Discovering Word Senses for Polysemous Words Using Feature Domain Similarity

Noriko Tomuro
DePaul University, School of Computer Science, Telecommunications and Information Systems,
Chicago, IL 60604 USA
tomuro@cs.depaul.edu

Kyoko Kanzaki, Hitoshi Isahara
National Institute of Information and Communications Technology,
Soraku-gun, Kyoto, 619-0289 Japan
{kanzaki,isahara}@nict.go.jp

Abstract

This paper presents a new clustering algorithm called *DSCBC* which is designed to automatically discover word senses for polysemous words. DSCBC is an extension of CBC (Pantel and Lin, 2002), and incorporates *feature domain similarity*: the similarity between the features *themselves*, obtained a priori from sources external to the dataset. By incorporating the feature domain similarity in clustering, DSCBC produces *monosemous clusters* (a cluster in one domain), thereby discovering individual senses of polysemous words. For evaluation, we apply the algorithm to Japanese and English *adjectives*, and compare the derived senses against manually created lexicons. The results show significant improvements over other clustering algorithms including CBC.

1 Introduction

Automatic acquisition of word senses from corpora has been an active topic of research in Natural Language Processing (NLP). Most conventional thesauri are manually compiled by lexicographers, therefore often contain inconsistencies. Word senses are also sometimes too fine-grained or specific, and not based on the common usage of the language, as frequently pointed out for WordNet (Miller, 1990). A better approach is to automatically derive word meanings from real data.

In the NLP research, many techniques have been developed which discover word senses or semantic classes from corpora (e.g. (Hindle, 1990; Lin and Pantel, 2001)). However, most of them have focused on nouns, which are largely monosemous, so the applicability of those techniques to polysemous words such as verbs and adjectives is unknown. Walde (2006) and Boleda et al. (2004) present clustering experiments which automatically derive semantic classes for German verbs and Catalan adjectives respectively. However, they both use standard clustering methods (such as K-means) only, and no attempt is made to develop new techniques tailored to polysemous words.

Pantel and Lin (2002) develop a clustering algorithm called *CBC* (Clustering By Committee), and report a superior performance of CBC over other clustering algorithms in discovering word senses from texts. However, their results are shown again mostly for nouns, and in fact, the algorithm does not include any special considerations to avoid generating *polysemous clusters*: a cluster in which several word meanings are mixed in. The problem of polysemous clusters also exists in other clustering algorithms when data contains many polysemous words. Indeed, some of the semantic classes derived in (Walde, 2006) are polysemous clusters.

In this paper, we present a new clustering algorithm called *DSCBC*, which is specifically designed to automatically discover senses of polysemous words (but works for monosemous words as well). DSCBC is an extension of CBC, and incorporates *feature domain similarity*: a prior knowledge about the similarity between features *themselves*. For example, if data instances x and y are represented by features $f1$ and $f2$, the feature domain similarity refers to the similarity between $f1$ and $f2$ (not between x and y). Feature domain similarity is obtained a priori from sources external to the dataset(s) used at hand. By incorporating the feature domain similarity in clustering, we are able to group data instances which have features in the same or similar *domain*, thereby producing monosemous clusters.

To test the effectiveness of DSCBC, we apply the algorithm to Japanese and English adjectives. In this work, we use *abstract nouns* as the features to represent adjectives. From a large corpus, we collect many instances where adjectives modify abstract nouns (e.g. "happy feeling") and construct a dataset for each language. Then after applying the algorithm, we compare the automatically derived word senses against manually created lexicons (*Daijirin* for Japanese and WordNet for English). The results show significant improvements by DSCBC over other clustering algorithms in discovering polysemous adjective senses.

2 Semantic Categories of Adjectives

In linguistics, there is a large body of work on adjectives. However, the attention they have received is much less than that for nouns and verbs. Likewise in NLP, there exist little other works which tackled adjectives specifically. Adjectives are usually considered in the context of nouns which they modify, and thought to add only auxiliary information to the nouns. Moreover, adjectives are generally difficult to incorporate in NLP applications because of their polysemous nature: not only do adjectives change or shift meanings depending on the nouns they modify (e.g. "warm temperature" vs. "warm person"), they are often used metaphorically (e.g. "dark conversation" – a discussion on grim topics, or to talk pessimistically). Most of the recent work on adjectives in NLP has focused either on specific applications (e.g. classifying documents according to subjectivity or sentiment (Wiebe, 2000)), or on specific types of adjectives (e.g. event adjectives (Lapata, 2001) and gradable adjectives (Hatzivassiloglou and Wiebe, 2000)).

Traditionally, meanings of a polysemous word are enumerated in dictionaries. For example, WordNet lists ten senses for "warm". Distinguishing different senses of polysemous words is a difficult problem for any part of speech. However for adjectives, salient homonym-level meanings are distinguishable based on the *domains* of the co-occurring nouns (Justeson and Katz, 1995).

