CHAPTER 4

DISTRIBUTIONAL SEMANTICS

This chapter describes the distributional semantics method as the corpus-based approach to solve the problem of word sense disambiguation. The distribution semantics method is an unsupervised learning and based on the distributional hypothesis. As stated in the hypothesis, words that are used in similar contexts will have the same or a closely related meaning. Our approach is the task of grouping the meaning of the target word that is used in similar contexts. The features that we use to represent these contexts are the words or word sequences frequently observed in the context of that word.

4.1 The Distributional Semantics

Distributional Semantics relies on the assumption that the meaning of words is related to their patterns of co-occurrence with other words in the text. This assumption formulated as the *Distributional Hypothesis* of the meaning foundation. The distributional hypothesis states that words with similar meanings tend to appear in similar contexts (Harris, 1968). The theory originates from the work of Zellis Harris written in his book entitled "Mathematical Structures of Language" (Harris, 1968). He states that:

"*The meaning of entities, and the meaning of grammatical relations among them, is related to the restriction on combinations of these entities relative to other entities.*"

For example, if we consider a polysemous word, "fly", that has a different meaning in a context of A (for example, in texts referring to pilot activity) an in the context of B (for example in texts about insects). The reason why the word has different meanings in these different contexts is that it is based on representing a word (or term) by the set of its word co-occurrence statistics.

Given the distributional hypothesis, we can expect that words that resemble each other in their meaning will have similar co-occurrence patterns with other words. For example, both nouns *bank* and *money* co-occur frequently with verbs like *deposit, withdraw, and exchange*. To capture this similarity, each word is represented by a word co-occurrence vector, which represents the statistics of its co-occurrence with all other words in the lexicon. The similarity of two words is then computed by applying some vector similarity measure to the two corresponding co-occurrence vectors.

That is the use of a word is explained by its distributional properties in the language that in its turn is defined by the contexts in which the word occurs. This means that the context can be utilized as a measure of distribution, by which the use of meaning of a word can be determined. Therefore, if the meaning of a word is determined by its use in the natural language, and its use is explained by its distribution, the distributional pattern as defined by the contexts of a word can be seen as a viable method for determining the meaning of that word.

Research in distributional semantics applied to Information Retrieval has been studied in several systems such as Distributional Semantics based Information Retrieval (DSIR) (Rungsawang and Rajman, 1995; Rungsawang, 1997) and Co-occurrence based IR (Schütze and Pedersen, 1997).

4.2 Types of Features

Features are the distinguishing attributes of objects that help to differentiate one object from others. In word sense disambiguation, the features that we use to represent the context of the target word in the test data are all surface level lexical features. These are word–based features that can be observed directly in whatever text is. We represent the context in which a target word occurs using co-occurrences. Co-occurrences mean the number of times the interest word co-occurs with the other adjacent words in the context (e.g. word, sentence, and paragraph).

4.2.1 Co–Occurrences

Two words are called co-occurrences of each other if they occur within some specified window of each other without regard to their order. In the corpus-based framework a word is represented by data about its joint co-occurrence with other words in the corpus. The representation can be constructed by deciding what counts as a co-occurrence of two words and specify types of relationship between occurrences of words.

We store co-occurrences in a matrix called a co-occurrence matrix, whose rows and columns represent the words and cell entries indicate the co-occurrence scores of the corresponding word pairs. For example, Table 4.1 illustrates co-occurrence counts for four words *judge*, *robe*, *legal* and *clothes* in a corpus. The word *judge* and *legal* co-occur 300 times with each other. The matrix entries show the co-occurrence frequency between the corresponding pair of words.

Table 4.1: An example of co-occurrence matrix.

	judge	robe
legal	300	120
clothes	60	200

4.2.1.1 Co-occurrence within a large window

A co-occurrence of words within a relatively large window in the text suggests that both words are related to the general topic in the text. This is for pairs of words that often co-occur in the same text. A special case for this type of relationship is co-occurrence within the entire document, which corresponds to a maximal window size.

