# Clustering Paraphrases by Word Sense

**Anne Cocos and Chris Callison-Burch**
Computer and Information Science Department, University of Pennsylvania

## Abstract

Automatically generated databases of English paraphrases have the drawback that they return a single list of paraphrases for an input word or phrase. This means that all senses of polysemous words are grouped together, unlike WordNet which partitions different senses into separate synsets. We present a new method for clustering paraphrases by word sense, and apply it to the Paraphrase Database (PPDB). We investigate the performance of hierarchical and spectral clustering algorithms, and systematically explore different ways of defining the similarity matrix that they use as input. Our method produces sense clusters that are qualitatively and quantitatively good, and that represent a substantial improvement to the PPDB resource.

## 1 Introduction

Many natural language processing tasks rely on the ability to identify words and phrases with equivalent meaning but different wording. These alternative ways of expressing the same information are called paraphrases. Several research efforts have produced automatically generated databases of English paraphrases, including DIRT (Lin and Pantel, 2001), the Microsoft Research Paraphrase Phrase Tables (Dolan et al., 2004), and the Paraphrase Database (Ganitkevitch et al., 2013; Pavlick et al., 2015a). A primary benefit of these automatically generated resources is their enormous scale, which provides superior coverage compared to manually compiled resources like WordNet (Miller, 1995). But automatically generated paraphrase resources currently have the drawback that they group all senses of polysemous words together, and do not partition paraphrases into groups like WordNet does with its synsets. Thus a search for paraphrases of the noun *bug* would yield a single list of paraphrases that includes *insect, glitch, beetle, error, microbe, wire, cockroach, malfunction, microphone, mosquito, virus, tracker, pest, informer, snitch, parasite, bacterium, fault, mistake, failure* and many others. The goal of this work is to group these paraphrases into clusters that denote the distinct senses of the input word or phrase, as shown in Figure 1.

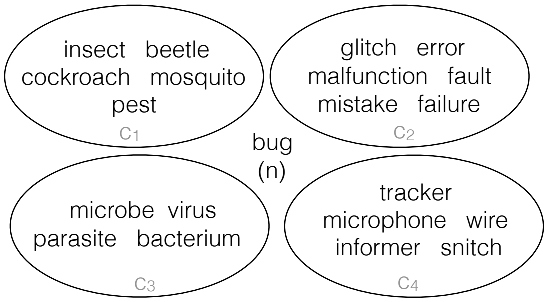


Figure 1: Our goal is to partition paraphrases of an input word like *bug* into clusters representing its distinct senses.

We develop a method for clustering the paraphrases from the Paraphrase Database (PPDB). PPDB contains over 100 million paraphrases generated using the bilingual pivoting method (Bannard and Callison-Burch, 2005), which posits that two English words are potential paraphrases of each other if they share one or more foreign translations. We apply two clustering algorithms, Hierarchical Graph Factorization Clustering (Yu et al., 2005; Sun and Korhonen, 2011) and Self-Tuning Spectral Clustering (Ng et al., 2001; Zelnik-Manor and Perona, 2004), and systematically explore different ways of defining the similarity matrix that they use as input. We exploit a variety of features from PPDB to cluster its paraphrases by sense, including its im-

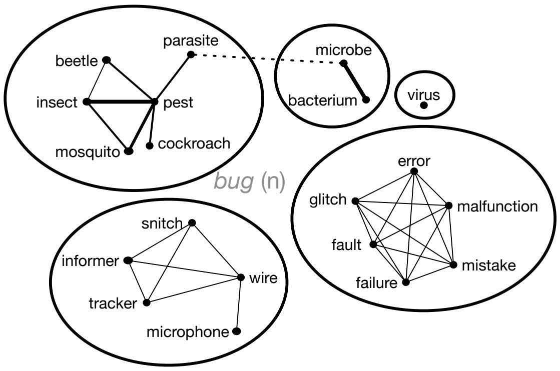


Figure 2: SEMCLUST connects all paraphrases that share foreign alignments, and cuts edges below a dynamically-tuned cutoff weight (dotted lines). The resulting connected components are its clusters.

plicit graph structure, aligned foreign words, paraphrase scores, predicted entailment relations, and monolingual distributional similarity scores.

Our goal is to determine which algorithm and features are the most effective for clustering paraphrases by sense. We address three research questions:

- Which similarity metric is best for sense clustering? We systematically compare different ways of defining matrices that specify the similarity between pairs of paraphrases.

- Are better clusters produced by comparing second-order paraphrases? We use PPDB's graph structure to decide whether *mosquito* and *pest* belong to the same sense cluster by comparing lists of paraphrases for the two words.

- Can entailment relations inform sense clustering? We exploit knowledge like *beetle* is-an *insect*, and that there is no entailment between *malfunction* and *microbe*.

Our method produces sense clusters that are qualitatively and quantitatively good, and that represent a substantial improvement to the PPDB resource.

