Context Analysis for Computer-Assisted Near-Synonym Learning

Liang-Chih Yu, Wei-Nan Chien and Kai-Hsiang Hsu

Abstract Despite their similar meanings, near-synonyms may have different usages in different contexts. For second-language learners, such differences are not easily grasped in practical use. This chapter introduces several context analysis techniques such as pointwise mutual information (PMI), *n*-gram language model, latent semantic analysis (LSA), and independent component analysis (ICA) to verify whether near-synonyms do match the given contexts. Applications can benefit from such techniques to provide useful contextual information for learners, making it easier for them to understand different usages of various near-synonyms. Based on these context analysis techniques, we build a prototype computer-assisted near-synonym learning system. In experiments, we evaluate the context analysis methods on both Chinese and English sentences, and compared its performance to several previously proposed supervised and unsupervised methods. Experimental results show that training on the independent components that contain useful contextual features with minimized term dependence can improve the classifiers' ability to discriminate among near-synonyms, thus yielding better performance.

1 Introduction

Near-synonym sets represent groups of words with similar meanings, which can be derived from the existing lexical ontologies such as WordNet (Fellbaum 1998), EuroWordNet (Rodríguez et al. 1998), and Chinese WordNet (Huang et al. 2008). These are useful knowledge resources for computer-assisted language learning

L.-C. Yu (✉) · W.-N. Chien
Yuan Ze University, Taoyuan, Taiwan
e-mail: lcyu@saturn.yzu.edu.tw

W.-N. Chien
e-mail: s986223@mail.yzu.edu.tw

K.-H. Hsu
Yuanze University, Taoyuan, Taiwan
e-mail: s986220@mail.yzu.edu.tw

© Springer Nature Singapore Pte Ltd. 2019

X. Lu and B. Chen (eds.), *Computational and Corpus Approaches to Chinese Language Learning*, Chinese Language Learning Sciences,
https://doi.org/10.1007/978-981-13-3570-9_7

(CALL) (Cheng 2004; Inkpen and Hirst 2006; Inkpen 2007; Ouyang et al. 2009; Wu et al. 2010) and natural language processing (NLP) applications such as information retrieval (IR) (Moldovan and Mihalcea 2000; Navigli and Velardi 2003; Shlrl and Revle 2006; Bhogal et al. 2007; Yu et al. 2009) and (near-)duplicate detection for text summarization (Vanderwende et al. 2007). For example, in composing a text, near-synonyms can be used to automatically suggest alternatives to avoid repeating the same word in a text when suitable alternatives are available in the near-synonym set (Inkpen and Hirst 2006). In information retrieval, systems can perform query expansion to improve the recall rate, for example, through recognizing that the weapon sense of "arm" corresponds to the weapon sense of "weapon" and of "arsenal".

Although the words in a near-synonym set have similar meanings, they are not necessarily interchangeable in practical use due to their specific usage and collocational constraints (Wible et al. 2003). Consider the following examples.

(E1) {strong, powerful} coffee
(E2) ghastly {error, mistake}
(E3) {bridge, overpass, tunnel} under the bay.

Examples (E1) and (E2) both present an example of collocational constraints for the given contexts. In (E1), the word "strong" in the near-synonym set {strong, powerful} is more suitable than "powerful" to fit the given context "coffee," since "powerful coffee" is an anti-collocation (Pearce 2001). Similarly, in (E2), "mistake" is more suitable than "error" because "ghastly mistake" is a collocation and "ghastly error" is an anti-collocation (Inkpen 2007). In (E3), the near-synonym set {bridge, overpass, tunnel} represents the meaning of a physical structure that connects separate places by traversing an obstacle. Suppose that the original word is "tunnel" in the context "under the bay". The word "tunnel" cannot be substituted by the other words in the same set because all the substitutions are semantically implausible (Yu et al. 2010). The above examples indicate that near-synonyms may have different usages in different contexts, and such differences are not easily captured by second-language learners. Therefore, we develop a computer-assisted near-synonym learning system to assist Chinese English-as-a-Second-Language (ESL) learners to better understand different usages of various English near-synonyms and use them appropriately in different contexts.

This chapter introduces the use of NLP techniques such as automatic near-synonym choice (Edmonds 1997; Gardiner and Dras 2007; Inkpen 2007; Islam and Inkpen 2010; Wang and Hirst 2010; Yu et al. 2010; Yu and Chien 2013; Yu et al. 2016) to verify whether near-synonyms match the given contexts. The problem of automatic near-synonym choice has been formulated as a "fill-in-the-blank" (FITB) task, as shown in Fig. 1. Given a near-synonym set and a sentence containing one of the near-synonyms, the near-synonym is first removed from the sentence to form a lexical gap. The goal is to predict an answer (best near-synonym) that can fill the gap from the given near-synonym set (including the original word). The systems can then be evaluated by examining their ability to restore the original word with the best near-synonym.

English Sentence:	This will make the ___ message easier to interpret.
Original word:	error
Near-synonym set:	{error, mistake, oversight}

Fig. 1 Example of FITB evaluation for automatic near-synonym choice

2 Automatic Near-Synonym Choice

Among many approaches to automatic near-synonym choice, Edmonds' pioneering study used a lexical co-occurrence network to determine the near-synonym that is most typical or expected in a given context (Edmonds 1997). Other proposed methods can generally be categorized as unsupervised (Gardiner and Dras 2007; Inkpen 2007; Islam and Inkpen 2010; Yu et al. 2010) and supervised methods (Wang and Hirst 2010; Yu and Chien 2013; Yu et al. 2016).

2.1 Unsupervised Methods

In the unsupervised methods, the pointwise mutual information (PMI) (Gardiner and Dras 2007; Inkpen 2007) and *n*-gram-based methods (Islam and Inkpen 2010; Yu et al. 2010) are the two major methods.

2.1.1 PMI-Based Method

The PMI is used to measure the strength of co-occurrence between a near-synonym and each individual word appearing in its context. A higher mutual information score indicates that the near-synonym fits well in the given context, and thus is more likely to be the correct answer. The pointwise mutual information (Church and Hanks 1990) between two words x and y is defined as

$$\mathrm{PMI}(x, y) = \log_2 \frac{P(x, y)}{P(x)P(y)}, \tag{1}$$

where $P(x, y) = C(x, y)/N$ denotes the probability that x and y co-occur; $C(x, y)$ is the number of times x and y co-occur in the corpus; and N is the total number of words in the corpus. Similarly, $P(x) = C(x)/N$, where $C(x)$ is the number of times x occurs in the corpus, and $P(y) = C(y)/N$, where $C(y)$ is the number of times y occurs in the corpus. Therefore, Eq. 1 can be rewritten as

$$\mathrm{PMI}(x, y) = \log_2 \frac{C(x, y) \cdot N}{C(x) \cdot C(y)}. \tag{2}$$

The frequency counts $C(\cdot)$ presented in Eq. 2 can be retrieved from a large corpus such as the Waterloo terabyte corpus used in (Inkpen 2007), and the Web 1T 5-gram corpus used in (Gardiner and Dras 2007).

Given a sentence s with a gap, $s = \ldots w_1 \ldots w_\ell \ldots w_{\ell+1} \ldots w_{2\ell} \ldots$, the PMI score for a near-synonym NS_j to fill the gap is computed from the words around the gap, defined as

$$\mathrm{PMI}(NS_j, s) = \sum_{i=1}^{2\ell} \mathrm{PMI}(NS_j, w_i). \tag{3}$$

where ℓ is a window size representing ℓ words to the left and right of the gap. Finally, the near-synonym with the highest score is considered to be the answer.