4.2.1.2 Co-occurrence within a small window

Co-occurrence of words within a small window captures a mixture of co-occurrences. Typically, only co-occurrence of content words is considered since

these words carry most semantic information. Direction of co-occurrence is considered, distinguishing between co-occurrences with words that appear to the left or to the right of the given word. The smaller the window is, the more associative relations between the words inside the window. If the window size of context is too large, the context cannot contain relevant information consistently (Kilgarriff and Rosenzweig, 2000).

In this thesis, we use co-occurrence within a small window as the local context with a window size ± 2 that is non-overlapping windows. When we slide windows through the sequence words of both left and right of given word, we will slide window until the last word of sentence. If the number of words in current slide window is less than fixed window size, we will not overlap to the first word in the next sentence. The description of window sliding can be illustrated in Figure 4.1. For example, we have two sentences. The first sentence contains seven words and the second sentence contains five words. If we use a window size ± 2 and the given word is w3, the next window slide will have w6 which is given word. Only w7 is spanned in window size.

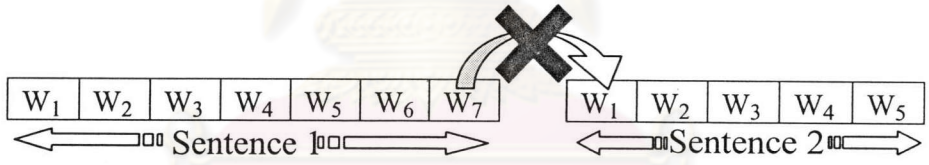

Figure 4.1: An Example of Window Sliding

4.3 Contextual Representation

4.3.1 Word Vector

A word *w* can be represented by a vector in which each component corresponds to a word *v* occurring in the corpus. The vector components represent frequencies of co-occurrence: the component associated with word *v* is the number of times that *v* occurs as a neighbor of *w* in the corpus. A neighbor is a content word occurring in a context window centered on *w*. Co–occurrence matrix is constructed to

show the frequency counts or statistical scores of association between all pairs of words that form co-occurrences in data. Thus, each word is represented by co-occurrence vector that shows the association of the other words in sentences containing the target word.

Instead of using word frequency as a component of vector directly, in this thesis, we compute the values in the matrix are the log–likelihood ratio between the corresponding pairs of words. The log-likelihood test was applied to all values in the matrices. The idea is to emphasize significant and to weaken incidental word pairs by comparing their observed co-occurrence counts with their expected co-occurrence counts. We selected the log-likelihood test instead of choosing the better known χ^2 test because Dunning (Dunning, 1993) showed that the log-likelihood test yields good results with relatively samples which are not larger size. The log–likelihood ratio for each pair of words can be formulated as follows:

$$2 \log \lambda = \sum_{i,j \in \{1,2\}} k_{ij} \log \frac{k_{ij} N}{C_i R_j} \tag{4.1}$$

$$= k_{11} \log \frac{k_{11} N}{C_1 R_1} + k_{12} \log \frac{k_{12} N}{C_1 R_2} \tag{4.2}$$

$$= k_{21} \log \frac{k_{21} N}{C_2 R_1} + k_{22} \log \frac{k_{22} N}{C_2 R_2} \tag{4.3}$$

where $C_1 = k_{11} + k_{12}$ $\quad C_2 = k_{21} + k_{22}$

$R_1 = k_{11} + k_{21}$ $\quad R_2 = k_{12} + k_{22}$

$N = k_{11} + k_{12} + k_{21} + k_{22}$

Parameters k_{ij} can be expressed in terms of corpus frequencies as follows:

k_{11} = frequency of common occurrence of word A and word B

k_{12} = corpus frequency of word A – k_{11}

k_{21} = corpus frequency of word B – k_{11}

k_{22} = size of corpus (no. of tokens) – corpus frequency of A – corpus frequency of B.