## 2 Related Work

The paraphrases in PPDB are already partitioned by syntactic type, following the work of Callison-Burch (2008). He showed that applying syntactic constraints during paraphrase extraction via the pivot method improves paraphrase quality. This means that paraphrases of the noun *bug* are separated from paraphrases of the verb *bug*, which consist of verbs like *bother, trouble, annoy, disturb*, and others. However, organizing paraphrases this way still leaves the issue of mixed senses within a single part of speech. This lack of sense distinction makes it difficult to decide when a paraphrase in PPDB would be an appropriate substitute for a word in a given sentence. Some researchers resort to crowdsourcing to determine when a PPDB substitution is valid (Pavlick et al., 2015c).

Our sense clustering work is closely related to the task of word sense induction (WSI), which aims to discover all senses of a target word from large corpora. One family of common approaches to WSI aims to discover the senses of a word by clustering the monolingual contexts in which it appears (Navigli, 2009). Another uncovers a word's senses by clustering its foreign alignments from parallel corpora (Diab, 2003). A more recent family of approaches to WSI represents a word as a feature vector of its substitutable words, i.e. paraphrases (Melamud et al., 2015; Yatbaz et al., 2012). In this paper we take inspiration from each of these families of approaches, and we explore them when measuring word similarity in sense clustering.

The work most closely related to ours is that of Apidianaki et al. (2014), who used a simple graph-based approach to cluster pivot paraphrases on the basis of contextual similarity and shared foreign alignments. Their method represents paraphrases as nodes in a graph and connects each pair of words sharing one or more foreign alignments with an edge weighted by contextual similarity. Concretely, for paraphrase set $P$, it constructs a graph $G = (V, E)$ where vertices $V = \{p_i \in P\}$ are words in the paraphrase set and edges connect words that share foreign word alignments in a bilingual parallel corpus. The edges of the graph are weighted based on their contextual similarity (computed over a monolingual corpus). In order to partition the graph into clusters, edges in the initial graph $G$ with contextual similarity below a threshold $T'$ are deleted. The connected components in the resulting graph $G'$ are taken as the sense clusters. The threshold is dynamically tuned using an iterative procedure (Apidianaki and He, 2010).

As evaluated against reference clusters derived from SEMEVAL 2007 Lexical Substitution gold data (McCarthy and Navigli, 2007), their method, which we call SEMCLUST, outperformed simple most-frequent-sense, one-sense-per-paraphrase, and random baselines. Apidianaki et al. (2014)'s work corroborated the existence of sense distinctions in the paraphrase sets, and highlighted the need for further work to organize them by sense. In this paper, we improve on their method using more advanced clustering algorithms, and by systematically exploring a wider range of similarity measures.

## 3 Graph Clustering Algorithms

To partition paraphrases by sense, we use two advanced graph clustering methods rather than using Apidianaki et al. (2014)'s edge deletion approach. Both of them allow us to experiment with a variety of similarity metrics.

### 3.1 Hierarchical Graph Factorization Clustering

The Hierarchical Graph Factorization Clustering (HGFC) method was developed by Yu et al. (2006) to probabilistically partition data into hierarchical clusters that gradually merge finer-grained clusters into coarser ones. Sun and Korhonen (2011) applied HGFC to the task of clustering verbs into Levin (1993)-style classes. Sun and Korhonen extended the basic HGFC algorithm to automatically discover the latent tree structure in their clustering solution and incorporate prior knowledge about semantic relationships between words. They showed that HGFC far outperformed agglomerative clustering methods on their verb data set. We adopt Sun and Korhonen's implementation of HGFC for our experiments.

HGFC takes as input a nonnegative, symmetric adjacency matrix $W = \{w_{ij}\}$ where rows and columns represent paraphrases $p_i \in P$, and entries $w_{ij}$ denote the similarity between paraphrases $sim_D(p_i, p_j)$. The algorithm works by factorizing $W$ into a bipartite graph, where the nodes on one side represent paraphrases, and nodes on the other represent senses. The output of HGFC is a set of clusterings of increasingly coarse granularity, which we can also represent with a tree structure. The algorithm automatically determines the number of clusters at each level. For our task, this has the benefit that a user can choose the cluster granularity most appropriate for the downstream task (as illustrated in Figure 5). Another benefit of HGFC is that it probabilistically assigns each paraphrase to a cluster at each level of the hierarchy. If some $p_i$ has high probability in multiple clusters, we can assign $p_i$ to all of them (Figure 3c).

