# PLSI Utilization for Automatic Thesaurus Construction

Masato Hagiwara, Yasuhiro Ogawa, and Katsuhiko Toyama

Graduate School of Information Science, Nagoya University,
Furo-cho, Chikusa-ku, Nagoya, JAPAN 464-8603
{hagiwara, yasuhiro, toyama}@kl.i.is.nagoya-u.ac.jp

**Abstract.** When acquiring synonyms from large corpora, it is important to deal not only with such surface information as the context of the words but also their latent semantics. This paper describes how to utilize a latent semantic model PLSI to acquire synonyms automatically from large corpora. PLSI has been shown to achieve a better performance than conventional methods such as tf·idf and LSI, making it applicable to automatic thesaurus construction. Also, various PLSI techniques have been shown to be effective including: (1) use of Skew Divergence as a distance/similarity measure; (2) removal of words with low frequencies, and (3) multiple executions of PLSI and integration of the results.

## 1 Introduction

Thesauri, dictionaries in which words are arranged according to meaning, are one of the most useful linguistic sources, having a broad range of applications, such as information retrieval and natural language understanding. Various thesauri have been constructed so far, including *WordNet* [6] and *Bunruigoihyo* [14]. Conventional thesauri, however, have largely been compiled by groups of language experts, making the construction and maintenance cost very high. It is also difficult to build a domain-specific thesaurus flexibly. Thus it is necessary to construct thesauri automatically using computers.

Many studies have been done for automatic thesaurus construction. In doing so, synonym acquisition is one of the most important techniques, although a thesaurus generally includes other relationships than synonyms (e.g., hypernyms and hyponyms). To acquire synonyms automatically, contextual features of words, such as co-occurrence and modification are extracted from large corpora and often used. Hindle [7], for example, extracted verb-noun relationships of subjects/objects and their predicates from a corpus and proposed a method to calculate similarity of two words based on their mutual information. Although methods based on such raw co-occurrences are simple yet effective, in a naive implementation some problems arise: namely, noises and sparseness. Being a collection of raw linguistic data, a corpus generally contains meaningless information, i.e., noises. Also, co-occurrence data extracted from corpora are often very sparse, making them inappropriate for similarity calculation, which is also known as the "zero frequency problem." Therefore, not only surface information but also latent semantics should be considered when acquiring synonyms from large corpora.

Several latent semantic models have been proposed so far, mainly for information retrieval and document indexing. The most commonly used and prominent ones are Latent Semantic Indexing (LSI) [5] and Probabilistic LSI (PLSI) [8]. LSI is a geometric model based on the vector space model. It utilizes singular value decomposition of the co-occurrence matrix, an operation similar to principal component analysis, to automatically extract major components that contribute to the indexing of documents. It can alleviate the noise and sparseness problems by a dimensionality reduction operation, that is, by removing components with low contributions to the indexing. However, the model lacks firm, theoretical basis [9] and the optimality of inverse document frequency (idf) metric, which is commonly used to weight elements, has yet to be shown [13].

On the contrary, PLSI, proposed by Hofmann [8], is a probabilistic version of LSI, where it is formalized that documents and terms co-occur through a latent variable. PLSI puts no assumptions on distributions of documents or terms, while LSI performs optimal model fitting, assuming that documents and terms are under Gaussian distribution [9]. Moreover, ad hoc weighting such as idf is not necessary for PLSI, although it is for LSI, and it is shown experimentally to outperform the former model [8].

This study applies the PLSI model to the automatic acquisition of synonyms by estimating each word's latent meanings. First, a number of verb-noun pairs were collected from a large corpus using heuristic rules. This operation is based on the assumption that semantically similar words share similar contexts, which was also employed in Hindle's work [7] and has been shown to be considerably plausible. Secondly, the co-occurrences obtained in this way were fit into the PLSI model, and the probability distribution of latent classes was calculated for each noun. Finally, similarity for each pair of nouns can be calculated by measuring the distances or the similarity between two probability distributions using an appropriate distance/similarity measure. We then evaluated and discussed the results using two evaluation criteria, discrimination rates and scores.

This paper also discusses basic techniques when applying PLSI to the automatic acquisition of synonyms. In particular, the following are discussed from methodological and experimental views: (1) choice of distance/similarity measures between probability distributions; (2) filtering words according to their frequencies of occurrence; and (3) multiple executions of PLSI and integration of the results.

This paper is organized as follows: in Sect. 2 a brief explanation of the PLSI model and calculation is provided, and Sect. 3 outlines our approach. Sect. 4 shows the results of comparative experiments and basic techniques. Sect. 5 concludes this paper.