2.1.2 5-Gram Language Model

N-grams can capture contiguous word associations within given contexts. Assume a sentence $s = \ldots w_{i-4}w_{i-3}w_{i-2}w_{i-1}w_iw_{i+1}w_{i+2}w_{i+3}w_{i+4} \ldots$, where w_i represents a near-synonym in a set. In computing the 5-gram scores for each near-synonym, only the five product items $P(w_i|w_{i-4}^{i-1})$, $P(w_{i+1}|w_{i-3}^{i})$, $P(w_{i+2}|w_{i-2}^{i+1})$, $P(w_{i+3}|w_{i-1}^{i+2})$, and $P(w_{i+4}|w_{i}^{i+3})$ are considered (Islam and Inkpen 2010). The other items are excluded because they do not contain the near-synonym and thus will have the same values. Accordingly, the 5-gram language model ($n = 5$) with a smoothing method can be defined as

$$\begin{aligned} P(s) &= \prod_{i=1}^{5} P(w_i|w_{i-n+1}^{i-1}) \\ &= \prod_{i=1}^{5} \frac{C(w_{i-n+1}^{i}) + (1+\alpha_n)M(w_{i-n+1}^{i-1})P(w_i|w_{i-n+2}^{i-1})}{C(w_{i-n+1}^{i-1}) + \alpha_n M(w_{i-n+1}^{i-1})} \end{aligned} \tag{4}$$

where $M(w_{i-n+1}^{i-1})$ denotes a missing count used in the smoothing method, defined as

$$M(w_{i-n+1}^{i-1}) = C(w_{i-n+1}^{i-1}) - \sum_{w_i} C(w_{i-n+1}^{i}) \tag{5}$$

where $C(\cdot)$ denotes an n-gram frequency, which can be retrieved from a large corpus such as the Web 1T 5-gram corpus used in (Islam and Inkpen 2010). The 5-gram language model is implemented as a back-off model. That is, if the frequency of a higher order n-gram is zero, then its lower order n-grams will be considered. Conversely, if the frequency of a higher order n-gram is not zero, then the lower order n-grams will not be included in the computation. Similar to the PMI-based method, the near-synonym with the highest score is considered to be the answer.

2.2 Supervised Methods

Supervised methods usually approach near-synonym choice tasks as classification tasks in which each near-synonym in a near-synonym set represents a class, and the features used for classification are the words which occur in the contexts of the near-synonyms. The near-synonyms and their context words are represented by vector-based representation which is frequently used in distributional models of lexical semantics (Harris 1954; Lin 1998; Roussinov and Zhao 2003; Weeds et al. 2004). Based on this representation, a co-occurrence matrix of the near-synonyms and their context words can be built from the training data, i.e., a collection of sentences containing the near-synonyms. Each entry in the matrix represents a co-occurrence frequency of a context word and a near-synonym. Different context analysis techniques such as latent semantic analysis (LSA) and independent component analysis (ICA) can then be applied to the context matrix to identify useful context features that contribute to the classification task (Wang and Hirst 2010; Yu and Chien 2013).

2.2.1 Latent Semantic Analysis (LSA)

LSA is a technique for analyzing the relationships between words and documents and has been widely used in many application domains such as information retrieval (Landauer et al. 1998), latent topic discovery (Cribbin 2011), and document clustering (Wei et al. 2008). For automatic near-synonym choice, LSA is used as a context analysis technique to identify useful latent context features for the near-synonyms through indirect associations between words and sentences.

The first step in LSA is to build a word-by-document matrix for near-synonyms and their context words (Wang and Hirst 2010). In addition to documents, for our task, other text units such as sentences or 5-grams could also be used to build the matrix because these text units also contain contextual information for near-synonyms. Figure 2 shows a sample matrix $\mathbf{X}$ built using the sentence as the unit. The columns in $\mathbf{X}_{v\times d}$ represent d sentences containing the near-synonyms in a near-synonym set and the rows represent v distinct words occurring in the near-synonyms' contexts in the corpus. Singular value decomposition (SVD) (Golub and Van Loan 1996) is then used to decompose the matrix $\mathbf{X}_{v\times d}$ into three matrices as follows:

$$\mathbf{X}_{v\times d} = \mathbf{U}_{v\times n} \sum_{n\times n} \mathbf{V}^T_{n\times d}, \tag{6}$$

where $\mathbf{U}$ and $\mathbf{V}$, respectively, consist of a set of latent vectors of words and sentences, $\sum$ is a diagonal matrix of singular values, and $n = \min(v, d)$ denotes the dimensionality of the latent semantic space. Additionally, each element in $\mathbf{U}$ represents the weight of a context word, and the higher weighted words are the useful context features for the near-synonyms. By selecting the largest k_1 ($\leq n$) singular values together with the first k_1 columns of $\mathbf{U}$ and $\mathbf{V}$, the near-synonyms and their context words

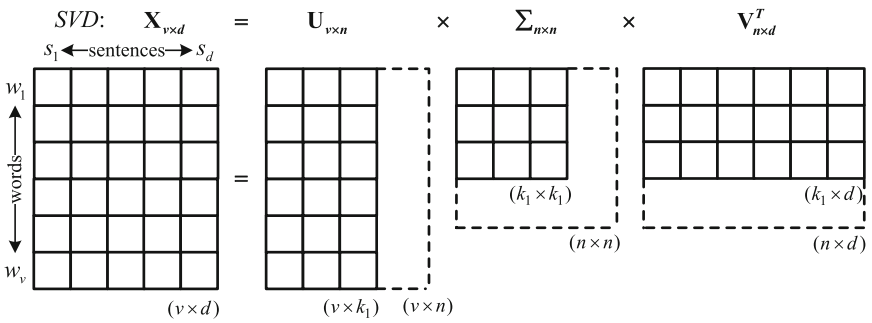

Fig. 2 Illustrative example of singular value decomposition for latent semantic analysis

can be represented in a low-dimensional latent semantic space. The original matrix can also be reconstructed with the reduced dimensions, as shown in Eq. 7

$$\widehat{\mathbf{X}}_{v\times d} = \mathbf{U}_{v\times k_1} \sum_{k_1\times k_1} \mathbf{V}^T_{k_1\times d}, \tag{7}$$

where $\widehat{\mathbf{X}}$ represents the reconstructed matrix.

In SVM training and testing, each input sentence with a lexical gap is first transformed into the latent semantic representation as follows:

$$\hat{\mathbf{t}}_{k_1\times 1} = \sum_{k_1\times k_1}^{-1} \mathbf{U}^T_{k_1\times v}\mathbf{t}_{v\times 1}, \tag{8}$$

where $\mathbf{t}_{v\times 1}$ denotes the vector representation of an input sentence and $\hat{\mathbf{t}}_{k_1\times 1}$ denotes the transformed vector in the latent semantic space. Each transformed training vector is then appended by the correct answer (the near-synonym removed from the sentence) to form a $(k_1 + 1)$-dimensional vector for SVM training.