### 3.2 Spectral Clustering

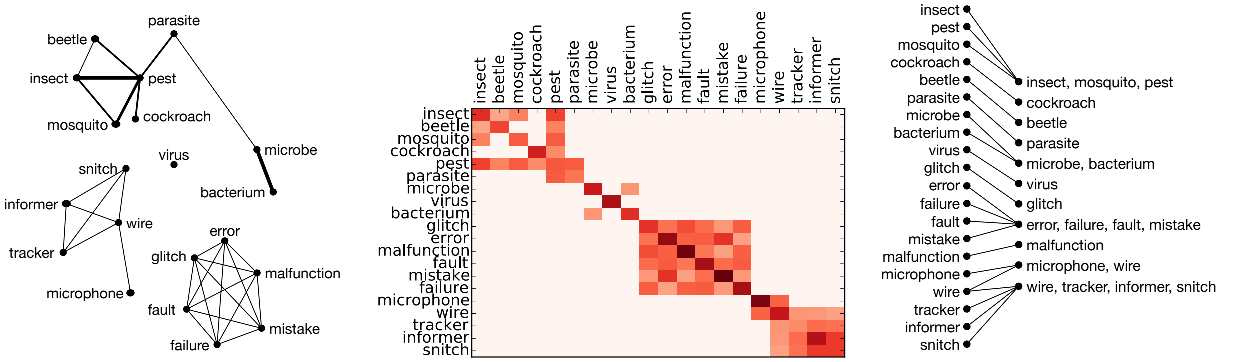
The second clustering algorithm that we use is Self-Tuning Spectral Clustering (Zelnik-Manor and Perona, 2004). Like HGFC, spectral clustering takes an adjacency matrix $W$ as input, but the similarities end there. Whereas HGFC produces a hierarchical clustering, spectral clustering produces a flat clustering with $k$ clusters, with $k$ specified at runtime. The Zelnik-Manor and Perona (2004)'s self-tuning method is based on Ng et al. (2001)'s spectral clustering algorithm, which computes a normalized Laplacian matrix $L$ from the input $W$, and executes K-means on the largest $k$ eigenvectors of $L$. Intuitively, the largest $k$ eigenvectors of $L$ should align with the $k$ senses in our paraphrase set.

## 4 Similarity Measures

Each of our clustering algorithms take as input an adjacency matrix $W$ where the entries $w_{ij}$ correspond to some measure of similarity between words $i$ and $j$. For the paraphrases in Figure 1, $W$ is a 20x20 matrix that specifies the similarity of every pair of paraphrases like *microbe* and *bacterium* or *microbe* and *malfunction*. We systematically investigated four types of similarity scores to populate $W$.

### 4.1 Paraphrase Scores

Bannard and Callison-Burch (2005) defined a *paraphrase probability* in order to quantify the goodness of a pair of paraphrases, based on the underlying translation probabilities used by the bilingual pivoting method. More recently, (Pavlick et al., 2015a) used supervised logistic regression to combine a variety of scores so that they align with human judgements of paraphrase quality. PPDB 2.0 provides this score for each pair of words in the database. The PPDB 2.0 score is a nonnegative real number that



(a) Undirected graph for query word *bug*. Wider lines signify stronger similarity.

(b) The corresponding adjacency matrix $W$. Darker cells signify stronger similarity.

(c) The bipartite graph induced by the first iteration of HGFC. Note *wire* is assigned to two clusters.

Figure 3: The graph, corresponding adjacency matrix $W$, and bipartite graph created by the first iteration of HGFC for query word *bug (n)*

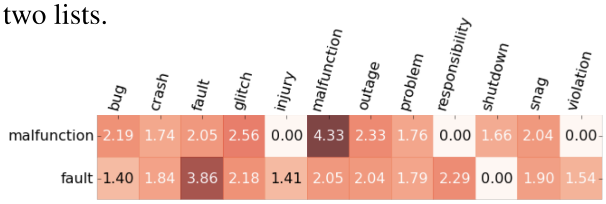
can be used directly as a similarity measure:

$$w_{ij} = \begin{cases} PPDB_{2.0}Score(i,j) & (i,j) \in \text{PPDB} \\ 0 & \text{otherwise} \end{cases}$$

PPDB 2.0 does not provide a score for a word with itself, so we set $PPDB_{2.0}Score(i,i)$ to be the maximum $PPDB_{2.0}Score(i,j)$ such that $i$ and $j$ have the same stem.

### 4.2 Second-Order Paraphrase Scores

Work by Rapp (2003) and Melamud et al. (2015) showed that comparing words on the basis of their *shared* paraphrases is effective for WSI. We define two novel similarity metrics that calculate the similarity of words $i$ and $j$ by comparing their second-order paraphrases. Instead of comparing *microbe* and *bacterium* directly with their PPDB 2.0 score, we look up all of the paraphrases of *microbe* and all of the paraphrases of *bacterium*, and compare those two lists.



| | bug | crash | fault | glitch | injury | malfunction | outage | problem | responsibility | shutdown | snag | violation |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| malfunction | 2.19 | 1.74 | 2.05 | 2.56 | 0.00 | 4.33 | 2.33 | 1.76 | 0.00 | 1.66 | 2.04 | 0.00 |
| fault | 1.40 | 1.84 | 3.86 | 2.18 | 1.41 | 2.05 | 2.04 | 1.79 | 2.29 | 0.00 | 1.90 | 1.54 |

Figure 4: Comparing second-order paraphrases for *malfunction* and *fault* based on *word-paraphrase* vectors. The value of vector element $v_{ij}$ is $PPDB_{2.0}Score(i,j)$.