4.3.2 Dimension Reduction

Because of its large size and sparseness, we employ Singular Value Decomposition (SVD) to reduce the dimensionality. We reduce the matrix to 10% of its original number of columns, or 300 columns, whichever is least. Thus, any matrix of 3,000 or more columns will be reduced to 300 columns, while those less than 3,000 columns are reduced to 10% of their number of columns. Note that SVD reduces the number of columns, but not the number of rows. The reduction has two effects. Firstly, it acts as a smoothing operation, where the resulting matrix will have very few (if any) zero values. Secondly, it has the effect of reducing the words that make up the columns from a word level feature space into a concept level semantic space. Although our approach is related to Schütze (Schütze, 1998), our use of log–likelihood scores makes it somewhat distinct, since the usual technique is to create a word co–occurrence matrix that employs frequency counts.

4.3.3 Context Vector

The context around of the target word is represented as a vector of features called a *context vector*. The context of each target word is then represented by the sum of all the vectors associated with the words in the context.

The context vector indirectly represents contexts in terms of the words that co-occur with the contextual words rather than with the target word. The reason behind this indirection is that a co-occurrence of a word is assumed to capture the meaning of that word in terms of the words that most frequently co-occur with it, and by averaging the word vectors of contextual words. We find the meanings of feature words found in any context by identifying the words that most commonly co-occur with them in the training data. For example, Figure 4.2 shows the context vector of an example context of *suit* containing the words *law*, *judge*, *statute*, and *suit*. The context vector is closer to the *legal* than to the *clothes* dimension, thus capturing that the context is a *legal* use of *suit*.

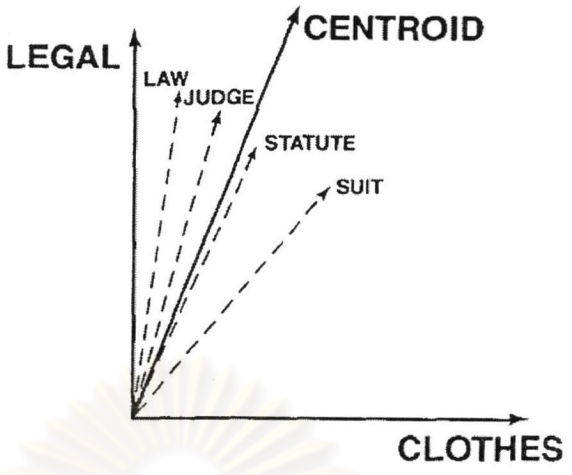

Figure 4.2: shows the context vector of an example context of *suit*

4.4 Sense Clusters

Similar context vectors can be seen as forming clusters in vector space. Each cluster represents one sense of an ambiguous word and can be characterized by its mean and covariance matrix. The sense of a new instance *w* is then assigned to the most similar cluster.

The parameter, i.e. number of clusters to be created, specifies a terminating condition to a clustering algorithm that will continue to cluster until the required numbers of clusters are formed. An agglomerative algorithm starts with *N* initial clusters (for *N* test instances) and merges the most similar pair of clusters at each iteration. The partitional algorithms divide the set of objects into a given number of clusters, and then iteratively refine those clusters.

4.5 Methodology

The steps of methodology can be illustrated in Figure 4.3.