## 2 The PLSI Model

This section provides a brief explanation of the PLSI model in information retrieval settings. The PLSI model, which is based on the aspect model, assumes that document $d$ and term $w$ co-occur through latent class $z$, as shown in Fig. 1 (a).

The co-occurrence probability of documents and terms is given by:

$$P(d, w) = P(d) \sum_z P(z|d) P(w|z). \tag{1}$$

Note that this model can be equivalently rewritten as

$$P(d, w) = \sum_z P(z) P(d|z) P(w|z), \tag{2}$$

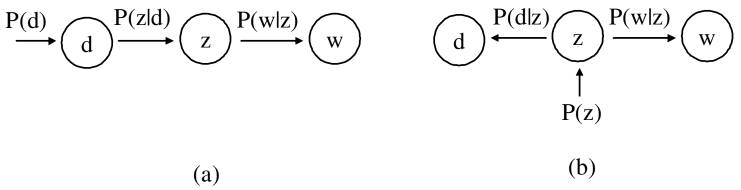


**Fig. 1.** PLSI model asymmetric (a) and symmetric (b) parameterization

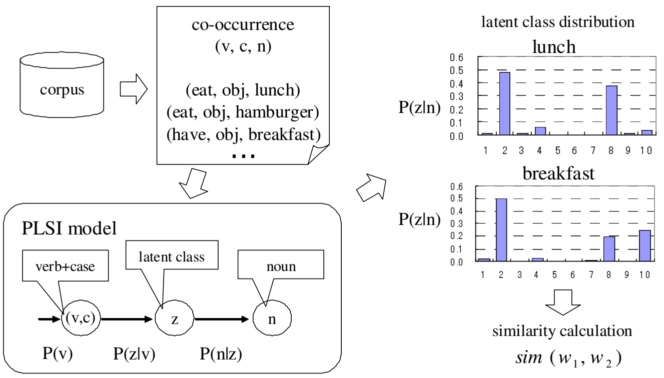


**Fig. 2.** Outline of our approach

whose graphical model representation is shown in Fig. 1 (b). This is a symmetric parameterization with respect to documents and terms. The latter parameterization is used in the experiment section because of its simple implementation.

Theoretically, probabilities $P(d)$, $P(z|d)$, $P(w|z)$ are determined by maximum likelihood estimation, that is, by maximizing the likelihood of document term co-occurrence:

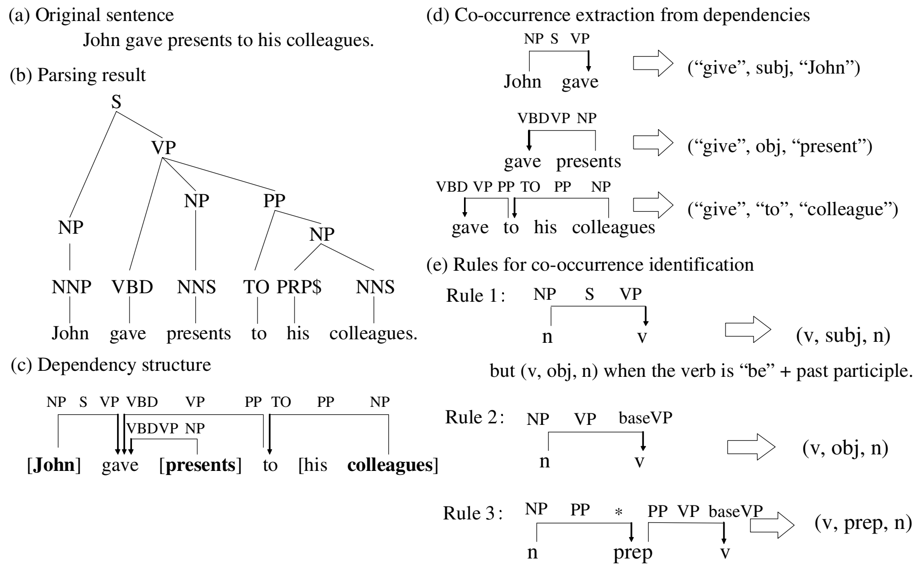
$$L = \sum_{d,w} N(d,w) \log P(d,w), \tag{3}$$

where $N(d,w)$ is the frequency document $d$ and term $w$ co-occur.

While the co-occurrence of document $d$ and term $w$ in the corpora can be observed directly, the contribution of latent class $z$ cannot be directly seen in this model. For the maximum likelihood estimation of this model, the EM algorithm [1], which is used for the estimation of systems with unobserved (latent) data, is used. The EM algorithm performs the estimation iteratively, similar to the steepest descent method.