Specifically, we form notional *word-paraphrase* feature vectors $v_i^p$ and $v_j^p$ where the features correspond to words with which each is connected in PPDB, and the value of the $k^{th}$ element of $v_i^p$ equals $PPDB_{2.0}Score(i,k)$. We can then calculate the cosine similarity or Jensen-Shannon divergence between vectors:

$$sim_{PPDB.cos}(i,j) = cos(v_i^p, v_j^p)$$

$$sim_{PPDB.js}(i,j) = 1 - JS(v_i^p, v_j^p)$$

where $JS(v_i^p, v_j^p)$ is calculated assuming that the paraphrase probability distribution for word $i$ is given by its normalized *word-paraphrase* vector $v_i^p$.

### 4.3 Similarity of Foreign Word Alignments

When an English word is aligned to several foreign words, sometimes those different translations indicate a different word sense (Yao et al., 2012). Using this intuition, Gale et al. (1992) trained an English WSD system on a bilingual corpus, using the different French translations as labels for the English word senses. For instance, given the English word *duty*, the French translation *droit* was a proxy for its *tax* sense and *devoir* for its *obligation* sense.

PPDB is derived from bilingual coropra. We recover the aligned foreign words and their associated translation probabilities that underly each PPDB entry. For each English word in our dataset, we get each foreign word that it aligns to in the Spanish and

Chinese bilingual parallel corpora used by Ganitkevitch and Callison-Burch (2014). We use this to define a novel foreign word alignment similarity metric, $sim_{TRANS}(i, j)$ for two English paraphrases $i$ and $j$. This is calculated as the cosine similarity of the *word-alignment* vectors $v_i^a$ and $v_j^a$ where each feature in $v^a$ is a foreign word to which $i$ or $j$ aligns, and the value of entry $v_{if}^a$ is the translation probability $p(f|i)$.

$$sim_{TRANS}(i, j) = cos(v_i^a, v_j^a)$$

### 4.4 Monolingual Distributional Similarity

Lastly, we populate the adjacency with a distributional similarity measure based on WORD2VEC (Mikolov et al., 2013). Each paraphrase $i$ in our data set is represented as a 300-dimensional WORD2VEC embedding $v_i^w$ trained on part of the Google News dataset. Phrasal paraphrases that did not have an entry in the WORD2VEC dataset are represented as the mean of their individual word vectors. We use the cosine similarity between WORD2VEC embeddings as our measure of distributional similarity.

$$sim_{DISTRIB}(i, j) = cos(v_i^w, v_j^w)$$

## 5 Determining the Number of Senses

The optimal number of clusters for a set of paraphrases will vary depending on how many senses there ought to be for an input word like *bug*. It is generally recognized that optimal sense granularity depends on the application (Palmer et al., 2001). WordNet has notoriously fine-grained senses, whereas most word sense disambiguation systems achieve better performance when using coarse-grained sense inventories (Navigli, 2009). Depending on the task, the sense clustering for query word *coach* in Figure 5b with $k = 5$ clusters may be preferable to the alternative with $k = 3$ clusters. An ideal algorithm for our task would enable clustering at varying levels of granularity to support different downstream NLP applications.

Both of our clustering algorithms can produce sense clusters at varying granularities. For HGFC this requires choosing which level of the resulting tree structure to take as a clustering solution, and for spectral clustering we must specify the number of clusters prior to execution.[1] To determine the optimal number of clusters, we use the mean Silhouette Coefficient (Rousseeuw, 1987) which balances optimal inter-cluster tightness and intra-cluster distance. The Silhouette Coefficient is calculated for each paraphrase $p_i$ as

$$s(p_i) = \frac{b(p_i) - a(p_i)}{max\{a(p_i), b(p_i)\}}$$

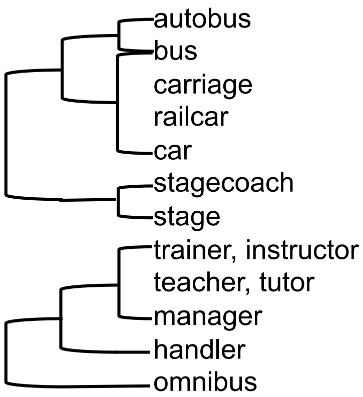
where $a(p_i)$ is $p_i$'s average intra-cluster distance (average distance from $p_i$ to each other $p_j$ in the same cluster), and $b(p_i)$ is $p_i$'s lowest average inter-cluster distance (distance from $p_i$ to the nearest external cluster centroid). For each clustering algorithm, we choose as the 'solution' the clustering which produces the highest mean Silhouette Coefficient. The Silhouette Coefficient calculation takes as input a matrix of pairwise distances, so we simply use $1 - W$ where the adjacency matrix $W$ is calculated using one of the similarity methods we defined.