Note that the premise of sense enumerability can be argued. Notably, Pustejovsky (1995) proposes a generative view of adjectival meanings: meanings of adjectives are fundamentally indifferentiable, and a specific meaning is generated (by coercion) when an adjective is combined with a noun in the context. However, knowing the specific meaning of an adjective in the given context would greatly help the understanding of the sentence in NLP tasks and applications, such as information extraction and question-answering. Our work in this paper aims to acquire such specific adjectival meanings that arise in the context as abstract *semantic categories* or *classes* of adjectives, automatically from corpora. Then the derived semantic adjectival categories can be used in constructing lexicons as well as investigating the polysemy of adjectives linguistically.

3 Datasets

In this work, we use *abstract nouns* as the features to represent adjectives, such as "gentle personality" (as versus concrete nouns such as "gentle goats"). Previous studies in linguistics have shown that, when an abstract noun is modified by an adjective, it often correlates with the semantic class of the adjective (e.g. "gentle" as a personality adjective) (Takahashi, 1975). Therefore, abstract nouns provide the meanings of adjectives literally and immediately, while with concrete nouns, the meanings must be inferred.

In the current work, we applied our sense discovery algorithm to two datasets. The first dataset is Japanese, extracted from a large volume of Japanese newspaper articles, and containing a large number of examples of adjectives with co-occurring abstract nouns. This dataset was previously used in (Kanzaki et al., 2004). From the original dataset, we randomly selected 1500 instances/adjectives to create the dataset. The number of co-occurring nouns/features was 361.

The second dataset is English, extracted from the Web n-gram corpus created by Google Inc. (Brants and Franz, 2006). This corpus contains English word n-grams (uni-grams to five-grams) and their frequency counts obtained from publicly accessible Web pages containing approximately 1 trillion word tokens. To extract adjectives co-occurring with abstract nouns, we first translated the nouns used in the Japanese dataset into English, and obtained a set of 277 English abstract nouns.[1] Then we extracted instances of the forms 'Adj AbN' and 'AbN BE Adj' (where BE stands for a copula verb, for instance "nature is gentle") from the bi-grams and tri-grams respectively.[2] Note that, although neither pattern guaran-

[1]Since there were several cases where two or more Japanese nouns corresponded to the same English noun (and less number of cases the other way around), the number of the English nouns was less than that of the Japanese nouns.

[2]The part of speech for Adj was checked against WordNet – we extracted instances where the word at the respective Adj

tees the matched AbN is truly the noun which the Adj modifies, no outstanding spurious instances were observed. There were over 15000 adjectives which co-occurred with the selected abstract nouns in the corpus. Then we randomly selected 1500 of them to create the dataset.

Besides language, the two datasets have a marked difference: the Japanese dataset is relatively sparse, while the English dataset is very dense. This is probably because newspapers use adjectives less than other genres, while web pages include a variety of genres and authorship.

For both datasets, we represented each adjective by a feature vector, where a feature was an abstract noun co-occurred with the adjective. The value was the *Mutual Information (MI)* computed from the frequency counts in the respective corpus.[3] The MI between two words x and y, denoted $I(x,y)$, is defined as:

$$I(x,y) = log\frac{p(x,y)}{p(x)p(y)}$$

where $p(z)$ is the probability of a word z, and $p(x,y)$ is the joint probability of x co-occurred with y. MI indicates the mutual dependence between two random variables, where $I(x,y) = 0$ if x and y are independent, or a positive value otherwise. In our case, $I(x,y)$ essentially indicates how well a feature noun predicts (or is correlated with) a given adjective. MI has been often used in NLP as a way to put weights on feature values (e.g. (Hindle, 1990)).

4 Clustering By Committee (CBC) Clustering

Our sense discovery algorithm is an extension of *Clustering By Committee (CBC) Clustering* (Pantel and Lin, 2002). We introduce CBC and discuss its limitations in this section, then describe the modifications we made to CBC to derive our algorithm in the next section.

4.1 The CBC Algorithm

CBC is an unsupervised clustering algorithm which automatically derives a set of *committees*. A committee is a cluster of words which are very similar to each other – similar in notion to a synonym set (synset). As with synset, each committee corresponds to a word sense. In CBC, a committee is represented by the centroid vector of the word vectors which comprise the cluster. The algorithm automatically derives committees by first finding a *tight* cluster of words which are similar to a given word, for every word in the dataset, then selecting a subset of the derived clusters whose centroid vectors are dissimilar/orthogonal to each other as committees. After obtaining the committees, CBC discovers the senses of a word by assigning the word to its most similar committees – all committees with which the similarity is above a threshold, and removing the features of the selected committees from the word as they are selected.

```
Phase I: Find clusters.
  Let $L$ be a list of clusters, initially empty.
  For each word $w \in W$ in the dataset,
    Select a set $c$ of at most $n$ words from $W$
      which are the most similar to $w$, and
      add $c$ to $L$ if it is not already in $L$.
  Sort $L$ in the descending order of the scores of
  the clusters.$^{(1)}$
Phase II: Find committees.
  Let $C$ be a list of committees, initially empty.
  For each cluster $c \in L$ (in the sorted order),
    Compute the centroid of $c$.$^{(2)}$
    If it is not similar to any other committee
      in $C$, add $c$ to $C$.
Phase III: Assign words to committees.
  For each word $w \in W$,
    Select all committees in $C$ whose centroids
      are similar to $w$.
```