## 3 Approach

The original PLSI model, as described above, deals with co-occurrences of documents and terms, but it can also be applied to verbs and nouns in the corpora. In this way, latent



(a) Original sentence
John gave presents to his colleagues.
(b) Parsing result
S
VP
NP
NP
PP
NP
NNP VBD NNS TO PRP$ NNS
John gave presents to his colleagues.
(c) Dependency structure
NP S VP VBD VP PP TO PP NP
VBDVP NP
[John] gave [presents] to [his colleagues]
(d) Co-occurrence extraction from dependencies
NP S VP
John gave
("give", subj, "John")
VBDVP NP
gave presents
("give", obj, "present")
VBD VP PP TO PP NP
gave to his colleagues
("give", "to", "colleague")
(e) Rules for co-occurrence identification
Rule 1: NP S VP
n v
(v, subj, n)
but (v, obj, n) when the verb is "be" + past participle.
Rule 2: NP VP baseVP
n v
(v, obj, n)
Rule 3: NP PP * PP VP baseVP
n prep v
(v, prep, n)

**Fig. 3.** Co-occurrence extraction

class distribution, which can be interpreted as latent "meaning" corresponding to each noun, is obtained. Semantically similar words are then obtained accordingly, because words with similar meaning have similar distributions. Fig. 2 outlines our approach, and the following subsections provide the details.

## 3.1 Extraction of Co-occurrence

We adopt triples $(v, c, n)$ extracted from the corpora as co-occurrences fit into the PLSI model, where $v$, $c$, and $n$ represent a verb, case/preposition, and a noun, respectively. The relationships between nouns and verbs, expressed by $c$, include case relation (subject and object) as well as what we call here "prepositional relation," that is, a co-occurrence through a preposition. Take the following sentence for example:

John gave presents to his colleagues.

First, the phrase structure (Fig. 3(b)) is obtained by parsing the original sentence (Fig. 3(a)). The resulting tree is then used to derive the dependency structure (Fig. 3(c)), using Collins' method [4]. Note that dependencies in baseNPs (i.e., noun phrases that do not contain NPs as their child constituents, shown as the groups of words enclosed by square brackets in Fig. 3(c)), are ignored. Also, we introduced baseVPs, that is, sequences of verbs [1], modals (MD), or adverbs (RB), of which the last word must be a verb. BaseVPs simplify the handling of sequences of verbs such as "might not be"

[1] Ones expressed as VB, VBD, VBG, VBN, VBP, and VBZ by the Penn Treebank POS tag set [15].

and "is always complaining." The last word of a baseVP represents the entire baseVP to which it belongs. That is, all the dependencies directed to words in a baseVP are redirected to the last verb of the baseVP.

Finally, co-occurrences are extracted and identified by matching the dependency patterns and the heuristic rules for extraction, which are all listed in Fig. 3 (e). For example, since the label of the dependency "John" $\rightarrow$"gave" is "NP S VP", the noun "John" is identified as the subject of the verb "gave" (Fig. 3(d)). Likewise, the dependencies "presents"$\rightarrow$"gave" and "his colleagues"$\rightarrow$"to"$\rightarrow$"gave" are identified as a verb-object relation and prepositional relation through "to".

A simple experiment was conducted to test the effectiveness of this extraction method, using the corpus and the parser mentioned in the experiment section. Co-occurrence extraction was performed for the 50 sentences randomly extracted from the corpus, and precision and recall turned out to be 88.6% and 78.1%, respectively. In this context, precision is more important than recall because of the substantial size of the corpus, and some of the extraction errors result from parsing error caused by the parser, whose precision is claimed to be around 90% [2]. Therefore, we conclude that this method and its performance are sufficient for our purpose.

### 3.2 Applying PLSI to Extracted Co-occurence Data

While the PLSI model deals with dyadic data $(d, w)$ of document $d$ and term $w$, the co-occurrences obtained by our method are triples $(v, c, n)$ of a verb $v$, a case/preposition $c$, and a noun $n$. To convert these triples into dyadic data (pairs), verb $v$ and case/preposition $c$ are paired as $(v, c)$ and considered a new "virtual" verb $v$. This enables it to handle the triples as the co-occurrence $(v, n)$ of verb $v$ and noun $n$ to which the PLSI model becomes applicable. Pairing verb $v$ and case/preposition $c$ also has a benefit that such phrasal verbs as "look for" or "get to" can be naturally treated as a single verb.