## 6 Incorporating Entailment Relations

Pavlick et al. (2015b) added a set of automatically predicted semantic entailment relations for each entry in PPDB 2.0. The entailment types that they include are *Equivalent, Forward Entailment, Reverse Entailment, Exclusive*, and *Independent*. While a negative entailment relationship (*Exclusive* or *Independent*) does not preclude words from belonging to the same sense of some query word, a positive entailment relationship (*Equivalent, Forward/Reverse Entailment*) does give a strong indication that the words belong to the same sense.

We seek a straightforward way to determine whether entailment relations provide information that is useful to the final clustering algorithm. Both of our algorithms take an adjacency matrix $W$ as input, so we add entailment information by simply

[1] For spectral clustering there has been significant study into methods for automatically determining the optimal number of clusters, including analysis of eigenvalues of the graph Laplacian, and finding the rotation of the Laplacian that brings it closest to block-diagonal (Zelnik-Manor and Perona, 2004). We experimented with these and other cluster analysis methods such as the Dunn Index (Dunn, 1973) in our work, but found that using the simple Silhouette Coefficient produced clusterings that were competitive with the more intensive methods, in far less time.



(a) HGFC clustering result

| | |
|---|---|
| | $c_1$: trainer, tutor, instructor, teacher |
| | $c_2$: stagecoach, stage |
| *k=5* | $c_3$: omnibus, bus, autobus |
| | $c_4$: car, carriage, railcar |
| | $c_5$: manager, handler |
| | $c_1$: trainer, tutor, instructor, teacher, manager, handler |
| *k=3* | $c_2$: stagecoach, stage |
| | $c_3$: omnibus, bus, autobus, car, carriage, railcar |

(b) Spectral clustering results

Figure 5: HGFC and Spectral Clustering results for *coach (n)*. Our silhouette optimization sets $k = 3$.

multiplying each pairwise entry by its entailment probability. Specifically, we set

$$w_{ij} = \begin{cases} (1 - p_{ind}(i,j))sim_D(i,j) & (i,j) \in \text{PPDB} \\ 0 & \text{otherwise} \end{cases}$$

where $p_{ind}(i,j)$ gives the PPDB 2.0 probability that there is an $Independent$ entailment relationship between words $i$ and $j$. Intuitively, this should increase the similarity of words that are very likely to be entailing like *fault* and *failure*, and decrease the similarity of non-entailing words like *cockroach* and *microphone*.

## 7 Experimental Setup

We follow the experimental setup of Apidianaki et al. (2014). We focus our evaluation on a set of query words drawn from the LexSub test data (McCarthy and Navigli, 2007), plus 16 additional handpicked polysemous words.

### 7.1 Gold Standard Clusters

One challenge in creating our clustering methodology is that there is no reliable PPDB-sized standard against which to assess our results. WordNet synsets provide a well-vetted basis for comparison, but only allow us to evaluate our method on the 38% of our PPDB dataset that overlaps it. We therefore evaluate performance on two test sets.

**WordNet+** Our first test set is designed to assess how well our solution clusters align with WordNet synsets. We chose 185 polysemous words from the SEMEVAL 2007 dataset and an additional 16 handpicked polysemous words. For each we formed a paraphrase set that was the intersection of their PPDB 2.0 XXXL paraphrases with their WordNet synsets, and their immediate hyponyms and hypernyms. Each reference cluster consisted of a WordNet synset, plus the hypernyms and hyponyms of words in that synset. On average there are 7.2 reference clusters per paraphrase set.

**CrowdClusters** Because the coverage of WordNet is small compared to PPDB, and because WordNet synsets are very fine-grained, we wanted to create a dataset that would test the performance of our clustering algorithm against large, noisy paraphrase sets and coarse clusters. For this purpose we randomly selected 80 query words from the SEMEVAL 2007 dataset and created paraphrase sets from their unfiltered PPDB2.0 XXL entries. We then iteratively organized each paraphrase set into reference senses with the help of crowd workers on Amazon Mechanical Turk. On average there are 4.0 reference clusters per paraphrase set. A full description of our method is included in the supplemental materials.

### 7.2 Evaluation Metrics

We evaluate our method using two standard metrics: the paired F-Score and V-Measure. Both were used in the 2010 SemEval Word Sense Induction Task (Manandhar et al., 2010) and by Apidianaki et al. (2014). We give our results in terms of weighted average performance on these metrics, where the score for each individual paraphrase set is weighted by the number of reference clusters for that query word.

**Paired F-Score** frames the clustering problem as a classification task (Manandhar et al., 2010). It generates the set of all word pairs belonging to the same reference cluster, $F(S)$, and the set of all word pairs belonging to the same automatically-generated cluster, $F(K)$. Precision, recall, and F-score can then be calculated in the usual way, i.e. $P = \frac{F(K)\cap F(S)}{F(K)}$, $R = \frac{F(K)\cap F(S)}{F(S)}$, and $F = \frac{2\cdot P\cdot R}{P+R}$.