The strength of LSA lies in discovering the latent context features for near-synonyms using SVD. Consider the example shown in Fig. 3. The original matrix, as shown in Fig. 3a, is built using five training sentences containing two different near-synonyms NS_i and NS_j. Suppose that the words w_1, w_2 are the useful features for NS_i, and w_3, w_4 are useful for NS_j, but w_4 is a latent feature because it does not frequently occur in the context of NS_j. After applying SVD, the latent features can be identified by replacing the zero entries in the original matrix with nonzero real values through the indirect associations between words and sentences. For instance, w_4 originally does not occur in s_3 and s_4, but it does co-occur with w_3 in the matrix (e.g., in s_5), which means that w_4 might also occur in the sentences where w_3 occurs (e.g., s_3 and s_4). Therefore, the zero entries (w_4, s_3) and (w_4, s_4) are replaced with a nonzero value through the indirect associations between of w_3 and w_4 in s_5, as shown in Fig. 3b. This helps identify a useful latent feature w_4 for NS_j. However, identi-

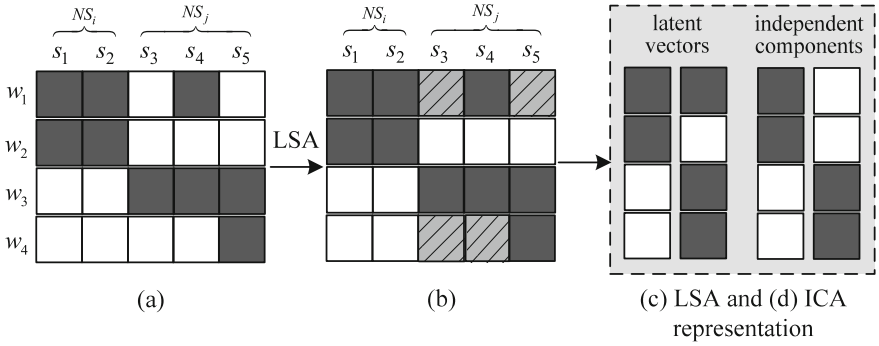

Fig. 3 Comparison of LSA and ICA for feature representation

fying latent features through the indirect associations cannot avoid feature overlap when different near-synonyms share common words in their contexts. This might be possible because near-synonyms usually have similar contexts. For instance, in Fig. 3a, w_1, which is useful for NS_i, still occurs in the context of NS_j (e.g., s_4). Through the indirect associations between of w_1 and w_3 in s_4, the frequency of w_1 increases in the context of NS_j because it may also occur in the sentences where w_3 occurs (e.g., s_3 and s_5), as shown in Fig. 3b. Therefore, when all word features are to be accommodated in a low-dimensional space reduced by SVD, term overlap may occur between the latent vectors. As indicated in Fig. 3c, the two sample latent vectors which contribute to two different near-synonyms share a common feature w_1. Classifiers trained on such latent vectors with term overlap may decrease their ability to distinguish among near-synonyms.

2.2.2 Independent Component Analysis (ICA)

ICA is a technique for extracting independent components from a mixture of signals and has been successfully applied to solve the blind source separation problem (Hyvärinen et al. 2001; Lee 1998). Recent studies have shown that ICA can also be applied to other application domains such as text processing (Kolenda and Hansen 2000; Rapp 2004; Sevillano et al. 2004). In contrast to LSA, ICA extracts independent components by minimizing the term dependence of the context matrix. Therefore, LSA and ICA can be considered complementary methods where LSA can be used to discover latent features that do not frequently occur in the context of near-synonyms, and ICA can be used to further minimize the dependence of the latent features such that overlapped features can be removed, as presented in Fig. 3d. Based on this complementariness, the ICA-based framework can be used to analyze the LSA output to discover more useful latent features for different near-synonyms, and the dependence between them can also be minimized. The discriminant power of classifiers can thus

be improved by training them on the independent components with minimized term overlap. The ICA model can be formally described as

$$\mathbf{X} = \mathbf{AS}, \tag{9}$$

where $\mathbf{X}$ denotes the observed mixture signals, $\mathbf{A}$ denotes a mixing matrix, and $\mathbf{S}$ denotes the independent components. The goal of ICA is to estimate both $\mathbf{A}$ and $\mathbf{S}$. Once the mixing matrix $\mathbf{A}$ is estimated, the demixing matrix can be obtained by $\mathbf{W} = \mathbf{A}^{-1}$, and Eq. 9 can be rewritten as

$$\mathbf{S} = \mathbf{WX}, \tag{10}$$

That is, the observed mixture signals can be separated into independent components using the demixing matrix.

For our problem, the context matrix can be considered as a mixture of signals because it consists of the contexts of different near-synonyms. Therefore, ICA used, herein is to estimate the demixing matrix so that it can separate the mixed contexts to derive the independent components for each near-synonym. Figure 4 shows the ICA-based framework combining LSA and ICA.

LSA decomposition and reconstruction In the training phase, the original context matrix $\mathbf{X}_{v\times d}$ is first decomposed by SVD using Eq. 6, and then reconstructed with reduced dimensions using Eq. 7. Useful latent features that do not frequently occur in the original matrix can thus be discovered in this step.

ICA decomposition and demixing To further minimize term dependence in deriving the independent components, the reconstructed matrix $\widehat{\mathbf{X}}_{v\times d}$ is passed to ICA to estimate the demixing matrix. ICA accomplishes this by decomposing $\widehat{\mathbf{X}}_{v\times d}$ using Eq. 11. Figure 5 shows an example of the decomposition.

$$\widehat{\mathbf{X}}_{v\times d} = \mathbf{A}_{v\times k_2}\mathbf{S}_{k_2\times d}. \tag{11}$$

Based on this decomposition, the demixing matrix can be obtained by $\mathbf{W}_{k_2\times v} = \mathbf{A}^{-1}_{v\times k_2}$, where k_2 ($\leq n$) is the number of independent components. Similar to the matrix $\mathbf{U}$ in LSA, each element in $\mathbf{W}$ also represents the weight of a context word, and the higher weighted words are useful context features for the near-synonyms. Therefore, the demixing matrix contains useful context features with minimized term dependence for different near-synonyms.

Once estimated, the demixing matrix is used to separate $\widehat{\mathbf{X}}_{v\times d}$ to derive the independent components as follows:

$$\mathbf{S}_{k_2\times d} = \mathbf{W}_{k_2\times v}\widehat{\mathbf{X}}_{v\times d}, \tag{12}$$