After the application of PLSI, we obtain probabilities $P(z), P(v|z)$, and $P(n|z)$. Using Bayes theorem, we then obtain $P(z|n)$, which corresponds to the latent class distribution for each noun. In other words, distribution $P(z|n)$ represents the features of meaning possessed by noun $n$. Therefore, we can calculate the similarity between nouns $n_1$ and $n_2$ by measuring the distance or similarity between the two corresponding distribution, $P(z|n_1)$ and $P(z|n_2)$, using an appropriate measure. The choice of measure affects the synonym acquisition results and experiments on comparison of distance/similarity measures are detailed in Sect. 4.3.

## 4 Experiments

This section includes the results of comparison experiments and those on the basic PLSI techniques.

### 4.1 Conditions

The automatic acquisition of synonyms was conducted according to the method described in Sect. 3, using WordBank (190,000 sentences, 5 million words) [3] as a corpus. Charniak's parser [2] was used for parsing and TreeTagger [16] for stemming. A total of 702,879 co-occurrences was extracted by the method described in Sect. 3.1.

When using EM algorithm to implement PLSI, overfitting, which aggravates the performance of the resultant language model, occasionally occurs. We employed the tempered EM (TEM) [8] algorithm, instead of a naive one, to avoid this problem. TEM algorithm is closely related to the deterministic annealing EM (DAEM) algorithm [17], and helps avoid local extrema by introducing inverse temperature $\beta$. The parameter was set to $\beta = 0.86$, considering the results of the preliminary experiments.

As the similarity/distance measure and frequency threshold $t_f$, Skew Divergence ($\alpha$ = 0.99) and $t_f = 15$ were employed in the following experiments in response to the results from the experiments described in Sects. 4.3 and 4.5. Also, because estimation by EM algorithm is started from the random parameters and consequently the PLSI results change every time it is executed, the average performance of the three executions was recorded, except in Sect. 4.6.

## 4.2 Measures for Performance

The following two measures, discrimination rate and scores, were employed for the evaluation of automated synonym acquisition.

*Discrimination rate* Discrimination rate, originally proposed by Kojima et al. [10], is the rate (percentage) of pairs $(w_1, w_2)$ whose degree of association between two words $w_1, w_2$ is successfully discriminated by the similarity derived by a method. Kojima et al. dealt with three-level discrimination of a pair of words, that is, highly related (synonyms or nearly synonymous), moderately related (a certain degree of association), and unrelated (irrelevant). However, we omitted the moderately related level and limited the discrimination to two-level: high or none, because of the high cost of preparing a test set that consists of moderately related pairs.

The calculation of discrimination rate follows these steps: first, two test sets, one of which consists of highly related word pairs and the other of unrelated ones, were prepared, as shown in Fig. 4. The similarity between $w_1$ and $w_2$ is then calculated for each pair $(w_1, w_2)$ in both test sets via the method under evaluation, and the pair is labeled highly related when similarity exceeds a given threshold $t$ and unrelated when the similarity is lower than $t$. The number of pairs labeled highly related in the highly related test set and unrelated in the unrelated test set are denoted $n_a$ and $n_b$, respectively. The discrimination rate is then given by:

$$\frac{1}{2}\left(\frac{n_a}{N_a} + \frac{n_b}{N_b}\right), \tag{4}$$

where $N_a$ and $N_b$ are the numbers of pairs in highly related and unrelated test sets, respectively. Since the discrimination rate changes depending on threshold $t$, maximum value is adopted by varying $t$.

We created a highly related test set using the synonyms in WordNet [6]. Pairs in a unrelated test set were prepared by first choosing two words randomly and then confirmed by hand whether the consisting two words are truly irrelevant. The numbers of pairs in the highly and unrelated test sets are 383 and 1,124, respectively.

| highly related | unrelated |
|---|---|
| (answer, reply) | (animal, coffee) |
| (phone, telephone) | (him, technology) |
| (sign, signal) | (track, vote) |
| (concern, worry) | (path, youth) |
| ⋮ | ⋮ |

**Fig. 4.** Test-sets for discrimination rate calculation

**Table 5.** Procedure for score calculation

base word: computer

| rank | synonym | $sim$ | $sim^*$ | rel.($p$) | $p \cdot sim^*$ |
|---|---|---|---|---|---|
| 1 | equipment | 0.6 | 0.3 | B(0.5) | 0.15 |
| 2 | machine | 0.4 | 0.2 | A(1.0) | 0.20 |
| 3 | Internet | 0.4 | 0.2 | B(0.5) | 0.10 |
| 4 | spray | 0.4 | 0.2 | C(0.0) | 0.00 |
| 5 | PC | 0.2 | 0.1 | A(1.0) | 0.10 |
| total | | 2.0 | 1.0 | | 0.55 |

*Scores* We propose a score which is similar to precision used for information retrieval evaluation, but different in that it considers the similarity of words. This extension is based on the notion that the more accurately the degrees of similarity are assigned to the results of synonym acquisition, the higher the score values should be.

