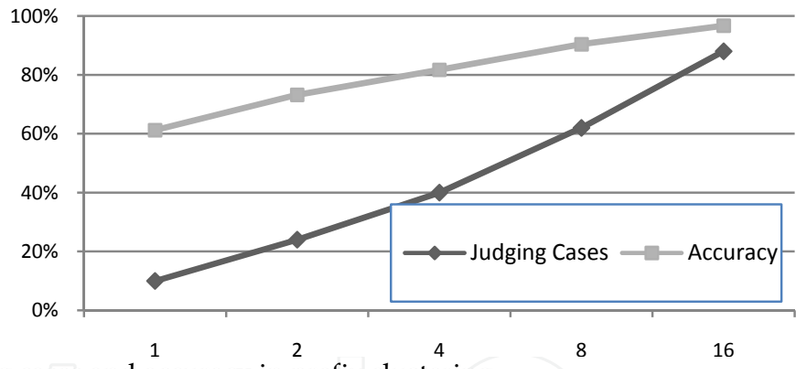


Fig. 5. Judging cases and accuracy in prefix clustering

This step is optional comparing to other modules employed in our framework, and sometimes it may lower the precision of the system. Figure 5 describes our judging cases and accuracy in a 1000-sized sub tree of a CST built by an 800,000 concept set. When setting the K-value to be 8, we will have an accuracy of 90.4% by judging 62% relation cases. Remarkably, not only does the percentage of judging cases depend on K-value, it also relates to the structure of targeting CST. However, prefix clustering will significantly improve the efficiency of human judgment during verification phase.

10. Discovering Hierarchical Lexical Hyponymy

Given a concept set ***C***, we use the CST clustering technique to build a CST. Then we compute the statistics of patterns described in Sect. 6 and store them in each CST node. We apply the

SCI rules to extract class concept candidates ***T′***, and add them to ***C***, enlarging our original concept set. We verify the unverified candidates in ***T′*** with the Google-base verification described in Sect. 7, and get a class concept set ***T***. In lexical hyponymy relation candidate set ***H′***, we remove all the relations that have hypernymy concepts in ***T***-***T′***.

Lexical hyponymy relations are generated as follows: For a given concept node <*cpt*>, set {<*s-cpt*$_1$>… <*s-cpt*$_n$>} is used to denote all the verified class concept nodes it goes through in CST, and we have *HISA*(<*s-cpt*$_i$>,<*s-cpt*$_j$>) (*i*<*j*). Put all generated relations to ***H′***. As the original concept set changed, we update statistical information of each node, and keep performing steps above until the status of each node remains unchanged. Finally we cluster the prefix according to Sect.7 and judge one relation candidate in each cluster in ***H′***, resulting our final *hierarchical* lexical hyponymy relation set ***H***. The pseudo code of acquiring process is given in Figure 6.

```
Acquiring a large-scale concept set C.
Constructing CST using CST clustering.
While(Convergence){
    Compute statistical information of inner nodes.
For each concept node <cpt> in CST {
Apply SCI rules.
        Get all class concept candidates T′
C←T′
        T← Verify unverified candidates in T′
        Remove hyponymy of invalid candidates
H′←All relation candidates in T
}
}
Perform Prefix Clustering in H′
Judging Relations in H′, resulting H
```