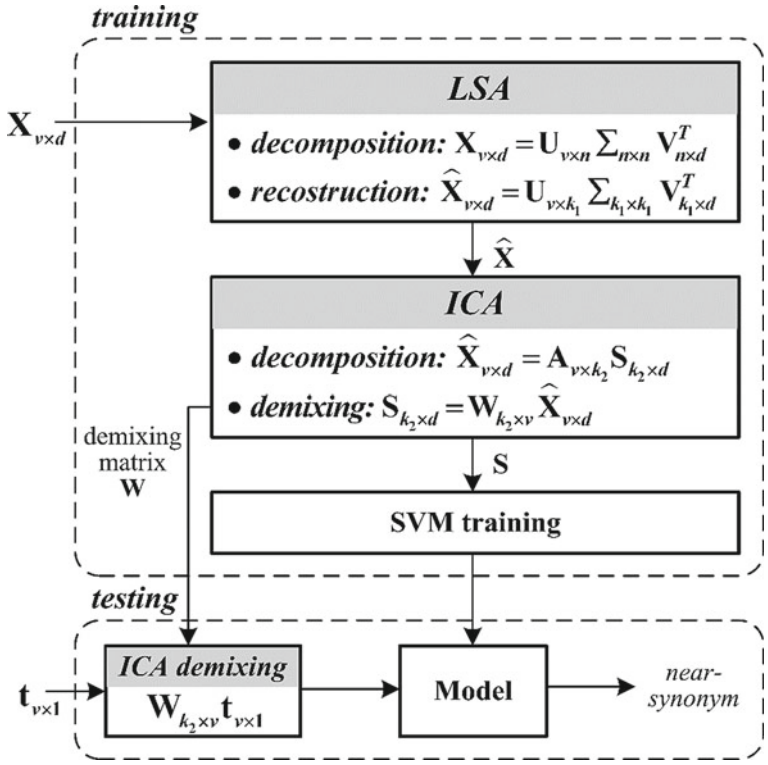

Fig. 4 ICA-based framework for near-synonym choice

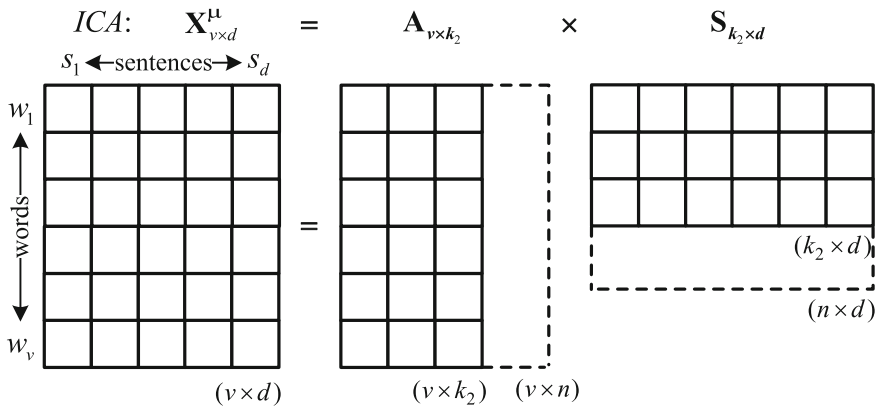

Fig. 5 Illustrative example of ICA decomposition

Each column vector in $\mathbf{S}_{k_2 \times d}$ is then appended by the correct answer for SVM training. Similarly, as shown in Fig. 4, each test instance $\mathbf{t}_{v \times 1}$ in the testing phase is also transformed by the demixing matrix, and then predicted with the trained SVM model.

3 Experimental Results

This section presents the evaluation results of different methods for near-synonym choice. Section 3.1 describes the experiment setup, including experimental data, implementation details of methods used, and the evaluation metric. Section 3.2 investigates the selection of optimal parameters for LSA and ICA-based methods. Section 3.3 compares the results obtained by the various methods. Section 3.4 presents a detailed analysis to examine the effect of term overlap on classification performance.

3.1 Experiment Setup

3.1.1 Data

As shown in Table 1, seven English and Chinese near-synonym sets was used for evaluation. For Chinese near-synonym choice evaluation, two test corpora were used: the Chinese News Corpus (CNC) and the Sinica Corpus (SC), both released by the Association for Computational Linguistics and Chinese Language Processing (ACLCLP). The test examples were collected from the two corpora by selecting sentences containing the near-synonyms in the seven Chinese near-synonym sets. A total of 36,427 (CNC) and 26,504 (SC) sentences were collected, where 20% of sentences from each corpus were randomly selected as a development set for parameter tuning of LSA and ICA-based methods, and the remaining 80% were used as the test set for performance evaluation. In addition, the classifiers (described in the next section) were built from the Chinese Web 5-gram corpus released by the Linguistic Data Consortium (LDC). For English near-synonym choice evaluation, the Web 1T 5-gram corpus released by LDC was used for both classifier training and evaluation using the cross-validation method.

3.1.2 Classifiers

The classifiers involved in this experiment included PMI, the 5-gram language model, cosine measure, LSA, and ICA-based methods (including stand-alone ICA and a combination of LSA and ICA). The implementation details for each classifier are as follows:

Table 1 English and Chinese near-synonym sets

No.	Near-synonym sets
1	Difficult, hard, tough 困難的, 艱難的, 艱苦的, 難懂的
2	Error, mistake, oversight 錯誤, 錯, 過失, 失察
3	Job, task, duty 工作, 任務, 義務
4	Responsibility, burden, obligation, commitment 責任, 職責, 職務, 約定
5	Material, stuff, substance 物質, 材料, 質料
6	Give, provide, offer 給, 給予, 給與, 供應, 供給
7	Settle, resolve 決定, 確定, 定奪, 終結

The English and Chinese near-synonyms in each set corresponds to the same sense

PMI: Given a near-synonym set and a test example with a gap, the PMI scores for each near-synonym were calculated using Eq. 3, where the size of the context window ℓ was set to 2. The frequency counts were retrieved from the Web 1T 5-gram corpus and Chinese Web 5-gram corpus, respectively, for English and Chinese near-synonym choice evaluation.

5 GRAM: The 5-gram scores for each near-synonym in a test example were calculated using Eq. 4. The frequency counts for *n*-grams were retrieved by querying Google (as in Yu et al. 2010) for English near-synonym choice evaluation, and from the Chinese Web 5-gram corpus for Chinese evaluation.

COS: Given a near-synonym set, all 5-grams containing the near-synonyms in the set were first extracted from the training data (i.e., from the Web 1T 5-gram corpus for English near-synonyms and from the Chinese Web 5-gram corpus for Chinese near-synonyms). The 5-grams with the near-synonyms are removed and were then used to build a word-by-class matrix for the near-synonym set. The best near-synonym for each test example was then predicted by comparing the cosine similarity of the test example and the near-synonyms in the matrix.

LSA: COS used all 5-grams to build the context matrix but, due to the efficiency consideration, we randomly selected only 20,000 5-grams to build the word-by-document (5-gram) matrix for each English and Chinese near-synonym set. The number of the 5-grams for each near-synonym in the matrix was selected according to their proportions in the corpus. Once the matrix was built, each training instance (i.e., each column vector of the matrix) was transformed into the latent space using

Eq. 8. The correct answer (the near-synonym removed from the training instance) was then appended to the corresponding transformed vector to form a (k_1 + 1)-dimensional vector for SVM training.

ICA: Stand-alone ICA was implemented using Eqs. 9 and 10. The input matrix was the same as that of LSA, and the demixing matrix was estimated using the FastICA package (Hyvärinen 1999). An SVM classifier was then trained using the independent components obtained using Eq. 10 with the correct answers appended.

LSA + ICA: The combination of LSA and ICA was implemented by taking the LSA result as input to estimate the demixing matrix in ICA, as shown in Eq. 11. An SVM classifier was then trained using the independent components obtained from Eq. 12 with the correct answers appended. For LSA, ICA, and LSA + ICA, the best near-synonym for each test example was predicted using the trained SVM models.

3.1.3 Evaluation Metric

In testing, this experiment followed the FITB evaluation procedure (Fig. 1) to remove the near-synonyms from the test samples. The answers proposed by each classifier are the near-synonyms with the highest score. The correct answers are the near-synonyms originally in the gap of the test samples. Performance is determined by accuracy, which is defined as the number of correct answers made by each classifier, divided by the total number of test examples.

3.2 Evaluation of LSA and ICA-Based Methods

Experiments were conducted to compare the performance of LSA, ICA, and LSA + ICA using different settings for the parameters k_1 and k_2, which, respectively represent the dimensionality of the latent semantic space and the number of independent components. The optimal settings of the two parameters were tuned by maximizing the classification accuracy on the development set. Figure 6 shows the classification accuracy of LSA, ICA, and LSA + ICA with different settings of dimensionality (k_1 or k_2). The accuracies were obtained by averaging the seven Chinese near-synonym sets. For LSA, the x-axis represents different values of k_1, with performance increasing with k_1 up to 2000. For ICA, the x-axis represents different values of k_2, with an optimal performance at $k_2 = 500$. For LSA + ICA, the performance was tuned by varying both k_1 and k_2. Due to the large number of combinations of k_1 and k_2, for LSA + ICA, Fig. 7 only plots the optimal accuracy for each value of k_1 on the x-axis, and the optimal accuracy for each value of k_1 (e.g., 500) was determined by increasing k_2 by increments of 100 to select the highest accuracy among the different settings of k_2. For instance, the accuracy of LSA + ICA at $k_1 = 500$ was selected

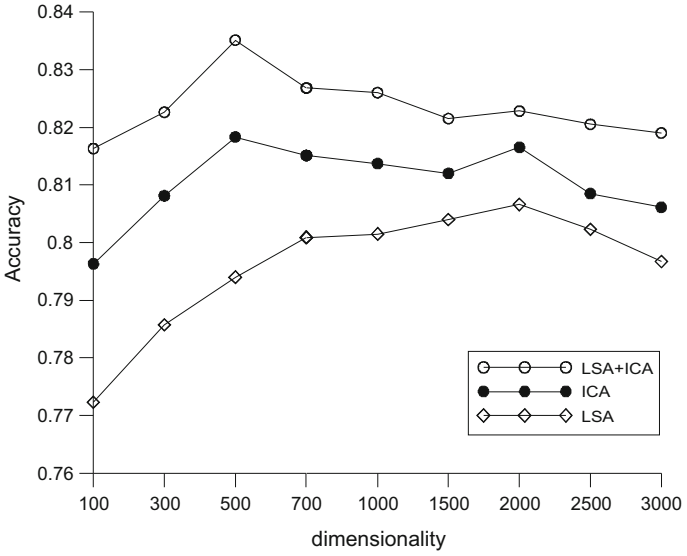

Fig. 6 Classification accuracy of LSA, ICA, and LSA + ICA on the development set, as a function of dimensionality

from the highest achieved at $k_2 = 500$, and this accuracy (i.e., that reached at $k_1 = 500$ and $k_2 = 500$) was also the optimal performance of LSA + ICA.