Described in the following, along with Table 5, is the procedure for score calculation. Table 5 shows the obtained synonyms and their similarity with respect to the base word "computer." Results are obtained by calculating the similarity between the base word and each noun, and ranking all the nouns in descending order of similarity $sim$. The highest five are used for calculations in this example.

The range of similarity varies based on such factors as the employed distance/ similarity measure, which unfavorably affects the score value. To avoid this, the values of similarity are normalized such that their sum equals one, as shown in the column $sim^*$ in Fig. 5. Next, the relevance of each synonym to the base word is checked and evaluated manually, giving them three-level grades: highly related (A), moderately related (B), and unrelated (C), and relevance scores $p = 1.0, 0.5, 0.0$ are assigned for each grade, respectively ("rel.($p$)" column in Fig. 5). Finally, each relevance score $p$ is multiplied by corresponding similarity $sim^*$ , and the products (the $p \cdot sim^*$ column in Fig. 5) are totaled and then multiplied by 100 to obtain a score, which is 55 in this case. In actual experiments, thirty words chosen randomly were adopted as base words, and the average of the scores of all base words was employed. Although this example considers only the top five words for simplicity, the top twenty words were used for evaluation in the following experiments.

## 4.3 Distance/Similarity Measures of Probability Distribution

The choice of distance measure between two latent class distributions $P(z|n_i)$, $P(z|n_j)$ affects the performance of synonym acquisition. Here we focus on the following seven distance/similarity measures and compare their performance.

- Kullback-Leibler (KL) divergence [12]: $KL(p \,||\, q) = \sum_x p(x) \log(p(x)/q(x))$
- Jensen-Shannon (JS) divergence [12]: $JS(p, q) = \{KL(p \,||\, m) + KL(q \,||\, m)\}/2$, $m = (p + q)/2$
- Skew Divergence [11]: $s_\alpha(p \,||\, q) = KL(p \,||\, \alpha q + (1 - \alpha)p)$
- Euclidean distance: $\mathrm{euc}(p, q) = ||p - q||$
- $L_1$ distance: $L_1(p, q) = \sum_x |p(x) - q(x)|$

– Inner product: $p \cdot q = \sum_x p(x)q(x)$
– Cosine: $\cos(p, q) = (p \cdot q)/||p|| \cdot ||q||$

KL divergence is widely used for measuring the distance between two probability distributions. However, it has such disadvantages as asymmetricity and zero frequency problem, that is, if there exists $x$ such that $p(x) \neq 0,\ q(x) = 0$, the distance is not defined. JS divergence, in contrast, is considered the symmetrized KL divergence and has some favorable properties: it is bounded [12] and does not cause the zero frequency problem. Skew Divergence, which has recently been receiving attention, has also solved the zero frequency problem by introducing parameter $\alpha$ and mixing the two distributions. It has shown that Skew Divergence achieves better performance than the other measures [11]. The other measures commonly used for calculation of the similarity/distance of two vectors, namely Euclidean distance, $L_1$ distance (also called Manhattan Distance), inner product, and cosine, are also included for comparison.

Notice that the first five measures are of distance (the more similar $p$ and $q$, the lower value), whereas the others, inner product and cosine, are of similarity (the more similar $p$ and $q$, the higher value). We converted distance measure $D$ to a similarity measure $sim$ by the following expression:

$$sim(p, q) = \exp\{-\lambda D(p, q)\}, \tag{5}$$

inspired by Mochihashi and Matsumoto [13]. Parameter $\lambda$ was determined in such a way that the average of $sim$ doesn't change with respect to $D$. Because KL divergence and Skew Divergence are asymmetric, the average of both directions (e.g. for KL divergence, $\frac{1}{2}(KL(p||q) + KL(q||p))$) is employed for the evaluation.

Figure 6 shows the performance (discrimination rate and score) for each measure. It can be seen that Skew Divergence with parameter $\alpha = 0.99$ shows the highest performance of the seven, with a slight difference to JS divergence. These results, along with several studies, also show the superiority of Skew Divergence. In contrast, measures for vectors such as Euclidean distance achieved relatively poor performance compared to those for probability distributions.