Figure 1: The CBC Algorithm

Figure 1 shows the overall steps of the CBC algorithm. A cluster found in Phase I is tight in that it consists of at most n similar words (where $n = 10$ is used in (Pantel and Lin, 2002)). Then, clusters are selected into the set of committees in the descending order of their scores (indicated with (1) in the algorithm). This way, CBC guarantees that only the tight clusters which are also well scattered in the feature space are selected as committees. The score of a cluster c is computed as $|c| \times$ avgsim(c), where $|c|$ is the number of words in c, and avgsim(c) is the average pairwise similarity between the words in c. Note that throughout

position is recorded as an adjective in WordNet.

[3] Actually for the English data, we took the log of base 10 of the frequency counts, since the values were significantly higher and their range was much wider than the Japanese dataset.

the CBC algorithm, the cosine coefficient (Salton and McGill, 1983) is used to measure the similarity between vectors, and MI is used to compute the values in the word vectors.

4.2 Limitations of CBC

Pantel and Lin (2002) applied this CBC algorithm to a large corpus (a parsed TREC collection), and reported its superior performance over other standard clustering algorithms as well as their predecessor algorithm called UNICON (Lin and Pantel, 2001). However, one problem with CBC is that it does not work well for polysemous words despite its claimed utility. The reason is because the algorithm uses the centroid to represent a committee (indicated with (2) in the algorithm). Consider the adjectives "warm" and "cold". These words have similar sense patterns, in particular, they both have the senses of temperature and personality. Many other adjectives which elaborate temperature such as "cool" have those two meanings as well – the pattern of temperature-personality is rather regular in adjectives (i.e., *systematic polysemy* (Pustejovsky, 1995)). Since their word vectors are similar, they are most likely grouped in the same cluster. And if this cluster is selected as a committee (where the algorithm has no mechanisms besides cosine to prevent it), the committee centroid will end up having both senses, thus failing to separate the two individual senses.

For CBC to produce monosemous committees, either the data contains mostly monosemous words, or sense patterns of the polysemous words in the data are not regular so that they won't constitute tight clusters. In the case of adjectives, words are highly polysemous,[4] and there are many patterns of polysemous senses that are regular.

The problem of polysemous clusters also arises in other clustering algorithms when the data contained many polysemous words. The problem will happen in other methods besides clustering in the same way as well. For example, Principal Component Analysis (PCA) finds a set of principal components (linear combinations of features) which account for the variance in the data. Principal components are orthogonal to each other, and in that sense, CBC has the same objective as PCA. However again, features involved in the regular polysemy patterns of the polysemous words are grouped into the same principal component, resulting in a heterogeneous, polysemous component in PCA.

[4]In WordNet, the proportion of polysemous words is 31% for adjectives, as compared to 16% for nouns. In our English dataset, 723 (= 48%) of the 1500 adjectives had more than one sense in WordNet.

5 Domain Similarity CBC (DSCBC) Clustering for Polysemous Words

5.1 Feature Domain Similarity

To overcome the polysemous cluster problem discussed above, we developed a new metric which measures the similarity between the features themselves, which we call the *Feature Domain Similarity*. This is the primary characteristics and the contribution of our algorithm.

Consider the following data:

```
   a b c d
x: 1 1 0 0
y: 1 0 1 0
z: 1 0 0 1
```

where *x*, *y*, *z* are data instances, and `a`, `b`, `c`, `d` are features. In most clustering algorithms, features are assumed to be independent to each other, or their dependencies are ignored. So in the example, *x* is equally likely clustered with *y* or *z*, because the similarity between *x* and *y*, and *x* and *z* are the same (based on the Euclidean distance, for example). However if we have a priori, general knowledge about the features that `b`'s domain is more similar to that of `c` than to `d`, it is better to cluster *x* and *y* instead of *x* and *z*, because the $\{x, y\}$ cluster is "tighter" than the $\{x, z\}$ cluster with respect to the domains of the features.