Fig. 6. Acquiring hierarchical lexical hyponymy relations

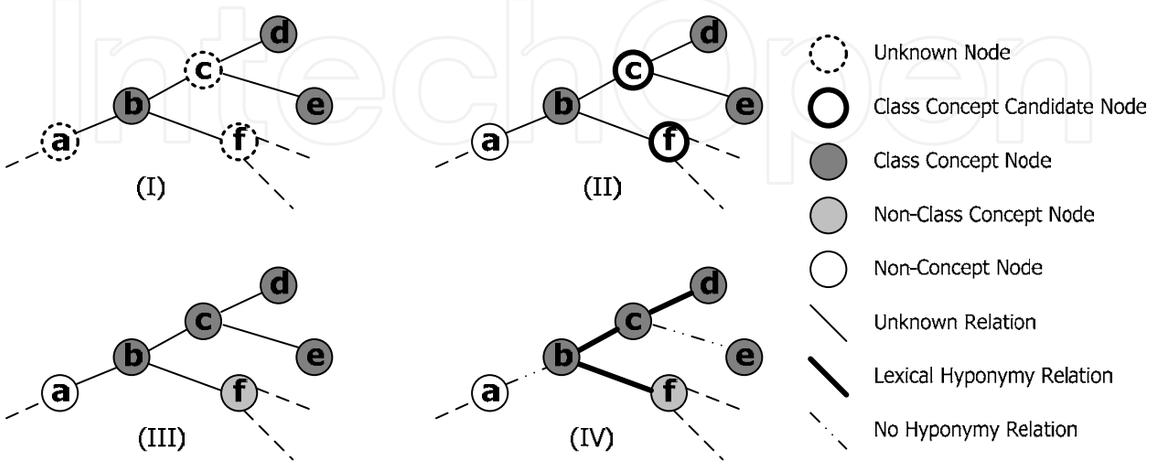


Fig. 7. An example of hierarchical acquisition process

To better illustrate this acquisition process, an example is given in Figure 7. Nodes {*a*, *b*, *c*, *d*, *e*, *f*} are *suffix chunk nodes* in a Common Suffix Tree. A suffix chunk node represents a lexical chunk of string starting from the corresponding CST node leading to the root. In (I), we have already known that *b*, *d*, *e* are class concept nodes and the rest are unknown nodes. Through suffix analysis, *a* is proved to be a non-concept and *b*, *c* are identified to be class concept candidates, as shown in (II). The candidates are then verified by the class concept classifier. In (III), *c* is classified as class concept and *d* is classified as non-class concept. Hyponymy relation candidates are *HISA(d, c)*, *HISA(e, c)*, *HISA(d, b)*, *HISA(e, b)*, *HISA(f, b)*, where *HISA(d, b)* and *HISA(e, b)* are derived from *transitivity* of hyponymy relation. *HISA(e, c)* is judged as a non-hyponymy relation, leading that *HISA(e, b)* to be removed, as shown in (IV).

11. Experiment

11.1 Concept Extraction

The concept extraction part of our system is called *Concept Extractor* (CptEx) and uses the following formulae to evaluate its performance:

$$p = \frac{\| m_a \cap m_m \|}{\| m_a \|}, r = \frac{\| m_a \cap m_m \|}{\| m_m \|}, F - Measure = \frac{2 \times p \times r}{p + r} \tag{8}$$

where m_a are the concepts CptEx extracts and m_{m} are the ones built manually. To calculate the performance, we selected 1000 chunks from the raw corpus and label the concepts in them manually. We compare the results based on CICMs with those based the Syntax Models and the POS Models as shown in Table 6:

Measurement	Syntax Models	POS Models	CICM
p	98.5%	86.1%	89.1%
r	1.2%	87.8%	84.2%
F-measure	2.3%	86.9%	86.6%

Table 6. Performance of CptEx

Having adopted CICMs to distinguish concepts from the chunks extracted by lexico-patterns, the precision rate drops down to 89.1% while the recall rate flies to 84.2%. The precision rate reduces because there are still some improper CICMs which will confirm fake concepts. Compared with POS Models, CICMs has a higher accuracy rate because we consider more factors to clarify the inner constructive rules rather than using part of speech only. On the other hand, our stricter models result in a lower recall rate.