**V-Measure** assesses the quality of a clustering solution against reference clusters in terms of clustering homogeneity and completeness (Rosenberg and Hirschberg, 2007). Homogeneity describes the extent to which each cluster is composed of paraphrases belonging to the same reference cluster, and completeness refers to the extent to which points in a reference cluster are assigned to a single cluster. Both are defined in terms of conditional entropy. V-Measure is the harmonic mean of homogeneity $h$ and completeness $c$; V-Measure $= \frac{2\cdot h\cdot c}{h+c}$.

### 7.3 Baselines

We evaluate the performance of HGFC on each dataset against the following baselines:

**Most Frequent Sense (MFS)** assigns all paraphrases $p_i \in P$ to a single cluster. By definition, the completeness of the MFS clustering is 1.

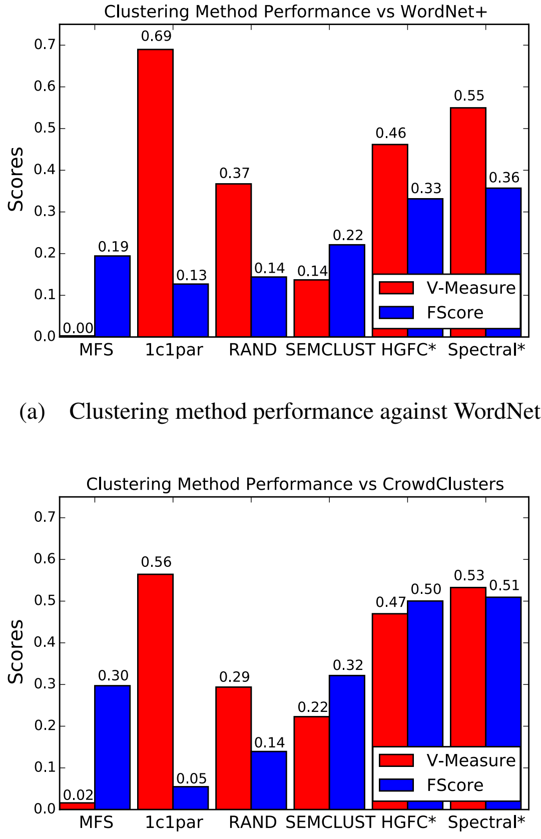
**One Cluster per Paraphrase (1C1PAR)** assigns each paraphrase $p_i \in P$ to its own cluster. By definition, the homogeneity of 1C1PAR clustering is 1.

**Random (RAND)** For each query term's paraphrase set, we generate five random clusterings of $k = 5$ clusters. We then take F-Score and V-Measure as the average of each metric calculated over the five random clusterings.

**SEMCLUST** We implement the SEMCLUST algorithm (Apidianaki et al., 2014) as a state-of-the-art baseline. Since PPDB contains only pairs of words that share a foreign word alignment, in our implementation we connect paraphrase words with an edge if the pair appears in PPDB. We adopt the WORD2VEC distributional similarity score $sim_{DISTRIB}$ for our edge weights.

## 8 Experimental Results

Figure 6 shows the performance of the two advanced clustering algorithms against the baselines. Our best configurations[2] for HGFC and Spectral significantly outperformed all baselines except 1C1PAR V-Measure, which is biased toward solutions with many small clusters (Manandhar et al., 2010). The dominance of 1C1PAR V-Measure is greater for the WordNet+ dataset which has smaller reference clusters than CrowdClusters. Qualitatively, we find that methods that strike a balance between high F-Score and high V-Measure tend to produce the 'best' clusters by human judgement.



(a) Clustering method performance against WordNet+

(b) Clustering method performance against CrowdClusters

Figure 6: Hierarchical Graph Factorization Clustering and Spectral Clustering both significantly outperform all baselines except 1C1PAR V-Measure.

On our dataset, the state-of-the-art SEMCLUST baseline tended to lump many senses of the query word together, and produced scores lower than in the original work. We attribute this to the fact that

[2] Our top-scoring Spectral method, Spectral*, uses entailments, $PPDB_{2.0}Score$ similarities, and $sim_{DISTRIB}$ to choose $k$. Our best HGFC method, HGFC*, uses entailments, $sim_{DISTRIB}$ similarities, and $PPDB_{2.0}Score$ to choose $k$.

| Method | F-Score | V-Measure | Avg # Clusters |
|---|---|---|---|
| $PPDB_{2.0}Score$ | 0.410 | 0.437 | 5.960 |
| $sim_{DISTRIB}$ | 0.376 | 0.440 | 5.707 |
| $sim_{PPDB.cos}$ | 0.389 | 0.428 | 7.204 |
| $sim_{PPDB.JS}$ | 0.385 | 0.425 | 7.143 |
| $sim_{TRANS}$ | 0.358 | 0.375 | 6.247 |
| SEMCLUST | 0.271 | 0.180 | 2.279 |
| Reference | 1.0 | 1.0 | 5.611 |

Table 1: Average performance and number of clusters produced by our different similarity methods.

the original work extracted paraphrases from EuroParl, which is much smaller than PPDB, and thus created adjacency matrices $W$ which were sparser than those produced by our method. Directly applied, SEMCLUST works well on small data sets, but does not scale well to the larger, noisier PPDB data. More advanced graph-based clustering methods produce better sense clusters for PPDB.