Feature domain similarity can be obtained from any linguistic resources (but should be external to the dataset(s) to be used in the experiments). For Japanese, we used the case frame data extracted automatically from the Web (Kawahara and Kurohashi, 2006). In our work, features are abstract nouns. So for each abstract noun (e.g. *kimochi* "feeling"), words which appeared in various case frame relations with the noun (largely verbs and adjectives, e.g. *kimochi ni* *naru* "come to bear feeling", *kimochi wo* *tsutaeru* "convey feeling") were used to represent the noun.[5] Then the cosine coefficient was used to compute the similarity between the nouns. For English, we used *javasimlib*:[6] a Java-based tool which computes the similarity between words (or synsets) over the Word-

[5]http://reed.kuee.kyoto-u.ac.jp/cf-search/

[6]http://wordnet.princeton.edu/links#extensions

Net hierarchies based on an information theoretic metric (Seco et al., 2004). Given two words or synsets, *javasimlib* returns a value between 0 and 1, where 1 indicates the highest similarity. For each noun, we considered the top (at most) two senses in the calculation in order to avoid spurious results arising from rare or unimportant senses encoded in WordNet. Then we defined the similarity between two nouns as the maximum value between their top two senses, that is,

$$fdsim(a,b) = \arg\max_{i,j \in \{1,2\}} jsim(a\#i, b\#j)$$

where a, b are nouns, $fdsim(a,b)$ denotes the feature domain similarity between a and b, $a\#i$ is the ith sense of a, and $jsim(x,y)$ is the similarity between two synsets x and y returned by *javasimlib*. Then for each dataset, we computed all pairwise similarities between the nouns/features and stored them in a matrix.

Next, we defined a notion of *domain tightness* for a single word vector. This metric indicates how "tight" the vector is with respect to the domains of its features (i.e., the degree of monosemy). For a given word vector v, the domain tightness of v, denoted $dt(v)$, is define as:

$$dt(v) = \frac{1}{n} \sum_{a,b \in F; a \neq b} \frac{v(a)+v(b)}{2} fdsim(a,b)$$

where F is the set of features used to represent v, $v(x)$ is the value in v for the feature $x \in F$, $fdsim(a,b)$ is the feature domain similarity between a and b described above, and n is the number of pairwise combinations of the features in F where their feature-values are greater than zero (i.e., n = count($\langle$a,b$\rangle$: $v(a) > 0$ and $v(b) > 0$)). Thus it is the average feature domain similarity between two non-zero features in v, weighted by the average of the two feature-values.

Finally, we defined a new similarity metric between two word vectors that incorporates the feature domain similarity, which we call *dsSim*, as follows. For given word vector $v1$ and $v2$, the similarity between them, denoted $dsSim(v1,v2)$ is:

$$dsSim(v1,v2) = w_0 \times cos(v1,v2) + w_1 \times dt(v3)$$

where $cos(v1,v2)$ is the cosine between $v1$ and $v2$, $v3$ is a vector in which $v1$ and $v2$ are merged (i.e., $v3 = v1 + v2$), and w_0, w_1 are weights which sum up to 1. We used $w_0 = 0.95$ and $w_1 = 0.05$ in our experiments. This new measure is essentially based on cosine, added with the domain tightness of the merged vector. We looked at the merged vector, because a cluster centroid is a vector in which all member word vectors are merged (and averaged), and that's precisely what we wanted to improve upon CBC – the domain tightness of the cluster/committee centroids.

5.2 The DSCBC Algorithm

In addition to feature domain similarity, we also incorporated a few more ideas which may help derive monosemous committees. One idea is to limit the number of features to be used in the committee centroids. By taking the top k features, only the most salient features are kept (which hopefully are features in one domain), and other insignificant features are discarded. Another idea is to use only the features that appeared with the majority of the words which make up the committee.

Furthermore, we derived the committees incrementally in two steps: first produce committees from the original dataset, then derive the final set of committees from those produced in the first step. This hierarchical scheme is aimed to find the second or higher-order features (such as those in a transitive similarity relation).

We present our algorithm named DSCBC (for *Domain Similarity CBC*) in Figure 2. Overall, it applies the (modified) Phase I and II of CBC twice successively. The modifications to the original CBC are: (1') use the new similarity measure *dsSim* instead of cosine to compute the scores of the derived clusters; and (2') remove from committee centroids the features which co-occurred with less than σ percent of the member words or which are not in the top k features. All other parts of the algorithm are unchanged from CBC.

Note that DSCBC may still group "warm" and "cold" in the same cluster. However, such a cluster would have a lower *dsSim* value (hence the cluster score) because the domains of temperature and personality are not similar themselves, thus is less likely to be selected as a committee. If it were, features in less dominant domains (i.e., extended meanings) are discarded because only the top k features are retained in the centroid, thereby resulting in a monosemous committee. In this sense, a DSCBC committee is not equivalent to a synset – rather, it is a list of features (abstract nouns) which explicitly describe the semantic domain(s) of the

```
Step 1: Derive committees from data.
  Apply Phase I and II of CBC to the dataset
  using:
    - dsSim in computing avgsim($c$)$^{(\mathbf{1'})}$
    - the top $k$ features that are also shared by
      more than $\sigma$ percent of the words in the
      committee to represent its centroid.$^{(\mathbf{2'})}$
Step 2: Derive committees from committees.
  Repeat Step 1, but with the committees
  derived in Step 1.
Step 3: Assign words to committees.
  Apply Phase III of the CBC algorithm to the
  dataset using the final set of committees
  derived in Step 2.
```

Figure 2: The DSCBC Algorithm

cluster.