## 4.4 Word Filtering by Frequencies

It may be difficult to estimate the latent class distributions for words with low frequencies because of a lack of sufficient data. These words can be noises that may degregate the results of synonym acquisition. Therefore, we consider removing such words with low frequencies before the execution of PLSI improves the performance. More specifically, we introduced threshold $t_f$ on the frequency, and removed nouns $n_i$ such that $\sum_j \mathrm{tf}^i_j < t_f$ and verbs $v_j$ such that $\sum_i \mathrm{tf}^i_j < t_f$ from the extracted co-occurrences.

The discrimination rate change on varying threshold $t_f$ was measured and shown in Fig. 7 for $d = 100, 200$, and 300. In every case, the rate increases with a moderate increase of $t_f$, which shows the effectiveness of the removal of low frequency words. We consequently fixed $t_f = 15$ in other experiments, although this value may depend on the corpus size in use.

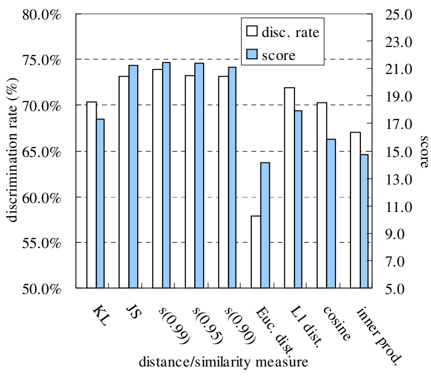


**Fig. 6.** Performances of distance/similarity measures

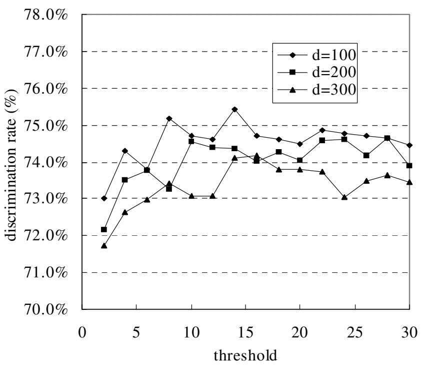


**Fig. 7.** Discrimination rate measured by varying threshold $t_f$

## 4.5 Comparison Experiments with Conventional Methods

Here the performances of PLSI and the following conventional methods are compared. In the following, $N$ and $M$ denote the numbers of nouns and verbs, respectively.

– tf: The number of co-occurrence $\mathrm{tf}^i_j$ of noun $n_i$ and verb $v_j$ is used directly for similarity calculation. The corresponding vector $\boldsymbol{n}_i$ to noun $n_i$ is given by:

$$\boldsymbol{n}_i = {}^t[\mathrm{tf}^i_1 \ \ \mathrm{tf}^i_2 \ \ ... \ \ \mathrm{tf}^i_M]. \tag{6}$$

– tf·idf: The vectors given by tf method are weighted by idf. That is,

$$\boldsymbol{n}^*_i = {}^t[\mathrm{tf}^i_1 \cdot \mathrm{idf}_1 \ \ \mathrm{tf}^i_2 \cdot \mathrm{idf}_2 \ ... \ \mathrm{tf}^i_M \cdot \mathrm{idf}_M], \tag{7}$$

where $\mathrm{idf}_j$ is given by

$$\mathrm{idf}_j = \frac{\log(N/\mathrm{df}_j)}{\max_k \log(N/\mathrm{df}_k)}, \tag{8}$$

using $\mathrm{df}_j$, the number of distinct nouns that co-occur with verb $v_j$.

– tf+LSI: A co-occurrence matrix $X$ is created using vectors $\boldsymbol{n}_i$ defined by tf:

$$X = [\boldsymbol{n}_1 \ \boldsymbol{n}_2 \ ... \ \boldsymbol{n}_N], \tag{9}$$

to which LSI is applied.

– tf·idf+LSI : A co-occurrence matrix $X^*$ is created using vectors $\boldsymbol{n}^*_i$ defined by tf·idf:

$$X^* = [\boldsymbol{n}^*_1 \ \boldsymbol{n}^*_2 \ ... \ \boldsymbol{n}^*_N], \tag{10}$$

to which LSI is applied.

– Hindle's method: The method described in [7] is used. Whereas he deals only with subjects and objects as verb-noun co-occurrence, we used all the kinds of co-occurrence mentioned in Sect. 3.1, including prepositional relations.