The results presented in Fig. 6 show that LSA + ICA improved the performance of LSA for all dimensionalities because the degree of term overlap in LSA was reduced by ICA. In addition, the performance difference between LSA + ICA and LSA was greater when the dimensionality was smaller, indicating that ICA was more effective in reducing the degree of term overlap in a low-dimensional space. More detailed analysis of the relationship between the term overlap and the classification performance is discussed in Sect. 3.4. The best settings of the parameters were used in the following experiments.

3.3 Comparative Results

This section reports the classification accuracy of supervised and unsupervised methods including PMI, 5GRAM, COS, LSA, ICA, and LSA + ICA. Table 2 shows the comparative results for the Chinese corpora including Chinese News Corpus (CNC), Sinica Corpus (SC), and both (ALL). The *binomial exact test* (Howell 2007) was used to determine whether the performance difference was statistically significant.

For the two unsupervised methods, 5GRAM outperformed PMI on both test corpora. One possible reason is that the 5-gram language model can capture contiguous word associations in a given context, whereas in PMI, words are considered inde-

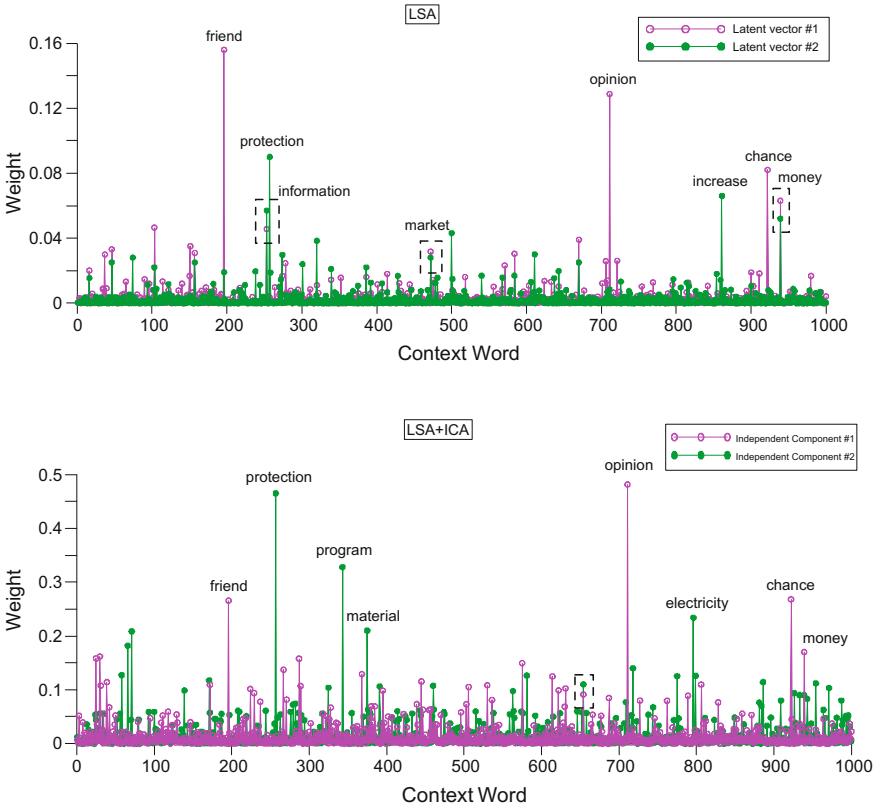

Fig. 7 Examples of latent vectors, selected from $\mathbf{U}_{v\times k}$, and independent components, selected from $\mathbf{W}^T_{v\times k}$, for the near-synonyms "give" and "provide". The weights shown in this figure are the absolute values of the weights in the latent vectors and independent components

pendently within the given context. In addition, all supervised methods (i.e., COS, LSA, ICA, and LSA + ICA) achieved better performance than the two unsupervised methods on both test corpora. In the supervised methods, COS provided the baseline results since it did not use any technique for context analysis. As indicated in Table 2, COS yielded higher average accuracy than the best unsupervised method (i.e., 5GRAM) by 2.55 and 7.83% on CNC and SC, respectively, and by 4.79% on ALL. Once context analysis techniques were employed, LSA, ICA, and LSA + ICA significantly improved COS, indicating that context analysis is a useful technique for near-synonym choice. For LSA, it improved the average accuracy of COS by 7.52, 4.29, and 6.16%, respectively, on CNC, SC, and ALL. The improvement mainly came from the discovery of useful latent features from the contexts of the near-synonyms. ICA also outperformed COS, with the improvement mainly coming from the discovery of independent components by minimizing the feature dependence among near-synonyms. After combining LSA and ICA, the performance of both LSA and

ICA was further improved because LSA + ICA cannot only discover useful latent features for different near-synonyms but also can minimize the dependence between them, thus improving the discriminant power of classifiers to distinguish between near-synonyms.

For the evaluation of English near-synonym choice, 20,000 5-grams for each English near-synonym set were randomly selected from the Web 1T 5-gram corpus (Web 1T), and the near-synonyms in the 5-grams were removed for the purpose of FITB evaluation. Ten-fold cross-validation was then used to determine the classification accuracy of the methods used, with a *t-test* to determine statistical significance. In addition, for PMI, frequency counts were retrieved from the whole Web 1T 5-gram corpus, and those for 5GRAM were retrieved by querying Google (as in Yu et al. 2010). Table 3 shows the comparative results of the various methods, showing that LSA + ICA improved the performance of both stand-alone LSA and ICA.

3.4 Term Overlap Analysis

This section investigates the effect of term overlap on classification performance. Term overlap in LSA and the ICA-based methods can be estimated from their respective corresponding matrices $\mathbf{U}_{v\times k}$ and $\mathbf{W}^T_{v\times k}$. Each column of $\mathbf{U}_{v\times k}$ and $\mathbf{W}^T_{v\times k}$ represents a latent vector/independent component of v words, and each element in the vector are a word weight representing its relevance to the corresponding latent vector/independent component. Therefore, the meaning of each latent vector/independent component can be characterized by its higher weighted words. Figure 7 shows two sample latent vectors for LSA and two independent components for LSA + ICA.

The upper part of Fig. 7 shows parts of the context words and their weights in the two latent vectors, where latent vector #1 can be characterized by *friend*, *opinion*, and *chance*, which are the useful features for identifying the near-synonym "give," and latent vector #2 can be characterized by *protection*, *information*, and *increase*, which are useful for identifying the near-synonym "provide". Although the two latent vectors contained useful context features for the respective different near-synonyms, these features still had some overlap between the latent vectors, as marked by the dashed rectangles. The overlapped features, especially those with higher weights, may reduce the classifier's ability to distinguish between the near-synonyms. The lower part of Fig. 7 also shows two independent components for the near-synonyms "give" and "provide". As indicated, the term overlap between the two independent components was relatively low.