11.2 Relation Mining

Our lexical hyponymy relation discovery is being evaluated through 5 concept sets of the size of 10000, 50000, 100000, 400000, and 800000, respectively. To compare their performances under different settings, we use the resulting lexical hyponymy relations acquired, followed by a human judgment with a k=10 prefix clustering. The same evaluation system as the last section is used to evaluate the performance of our system:

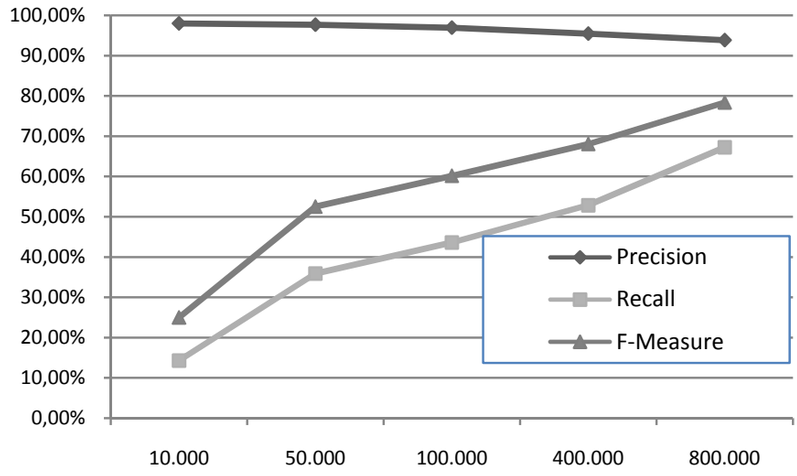


Fig. 8. System performance with different concept set size

From the acquisition result shown in Fig.8, we can discover that F-measure incrementally increases coincides the larger concept-set size, from 24.93 in 10000-sized concept set, climbing to 78.34 in 800000 one. Precision lower slightly and recall increase significantly with a larger concept set. As the size of concept set enlarges, more statistical information emerges, and at the same time more suffix concepts are extracted as class concepts, some of which form lexical hyponymy relations, causing a higher recall, while some other relations are invalid, leading to a lower precision. Under the concept-set with a size of 800000, the precision is 93.8% and recall reaches to 67.24%. The recall can be even higher when given a larger concept set.

In our concept set, we discover noise due to exocentric compounds, in which the suffix concepts are not hypernymy concepts. So far, no effort has been done to verify Chinese exocentric structures and the difficulty of linguistic usage makes it hard to analyze semantic relation within Chinese lexical concepts, which inevitably lower the precision of our framework.

Single-gram hypernymy concepts, such as '计', are likely to cause ambiguity. In our concept set, we find a large number of concepts ended with suffixes like {"硬度计","光度计", "温度计", "速度计", "长度计", "高度计"}. The mutual information between "度" and "计" is very high, leading the algorithm adopting SPI rule to wrongly mark the chunk "度计", rather than "计", as a class concept candidate. This problem might be solved if we could avoid the information sparsity by further enlarging the concept set.

The precision of class concept verification module is an important factor to the performance of whole system. We can further obtain a larger feature space and enhance the performance by employing advanced learning techniques such as SVM and Naïve Bayes Network.

Final precision of the framework is affected by our prefix clustering judgment, however, when the concept set becomes larger and thus more relations are extracted, it is inevitable for us to adopt that judgment.

12. Conclusion

We have described a new approach for automatic acquisition of concepts from text based on Syntax Models and CICMs of concepts. This method extracted a large number of candidate concepts using lexico-patterns firstly, and then learned CICMs to identify more concepts accordingly. Experiments have shown that our approach is efficient and effective. We test the method in a 160G free text corpus, and the outcome indicates the utility of our method.

To discover the inner relationships of the concept set, we propose a novel approach to discover lexical hyponymy relations in a large-scale concept set and make the acquisition of lexical hyponymy relations possible. In this method we cluster a concept set into a common suffix tree firstly, and then use the proposed statistical suffix identification rules to extract class concept candidates in the inner nodes of the common suffix tree. We then design a Google-base symbolic class concept verifier. Finally we extract Lexical hyponymy relations and judge them after the prefix clustering process. Experimental result has shown that our approach is efficient and can correctly acquire most lexical hyponymy relations in a large-scale concept set.