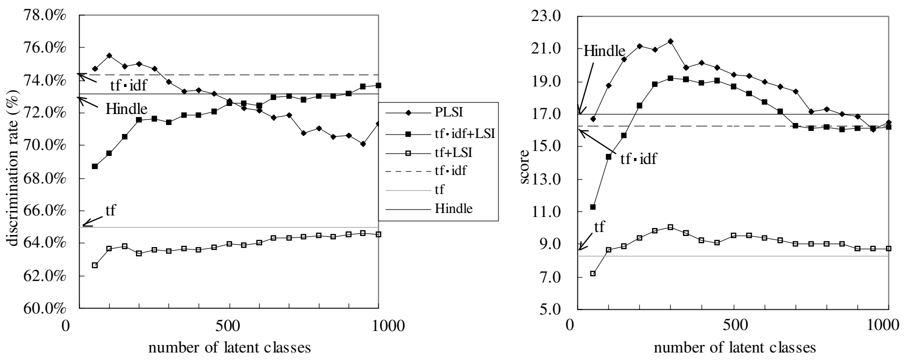


**Fig. 8.** Performances of PLSI and conventional methods

The values of discrimination rate and scores are calculated for PLSI as well as the methods described above, and the results are shown in Fig. 8. Because the number of latent classes $d$ must be given beforehand for PLSI and LSI, the performances of the latent semantic models are measured varying $d$ from 50 to 1,000 with a step of 50. The cosine measure is used for the similarity calculation of tf, tf·idf, tf+LSI, and tf·idf+LSI.

The results reveal that the highest discrimination rate is achieved by PLSI, with the latent class number of approximately 100, although LSI overtakes with an increase of $d$. As for the scores, the performance of PLSI stays on top for almost all the values of $d$, strongly suggesting the superiority of PLSI over the conventional method, especially when $d$ is small, which is often.

The performances of tf and tf+LSI, which are not weighted by idf, are consistently low regardless of the value of $d$. PLSI and LSI distinctly behave with respect to $d$, especially in the discrimination rate, whose cause require examination and discussion.

## 4.6 Integration of PLSI Results

In maximum likelihood estimation by EM algorithm, the initial parameters are set to values chosen randomly, and likelihood is increased by an iterative process. Therefore, the results are generally local extrema, not global, and they vary every execution, which is unfavorable. To solve this problem, we propose to execute PLSI several times and integrate the results to obtain a single one.

To achieve this, PLSI is executed several times for the same co-occurrence data obtained via the method described in Sect. 3.1. This yields $N$ values of similarity $sim_1(n_i, n_j), ..., sim_N(n_i, n_j)$ for each noun pair $(n_i, n_j)$. These values are integrated using one of the following four schemes to obtain a single value of similarity $\overline{sim}(n_i, n_j)$.

- arithmetic mean: $\overline{sim}(n_i, n_j) = \frac{1}{N}\sum_{k=1}^{N} sim_k(n_i, n_j)$
- geometric mean:$\overline{sim}(n_i, n_j) = \sqrt[N]{\prod_{k=1}^{N} sim_k(n_i, n_j)}$
- maximum: $\overline{sim}(n_i, n_j) = \max_k sim_k(n_i, n_j)$
- minimum: $\overline{sim}(n_i, n_j) = \min_k sim_k(n_i, n_j)$

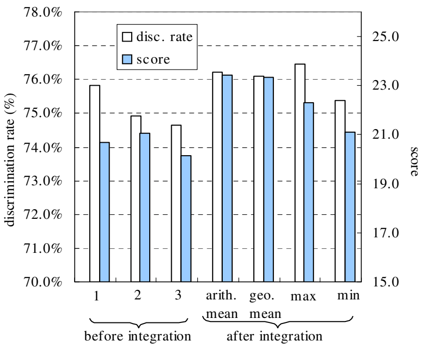


**Fig. 9.** Integration result for $N = 3$

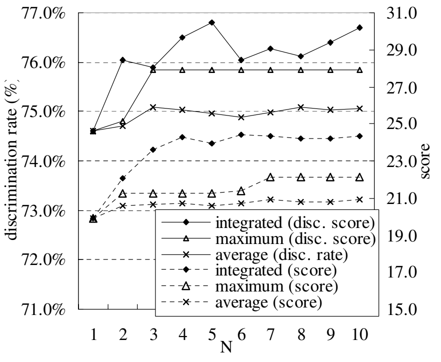


**Fig. 10.** Integration results varying $N$

Integration results are shown in Fig. 9, where the three sets of performance on the left are the results of single PLSI executions, i.e., before integration. On the right are the results after integration by the four schemes. It can be observed that integration improves the performance. More specifically, the results after integration are as good or better than any of the previous ones, except when using the minimum as a scheme.