To formally compare the degree of term overlap of LSA and the ICA-based methods, we used a measure, *overlap@n*, to calculate the degree of overlap of the top n ranked words among the latent vectors in $\mathbf{U}_{v\times k}$ and independent components in $\mathbf{W}^T_{v\times k}$. First, the words in each latent vector (or independent component) were ranked according to the descending order of their weights. The top n words were then selected to form a word set. Let s^n_i and s^n_j be the two word sets of the top n words for any two

Table 2 Classification accuracy for Chinese corpora

CNC	Accuracy					
	PMI	5GRAM	COS	LSA	ICA	LSA + ICA
1	72.72	71.33	67.47	71.49	72.81	76.55
2	59.70	53.70	65.05	65.97	68.65	69.36
3	67.90	70.68	81.40	87.02	88.38	89.38
4	56.35	56.87	59.90	69.62	70.53	72.63
5	75.85	64.39	78.96	83.77	83.69	86.32
6	51.85	57.71	57.67	65.98	66.35	67.99
7	60.81	72.27	63.69	73.53	79.98	82.03
Average	61.49	66.41	68.96	76.48	79.02*	80.58†
Data size	29,141		28,694			
SC	Accuracy					
	PMI	5GRAM	COS	LSA	ICA	LSA + ICA
1	68.84	70.66	69.35	68.08	72.37	73.31
2	67.51	52.47	76.12	77.40	79.02	80.11
3	64.67	76.72	84.96	90.01	90.71	92.06
4	50.14	68.58	68.86	75.70	77.51	79.10
5	73.29	59.52	76.13	72.13	75.96	78.63
6	69.85	65.68	76.72	81.52	82.66	84.57
7	61.58	71.98	70.17	74.88	76.79	78.75
Average	65.26	70.11	77.94	82.23	83.60*	85.28†
Data size	21,192		21,098			
ALL	Accuracy					
	PMI	5GRAM	COS	LSA	ICA	LSA + ICA
1	70.35	70.92	68.63	69.35	72.54	74.55
2	63.48	53.11	70.41	71.51	73.67	74.57
3	66.52	73.25	82.92	88.30	89.38	90.53
4	54.17	60.98	63.06	71.76	72.99	74.91
5	74.26	61.37	77.20	76.53	78.88	81.54
6	60.81	61.68	67.25	73.80	74.55	76.33
7	61.05	72.18	65.72	73.95	78.98	81.00
Average	63.07	67.97	72.76	78.92	80.96*	82.57†
Data size	50,333		49,792			

All figures are in %

*ICA versus LSA significantly different ($p < 0.05$)

†LSA + ICA versus ICA significantly different ($p < 0.05$)

Table 3 Classification accuracy for the English corpus

Web 1T	Accuracy					
	PMI	5GRAM	COS	LSA	ICA	LSA + ICA
1	60.36	61.36	60.88	62.68	63.57	65.29
2	76.62	72.67	76.39	79.16	79.85	82.05
3	70.67	71.33	76.17	78.95	80.30	80.97
4	68.75	70.25	67.25	68.50	72.21	73.50
5	70.58	70.35	71.53	74.79	77.25	79.50
6	65.93	61.98	66.25	72.53	73.22	75.08
7	71.29	68.50	76.56	77.89	79.06	81.06
Average	69.17	68.06	70.72	73.50	75.07*	76.78†

All figures are in %
*ICA versus LSA significantly different ($p < 0.05$)
†LSA + ICA versus ICA significantly different ($p < 0.05$)

Fig. 8 Example of computing the degree of term overlap

k=1		k=2		k=3	
A	0.4381	F	0.3678	F	0.2218
B	0.2708	A	0.2342	H	0.1582
C	0.2532	G	0.2095	E	0.1416
D	0.2342	H	0.1972	I	0.1332
E	0.2104	I	0.1895	C	0.1276

latent vectors (or independent components). The degree of term overlap between them was calculated by the number of intersections between their corresponding word sets divided by n, i.e., $\left|s_i^n \cap s_j^n\right| \Big/ n$. Therefore, the degree of term overlap for a whole matrix, namely $overlap_{\mathbf{U}_{v\times k}}$ (or $overlap_{\mathbf{W}_{v\times k}^T}$), can be calculated by the average of the degrees of term overlap between all latent vectors (or independent components) in $\mathbf{U}_{v\times k}$ (or $\mathbf{W}_{v\times k}^T$). That is,

$$overlap_{\mathbf{U}_{v\times k}}@n = \frac{1}{C_2^k}\sum_{i=1}^{k}\sum_{j=i+1}^{k}\frac{\left|s_i^n \cap s_j^n\right|}{n}, \tag{13}$$

where C_2^k denotes the number of combinations of any two vectors in $\mathbf{U}_{v\times k}$ (or $\mathbf{W}_{v\times k}^T$). Figure 8 presents a sample matrix for computing the degree of term overlap, consisting of three vectors of the top five terms. The respective *overlap@5* between the vectors (1, 2), (1, 3), and (2, 3), was 1/5, 2/5, and 3/5, yielding an average 2/5 as the *overlap@5* for the matrix.

By averaging the degree of term overlap over the matrices corresponding to the near-synonym sets, we can obtain the degree of term overlap of LSA, ICA, and LSA + ICA, respectively, defined as $overlap_{LSA}@n$, $overlap_{ICA}@n$, and $overlap_{LSA+ICA}@n$.

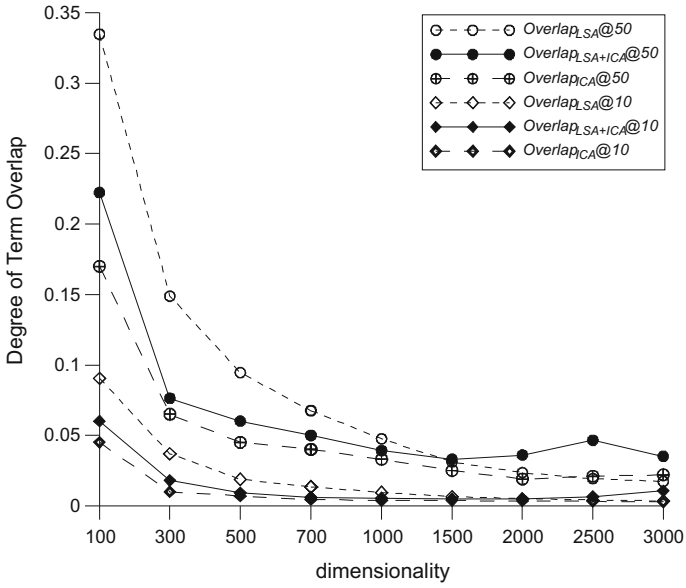

Fig. 9 Degree of term overlap of LSA, ICA, and LSA + ICA, as a function of dimensionality

Figure 9 shows the degree of term overlap for LSA, ICA, and LSA + ICA averaged over the seven Chinese near-synonym sets against various dimensionality values. The results show that ICA achieved the lowest degree of term overlap for both *overlap@10* and *overlap@50*. In addition, combining LSA and ICA reduced the degree of term overlap of using LSA alone, especially for a smaller dimensionality. As indicated in Fig. 9, the difference between both $overlap_{LSA+ICA}@10$ and $overlap_{LSA}@10$ and $overlap_{LSA+ICA}@50$ and $overlap_{LSA}@50$ increased with smaller dimensionality, mainly due to the fact that the features discovered by LSA were not easily separable in a lower dimensional space. In this circumstance, incorporating ICA can more effectively reduce the degree of term overlap. As the dimensionality increased, the difference between LSA and LSA + ICA gradually decreased, mainly because the increase in dimensionality decreased the degree of term overlap in LSA, thus ICA only produces a limited reduction of term overlap in LSA. As indicated, both $overlap_{LSA+ICA}@10$ and $overlap_{LSA+ICA}@50$ yielded a small decrease or increase of overlap when the dimensionality exceeded 1000.