In the concept extraction part there are still some more works be done to get better performance for there are some improper CICMs. We plan to validate concepts in an open corpus such as in the World Wide Web in the future. In the relation discovery future work will be concentrated on the extraction of single-gram suffixes, which covers a large part of lexical hyponymy relations. On the other hand, through inner cross verification within a concept set, an approach that automatically verifies hyponymy relation is coming soon.

13. Reference

Acquemin, C. & Bourigault, D. (2000). Term Extraction and Automatic Indexing. Oxford University Press, Oxford(2000)

Agirre, E.; Ansa, O.; Hovy, E. & Martinez, D. (2004). Enriching very large ontologies using the WWW. In: Proc. of the ECAI 2004 Workshop on Ontology Learning.

Ando, M.; Sekine, S. & Ishizaki, S.(2003). "Automatic Extraction of Hyponyms from Newspaper Using Lexicon-syntactic Patterns", In IPSJ SIG Technical Report 2003-NL-157, pp.77-83, 2003

Cao C., Feng, Q. and et al. (2002). Progress in the Development of National Knowledge Infrastructure. Journal of Computer Science & Technology, Vol.17, No.5, 1¨C16, May, 2002.

Caraballo, S. (1999). "Automatic Construction of a Hypernym-babeled Noun Hierarchy from Text", In proceedings of 37th Annual Meeting of the Association for Computational Linguistics, Maryland, pp120-126, Jun 1999.

Chen, W.; Zhu, J. & Yao, T. (2003). Automatic learning field words by bootstrapping. In: Proc. of the JSCL. Beijing: Tsinghua University Press, 2003. pp. 67--72

Cilibrasi, R.L.& Vitanyi, P.(2007). The Google Similarity Distance. Knowledge and Data Engineering. IEEE Transactions 19(3), pp370–383, 2007

Dong, Z. & Dong, Q. (2006). HowNet and the computation of meaning. World Scientific Publishing Co., Inc. 2006.

Du, B.; Tian, H.; Wang, L. & Lu, R. (2005), Design of domain-specific term extractor based on multi-strategy. Computer Engineering, 31(14):159¨C160, 2005

Du, X. & Li, M. (2006). "A Survey of Ontology Learning Research", Journal of Software, Vol. 17 No.9 pp. 1837-1847, Sep. 2006.

Gelfand, B.; Wulfekuler, M. & Punch. W. F. (1998). Automated concept extraction from plain text. In: AAAI 1998 Workshop on Text Categorization, pp. 13--17, Madison, WI, 1998.

Gusfield,D. (1997). Algorithms on Strings, Trees and Sequences : Computer Science and Computational Biology[M]. Cambridge University Press, 1997.

Hearst, M.A. (1992). Automatic acquisition of hyponyms from large text corpora. Proceedings of the 14th International Conference on Computational Linguistics (COLING), pp. 539--545, 1992.

Hinneburg, A. & Keim, D. (1998). An efficient approach to clustering in large multimedia databases with noise. In: Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining, 1998.

Laurence, S. & Margolis, E. (1999). Concepts: Core Readings, Cambridge, Mass. MIT Press. 1999.

Liu L. & et al. (2005). Acquiring Hyponymy Relations from Large Chinese Corpus, WSEAS Transactions on Business and Economics, Vol.2, No.4, pp.211-218, 2005

Liu, L.; Cao, C.; Wang, H.& Chen, W. (2006). A Method of Hyponym Acquisition Based on "isa" Pattern, J. of Computer Science. pp146-151.2006

Lu, C.; Liang, Z.& Guo, A. (1992). The semantic networks: a knowledge representation of Chinese information process. In: ICCIP'92. pp. 50--57

MacQueen,J.(1967). Some Methods for classification and Analysis of Multivariate Observations. In proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability, Berkeley, University of California Press, pp281-297, 1967.