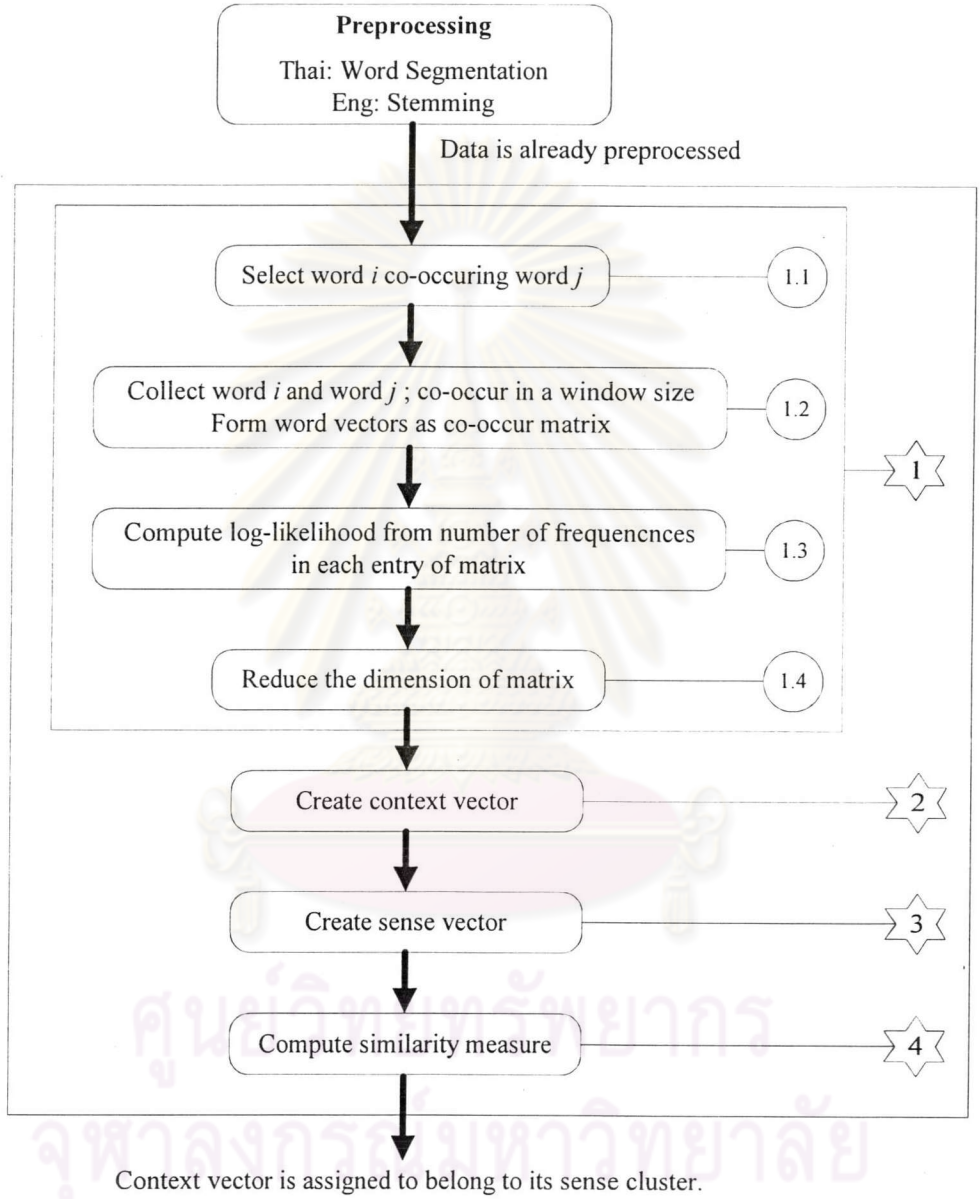

Figure 4.3: The Steps of Methodology

Each step which is shown in Figure 4.3 can be described below.

Step 1

1.1 Find feature vector as *word vector.* Word vectors are derived from neighbors in the corpus. A vector for word *i* is derived from the close neighbors of *i* in the corpus. Close neighbors are all words that co-occur with *i* in a sentence or a larger context. The entry for word *j* (e.g. money) co-occurs in the vector for *i* (e.g. bank). The number of times that word *j* occurs close to *i* in the corpus is recorded.