In summary, the key to overcome the polysemous cluster problem is to use externally obtained information about the similarity/dependency between features and incorporate the information in the clustering process.

5.3 Related Work

There are only few other works which attempted to automatically derive semantic categories of adjectives. Tanaka and Hirai (2001) used Latent Semantic Analysis (LSA) to cluster (Japanese) adjectives. However, they used individual senses of the adjectives recorded in a dictionary (*Kojien*) as the data instances, thus did not deal with the polysemy of adjectives.

Hatzivassiloglou and McKeown (1993) cluster adjectives by their properties or *scales*. For example, a set of adjectives {cold, cool, warm} is a linguistic scale which indicates a variation in the intensity of temperature. They extracted instances of adjective-noun pairs from a parsed corpus, and used Kendall's τ coefficient as the similarity measure and a hill-climbing partitive algorithm to cluster the extracted adjectives. However, their work focuses only on scalar adjectives, and does not concern groupings which indicate non-scalar properties (such as personality).

6 Evaluation

To evaluate our algorithm DSCBC, we applied it to the two datasets described earlier and compared the results with other clustering algorithms, in particular CBC, a graph-based clustering and K-means. For the graph and K-means algorithms, we used a tool called CLUTO.[7] Also for those two algorithms, we made some modifications so that they do soft-clustering (which assigns an instance to one or more clusters) in order to make them comparable with CBC and DSCBC. To make modifications, we closely followed the way described in (Pantel and Lin, 2002): first apply the algorithm to the dataset and obtain clusters, then apply *MK-means* (Zhong and Ghosh, 2003) using the centroids of those clusters as the initial centroids. MK-means is a generalized version of the standard K-means algorithm, and assigns each instance to one or more clusters with which it has the similarity greater than a pre-specified threshold. As with K-means, MK-means performs several iterations until a pre-specified number of iterations is reached. In our experiments, we set the maximum number of iterations to be 5, as with (Pantel and Lin, 2002).

Furthermore, since DSCBC is an extension of CBC with the domain feature similarity, we also implemented the versions of the graph and K-means algorithms which extend the base algorithms in the same way. So in all, we compared a total of six algorithms: DSCBC, CBC, DSGr, Gr, DSKmeans and Kmeans.

For the parameters in the DS extended algorithms (DSCBC, DSGr, DSKmeans), for the English dataset, we set $k = 10$ and 6, and $\sigma = 0.8$ and 0.6 in Step 1 and Step 2 respectively. We reduced k from Step 1 to Step 2 in order to refine the salient features. For σ, we used somewhat higher values, especially in Step 1, because the dataset was relatively dense (since word usage is extremely diverse in web pages). For the Japanese dataset, we set $k = 6$ in both steps, and $\sigma = 0.4$ and 0.2 in Step 1 and Step 2 respectively. We used the same k and lower σ values because the Japanese dataset was rather sparse. We decided on those parameter values based on our intuitions and quick preliminary inspections; they were not optimized for any specific algorithm tested.

6.1 Derived Committees

First we examined the committees derived by the algorithms. Table 1 shows some statistics. DSCBC produced 38 committees for both datasets (coincidentally), while CBC produced 35 and 50 committees for the Japanese and English

[7]http://glaros.dtc.umn.edu/gkhome/views/cluto

Table 1: Some Committee Statistics

	Total # comm.	Ave # feat.	Ave Domain Tightness (p-value)	Ave Cosine (p-value)
Japanese				
DSCBC	38	5.7	0.172 (–)	0.026 (–)
CBC	35	5.8	0.138 (*)	0.025 (0.447)
DSGr	38	6.0	0.153 (0.08)	0.069 (*)
Gr	35	6.0	0.154 (0.09)	0.056 (*)
DSKmeans	38	6.0	0.146 (*)	0.044 (*)
Kmeans	35	6.0	0.142 (*)	0.037 (*)
English				
DSCBC	38	3.3	0.470 (–)	0.010 (–)
CBC	50	4.9	0.196 (*)	0.012 (0.18)
DSGr	38	6.0	0.224 (*)	0.018 (*)
Gr	50	6.0	0.221 (*)	0.022 (*)
DSKmeans	38	6.0	0.209 (*)	0.012 (0.20)
Kmeans	50	6.0	0.198 (*)	0.011 (0.37)

Japanese

{*shinjo* "feeling", *kimochi* "feeling", *kibun* "mood", *kanjo* "emotion", *omoi* "feeling"}
{*imiai* "meaning", *shikisai* "tint", *imi* "meaning", *sokumen* "side", *men* "side"}
{*kibo* "scale", *gaku* "amount", *ryo* "volume", *kazu* "quantity", *aida* "interval", *deido* "degree"}
{*kuuki* "air", *kikou* "climate", *tenki* "weather", *seikaku* "personality", *imeji* "image"}

English

{smell, aroma}
{appearance, look}
{quantity, amount, number}
{attitude, countenance, nature}
{day, shade, room, light, face, color}

Figure 3: Example DSCBC Committees

dataset respectively. Other DS algorithms (DSGr and DSKmeans) and non-DS algorithms (Gr and Kmeans) were pre-specified to produce the same number of committees as DSCBC and CBC respectively. The average number of (non-zero) features was 5.7 and 3.3 by DSCBC, and 5.8 and 4.9 by CBC, for Japanese and English respectively. For all other algorithms, the average number of features was 6 (= k, as described in section 5.2).