An additional experiment was conducted that varied $N$ from 1 to 10 to confirm that such performance improvement is always achieved by integration. Results are shown in Fig. 10, which includes the average and maximum of the $N$ PLSI results (unintegrated) as well as the performance after integration using arithmetic average as the scheme. The results show that the integration consistently improves the performance for all $2 \leq N \leq 10$. An increase of the integration performance was observed for $N \leq 5$, whereas increases in the average and maximum of the unintegrated results were relatively low. It is also seen that using $N > 5$ has less effect for integration.

## 5 Conclusion

In this study, automatic synonym acquisition was performed using a latent semantic model PLSI by estimating the latent class distribution for each noun. For this purpose, co-occurrences of verbs and nouns extracted from a large corpus were utilized. Discrimination rates and scores were used to evaluate the current method, and it was found that PLSI outperformed such conventional methods as tf·idf and LSI. These results make PLSI applicable for automatic thesaurus construction. Moreover, the following techniques were found effective: (1) employing Skew Divergence as the distance/similarity measure between probability distributions; (2) removal of words with low frequencies, and (3) multiple executions of PLSI and integration of the results.

As future work, the automatic extraction of the hierarchical relationship of words also plays an important role in constructing thesauri, although only synonym relationships were extracted this time. Many studies have been conducted for this purpose, but extracted hyponymy/hypernymy relations must be integrated in the synonym relations to construct a single thesaurus based on tree structure. The characteristics of the latent class distributions obtained by the current method may also be used for this purpose.

In this study, similarity was calculated only for nouns, but one for verbs can be obtained using an identical method. This can be achieved by pairing noun $n$ and case / preposition $c$ of co-occurrence $(v, c, n)$, not $v$ and $c$ as previously done, and executing PLSI for the dyadic data $(v, (c, n))$. By doing this, the latent class distributions for each verb $v$, and consequently the similarity between them, are obtained.

Moreover, although this study only deals with verb-noun co-occurrences, other information such as adjective-noun modifications or descriptions in dictionaries may be used and integrated. This will be an effective way to improve the performance of automatically constructed thesauri.

## References

1. Bilmes, J. 1997. A gentle tutorial on the EM algorithm and its application to parameter estimation for gaussian mixture and hidden markov models. *Technical Report* ICSI-TR-97-021, International Computer Science Institute (ICSI), Berkeley, CA.
2. Charniak, E. 2000. A maximum-entropy-inspired parser. NAACL 1, 132–139.
3. Collins. 2002. Collins Cobuild Major New Edition CD-ROM. HarperCollins Publishers.
4. Collins, M. 1996. A new statistical parser based on bigram lexical dependencies. *Proc. of 34th ACL*, 184–191.
5. Deerwester, S., et al. 1990. Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science*, 41(6):391–407.
6. Fellbaum, C. 1998. *WordNet: an electronic lexical database.* MIT Press.
7. Hindle, D. 1990. Noun classification from predicate-argument structures. *Proc. of the 28th Annual Meeting of the ACL*, 268–275.
8. Hofmann, T. 1999. Probabilistic Latent Semantic Indexing. *Proc. of the 22nd International Conference on Research and Development in Information Retrieval (SIGIR '99)*, 50–57.
9. Hofmann, T. 2001. Unsupervised Learning by Probabilistic Latent Semantic Analysis. *Machine Learning*, 42:177–196.
10. Kojima, K., et. al. 2004. Existence and Application of Common Threshold of the Degree of Association. *Proc. of the Forum on Information Technology (FIT2004)* F-003.
11. Lee, L. 2001. On the Effectiveness of the Skew Divergence for Statistical Language Analysis. *Artificial Intelligence and Statistics 2001*, 65–72.
12. Lin, J. 1991. Divergence measures based on the shannon entropy. *IEEE Transactions on Information Theory*, 37(1):140–151.
13. Mochihashi, D., Matsumoto, Y. 2002. Probabilistic Representation of Meanings. *IPSJ SIG-Notes Natural Language*, 2002-NL-147:77–84.
14. The National Institute of Japanese Language. 2004. *Bunruigoihyo.* Dainippontosho.
15. Santorini, B. 1990. Part-of-Speech Tagging Guidelines for the Penn Treebank Project. ftp://ftp.cis.upenn.edu/pub/treebank/doc/tagguide.ps.gz
16. Schmid, H. 1994. Probabilistic Part-of-Speech Tagging Using Decision Trees. *Proc. of the First International Conference on New Methods in Natural Language Processing (NemLap-94)*, 44–49.
17. Ueda, N., Nakano, R. 1998. Deterministic annealing EM algorithm. *Neural Networks*, 11:271–282.