The first question we sought to address with this work was which similarity metric is the best for sense clustering. Table 1 reports the average F-Score and V-Measure across 40 test configurations for each similarity calculation method.[3] On average across test sets and clustering algorithms, the paraphrase similarity score ($PPDB_{2.0}Score$) performs better than monolingual distributional similarity ($sim_{DISTRIB}$) in terms of F-Score, but the results are reversed for V-Measure. This is also shown in the best HGFC and Spectral configurations, where the two similarity scores are swapped between them.

Next, we investigated whether comparing second-order paraphrases would produce better clusters than simply using $PPDB_{2.0}Score$ directly. Table also 1 compares the two methods that we had for computing the similarity of second order paraphrases – cosine similarity ($sim_{PPDB.cos}$) and Jensen-Shannon divergence ($sim_{PPDB.JS}$). On average using the direct paraphrase score gives the best balance of high V-Measure and F-score across test sets and clustering algorithms. It produces coarser clusters than the second-order PPDB similarity methods.

Finally, we investigated whether incorporating automatically predicted entailment relations would improve cluster quality, and we found that it did. All other things being equal, adding entailment information increases F-Score by .013 and V-Measure by .020 on average (Figure 7). Adding entailment information had the greatest improvement to HGFC methods with $sim_{DISTRIB}$ similarities, where it improved F-Score by an average of .03 and V-Measure by an average of .05.

## 9 Discussion and Future Work

We have presented a novel method for clustering paraphrases in PPDB by sense. When evaluated against WordNet synsets, the sense clusters produced by the Spectral Clustering algorithm give a 64% relative improvement in F-Score over the closest baseline, and the HGFC algorithm gives a 50% improvement in F-Score. We systematically analyzed a variety of similarity metrics as input to HGFC and Spectral Clustering, and showed that incorporating predicted entailment relations from PPDB boosts the performance of sense clustering.

Our sense clustering provides a significant improvement to the PPDB resource that may improve its applicability to downstream NLP tasks. One possible application of sense-clustered PPDB entries is the lexical substitution task, which seeks to identify appropriate word substitutions. Given a target word in context, it would be reasonable to suggest substitutes from the target word's PPDB sense cluster most closely related to the target context. There are many possible ways to choose the best cluster for a given context, ranging from simply choosing the cluster whose members have highest average pointwise mutual information with the context, to a more complex approach based on training cluster representations using a pseudo-word approach as in Melamud et al. (2015). We leave this application for future work.

## 10 Software and Data Release

Upon publication we will release paraphrase clusters for all PPDB entries, clustering code, and an interface for crowdsourcing paraphrase clusters using Amazon Mechanical Turk.

## 11 Supplementary Material

Our Supplementary Material provides additional detail on our similarity metric calculation, clustering

[3] Our Supplementary Materials file provides the full set of results for all 200 configurations that we tested.

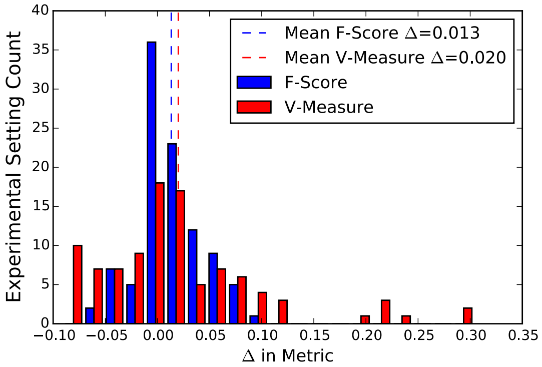


Figure 7: Histogram of metric change by adding entailment information across all experiments.

algorithm implementation, and CrowdCluster reference cluster data development. We also provide full evaluation results across the entire range of our experiments, a selection of sense clusters output by our methods, and example content of our WordNet+ and CrowdCluster paraphrase sets.

## Acknowledgments

This research was supported by the Allen Institute for Artificial Intelligence (AI2), the Human Language Technology Center of Excellence, and by gifts from the Alfred P. Sloan Foundation, Google, and Facebook. This material is based in part on research sponsored by the NSF grant under IIS-1249516 and DARPA under number FA8750-13-2-0017 (the DEFT program). The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes. The views and conclusions contained in this publication are those of the authors and should not be interpreted as representing official policies or endorsements of DARPA and the U.S. Government.

We would like to thank Marianna Apidianaki and Alex Harelick for sharing code used in this research, and Ellie Pavlick for her substantive input. We are grateful to our anonymous reviewers for their constructive comments.