The column "Ave Domain Tightness" indicates the average domain tightness (dt) of the committees derived by each algorithm. Here, a higher value means the feature domain similarity of the committees are overall more similar, therefore the committees are less polysemous. As you can see, DSCBC showed the highest tightness (0.172 and 0.470 respectively). The results by all other algorithms, including other DS algorithms, were much lower than DSCBC, and the differences were statistically significant, as evidenced by the p-values[8] (where a symbol (*) indicates < 0.05). As a note, the domain tightness was in general much lower for the Japanese dataset. It is probably due to the difference in the sources used to compute the feature domain similarity – for Japanese, the case frame relations were collected from the Web and the words were not disambiguated, so the data is somewhat noisy, whereas for English we used the first two senses of a word in *javasimlib*.

The column "Ave Cosine" indicates the average pairwise cosine between the committees for each algorithm. Here, a lower value means the committees are dissimilar to each other, thus scattered well in the feature space. The result shows that DSCBC had the lowest value, although the differences were not statistically significant in some cases. As for the graph and K-means algorithms, the graph in general seems to produce more monosemous but more correlated clusters than K-means, as observed for both datasets.

Figure 3 shows some example committees derived by DSCBC for both datasets. Most of the committees seem to pick out a single semantic domain fairly well, but some are still polysemous clusters. For example, {*kuuki* "air", *kikou* "climate", *tenki* "weather", *seikaku* "personality", *imeji* "image"} in Japanese is a mixture of weather and personality (as we have been using as an example in this paper). Similarly, {day, shade, room, light, face, color} in English seems to group different domains – probably caused by the polysemy of color or luminance adjectives such as "bright" and "dark".

6.2 Assigned Word Senses

Next we inspected the senses/committees assigned to words. For each word, the algorithms assign one or more committees. Here, each committee should correspond to a sense of the word. Figure 4 shows some examples of the assigned senses by DSCBC. To determine whether or not a committee indeed corresponds to a correct sense of a word, we compared it against the sense encoded in manually created lexicons as gold standards: *Daijirin*[9] for Japanese, and WordNet for English. Then the evaluation was measured with respect to precision and recall.

[8]The p-values shown in Table 1 are obtained by one-sided t-tests against DSCBC.

[9]The web version available at http://dictionary.goo.ne.jp/

Japanese
atatakai "warm"
{*kuuki* "air", *kikou* "climate", *tenki* "weather", *seikaku* "character", *imeji* "image"}
{*hitogara* "personality", *seishitsu* "nature", *monogoshi* "manner"}
yawarakai "soft"
{*kankaku* "sensation", *kanshoku* "touch", *aji* "taste", *yosa* "merit"}
{*hitogara* "personality", *seishitsu* "nature", *monogoshi* "manner"}
hiroi "wide"
{*kibo* "scale", *gaku* "amount", *ryo* "volume", *kazu* "quantity", *aida* "interval", *deido* "degree"}
{*shiya* "view", *kenchi* "viewpoint", *kanten* "viewpoint", *kakudo* "angle"}
English
"cold"
{complexion, color, face}
{attitude, countenance, nature}
"democratic"
{framework, concept, approach, idea, model}
{tendency, view, nature}

Figure 4: Example Sense Assignments by DSCBC

6.2.1 The English Dataset

For the English dataset, we first looked up each of the 1500 adjectives in the dataset in WordNet and obtained its senses/synsets. Then for each synset, we mapped it to its related noun so that we can utilize the WordNet noun hierarchy to compute the similarity/correspondence.[10] For a given synset, we traversed the *attribute* relation encoded in WordNet, for example, "warm#1" → "temperature#1". If the attribute relation was not available, we traversed the *derivationally related form* relation, for example, "warm#4" → "warmness#1". Word senses which are not indicated with either relation were ignored in the evaluation. The average number of senses (of the 1500 adjectives) which were associated with either relation was 1.63. Finally, we determined that an automatically derived committee corresponds to the mapped noun synset if the average similarity between the nouns in the committee and the synset is above a threshold, that is,

$$\frac{1}{|c|}\sum_{f\in F(c)} jsim(f,s) \geq \theta$$

where c is a committee, f is an abstract noun in $F(c)$ (the set of (non-zero) features in c), s is the mapped WordNet noun synset, $jsim(f,s)$ is the similarity between f and s obtained through *javasimlib*, and θ is the threshold. For $jsim(f,s)$, the top (at most) two senses were considered for the feature noun f, and the maximum of the two values returned by *javasimlib* ($jsim(f\#1,s)$ and $jsim(f\#2,s)$) was used.