1.2 Form all feature vectors in matrix representation. The feature vector is co-occurrence of a word within windows size ±2. Word vectors are formed by collecting words *i* and words *j* co-occur in a window size. Words that are represented as word vectors are also formed as the dimensions space which represented by the matrix form. This matrix is called *co-occurrence matrix* whose rows and columns represent the words and element entries indicate the number of times of the corresponding word pairs. As the same word represents the row/column and as the value at (*i*,*j*) is the same as that at (*j*,*i*), the co-occurrence matrix is always square and symmetric.

1.3 Represent feature value in term of measures scores. A real valued feature can be represented by the scores of measures of association which is the log–likelihood ratio. The entry values which are the number of times of the corresponding word pairs in the matrix are computed to be the log–likelihood ratio between the corresponding pairs of words.

1.4 Reduce the dimension of word vectors. We use Singular Value Decomposition (SVD) to reduce the dimension of word vectors. We reduce the matrix to 10% of its original number of columns, or 300 columns.

Step 2

2.1 Create context vector. The context vectors are derived from word vectors. A context vector is the centroid (or sum) of the vectors of the words occurring in the context.

Step 3

3.1 Create sense vector. The sense cluster can be created by grouping similar contexts. The centroid of cluster is the representation of a sense. Sense representations are computed as groups of similar contexts. This set of context vectors is then clustered into a predefined number of clusters or context groups. The representation of a sense is simply the centroid of its cluster. We chose the K-Means algorithm for clustering since it does not require an exhaustive series of pairwise comparisons like the agglomerative methods do. In our case, we use Euclidean Distance to find the distance between context vectors and their centroids. The context vectors which have the shortest distance is assigned to the closest cluster. In other words, the centroids are representatives for the context vectors in their cluster. An example is shown in Figure 4.4, the clustering step has grouped context vectors cl, c2, c3, and c4 in the first group and c5, c6, c7, and c8 in the second group. The sense vector of the first group is the centroid labeled SENSE 1, the sense vector of the second group is the centroid labeled SENSE 2.

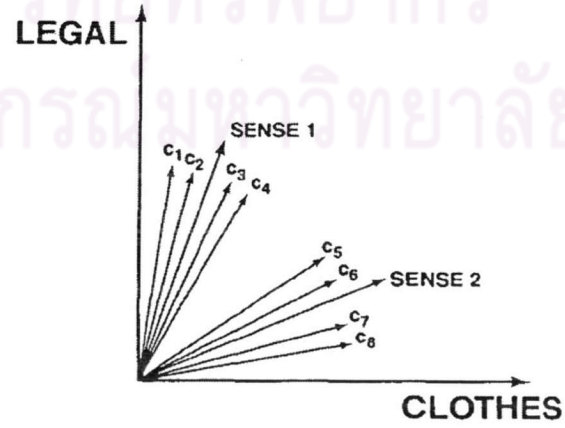

Figure 4.4: The derivation of sense vectors.

Step 4

4.1 Compute similarity measure. Having created the feature vectors for each of the selected feature word and get sense vector in the training data, we then build a context vector for each target word in the test data. The similarity measure between context vector of test data and sense vector which is created from training data is computed via cosine similarity measurement. The context vector which is closest to the sense vector is disambiguated and is assigned to belong to that sense cluster.

4.6 Comparison between Schütze and Thesis Approach

The following Table 4.2 gives a short summary of the differences between Schütze (Schütze, 1998) and thesis approach.

Table 4.2: Summary of the differences between Schütze (Schütze, 1998) and thesis approach.

	Schütze's work	**Thesis Approach**
Co-Occurrence Feature		
1. Feature selection	1. Using frequency counts and χ^2 test for likelihood estimation.	1. Using the log–likelihood ratios between the word pairs.
2. Windows size (Dimension of word space)	2. Co-occurrence within a large window size ± 25 as a local context.	2. Co-occurrence within a small window size ± 2
Data	1. English words	1. Thai and English words