To further analyze the relationship between term overlap and classification performance, Fig. 10 compares the classification accuracy of LSA, ICA, and LSA + ICA from Fig. 6 and $overlap_{LSA}@50$, $overlap_{ICA}@50$, and $overlap_{LSA+ICA}@50$ from Fig. 9. Comparing the degree of term overlap and classification performance of LSA + ICA and LSA found that reducing the degree of term overlap improved classification performance. Given a small dimensionality, the performance of LSA was low due to the high degree of term overlap. Combining LSA and ICA in this stage yielded a greater performance improvement because LSA + ICA effectively reduced

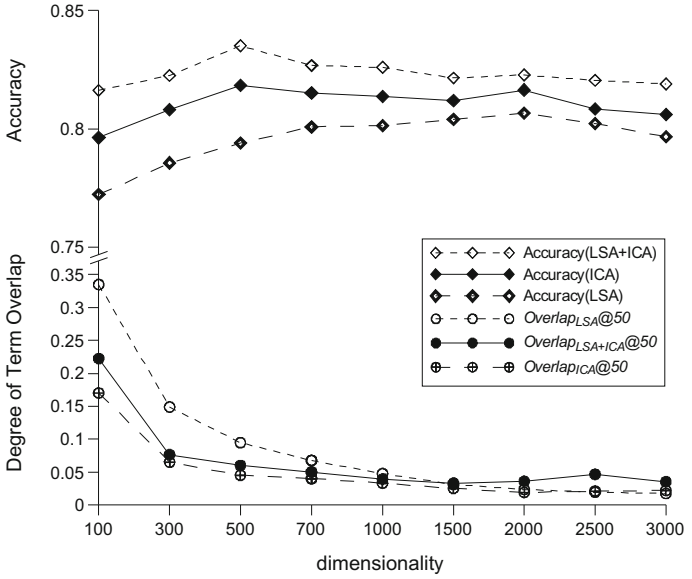

Fig. 10 Comparison of the classification accuracy and the degree of term overlap of LSA, ICA, and LSA + ICA, as a function of dimensionality

the degree of term overlap in LSA such that useful context features could be separated into different independent components according to their contribution to different near-synonyms. An increase in the dimensionality improves the performance of LSA due to the reduced degree of term overlap. Meanwhile, the performance of LSA + ICA was not similarly improved due to the small reduction of the degree of term overlap, resulting in a gradual decrease in the performance difference between LSA + ICA and LSA. Another observation is that LSA + ICA also outperformed ICA, despite ICA having a lower degree of term overlap than LSA + ICA. This is mainly due to the fact that LSA + ICA can discover more useful context features than ICA, and also minimizes feature dependence.

4 Applications

Based on the contextual information provided by the PMI and n-gram, we implement a prototype system with two functions: contextual statistics and near-synonym choice, both of which interact with learners.

4.1 Contextual Statistics

This function provides the contextual information retrieved by PMI and n-gram. This prototype system features a total of 21 English near-synonyms grouped into seven near-synonym sets, as shown in Table 1. Figure 11 shows a screenshot of the interface for contextual information lookup. Once a near-synonym set is selected, the 100 top-ranked context words and *n*-grams are retrieved for each near-synonym in the set. For PMI, both proportion-based PMI scores and co-occurrence frequencies between near-synonyms and their context words are presented. For n-gram, the 100 top-ranked *n*-grams with their frequencies are presented. Through this function, learners learn to determine the most frequently co-occurring and discriminative words and *n*-grams for different near-synonyms.

4.2 Near-Synonym Choice

This function assists learners in determining suitable near-synonyms when they are not familiar with the various usages of the near-synonyms in a given context. Learners can specify a near-synonym set and then input a sentence with "*" to represent any near-synonym in the set. The system will replace "*" with each near-synonym, and

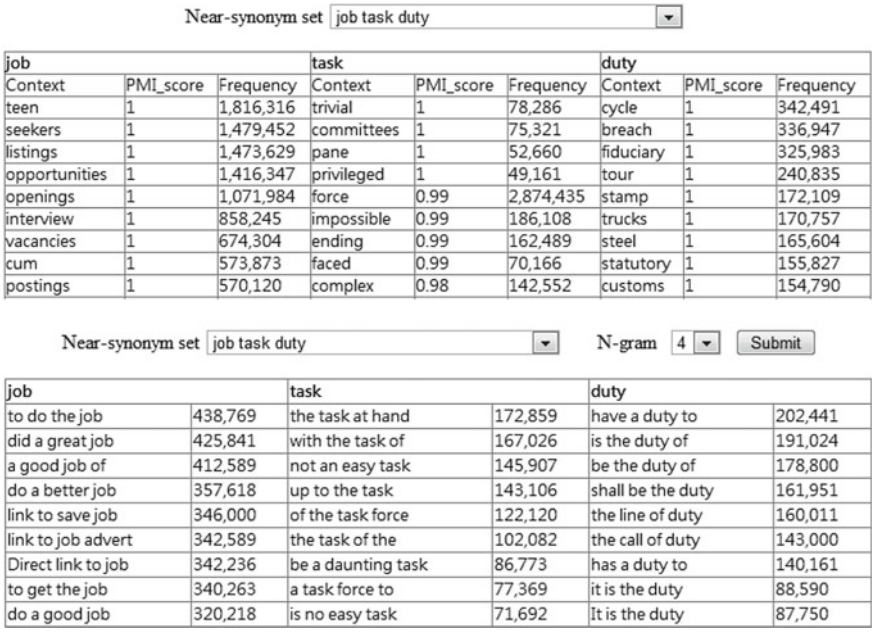

Fig. 11 Screenshot of contextual statistics

Near-Synonym set: material stuff substance

It was found that the * of the matter and not only mere theory was to be regarded

submit

Window size: 3

PMI	material	stuff	substance
found	0.35	0.34	0.31
that	0.24	0.49	0.28
the	0.31	0.27	0.42
of	0.28	0.28	0.44
the	0.31	0.27	0.42
matter	0.15	0.08	0.77
	1.64	1.73	2.64

Bi-gram	the material	7,488,173	the stuff	2,581,457	the substance	1,319,583
	material of	817,776	stuff of	392,805	substance of	848,188
Tri-gram	that the material	237,330	that the stuff	25,305	that the substance	52,803
	the material of	129,580	the stuff of	254,931	the substance of	545,206
	material of the	179,643	stuff of the	31,130	substance of the	341,962
4-gram	found that the material	1,706	found that the stuff	211	found that the substance	623
	that the material of	3,082	that the stuff of	910	that the substance of	15,240
	the material of the	48,148	the stuff of the	9,307	the substance of the	242,832
	material of the matter	0	stuff of the matter	0	substance of the matter	6,205
5-gram	found that the material of	0	found that the stuff of	0	found that the substance of	110
	that the material of the	1,229	that the stuff of the	150	that the substance of the	7,898
	the material of the matter	0	the stuff of the matter	0	the substance of the matter	5,427

Fig. 12 Screenshot of near-synonym choice

then retrieve the contextual information around "*" using PMI and n-gram. Figure 12 shows a sample sentence (the original word *substance* has been replaced with *) along with its contextual information retrieved by the system. For PMI, at most five context words (window size) before and after "*" are included to compute proportion-based PMI scores for each near-synonym. In addition, the sum of all PMI scores for each near-synonym is also presented to facilitate learner decisions. For n-gram, the frequencies of the *n*-grams (2–5) containing each near-synonym are retrieved. In the example shown in Fig. 12, learners can learn useful word pairs such as (substance, matter) and *n*-grams such as "substance of the matter," thus learning to discriminate between *substance*, *material*, and *stuff*.