Then we computed the *precision* of an adjective a as the ratio of the correctly assigned committees of all committees assigned by the algorithm for a. The precision of an algorithm was the average precision of all adjectives in the dataset.

The *recall* of an adjective a, on the other hand, was computed as the ratio of the correctly discovered senses of all senses encoded in WordNet for a. This recall actually is a tough measure, because the coverage of our dataset is much more limited than WordNet. But we thought this measure could provide some indication on the coverage of the algorithms and be utilized to rank the algorithms. The recall of the algorithm was the average recall of all adjectives in the dataset.

Table 2 shows the precision, recall and F-measure for the English dataset when θ = 0.25. The F-measure was computed as standard:

$$F = \frac{2RP}{R+P}$$

where R is the recall, P is the precision. As you can see, DSCBC produced the highest precision as well as recall. Also, the DS-extended algorithms performed considerably better than their non-DS counterparts for both precision and recall – verifying the positive effects made by the incorporation of domain feature similarity. Also notice CBC showed a better performance over other standard algorithms (graph and K-means), including their DS-extended versions. This verifies that CBC is indeed an effective tool for word sense discovery, and that using CBC as the base algorithm for extension in our work was a good choice.

One thing to note about the results is that precision is overall rather low. The main reason would be because we only examined a subset of the WordNet senses. When an assigned sense indeed corresponded to a WordNet sense but if it is not mapped to a noun synset in WordNet, the assigned sense ended up having no matches, thus ultimately was considered incorrect.

To investigate further, we also inspected the performance of the algorithms with different values of the threshold θ, since the determination of a correct assignment is dependent on this value. As θ is raised, the precision and recall (thus F-measure)

[10]In WordNet, adjectives are not organized hierarchically; instead they are simply categorized into two large groups (descriptive and relational adjectives).

Table 2: Precision, Recall and F-measure for the English Dataset (when $\theta = 0.25$)

	Precision	Recall	F-measure
DSCBC	0.397	0.397	0.397
CBC	0.304	0.264	0.283
DSGr	0.214	0.251	0.231
Gr	0.192	0.202	0.197
DSKmeans	0.254	0.213	0.232
Kmeans	0.169	0.162	0.166

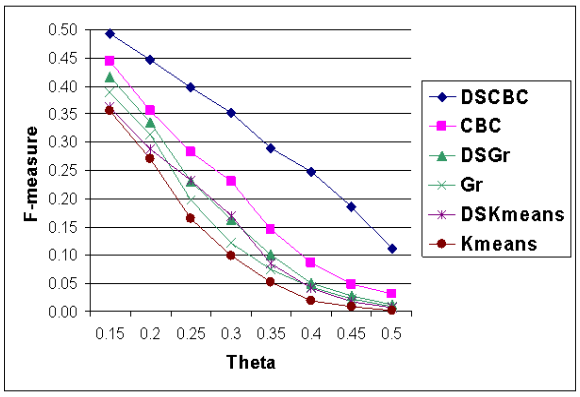

Figure 5: F-measure for varying θ for the English Dataset

will decrease, because only the higher correspondence values are considered as correct. Figure 5 shows the F-measure for varying θ for all algorithms. DSCBC has the highest F-measure consistently for all θ by a large margin. The DS algorithms are showing a better performance over their non-DS counterparts here as well for all θ, although the differences (between DSGr and Gr, and DSKmeans and Kmeans) are not as dramatic as the difference between DSCBC and CBC.

6.2.2 The Japanese Dataset

For the Japanese dataset, we were unable to do an automatic evaluation, because word senses in *Daijirin* are only enumerated with no particular organization. Instead, we randomly selected 60 words from the dataset, and manually compared the committees assigned by the algorithms against the senses encoded in *Daijirin*. The average number of senses of the selected 60 words was 4.05. For the algorithms, we focused on three algorithms: DSCBC and CBC, plus Gr which showed the tightest domain feature similarity among the non-CBC algorithms according to our inspection.

For each algorithm, we manually matched up the assigned committees to the word senses which we determined are correct. To judge the correctness, we looked at the top 3 features in a committee and determined correct if at least one of them corresponded to the sense based on our linguistic intuitions. Then we computed the precision as the ratio of the correctly assigned committees over all assignments (thus each sense assignment was taken individually). Similarly we computed the recall as the ratio of the correctly assigned committees over all word senses listed in the lexicon. Table 3 shows the results. As you can see, DSCBC produced the best result for this dataset as well.