## References

Marianna Apidianaki and Yifan He. 2010. An algorithm for cross-lingual sense clustering tested in a MT evaluation setting. In *Proceedings of the 7th International Workshop on Spoken Language Translation (IWSLT-10)*.

Marianna Apidianaki, Emilia Verzeni, and Diana McCarthy. 2014. Semantic clustering of pivot paraphrases. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC 2014)*.

Colin Bannard and Chris Callison-Burch. 2005. Paraphrasing with bilingual parallel corpora. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL 05)*.

Chris Callison-Burch. 2008. Syntactic constraints on paraphrases extracted from parallel corpora. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 196–205, Honolulu, Hawaii, October. Association for Computational Linguistics.

Mona Talat Diab. 2003. *Word sense disambiguation within a multilingual framework*. Ph.D. thesis, University of Maryland.

William Dolan, Chris Quirk, and Chris Brockett. 2004. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *Proceedings of the International Conference of Computational Linguistics (COLING 2004)*.

Joseph C Dunn. 1973. A fuzzy relative of the isodata process and its use in detecting compact well-separated clusters.

William A. Gale, Kenneth W. Church, and David Yarowsky. 1992. Using bilingual materials to develop word sense disambiguation methods. In *Proceedings of the Fourth International Conference on Theoretical and Methodological Issues in Machine Translation*.

Juri Ganitkevitch and Chris Callison-Burch. 2014. The multilingual paraphrase database. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014), Reykjavik, Iceland*, pages 4276–4283.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. PPDB: The paraphrase database. In *Proceedings of NAACL-HLT 2013*.

Beth Levin. 1993. *English verb classes and alternations: A preliminary investigation*. University of Chicago press.

Dekang Lin and Patrick Pantel. 2001. Discovery of inference rules for question answering. *Natural Language Engineering*.

Suresh Manandhar, Ioannis Klapaftis, Dmitriy Dligach, and Sameer Pradhan. 2010. SemEval-2010 Task 14: Word sense induction & disambiguation. In *Proceedings of the Fifth International Workshop on Semantic Evaluations (SemEval-2010)*.

Diana McCarthy and Roberto Navigli. 2007. Semeval-2007 task 10: English lexical substitution task. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*.

Oren Melamud, Ido Dagan, and Jacob Goldberger. 2015. Modeling word meaning in context with substitute vectors. In *Human Language Technologies: The 2015 Annual Conference of the North American Chapter of the ACL*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of NIPS*.

George A Miller. 1995. WordNet: a lexical database for English. *Communications of the ACM*, 38(11):39–41.

Roberto Navigli. 2009. Word sense disambiguation: A survey. *ACM Computing Surveys*.

Andrew Ng, Michael Jordan, and Y. Weiss. 2001. On spectral clustering: Analysis and an algorithm. *Advances in Neural Information Processing Systems*.

Martha Palmer, Hoa Trang Dang, and Christiane Fellbaum. 2001. Making fine-grained and coarse-grained sense distinctions, both manually and automatically. *Natural Language Engineering*.

Ellie Pavlick, Johan Bos, Malvina Nissim, Charley Beller, Benjamin Van Durme, and Chris Callison-Burch. 2015a. PPDB 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (ACL 2015)*.

Ellie Pavlick, Johannes Bos, Malvina Nissim, Charley Beller, and and Chris Callison-Burch Benjamin Van Durme. 2015b. Adding semantics to data-driven paraphrasing. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (ACL 2015)*.

Ellie Pavlick, Travis Wolfe, Pushpendre Rastogi, Chris Callison-Burch, Mark Drezde, and Benjamin Van Durme. 2015c. FrameNet+: Fast paraphrastic tripling of FrameNet. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (ACL 2015)*, Beijing, China, July. Association for Computational Linguistics.

Reinhard Rapp. 2003. Word sense discovery based on sense descriptor dissimilarity. In *Proceedings of the Ninth Machine Translation Summit*, pages 315–322.

Andrew Rosenberg and Julia Hirschberg. 2007. V-measure: A conditional entropy-based external cluster evaluation measure. In *EMNLP-CoNLL*, volume 7, pages 410–420.

Peter J Rousseeuw. 1987. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65.

Lin Sun and Anna Korhonen. 2011. Hierarchical verb clustering using graph factorization. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1023–1033. Association for Computational Linguistics.

Xuchen Yao, Benjamin Van Durme, and Chris Callison-Burch. 2012. Expectations of word sense in parallel corpora. In *The 2012 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 621–625, Montréal, Canada, June. Association for Computational Linguistics.

Mehmet Ali Yatbaz, Enis Sert, and Deniz Yuret. 2012. Learning syntactic categories using paradigmatic representations of word context. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 940–951. Association for Computational Linguistics.

Kai Yu, Shipeng Yu, and Volker Tresp. 2005. Soft clustering on graphs. In *Advances in neural information processing systems*, pages 1553–1560.

Lihi Zelnik-Manor and Pietro Perona. 2004. Self-tuning spectral clustering. In *Advances in neural information processing systems*, pages 1601–1608.