Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI-10)

Commonsense Knowledge Mining from the Web

Chi-Hsin Yu and **Hsin-Hsi Chen**
Department of Computer Science and Information Engineering, National Taiwan University
No. 1, Sec. 4, Roosevelt Road, Taipei, 10617 Taiwan (R.O.C)
jsyu@nlg.csie.ntu.edu.tw, hhchen@ntu.edu.tw

Abstract

Good and generous knowledge sources, reliable and efficient induction patterns, and automatic and controllable quality assertion approaches are three critical issues to commonsense knowledge (CSK) acquisition. This paper employs Open Mind Common Sense (OMCS), a volunteers-contributed CSK database, to study the first and the third issues. For those stylized CSK, our result shows that over 40% of CSK for four predicate types in OMCS can be found in the web, which contradicts to the assumption that CSK is not communicated in texts. Moreover, we propose a commonsense knowledge classifier trained from OMCS, and achieve high precision in some predicate types, e.g., 82.6% in *HasProperty*. The promising results suggest new ways of analyzing and utilizing volunteer-contributed knowledge to design systems automatically mining commonsense knowledge from the web.

Introduction

Knowledge acquisition bottleneck is an important issue in commonsense reasoning. Approaches that acquire commonsense knowledge from different sources (Singh et al. 2002; Chklovski 2003; Schubert and Tong 2003) have been proposed, but few focus on studying the properties of CSK (Chklovski and Gil 2005). While good CSK acquisition approaches may result in large or high quality CSKs, mastering the proprieties of CSK is essential and can be beneficial for designing new and efficient acquisition methods.

Chklovski (2005) roughly classified CSK acquisition approaches into three categories according to the knowledge sources. The approaches of the first category collect CSK from experts. WordNet (Fellbaum 1998) and CYC (Lenat and Guha 1989) are typical examples. A lot of knowledge is collected by linguists or by knowledge engineers. These approaches result in well-organized and high quality knowledge base, but the cost is expensive and then limits the scalability. The approaches of the second category collect CSK from untrained volunteers. Open Mind Common Sense (OMCS) (Singh et al. 2002) and LEARNER (Chklovski 2003) are of this type. These approaches employ the vast volunteers in the Web to contribute CSK and correct the input CSK. The resulting CSK assertions can be in the order of million, but its quality is not as high as WordNet. The last approaches collect CSK from texts/the Web using algorithms. TextRunner (Etzioni et al. 2008), KNEXT (Schubert 2009), and the systems (Clark and Harrison 2009; Cankaya and Moldovan 2009; Girju, Badulescu, and Moldovan 2006) are of this type. These approaches usually process texts or web pages first, and then use lexical or syntactical patterns to extract facts or general knowledge. Because the CSK mined is large, it is not feasible to examine all CSK assertions manually. The performance of a knowledge acquisition algorithm is evaluated directly by assessors' small sampling or indirectly by employing the extracted knowledge to some tasks such as word sense disambiguation and examining the performance of applications. These approaches have the feasibility of controllable knowledge domain and scalability of extracted knowledge base, but the quality of resulting CSK is hard to control.

Algorithm 1 Automatic CSK mining from texts

```
Let K be a seed set of CSK
Build classifier C from K
repeat
  Identify CSK in texts and analyze its context
  Induce new lexical or syntactical patterns P
  Extract new CSK from texts using patterns P
  Assert the quality of new CSK using classifier C
  Let K be union of new CSK set and K
  Build new classifier C from K
until No more qualified CSK is extracted
```

This paper adopts a general CSK mining algorithm shown in Algorithm 1. To generate a large knowledge base of high quality from a seed CSK set, the knowledge source, the induction of mining patterns, and the quality assertion of CSK are three major issues. In this paper, we experiment with OMCS knowledge base and conclude that large portion of high quality CSK is explicitly stated in the web for some predicate types. It resolves the debate that CSK is not communicated in the texts–at least for those stylized CSK. Besides, we also introduce CSK classifiers to verify the correctness of assertions. This approach opens new ways to assert the quality of a large CSK set automatically.

Commonsense Knowledge in Texts

Although algorithms that use lexical patterns to mine knowledge from texts are very common, the CSK-in-text assumption behind the algorithms is never asserted or analyzed. Some researches take CSK-in-text for granted (Schwartz and Gomez 2009; Clark and Harrison 2009), while some researches assume that most of CSK are un-stated and need to be taught by human (Singh et al. 2002). We analyze the **"explicitly stated"** CSK from OMCS. For example, *"A car is a vehicle"* is an assertion in OMCS, and this assertion is stated in sentence *"**A car is a vehicle** with specific characteristics."* in the web. Therefore, this assertion is defined as an explicitly stated CSK (abbreviated as e-CSK hereafter).

Knowing explicitly stated CSK has some practical uses. For example, we can collect context for an e-CSK, analyze and further classify its context, and apply the analyzed context to mine new CSK of similar types. This kind of knowledge is also important for the study of commonsense knowledge usage in language.

OMCS Database

OMCS project[1] in MIT has announced a public available database which contains CSK all contributed by volunteers. In this database, sentences entered by users are processed to a predefined format. Each sentence has two concepts corresponding to two arguments with a specific predicate. There are 18 predicate types in this database. Table 1 lists some examples.

Predicate	Concept 1	Concept 2
CausesDesire	the need for money	apply for a job
HasProperty	Stones	hard
Causes	making friends	a good feeling
HasPrerequisite	having a party	inviting people
CapableOf	a cat	catch a mouse
HasSubevent	having fun	laughing
UsedFor	a clothing store changing room	trying on clothes
IsA	a swiss army knife	a practical tool
AtLocation	a refrigerator freezer	the kitchen

Table 1: Examples from OMCS database

Besides, each sentence has a confidence score determined by users collaboratively. When a user asserts a sentence as a valid CSK, the corresponding score is increased by one. On the contrary, if a user negates this sentence as a valid CSK, its score is decreased by one. In this way, the confidence score of a CSK can be considered as an indicator of its quality.

Estimation Method of e-CSK

We use the OMCS dataset to estimate the number of e-CSKs in texts. An assertion in OMCS is submitted to a search engine as a single string to get its occurrence count in the web. But search results may have noises. For example, the search results of *"A car is a vehicle"* may be as follows, where (a) is a noise and (b) is a valid e-CSK:

(a) An extended warranty on *a car is a vehicle* service contract between the warranty company and you.

(b) Because *a car is a vehicle*, it moves and carries things.

To decrease the effects of noises, we consider frequency as a filter. If the number of occurrences of an assertion is larger than a given threshold, this assertion is regarded as an e-CSK. Instead of Google[2], we use Google AJAX[3] for estimation. Google AJAX has no limitation on the number of submitted sentences, but the returned search results are usually less than those returned from Google. To estimate the search results in Google by using the search results in Google AJAX, we adopt linear measurement model (linear regression) as follows:

$$y = a \times x - b \tag{1}$$

$$x = \sqrt{\text{search result in AJAX}} \tag{2}$$

$$y = \sqrt{\text{search result in Google}} \tag{3}$$

In equation (2) and (3), we use square root on search results to decrease the weights of sentences of high frequency.

Experiment Settings Three datasets are selected from OMCS for experiments. Dataset A has 630,464 English sentences which are used for queries to Google AJAX. Dataset B consists of 12,049 sentences randomly selected from dataset A. These sentences are used for queries to Google. Dataset C, which is a part of Dataset A, is composed of sentences having confidence scores in OMCS database. For comparison, we also randomly generate a data set consisting of 299,454 sentences and having similar sentence length distribution to Dataset C. These sentences can't be found by searching Google AJAX, so that the result is not reported hereafter.

Threshold Selection The parameters a and b of the resulting linear regression model specified in equation (1) are 4.1425 and 2.6657, respectively. Before determining a specific threshold for e-CSK, we have to know whether the AJAX search result is well-behaved as expected. We use language models to model its behavior.

Suppose both CSK and e-CSK are generated by language models M_{CSK} and $M_{\text{e-CSK}}$, respectively, and the two models have the same vocabulary. Using naïve Bayes model, we have the first equality in (4), in which w_i is i-th word in a sentence. To simplify equation (4), we assume all words have equal probability to be drawn from vocabulary, and we have the second equality in (4), in which w is any word from vocabulary. We can conclude that the ratio in (4) decrease exponentially to the sentence length in word.

$$\begin{aligned} &\frac{\text{Number of explicitly stated CSK with } L \text{ words}}{\text{Number of CSK with } L \text{ words}} \\ \simeq\ & \frac{P(\text{sentence with } L \text{ words} \mid M_{\text{e-CSK}})}{P(\text{sentence with } L \text{ words} \mid M_{\text{CSK}})} \\ =\ & \frac{\prod_{i=1}^{L} p(w_i \mid M_{\text{e-CSK}})}{\prod_{j=1}^{L} p(w_j \mid M_{\text{CSK}})} = \left(\frac{p(w \mid M_{\text{e-CSK}})}{p(w \mid M_{\text{CSK}})}\right)^{L} \end{aligned} \tag{4}$$

$$f(L) = c \times 2^{-d \times L + e} + g \tag{5}$$

To verify the derived model in (4) by experimental data, we use function $f(L)$ in (5) to fit the search results of dataset A. The fitting between AJAX search result ≥ 7 and function $f(L)$ that uses the nonlinear least-squares (NLLS) Marquardt-Levenberg algorithm is shown in Fig. 1.

In Fig. 1, y-axis denotes the ratio in (4), and x-axis is the sentence length L in word. The function $f(L)$ perfectly fits experimental data that has at least 7 AJAX search results. It indicates that the AJAX search results can be modeled by (4) and is well-behaved. Next, we want to decide an e-CSK threshold in such a way that a sentence is regarded as an e-CSK when its AJAX search result is larger than this threshold. Fig. 2 shows how to determine the threshold.

[1] http://conceptnet.media.mit.edu/dist/

[2] http://www.google.com

[3] http://ajax.googleapis.com

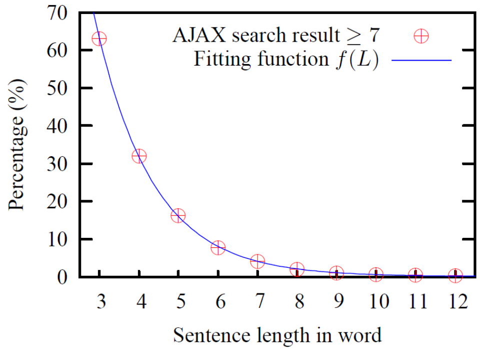

Figure 1: Fitting function $f(L)$ to AJAX search result ≥7

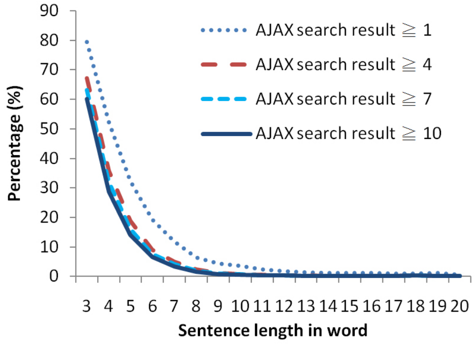

Figure 2: Percentage of sentences of different search results given different sentence length

In Fig. 2, the lines are gradually converged as the minimal number of Google AJAX search results increases from 1 to 10. Besides, if the number of the search results in Google AJAX is 10, the number of search results in Google equals to 109, which has high probability to be an e-CSK with these occurrences. Therefore, 10 is considered as the threshold of e-CSK.

Statistics about e-CSK

We use dataset C to derive some statistics of e-CSK. Fig. 3 shows the relationship between confidence score (CS) and e-CSK.

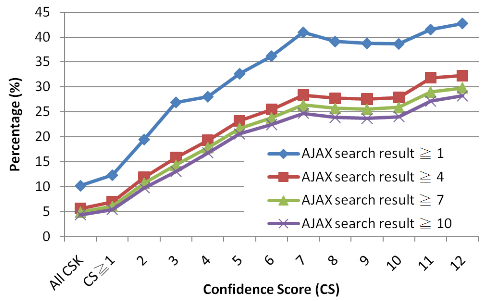

Figure 3: Relationship between confidence score and e-CSK

$$\frac{\text{Num. of CSK with (CS} \geq q \text{ and AJAX results} \geq p)}{\text{Number of CSK with CS} \geq q} \qquad (6)$$

In Fig. 3, y-axis is the ratio defined in (6), where $p \in \{1, 4, 7, 10\}$ and $q \in \{1, ..., 12\}$. "All CSK" in Fig. 3 is the ratio without the limitation on CS, and its denominator of the percentage is the size of dataset C. Besides e-CSK (i.e., AJAX search result $\geq$ 10), we also plot other AJAX search results (1, 4, and 7) for reference. We can see that the percentage increases along with confidence scores. In other words, sentences of higher confidence scores tend to have higher probability to be mentioned explicitly. When CS is larger than four, 16.8% of sentences are e-CSK. That means a large portion of high quality CSK can be found in texts. Because OMCS records person's CSK, it reverses the usual assumption that CSK is not communicated.

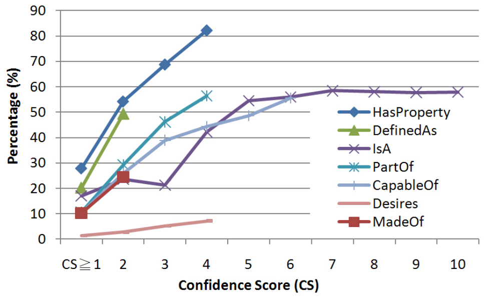

Figure 4: Relationship between predicate types and e-CSK

Fig. 4 further tells the relationship between predicate types and e-CSK. Y-axis is interpreted similarly as that in Fig. 3 except that we calculate the ratio by different predicate types. In Fig. 4, total 7 of 18 predicate types in dataset C are shown here. The remaining 11 predicate types whose ratios are smaller than 3% are ignored. All points in this figure have at least 90 CSK sentences to maintain the statistical significance. We can find that more than 40 percents of sentences are e-CSK in four predicate types, i.e., *HasProperty, IsA, PartOf*, and *CapableOf*, when CS is larger than four. Especially, 82.2% of sentences of *HasProperty* type can be found in texts.

There are two main reasons why the ratio of the remaining 11 predicate types are too small to be included in the figure. First, some patterns used in OMCS are not very common in language. For example, *"If you want to go for a drive then you should get a car"* has high CS but has low Google search results. This sentence is collected by using pattern *"If you want to * then you should *"*. Second, some CSK are usually mentioned implicitly. For example, *"a lobby is used for waiting"* and *"Beds are used to sleep in"* are two sentences of high confidence socres but of 0 Google search result. Instead, *"waiting in the lobby"* and *"sleep in bed"* have 806,000 and 13,800,000 Google search results, respectively.

Commonsense Knowledge Classification

The second goal of this paper is to build a general purpose CSK classifier with broad coverage. Classification algorithms are cost effective and easy to control, comparing to human-involved acquisition approaches. We focus on binary classifier which labels a given assertion with CSK or non-CSK.

Representation Scheme

For CSK classification, we propose a representation scheme to denote an assertion. In OMCS database, an assertion is already preprocessed to a tuple, i.e., (PredicateType, Concept1, Concept2), which have been shown in Table 1. Because a concept is usually a phrase, a concept is specified by slots. The number of slots depends on different approaches shown as follows. A slot in turn contains words, and a word is represented by a co-occurrence vector.

Word Representation: We use a co-occurrence vector W_i in (7) to represent a word w_i in a slot, where D is a dictionary with size $|D|$, d_j is the j-th entry in D, and $f(d_j, w_i)$ is the co-occurrence frequency of entry d_j and w_i in a corpus.

Concept Representation: We propose three approaches to determine the number of slots and how to place words in a concept into slots. The three approaches are discussed as follows:

1. **Bag-of-Words (BoW) Approach**: All words are placed in one slot. BoW is considered as a baseline.
2. **N-V-OW Approach**: All words are categorized into three slots, named HeadNoun, FirstVerb, and OtherWords. HeadNoun and FirstVerb are the head nominal of a phrase and the first verb of a phrase, respectively. Those words that are not in the two slots are put into OtherWords slot.
3. **N-V-ON-OW Approach**: All words are categorized into four slots, named HeadNoun, FirstVerb, Other-Nouns, and OtherWords. HeadNoun and FirstVerb are interpreted the same as those in the second approach. We further distinguish other words by their parts of speech (i.e., noun vs. non-noun).

$$W_i = (f(d_1, w_i), f(d_2, w_i), ..., f(d_{|D|}, w_i)) \tag{7}$$

$$S_{j,k} = \sum_{w_i \,\in\, \text{slot } k \text{ of concept } j} W_i \tag{8}$$

$$C_1 = (S_{1,1}, S_{1,2}, S_{1,3}) \tag{9}$$

$$C_2 = (S_{2,1}, S_{2,2}, S_{2,3}) \tag{10}$$

With the approaches defined above, the vector $S_{j,k}$ of slot k in concept j is described in (8). Vector C_1 of Concept1 and vector C_2 of Concept2 by using N-V-OW approach is described in (9) and (10), respectively. The concept vectors for BoW and N-V-ON-OW approaches are defined in the similar way.

Assertion Representation: An assertion is a tuple as described, but we ignore the PredicateType information here because it is usually the same within a predicate. For example, in *IsA* predicate, the keywords are ”is” or ”are” which did not help much for binary classification in our setting. Hence, an assertion is a vector (C_1, C_2) in which C_1 and C_2 come from Concept1 and Concept2, respectively.

CSK Classification Algorithm

Algorithm 2 CSK Classification Algorithm

```
   Preprocessing
1: Use POS Tagger to tag an assertion
2: Place words of concepts into slots
3: Derive vector of word $W_i$ from a corpus
4: Represent an assertion
   Feature Selection
5: Normalize concepts $C_1$ and $C_2$ to 1, respectively
6: Calculate Pearson correlation coefficient of each feature
   in slot
7: Select the first 10% features in each slot
   Classification
8: Use support vector machine to classify assertions
```

CSK classification algorithm is described in Algorithm 2. In step 1, we use Stanford POS tagger (Toutanova et al. 2003) to get tags. In step 2, we identify the head noun and the first verb of concepts by heuristic rules. For example, the first appearance of a verb in a tagged sequence is regarded as the first verb, and the last noun in a phrase is considered as the head noun. We distinguish noun and non-noun by parts of speech. In step 3, we consider Google Web 1T 5-Gram as our reference corpus (Brants and Franz 2006), and employ only 5-gram entries in this corpus. This corpus consists of approximately 1 trillion English word tokens from web pages to generate word n-grams and their frequency count. Take a 5-gram entry in Google 1T5G as an example:

last moments of your life   280

This 5-gram occurs 280 times, so it contributes co-occurrence frequency 280 to word pair moments and life. We sum all related frequencies in all 5-grams to determine the final co-occurrence count of this word pair.

The dictionary D in step 3 is a combination of WordNet 3.0 and Webster online dictionary[4] (noun, verb, adjective, and adverb). The resulting lexicon contains 236,775 entries. In step 5, the concept vectors are normalized to 1 respectively to equally emphasize on the two concepts. Only top 10% of features are selected.

Experiment Settings

Datasets: We select positive assertions from OMCS and automatically generate negative assertions to produce a balance dataset for a predicate type. The positive assertions must meet the following four criteria: (1) the confidence score of an assertion must be at least 4; (2) the polarity must be positive (Note that there are positive and negative polarities in OMCS, and only 1.8% of assertions with CS ≥ 4 have negative polarity.); (3) a concept contains no relative clause, conjunct, or disjunct; and (4) the length of a concept is less than 8 words for simplicity. The negative assertions are generated by randomly selecting and merging concepts from the OMCS database. We ignore datasets of size smaller than 200. Table 2 lists the resulting nine datasets among 18 predicate types in OMCS.

Predicate	Size	Predicate	Size
CausesDesire	204	*HasSubevent*	1026
HasProperty	254	*UsedFor*	1442
Causes	510	*IsA*	1818
HasPrerequisite	912	*AtLocation*	2580
CapableOf	916		

Table 2: Datasets for CSK classification

[4] http://www.mso.anu.edu.au/∼ralph/OPTED/index.html

Experiment Procedure: For each dataset, we randomly split 90% for training and 10% for testing. Next, we use LibSVM for classification. In SVM training, we adopt radial basis for kernel function, grid search in parameters (c, g) (80 pairs). After the best parameters are obtained, we train a model on training set by using these parameters, and apply trained model to test set. The same procedure is repeated ten times to obtain statistically significant results.

Note the performance variation of a classifier in classifying commonsense knowledge. We can view a train set as a knowledgebase that one person owns, and this person may be good at some aspects but bad at other aspects. This kind of train set may over-fit on some aspects and miss-classify other valid CSK. Because we aim to obtain a general purpose CSK classifier with broad coverage, the performance variation is an important indicator to evaluate a CSK classification algorithm.

Experiment Results

The test performance of classifiers is shown in Fig. 5. The standard deviation and database size are also shown in this figure.

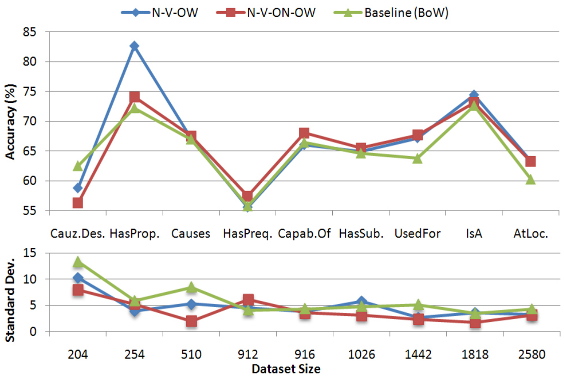

Figure 5: Classifiers' testing accuracy on nine datasets

In Fig. 5, N-V-OW approach and N-V-ON-OW approach are better than BoW approach except in *CausesDesire* predicate type. N-V-ON-OW approach tends to have smaller performance variation than BoW and N-V-OW approaches. N-V-OW approach has the best accuracy (82.6%) and variation in *HasProperty* predicate. The best result of *IsA* predicate is 74.4%, which is comparable to similar problems in SemEval-2007 Task 4, Classifcation of Semantic Relations between Nominals (Girju et al. 2007). Although the accuracy variation is reduced when dataset size increases, it is still large. We analyze the percentage of support vectors and show it in Fig. 6.

Fig. 6 shows the percentage of support vectors is usually larger than 60%. The consequence is that the behavior of the resulting SVM classifiers is similar to k-nearest neighbor algorithm. We can see this in the decision function of SVM classifier. When classifiers' behavior is like the kNN algorithm, the variation on train set has a great impact on the performance of test set. That results in large accuracy variation. This result can be read from two perspectives: first, CSK in OMCS database has great diversity which is good for CSK acquisition approach; second, the large percentage of support vectors will result in expensive computation in runtime which is bad for using SVM classifiers in large scale CSK acquisition. Reducing the percentage of support vectors is a very important issue if we use SVM classifier. One solution for this problem is to learn a specific classifier for one domain instead of a general purpose classifier. In this approach, a domain detection algorithm is needed for domain switching, and errors may be introduced.

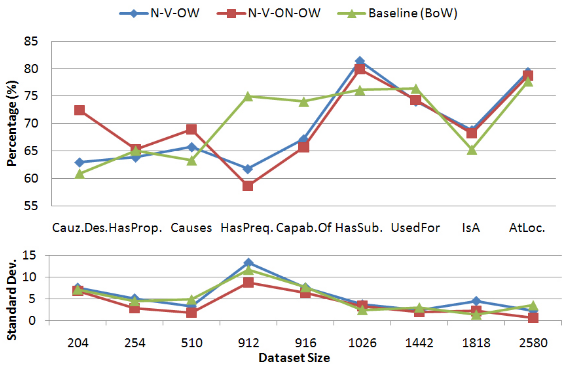

Figure 6: The percentage of support vectors in trained classifiers

Conclusions and Future Work

In this paper, we study two important issues in automatic CSK mining. The experimental results show that more than 40% of assertions are e-CSK (explicitly stated CSK in text) for four predicate types in OMCS including *HasProperty, IsA, PartOf,* and *CapableOf.* This exciting result opens new directions to study the properties of volunteers-contributed datasets. Researchers can distinguish e-CSK and non-e-CSK by using OMCS database, and then focus their attentions on mining non-e-CSK from users in public domain, studying the differences between e-CSK and non-e-CSK, and studying what kind of knowledge in what kind of situations may be stated in texts. That will be important to know the properties of CSK and will be also useful to boost the knowledge mining approaches.

The use of classifier in CSK acquisition is new, and a lot of issues are worthy of investigation. For example, what kind of classifier is better for CSK classification, general purpose or special purpose? How do we integrate CSK classifier into volunteers-contributed acquisition systems to assert user input knowledge or to filter out nonsense CSK generated by systems such as AnalogSpace (Speer, Havasi, and Lieberman 2008)? And how do we use classifier to gradually improve the quality of mined CSK? Those problems are not only useful for CSK classifier, but also important for CSK analysis.

Acknowledgments

This work was supported in part by the Excellent Research Projects of National Taiwan University, 98R0062-04.

References

Brants, T., and Franz, A. 2006. Web 1T 5-gram Version 1.

Cankaya, H. C., and Moldovan, D. 2009. Method for extracting commonsense knowledge. In *K-CAP '09*, 57–64.

Chklovski, T., and Gil, Y. 2005. An analysis of knowledge collected from volunteer contributors. In *AAAI 2005*, 564–570.

Chklovski, T. 2003. Learner: A system for acquiring commonsense knowledge by analogy. In *K-CAP '03*, 4–12.

Clark, P., and Harrison, P. 2009. Large-scale extraction and use of knowledge from text. In *K-CAP '09*, 153–160.

Etzioni, O.; Banko, M.; Soderland, S.; and Weld, D. S. 2008. Open information extraction from the web. *Commun. ACM*.

Fellbaum, C., ed. 1998. *WordNet: An Electronic Lexical Database*. MIT Press.

Girju, R.; Badulescu, A.; and Moldovan, D. 2006. Automatic discovery of part-whole relations. *Comput. Linguist.*

Girju, R.; Nakov, P.; Nastase, V.; Szpakowicz, S.; Turney, P.; and Yuret, D. 2007. Semeval-2007 task 04: Classification of semantic relations between nominals. In *SemEval '07*, 13–18.

Lenat, D. B., and Guha, R. V. 1989. *Building Large Knowledge-Based Systems, Representation and Inference in the Cyc Project*. Addison-Wesley Longman Publishing Co., Inc.

Schubert, L., and Tong, M. 2003. Extracting and evaluating general world knowledge from the brown corpus. In *Proceedings of the HLT-NAACL 2003 Workshop on Text Meaning*, 7–13.

Schubert, L. K. 2009. From generic sentences to scripts. In *IJCAI'09 Workshop W22, LSIR 2*.

Schwartz, H. A., and Gomez, F. 2009. Acquiring applicable common sense knowledge from the web. In *Proceedings of the Workshop on Unsupervised and Minimally Supervised Learning of Lexical Semantics*, 1–9.

Singh, P.; Lin, T.; Mueller, E. T.; Lim, G.; Perkins, T.; and Zhu, W. L. 2002. Open mind common sense: Knowledge acquisition from the general public. In *On the Move to Meaningful Internet Systems*.

Speer, R.; Havasi, C.; and Lieberman, H. 2008. Analogyspace: Reducing the dimensionality of common sense knowledge. In *AAAI 2008*, 548–553.

Toutanova, K.; Klein, D.; Manning, C. D.; and Singer, Y. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *HLT-NAACL 2003*, 173–180.