Table 3: Precision, Recall and F-measure for the Japanese Dataset

	Precision	Recall	F-measure
DSCBC	0.571	0.576	0.574
CBC	0.532	0.440	0.482
Gr	0.590	0.449	0.511

Table 4: Average Ranks by Algorithm for the Japanese Dataset

	Recalled by All 3 Algo.	Recalled by At least 2 Algo.
DSCBC	1.302	1.190
CBC	2.206	1.638
Gr	2.492	1.802

To further investigate the quality of the derived committees, we also ranked the committees assigned to the same word sense by the three algorithms. Since not all word senses were discovered by all three algorithms, we looked at the ones that were recalled by all three and at least two algorithms. Table 4 shows the average ranks by the algorithms. Ranks were given in the descending order (from 1 to 3), so a lower value means a better correspondence to the lexicon senses. For both cases, DSCBC had the lowest rank, suggesting that DSCBC's committees were the closest to the gold standards. On the other hand, the quality of the committees derived by Gr is not as good as DSCBC or CBC – although Gr showed the highest precision, its ranks were the lowest.

7 Conclusions and Future Work

In this paper, we presented a new algorithm for discovering word senses for polysemous words, and showed the improved performance over other clustering algorithms. By incorporating the feature domain similarity, the algorithm produces clusters which are more "tight" with respect to the domains, that is, more monosemous clusters.

For future work, an immediate task is to do a more thorough evaluation, manual as well as automatic, including an investigation on exactly which element(s) in the algorithm contributed the most to the improved performance. Our preliminary inspection indicates the feature domain similarity was the most significant factor. Finally, we would like to investigate the usefulness of the derived word senses in practical applications such as word sense disambiguation and question-answering.

References

G. Boleda, T. Badia, and E. Batlle. 2004. Acquisition of Semantic Classes for Adjectives from Distributional Evidence. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING-04)*.

T. Brants and A. Franz. 2006. *Web 1T 5-gram Version 1*. Linguistic Data Consortium. Catalog No. LDC2006T13, ISBN 1-58563-397-6.

V. Hatzivassiloglou and K. McKeown. 1993. Towards the Automatic Identification of Adjectival Scales: Clustering Adjectives According to Meaning. In *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics (ACL-93)*.

V. Hatzivassiloglou and J. Wiebe. 2000. Effects of Adjective Orientation and Gradability on Sentence Subjectivity. In *Proceedings of the 18th International Conference on Computational Linguistics (COLING-2000)*.

D. Hindle. 1990. Noun Classification from Predicate-argument Structures. In *Proceedings of the 28th Annual Meeting of the Association for Computational Linguistics (ACL-90)*.

J. Justeson and S. Katz. 1995. Principled Disambiguation: Discriminating Adjective Senses with Modified Nouns. *Computational Linguistics*, 21(1):1 – 27.

K. Kanzaki, E. Yamamoto, Q. Ma, and H. Isahara. 2004. Construction of an Objective Hierarchy of Abstract Concepts via Directional Similarity. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING-04)*.

D. Kawahara and S. Kurohashi. 2006. A Fully-Lexicalized Probabilistic Model for Japanese Syntactic and Case Structure Analysis. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT/NAACL-2006)*.

M. Lapata. 2001. A Corpus-based Account of Regular Polysemy: The Case of Context-sensitive Adjectives. In *Proceedings of the 2nd Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

D. Lin and P. Pantel. 2001. Induction of Semantic Classes from Natural Language Text. In *Proceedings of the 7th ACM Conference on Knowledge Discovery and Data Mining (KDD-01)*.

G. Miller. 1990. WordNet: An Online Lexical Database. *International Journal of Lexicography*, 3(4).

P. Pantel and D. Lin. 2002. Discovering Word Senses from Text. In *Proceedings of the 8th ACM Conference on Knowledge Discovery and Data Mining (KDD-02)*.

J. Pustejovsky. 1995. *The Generative Lexicon*. The MIT Press.

G. Salton and M. McGill. 1983. *Introduction to Modern Information Retrieval*. McGraw Hill.

N. Seco, T. Veale, and J. Hayes. 2004. An Intrinsic Information Content Metric for Semantic Similarity in WordNet. In *Proceedings of the 16th European Conference on Artificial Intelligence (ECAI-04)*.

T. Takahashi. 1975. Various Phases Related to the Part-whole Relationship Investigated in Sentences. *Studies in the Japanese Language*, 103:1–16. In Japanese.

M. Tanaka and Y. Hirai. 2001. Clustering Adjectives Based on Latent Semantic Analysis. Technical Report 2001-92, The Institute of Electronics, Information and Communication Engineers. In Japanese.

S. Walde. 2006. Experiments on the Automatic Induction of German Semantic Verb Classes. *Computational Linguistics*, 32(2):159 – 194.

J. Wiebe. 2000. Learning Subjective Adjectives from Corpora. In *Proceedings of the 17th National Conference on Artificial Intelligence (AAAI-2000)*.

S. Zhong and J. Ghosh. 2003. A Unified Framework for Model-based Clustering. *Journal of Machine Learning Research*, 4:1001–1037.