5 Conclusion

A framework that incorporates LSA and ICA for near-synonym choice is presented. Both LSA and ICA are used to analyze the contexts of near-synonyms. LSA is used to discover useful latent contextual features, while ICA is used to estimate the independent components with minimal dependence between the features. Experiments

compared several supervised and unsupervised methods on both Chinese and English corpora. Results show that the ICA-based methods can reduce the degree of term overlap to improve the classifiers' ability to distinguish among near-synonyms, thus yielding higher classification accuracy.

Future work will focus on improving classification performance by combining multiple features such as predicate-argument structure and named entities occurring in the context of near-synonyms. In addition, current near-synonym choice evaluation is carried out on several preselected near-synonym sets. To make the near-synonym choice task more practical (similar to all-words word sense disambiguation), all-words near-synonym choice evaluation could also be designed and implemented to verify whether every word in a text fits the context well.

References

Bhogal, J., Macfarlane, A., & Smith, P. (2007). A review of ontology based query expansion. *Information Processing and Management, 43*(4), 866–886.

Cheng, C.-C. (2004). Word-focused extensive reading with guidance. In *Proceedings of the 13th International Symposium on English Teaching* (pp. 24–32).

Church, K., & Hanks, P. (1990). Word association norms, mutual information and lexicography. *Computational Linguistics, 16*(1), 22–29.

Cribbin, T. (2011). Discovering latent topical structure by second-order similarity analysis. *Journal of the American Society for Information Science and Technology, 62*(6), 1188–1207.

Edmonds, P. (1997). Choosing the word most typical in context using a lexical co-occurrence network. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics* (pp. 507–509).

Fellbaum, C. (1998). *WordNet: An electronic lexical database*. Cambridge, MA: MIT Press.

Gardiner, M., & Dras, M. (2007). Exploring approaches to discriminating among near-synonyms. In *Proceedings of the Australasian Language Technology Workshop* (pp. 31–39).

Golub, G. H., & Van Loan, C. F. (1996). *Matrix computations* (3rd ed.). Baltimore, MD: Johns Hopkins University Press.

Harris, Z. (1954). Distributional structure. *Word, 10*(2–3), 146–162.

Howell, D. C. (2007). *Statistical methods for psychology* (6th ed.). Belmont, CA: Thomson.

Huang, C.-R., Hsieh, S.-K., Hong, J.-F., Chen, Y.-Z., Su, I.-L., Chen, Y.-X., & Huang, S.-W. (2008). Chinese Wordnet: Design, implementation, and application of an infrastructure for cross-lingual knowledge processing. In *Proceedings of the 9th Chinese Lexical Semantics Workshop*.

Hyvärinen, A. (1999). Fast and robust fixed-point algorithms for independent component analysis. *IEEE Transactions on Neural Networks, 10*(3), 626–634.

Hyvärinen, A., Karhunen, J., & Oja, E. (2001). *Independent component analysis*. New York: Wiley.

Inkpen, D. (2007). A statistical model of near-synonym choice. *ACM Transactions on Speech and Language Processing, 4*(1), 1–17.

Inkpen, D., & Hirst, G. (2006). Building and using a lexical knowledge-base of near-synonym differences. *Computational Linguistics, 32*(2), 1–39.

Islam, A., & Inkpen, D. (2010). Near-synonym choice using a 5-gram language model. *Research in Computing Science, 46,* 41–52.

Kolenda, T., & Hansen, L. K. (2000). Independent components in text. *Advances in Neural Information Processing Systems, 13,* 235–256.

Landauer, T. K., Foltz, P. W., & Laham, D. (1998). An introduction to latent semantic analysis. *Discourse Processes, 25*(2–3), 259–284.

Lee, T. W. (1998). *Independent component analysis—Theory and applications*. Norwell, MA: Kluwer.

Lin, D. (1998). Automatic retrieval and clustering of similar words. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics* (pp. 768–774).

Moldovan, D., & Mihalcea, R. (2000). Using Wordnet and lexical operators to improve internet searches. *IEEE Internet Computing, 4*(1), 34–43.

Navigli, R., & Velardi, P. (2003). An analysis of ontology-based query expansion strategies. In *Proceedings of the Workshop on Adaptive Text Extraction and Mining*.

Ouyang, S., Gao, H.-H., & Koh, S.-N. (2009). Developing a computer-facilitated tool for acquiring near-synonyms in Chinese and English. In *Proceedings of the 8th International Conference on Computational Semantics* (pp. 316–319).

Pearce, D. (2001). Synonymy in collocation extraction. In *Proceedings of the Workshop on WordNet and Other Lexical Resources*.

Rapp, R. (2004). Mining text for word senses using independent component analysis. In *Proceedings of the 4th SIAM International Conference on Data Mining* (pp. 422–426).

Rodríguez, H., Climent, S., Vossen, P., Bloksma, L., Peters, W., Alonge, A., et al. (1998). The top-down strategy for building EuroWordNet: vocabulary coverage, base concepts and top ontology. *Computers and the Humanities, 32,* 117–159.

Roussinov, D., & Zhao, J. L. (2003). Automatic discovery of similarity relationships through Web mining. *Decision Support Systems, 35*(1), 149–166.

Sevillano, X., Alías, F., & Socoró, J. C. (2004). Reliability in ICA-based text classification. In *Proceedings of the 5th International Conference on Independent Component Analysis and Blind Signal Separation* (pp. 1213–1220).

Shlrl, A., & Revle, C. (2006). Query expansion behavior within a thesaurus-enhanced search environment: A user-centered evaluation. *Journal of the American Society for Information Science and Technology, 57*(4), 462–478.

Vanderwende, L., Suzuki, H., Brockett, C., & Nenkova, A. (2007). Beyond SumBasic: Task-focused summarization with sentence simplification and lexical expansion. *Information Processing and Management, 43*(6), 1606–1618.

Wang, T., & Hirst, G. (2010). Near-synonym lexical choice in latent semantic space. In *Proceedings of the 23rd International Conference on Computational Linguistics* (pp. 1182–1190).

Weeds, J., Weir, D., & McCarthy, D. (2004). Characterising measures of lexical distributional similarity. In *Proceedings of the 20th International Conference on Computational Linguistics* (pp. 1015–1021).

Wei, C. P., Yang, C. C., & Lin, C. M. (2008). A Latent semantic indexing-based approach to multilingual document clustering. *Decision Support Systems, 45*(3), 606–620.

Wible, D., Kuo, C.-H., Tsao, N.-L., Liu, A., & Lin, H.-L. (2003). Bootstrapping in a language learning environment. *Journal of Computer Assisted learning, 19*(1), 90–102.

Wu, C.-H. Liu, C.-H., Matthew, H., & Yu, L.-C. (2010). Sentence correction incorporating relative position and parse template language models. *IEEE Transactions* on *Audio, Speech and Language Processing, 18*(6), 1170–1181.

Yu, L.-C., & Chien, W.-N. (2013). Independent component analysis for near-synonym choice. *Decision Support Systems, 55*(1), 146–155.

Yu, L.-C., Lee, L.-H., Yeh, J.-F., Shih, H.-M., & Lai, Y.-L. (2016). Near-synonym substitution using a discriminative vector space model. *Knowledge-Based Systems, 106,* 74–84.

Yu, L.-C., Wu, C.-H., Chang, R.-Y., Liu, C.-H., & Hovy, E. H. (2010). Annotation and verification of sense pools in OntoNotes. *Information Processing and Management, 46*(4), 436–447.

Yu, L.-C., Wu, C.-H., & Jang, F.-L. (2009). Psychiatric document retrieval using a discourse-aware model. *Artificial Intelligence, 173*(7–8), 817–829.