Nakata, K.; Voss, A.; Juhnke, M. & Kreifelts, T. (1998). Collaborative Concept Extraction from Documents. In U. Reimer, editor, Proc. Second International Conference on Practical Aspects of Knowledge Management (PAKM 98), Basel, 1998.

Ramirez, P.M. & Mattmann, C.A. (2004). ACE: improving search engines via Automatic Concept Extraction. In: Proceedings of the 2004 IEEE International Conference(2004) pp. 229--234.

Rydin,S. (2002). Building a hyponymy lexicon with hierarchical structures, In Proceedings of the Workship of the ACL Special Interest Group on the lexicon (SIGLEX), Philadelphia, July 2002, pp. 26-33, 2002

Tian, G. (2007). Research os Self-Supervised Knowledge Acquisition from Text based on Constrained Chinese Corpora. A dissertation submitted to Graduate University of the Chinese Academy of Sciences for the degree of Doctor of Philosophy. Beijing China, May 2007.

Velardi, P.; Fabriani, P. & Missikoff, M. (2001). Using text processing techniques to automatically enrich a domain ontology. In: Proc. Of the FOIS. New York: ACM Press, 2001. pp. 270--284.

Yu. L. (2006). A Research on Acquisition and Verification of Concepts from Large-Scale Chinese Corpora. A dissertation Submitted to Graduate School of the Chinese academy of Sciences for the degree of master. Beijing China, May 2006

Zelenko, D.; Aone, C. & Richardella, A.(2003). "Kernel Methods for Relation Extraction", Journal of Machine Learning Research, No.3, pp .1083-1106, 2003

Zhang, C. & Hao, T. (2005). The State of the Art and Difficulties in Automatic Chinese Word Segmentation. Journal of Chinese System Simulation. Vol.17 No.1 138¨C147. 2005.

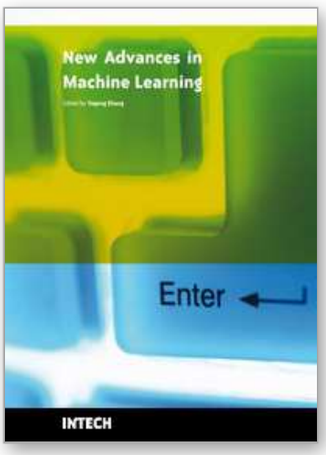
Zhang, C. et al. (2007). Extracting Hyponymy Relations from domain-specific free texts, In proceedings of the 9th Int. Conf. on Machine Learning and Cybernetics, Hong Kong, 19-22 August 2007.

Zhang, Y.; Gong, L.; Wang, Y. & Yin, Z. (2003). An Effective Concept Extraction Method for Improving Text Classification Performance. Geo-Spatial Information Science. Vol. 6, No.4, 2003

Zheng J. & Lu J. (2005). Study of an improved keywords distillation method. Computer Engineering, 31:194¨C196, 2005

Zhou, J.; Wang, S. & Cao,C. (2007), A Google-Based Statistical Acquisition Model of Chinese Lexical Concepts. In proceedings of KSEM07, pp243-254. 2007

IntechOpen



New Advances in Machine Learning

Edited by Yagang Zhang

ISBN 978-953-307-034-6
Hard cover, 366 pages
Publisher InTech
Published online 01, February, 2010
Published in print edition February, 2010

The purpose of this book is to provide an up-to-date and systematical introduction to the principles and algorithms of machine learning. The definition of learning is broad enough to include most tasks that we commonly call “learning” tasks, as we use the word in daily life. It is also broad enough to encompass computers that improve from experience in quite straightforward ways. The book will be of interest to industrial engineers and scientists as well as academics who wish to pursue machine learning. The book is intended for both graduate and postgraduate students in fields such as computer science, cybernetics, system sciences, engineering, statistics, and social sciences, and as a reference for software professionals and practitioners. The wide scope of the book provides a good introduction to many approaches of machine learning, and it is also the source of useful bibliographical information.

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen