In *Proceedings of the International Conference on Natural Language Processing (ICON), 2004*.

Mining Word Senses from Text for Corpus-Based Lexicography

Canasai Kruengkrai, Thatsanee Charoenporn,
Virach Sornlertlamvanich, Hitoshi Isahara
Thai Computational Linguistics Laboratory
National Institute of Information and Communications Technology
112 Paholyothin Road, Klong 1, Klong Luang, Pathumthani 12120, Thailand
Email: {*canasai,thatsanee,virach*}*@tcllab.org, isahara@nict.go.jp*

Abstract

This paper discusses the problem of automated lexicography. In the corpus-based approach, a lexicographer has to manually group contexts of a target word into clusters in order to identify word senses. When a large number of the contexts is given, this process becomes a tedious and time-consuming task. To overcome this problem, we propose an efficient technique based on unsupervised clustering. We present the spherical Gaussian EM algorithm that can be enhanced by combining a robust initialization method based on Principal Component Analysis. The resulting clusters can provide a structure for analyzing the underlying senses of the target word found in a text corpus. Experimental results on two different data sets of polysemous words indicate that our proposed algorithm is a promising technique for corpus-based lexicography.

1 Introduction

The use of a corpus has played an important role in the modern dictionary construction. One of the successful developments is LEX*i*TRON[1] [Palingoon *et al.*, 2002] that was constructed based on large textual corpora of Thai words, containing a wide variety of electronically stored text such as newspapers, books, etc. The example sentence for each word is extracted directly from the corpus that shows how a word is really used. With the advent of the computational lexicon, such as TCL's Computational Lexicon[2] [Charoenporn *et al.*, 2004], the corpus is becoming more and more important.

In corpus-based lexicography, the task can be summarized as the following steps [Stevenson, 2003]. It starts by collecting all occurrences of a target word in the corpus using a concordance program. The words are represented as Key Word in Context (KWIC) concordance lines, which

[1] `http://lexitron.nectec.or.th`.
[2] `http://www.tcllab.org/tcllex`.

are corpus lines presented with the target word in the center. Then, a lexicographer manually groups these concordance lines into clusters based on certain criteria such as part of speech category, grammatical behavior, and semantic and topical considerations. Once the lexicographer is satisfied that the groupings are correct, a definition is written for each group representing a different sense. When a large number of the concordance lines is given, it is a tedious and time-consuming task for the lexicographer to group these concordance lines. The lexicographer may not have the patience to do such task.

To alleviate the bottleneck of corpus-based lexicography as described above, the automatic process is required. Consequently, we have to consider two main problems: (1) how to represent contexts of the target word that are suitable for computation, and (2) how to automatically cluster those contexts into groups. For the former problem, Leacock et al. [1998] proposed the use of the topical context and the local context for word sense identification. The topical context is based on substantive words that co-occur with a given sense of the target word, whereas the local context tries to capture possible syntactic structures of the target word. Schütze [1998] addressed the problem of data sparseness, and then proposed the second-order representation. The idea is to create a context vector using the average of co-occurring word vectors. However, Purandare and Pedersen [2004] showed empirical evidences that using the second-order representation has no significant improvement over the first-order representation using terms as features. For the latter problem, a variety of unsupervised learning algorithms has also been examined, e.g. the Expectation-Maximization (EM) algorithm and its variants [Schütze, 1998] [de Marneffe and Dupont, 2004], and decision lists [Yarowsky, 1995].

In this paper, we propose an efficient technique to mine word senses for corpus-based lexicography. Our objective is to group the contexts of the target word into clusters according to their senses. The resulting clusters can provide a structure for analyzing the underlying senses of the target word found in a corpus. We present an unsupervised clustering algorithm named the spherical Gaussian EM algorithm, and provide a mechanism for finding a good initialization using the idea of Principal Component Analysis (PCA). Since it is commonly known that the EM algorithm often gets stuck in local optima depending on the initial random partitioning, applying PCA may help to avoid this problem. For the contextual representation, we focus on the vector space model, whose dimensions combine several useful information extracted from the contexts. Experimental results on two different data sets of polysemous words are given to show the performance of the proposed algorithm.

The remainder of this paper is organized as follows. Section 2 presents our contextual representation scheme. In Section 3, we describe the clustering algorithm in detail. Section 4 explains the data sets and the evaluation methods, and gives the experimental results. Finally, we conclude in Section 5 with some directions of future work.

2 Contextual Representation

Our contextual representation scheme is mainly based on the vector space model, which is introduced in the field of information retrieval. One major advantage of using this model is that any clustering algorithm can be applied to solve the problem. The idea is to represent the contexts

of the target word with feature vectors. Features are extracted from contextual words. Using different patterns of the contextual words can distinguish meanings of the target word in some extent. This is closely related to the idea that humans interpret meanings of words by the contexts in which they are used [Manning and Schütze, 1999]. The feature vector of the context i can be expressed as:

$$\mathbf{x}_i = (x_{i1}, x_{i2}, \ldots, x_{im})^T \ ,$$

where x_{ij} is the number of the contextual word j that co-occurs with the target word in the context i. In our work, we classify the contextual words into two types: topical and local contexts [Leacock *et al.*, 1998].

For the topical context, we extract open-class words (nouns, verbs, adjectives, and adverbs) that broadly co-occur with the target word. Since the co-occurring words are extracted from a wide window size, many common words are also included. Thus, we remove these common words by using a list of stopwords, consisting of 571 frequently used English words. Although this seems to be a very simple representation, it can still show different topical aspects of the target word. For example, as shown in [Yarowsky, 1995], if a polysemous word 'plant' occurs in a context with 'life', it is being used to express a different sense of 'plant' than if it occurred with 'manufacturing'.

For the local context, we consider both open-class words and close-class words (non-open-class words, e.g., prepositions, and determiners) that narrowly co-occur with the target word. We first extract the local open-class words occurring within a window size ± 3. We then extract the local close-class words selected in a narrower window size, say ± 2. In this case, we do not eliminate the stopwords, because they can also help to discriminate senses based on the syntactic structure of the target word. For example, 'a line from' does not suggest the same sense as 'in line for' [Leacock *et al.*, 1996].

3 Clustering Algorithm

3.1 Spherical Gaussian EM

We formulate our problem as a finite mixture model, and apply a variant of the EM algorithm for learning the model. We assume that each feature vector $\mathbf{x}_i$ is generated independently from a probability density function:

$$p(\mathbf{x}_i|\mathbf{\Theta}) = \sum_{j=1}^{k} p(\mathbf{x}_i|c_j;\boldsymbol{\theta}_j)P(c_j) \ , \tag{1}$$

where k is the number of clusters, the conditional probability density function $p(\mathbf{x}_i|c_j;\boldsymbol{\theta}_j)$ is the cluster density, and the prior probability $P(c_j)$ is the mixing parameter. Thus, we have the model parameters: $\mathbf{\Theta} = \{P(c_1), \ldots, P(c_k); \boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_k\}$.

To estimate the model parameters, we typically work with the maximum likelihood (ML) estimation. We introduce the variable of the missing data $\mathcal{Z} = \{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_n\}$, where $\mathbf{z}_i = (z_{i1}, \ldots, z_{ik})^T$

Algorithm 1: The spherical Gaussian EM algorithm.

begin

Initialization: Set $(\mathbf{z}_i)_j^{\langle 0 \rangle}$ from a partitioning of the data, and $t \leftarrow 0$.

repeat

E-step: For each $\mathbf{x}_i, 1 \leq i \leq n$ and $c_j, 1 \leq j \leq k$, find its new cluster index as:

$$(\mathbf{z}_i)_j^{\langle t+1 \rangle} = \begin{cases} 1, & \text{if } j^* = \text{argmax}_j \log(P^{\langle t \rangle}(c_j|\mathbf{x}_i; \boldsymbol{\theta}_j)), \\ 0, & \text{otherwise.} \end{cases}$$

M-step: Re-estimate the model parameters:

$$\begin{aligned} P(c_j)^{\langle t+1 \rangle} &= \frac{1}{n} \sum_{i=1}^{n} (\mathbf{z}_i)_j^{\langle t+1 \rangle} \\ \mathbf{m}_j^{\langle t+1 \rangle} &= \sum_{i=1}^{n} \mathbf{x}_i (\mathbf{z}_i)_j^{\langle t+1 \rangle} / \sum_{i=1}^{n} (\mathbf{z}_i)_j^{\langle t+1 \rangle} \\ \sigma^{2\langle t+1 \rangle} &= \frac{1}{n \cdot d} \sum_{i=1}^{n} \sum_{j=1}^{k} \parallel \mathbf{x}_i - \mathbf{m}_j \parallel^2 (\mathbf{z}_i)_j^{\langle t+1 \rangle} . \end{aligned}$$

until $\Delta \log L_c(\boldsymbol{\Theta}) < \delta$;

end

is a vector of binary indicator variables, and $(\mathbf{z}_i)_j = z_{ij} = 1$ if the vector $\mathbf{x}_i$ was generated from the cluster c_j; otherwise $z_{ij} = 0$. Thus, we can write the complete-data log likelihood as:

$$\log L_c(\boldsymbol{\Theta}) = \sum_{i=1}^{n} \sum_{j=1}^{k} z_{ij} \log(p(\mathbf{x}_i|c_j; \boldsymbol{\theta}_j) P(c_j)) . \tag{2}$$

We assume that the data samples are drawn from the multivariate normal density in $\Re^d$. We also assume that features are statistically independent, and the cluster c_j generates its members from the spherical Gaussian with the same covariance matrix. Thus, the parameter for each cluster c_j becomes $\boldsymbol{\theta}_j = (\mathbf{m}_j, \sigma^2 \mathbf{I}_d)$, and the cluster density can be calculated by:

$$p(\mathbf{x}_i|c_j; \boldsymbol{\theta}_j) = \frac{1}{(2\pi)^{d/2} \sigma^d} \exp\Big(- \frac{\parallel \mathbf{x}_i - \mathbf{m}_j \parallel^2}{2\sigma^2} \Big) . \tag{3}$$

Based on Bayes' theorem, the probability that the vector $\mathbf{x}_i$ falls into the cluster c_j can be defined as the product of the cluster density and the mixing parameter:

$$P(c_j|\mathbf{x}_i; \boldsymbol{\theta}_j) = p(\mathbf{x}_i|c_j; \boldsymbol{\theta}_j) P(c_j) . \tag{4}$$

Note that we estimate this probability with $\log(P(c_j|\mathbf{x}_i; \boldsymbol{\theta}_j))$ that corresponds to the $\log$ term in Equation 2. Algorithm 1 shows an outline of a simplified version of the EM algorithm for estimating the model parameters. From the assumptions of the model, we refer to this algorithm

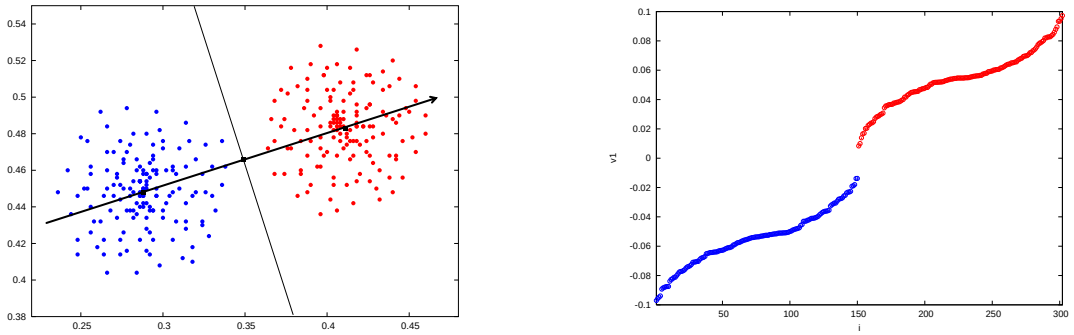

Figure 1: The partitioning result on a 2-dimensional data set containing 302 points distributed in 2 Gaussians (left), and corresponding projected values of the data points (right).

as the spherical Gaussian EM algorithm (or sGEM for short). The algorithm tries to maximize $\log L_c$ at very step, and iterates until convergence. For example, the algorithm terminates when $\Delta \log L_c < \delta$, where δ is a predefined threshold. In this paper, we set $\delta = 0.001$.

3.2 Finding Good Initialization

The sGEM algorithm we just described requires some approach to construct an initial partitioning of the data. A common approach is to randomly partition data samples into k disjoint subsets. However, this approach is pron to a bad local maximum. Here we describe a mechanism to find a good initialization for the sGEM algorithm based on the idea of PCA.

Given a data set represented by a matrix $\mathbf{M} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$, we obtain the covariance matrix $\mathbf{C} = (\mathbf{M} - \mathbf{m}e^T)(\mathbf{M} - \mathbf{m}e^T)^T = \mathbf{A}\mathbf{A}^T$, where $\mathbf{A} = \mathbf{M} - \mathbf{m}e^T$, $\mathbf{m}$ is the mean vector of $\mathbf{M}$, and e is the vector of ones, $(1, \ldots, 1)^T$. To split the data set into two sub-clusters, $k = 2$, we first find the principal direction derived from the covariance matrix, and project all the data samples onto this principal direction. Then, the splitting process can be performed based on the projected values.

By writing $\mathbf{A}$ in terms of Singular Value Decomposition (SVD), we get $\mathbf{A}\mathbf{A}^T = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T\mathbf{V}\mathbf{\Sigma}^T\mathbf{U}^T = \mathbf{U}\mathbf{\Sigma}\mathbf{I}\mathbf{\Sigma}^T\mathbf{U}^T = \mathbf{U}\mathbf{\Sigma}^2\mathbf{U}^T$. The principal direction is equivalent to the first left singular vector, denoted by $\mathbf{u}_1$. As a result, each $\mathbf{x}_i$ is assigned a label according to its projected value on $\mathbf{u}_1$. Since we can write $\mathbf{v}_1 = \mathbf{A}^T\mathbf{u}_1/\sigma_1$ where σ_1 is the largest singular value of $\mathbf{A}$, the projected values are equal to the elements of the first right singular vector. Thus, the two sub-clusters c_1 and c_2 are defined as:

$$c_1 = \{\mathbf{x}_i | \ (\mathbf{v}_1)_i \leq 0\}, c_2 = \{\mathbf{x}_i | \ (\mathbf{v}_1)_i > 0\} \ . \tag{5}$$

We can also interpret that $\mathbf{v}_1$ is the continuous solution of the cluster membership indicator vector (see [Ding and He, 2004] for more details). Figure 1 shows the partitioning result on a 2-dimensional data set containing 302 data points distributed in 2 Gaussians (left), and corresponding projected values of the data points (right).

We can start with all the feature vectors in a large single cluster, and proceed by recursively

Word	Sense	n_h	Percentage
line (noun)	product	2217	53.47%
	telephone connection	429	10.35%
	written or spoken text	404	9.74%
	division	374	9.02%
	cord	373	9.00%
	formation	349	8.42%
serve (verb)	supply with food	1814	41.43%
	hold an office	1272	29.05%
	function as something	853	19.48%
	provide a service	439	10.03%

Table 1: Data sets of polysemous words.

splitting the cluster into two sub-clusters. The stop splitting criterion is based on some heuristic, e.g. a predefined number of clusters. We then obtain a binary tree, whose leaf nodes form the resulting clusters. To keep the binary tree balanced, we select an un-split cluster to split by using the scatter value, measuring the average distance from the data points in the cluster to their centroid. This technique is also applied in the PDDP algorithm [Boley, 1998] and its refinement version [Kruengkrai *et al.*, 2003]. Finally, we can use these resulting clusters as the initial partitioning for the sGEM algorithm, which is more reliable than using random initialization.

4 Experiments

4.1 Data Sets and Experimental Setup

We used standard data sets for evaluating word sense disambiguation rather than making the concordance lines from scratch. According to [Leacock *et al.*, 1998], the *line* (noun) and *serve* (verb) data sets[3] were taken from the 1987-89 *Wall Street Journal* corpus and from the *American Printing House of the Blind* corpus. Each instance is composed of the sentence containing the target word and one sentence preceding it, and manually assigned a single sense from WordNet. Table 1 summarizes the senses of each word and their distributions found in the corresponding corpora.

For the purpose of comparison, we varied our algorithm by performing PCA and sGEM individually, and the combination of the two refereed to as PCA-sGEM (using PCA for finding the initial partitioning and running sGEM until convergence). In preprocessing, we extracted the contextual words as described in Section 2. We removed words that occurred less than 5, 10, and 25 times for the topical open-class words, the local open-class words, and the local close-class words, respectively. These thresholds are empirically selected from our experiments. We created vectors of instances by parsing their contexts according to extracted features. Finally, all the

[3]The data sets are publicly available at `http://www.d.umn.edu/~tpederse/data.html`.

feature vectors were weighted by using the tf·idf term weighting technique [Salton and Buckley, 1988].

4.2 Evaluation Methods

Since all instances are already categorized, we can perform evaluation by comparing clustering results with the true sense labels. In our experiments, we used the normalized mutual information (NMI) [Strehl and Ghosh, 2002]. When the number of clusters k differs from the actual number of senses g, this measure is very useful without a bias towards smaller clusters. By normalizing this criterion to take values between 0 and 1, the NMI can be calculated as follows:

$$NMI = \frac{\sum_{h,l} n_{h,l} \log(n \cdot n_{h,l}/n_h n_l)}{\sqrt{(\sum_h n_h \log(n_h/n))(\sum_l n_l \log(n_l/n))}} , \tag{6}$$

where n_h is the number of instances in the sense h, n_l is the number of instances in the cluster l, and $n_{h,l}$ is the number of instances in the sense h that is assigned to the cluster l. We can interpret that the NMI value is 1 when the results exactly match the true sense labels, and close to 0 for a random partitioning.

For an individual cluster, the cluster's quality can be investigated through the purity and entropy measures. The purity of the cluster c_l can be calculated by:

$$P(c_l) = \frac{1}{n_l} \max_h (n_{h,l}) , \tag{7}$$

and the entropy of the cluster c_l is defined as:

$$H(c_l) = \frac{1}{\log g} \sum_{h=1}^{g} \frac{n_{h,l}}{n_l} \log \left(\frac{n_{h,l}}{n_l} \right) . \tag{8}$$

Both measures are normalized to take values between 0 and 1. The entropy value near 0 indicates a good clustering result where the cluster is almost composed of one sense, while the entropy value near 1 indicates a bad clustering result where the cluster contains a uniform mixture of different senses.

4.3 Experimental Results

We now show experimental results obtained by performing variants of our proposed method. The results of sGEM are averaged over 10 trials of random initializations. PCA and PCA-sGEM are deterministic and produce the same clustering output.

Figure 2 (left) shows the clustering performance on the *line* data set for the algorithms considered. The horizontal axis indicates the number of clusters, while the vertical axis shows the NMI values. We clearly see that PCA performs better than sGEM and PCA-sGEM at $k = 2$, since PCA can split one large cluster, mostly containing instances of the 'product' sense. However, as the number of clusters increases, sGEM and PCA-sGEM constantly outperform PCA.

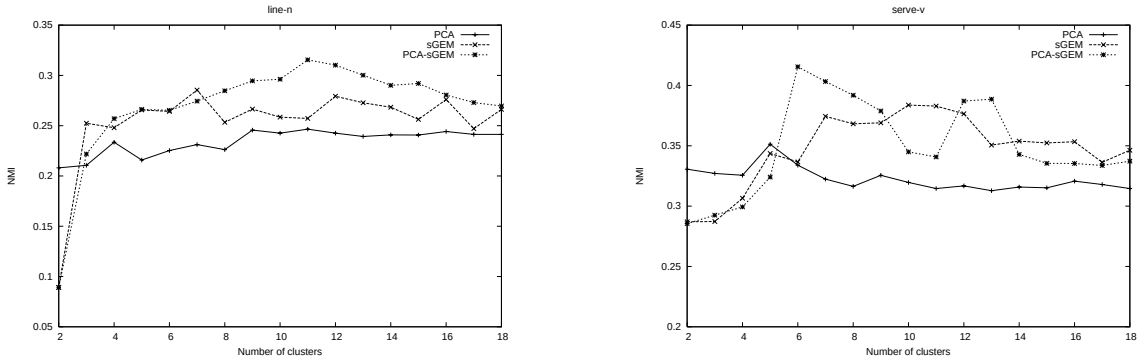

Figure 2: Clustering performance on the *line* data set (left), and the *serve* data set (right).

It is interesting to note that even sGEM simply uses the random initialization, it performs surprisingly well, compared to PCA. Estimating the initial partitioning with PCA as performed in PCA-sGEM can improve the performance with a reasonable gap.

Figure 2 (right) shows the clustering performance on the *serve* data set. PCA works well for small numbers of clusters, but the performance tends to decrease after 5 clusters. In contrast, sGEM gradually improves the performance, and significantly outperforms PCA after $k = 7$. The performance of PCA-sGEM is sometimes considerably better, and also sometimes worse. We hypnotize that it is due to an inappropriate initialization produced by PCA. Further refinement techniques may be required, e.g. k-way projections as described in [Ding and He, 2004].

We move on to demonstrate the clustering results more clearly by presenting confusion matrices. Due to the limited space, we only show the results performed by PCA-sGEM with $k = 12$. Our interest is that whether the algorithm can discover the underlying senses in the data sets or not. Columns of the confusion matrix show the actual senses of the target word, while rows show the predicted clusters. We just rank the rows according to the purity values. We anticipate that each cluster should be dominated by one sense.

Table 2 shows the confusion matrix for the *line* data set. We can see that every sense of the target word can be revealed. The 'cord' and 'division' senses are discovered in the clusters c_5 and c_9, respectively. Although the 'product' sense is divided into several clusters since it has a large number of instances, these clusters are quite pure, e.g. c_2, c_6, and c_{11}. The remaining senses, 'formation', 'phone', and 'text', are also clustered with some undesired noise of other senses. Table 3 shows the confusion matrix for the *serve* data set. Again, every sense of the word can be reasonably revealed with impressive purity and entropy values, e.g. c_0 for 'food', c_3 for 'office', c_{11} for 'function', and c_4 for 'service'.

c_j	$P(c_j)$	$H(c_j)$	cord	division	formation	phone	product	text
9	0.978	0.067	1	**133**	.	.	.	2
11	0.970	0.086	.	2	.	4	**193**	.
2	0.951	0.131	.	2	7	.	**194**	1
10	0.949	0.127	.	1	1	11	**243**	.
5	0.892	0.282	**149**	2	3	2	5	6
6	0.885	0.279	4	17	23	73	**1026**	16
7	0.880	0.240	.	2	.	7	**66**	.
3	0.802	0.372	1	3	4	36	**190**	3
0	0.613	0.655	33	8	7	**95**	6	6
1	0.535	0.754	14	10	**176**	64	33	32
8	0.353	0.889	57	69	61	51	188	**232**
4	0.219	0.986	114	125	67	86	73	106

Table 2: Confusion matrix for the *line* data set.

c_j	$P(c_j)$	$H(c_j)$	food	office	function	service
0	1.000	0.000	**154**	.	.	.
1	0.997	0.015	**316**	.	1	.
2	0.997	0.013	**392**	1	.	.
11	0.987	0.051	2	.	**148**	.
3	0.937	0.185	.	**134**	1	8
8	0.857	0.395	7	**275**	27	12
5	0.846	0.419	27	**336**	26	8
6	0.845	0.426	**714**	66	39	26
7	0.694	0.629	**109**	22	24	2
10	0.662	0.690	55	146	**472**	40
9	0.510	0.788	11	**153**	38	98
4	0.502	0.833	27	139	77	**245**

Table 3: Confusion matrix for the *serve* data set.

5 Conclusion and Future Work

This paper has described our first attempt to cope with the bottleneck in corpus-based lexicography. We present an efficient algorithm for clustering the contexts of the target word. The sGEM algorithm is an iterative optimization clustering algorithm that can be enhanced by combining the robust initialization method based on PCA. Preliminary results show that our algorithm can discover the clusters reflecting the underlying word senses found in the corpus. These results can illustrate the performance of the proposed algorithm, even though the window sizes of the contexts are relatively small.

Many issues of future work remain. We intend to develop the kernelized version of our algorithm using the idea of kernel PCA [Schölkopf *et al.*, 1998,] and the kernel k-means algorithm [Dhillon *et al.*, 2004]. For the real-world application, we will implement a graphic user interface for our software. This can help the lexicographer to explore and select good examples for each word sense. Not only these examples can be used for describing the sense itself, but also they can be used as the *seed* examples for semi-supervised and supervised learning algorithms.

References

[Boley, 1998] Daniel Boley. Principal direction divisive partitioning. *Data Mining and Knowledge Discovery*, 2(4):325–344, 1998.

[Charoenporn *et al.*, 2004] Thatsanee Charoenporn, Canasai Kruengkrai, Virach Sornlertlamvanich, and Hitoshi Isahara. Acquiring semantic information in the tcl's computational lexicon. In *The 4th Workshop on Asian Language Resources (ALR-04)*, 2004.

[de Marneffe and Dupont, 2004] Marie-Catherine de Marneffe and Pierre Dupont. Comparative study of statistical word sense discrimination. In *Proceedings of the 7th International Conference on Statistical Analysis of Textual Data*, pages 270–281, 2004.

[Dhillon *et al.*, 2004] Inderjit S. Dhillon, Yuqiang Guan, and Brian Kulis. Kernel k-means, spectral clustering and normalized cuts. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2004.

[Ding and He, 2004] Chris Ding and Xiaofeng He. K-means clustering via principal component analysis. In *Proceedings of the 21st International Conference on Machine Learning*, 2004.

[Kruengkrai *et al.*, 2003] Canasai Kruengkrai, Virach Sornlertlamvanich, and Hitoshi Isahara. Refining a divisive partitioning algorithm for unsupervised clustering. In *Proceedings of the 3rd International Conference on Hybrid Intelligent Systems*, pages 535–542, 2003.

[Leacock *et al.*, 1996] Claudia Leacock, Geoffrey Towell, and Ellen M. Voorhees. *Towards building contextual representations of word senses using statistical models*. In B. Boguraev and J. Pustejovsky, editors, Corpus Processing for Lexical Acquisition. MIT Press, Cambridge, 1996.

[Leacock *et al.*, 1998] Claudia Leacock, George A. Miller, and Martin Chodorow. Using corpus statistics and wordnet relations for sense identification. *Computational Linguistics*, 24(1):147–165, 1998.

[Manning and Schütze, 1999] Christopher D. Manning and Hinrich Schütze. *Foundations of statistical natural language processing*. MIT Press. Cambridge, MA, 1999.

[Palingoon *et al.*, 2002] Pornpimon Palingoon, Pornchan Chantanapraiwan, Supranee Theerawattanasuk, Thatsanee Charoenporn, and Virach Sornlertlamvanich. Qualitative and quantitative approaches in bilingual corpus-based dictionary. In *The 5th Symposium on Natural Language Processing 2002 & Oriental COCOSDA Workshop 2002*, 2002.

[Purandare and Pedersen, 2004] Amruta Purandare and Ted Pedersen. Word sense discrimination by clustering contexts in vector and similarity spaces. In *Proceedings of CoNLL-2004*, pages 41–48, 2004.

[Salton and Buckley, 1988] Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing and Management: an International Journal*, 24(5):513–523, 1988.

[Schölkopf *et al.*, 1998] Bernhard Schölkopf, Alexander Smola, and Klaus-Rober Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1998.

[Stevenson, 2003] Mark Stevenson. *Word Sense Disambiguation: The Case for Combinations of Knowledge Sources*. CSLI Publications, California, 2003.

[Strehl and Ghosh, 2002] Alexander Strehl and Joydeep Ghosh. Cluster ensembles - a knowledge reuse framework for combining multiple partitions. *Journal on Machine Learning Research*, 3:583–617, 2002.

[Schütze, 1998] Hinrich Schütze. Automatic word sense discrimination. *Computational Linguistics*, 24(1):97–124, 1998.

[Yarowsky, 1995] David Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, pages 189–196, 1995.