Discriminative Learning of Selectional Preference from Unlabeled Text

Shane Bergsma
Department of Computing Science
University of Alberta
Edmonton, Alberta
Canada, T6G 2E8
bergsma@cs.ualberta.ca

Dekang Lin
Google, Inc.
1600 Amphitheatre Parkway
Mountain View
California, 94301
lindek@google.com

Randy Goebel
Department of Computing Science
University of Alberta
Edmonton, Alberta
Canada, T6G 2E8
goebel@cs.ualberta.ca

Abstract

We present a discriminative method for learning selectional preferences from unlabeled text. Positive examples are taken from observed predicate-argument pairs, while negatives are constructed from unobserved combinations. We train a Support Vector Machine classifier to distinguish the positive from the negative instances. We show how to partition the examples for efficient training with 57 thousand features and 6.5 million training instances. The model outperforms other recent approaches, achieving excellent correlation with human plausibility judgments. Compared to Mutual Information, it identifies 66% more verb-object pairs in unseen text, and resolves 37% more pronouns correctly in a pronoun resolution experiment.

1 Introduction

Selectional preferences (SPs) tell us which arguments are plausible for a particular predicate. For example, Table 2 (Section 4.4) lists plausible and implausible direct objects (arguments) for particular verbs (predicates). SPs can help resolve syntactic, word sense, and reference ambiguity (Clark and Weir, 2002), and so gathering them has received a lot of attention in the NLP community.

One way to determine SPs is from co-occurrences of predicates and arguments in text. Unfortunately, no matter how much text we use, many acceptable pairs will be missing. Bikel (2004) found that only 1.49% of the bilexical dependencies considered by Collins' parser during decoding were observed during training. In our parsed corpus (Section 4.1), for example, we find *eat* with *nachos*, *burritos*, and *tacos*, but not with the equally tasty *quesadillas*, *chimichangas*, or *tostadas*. Rather than solely relying on co-occurrence counts, we would like to use them to generalize to unseen pairs.

In particular, we would like to exploit a number of arbitrary and potentially overlapping properties of predicates and arguments when we assign SPs. We do this by representing these properties as features in a linear classifier, and training the weights using discriminative learning. Positive examples are taken from observed predicate-argument pairs, while pseudo-negatives are constructed from unobserved combinations. We train a Support Vector Machine (SVM) classifier to distinguish the positives from the negatives. We refer to our model's scores as Discriminative Selectional Preference (DSP). By creating training vectors automatically, DSP enjoys all the advantages of supervised learning, but without the need for manual annotation of examples.

We evaluate DSP on the task of assigning verb-object selectional preference. We encode a noun's textual distribution as feature information. The learned feature weights are linguistically interesting, yielding high-quality similar-word lists as latent information. Despite its representational power, DSP scales to real-world data sizes: examples are partitioned by predicate, and a separate SVM is trained for each partition. This allows us to efficiently learn with over 57 thousand features and 6.5 million examples. DSP outperforms recently proposed alternatives in a range of experiments, and better correlates with human plausibility judgments. It also shows strong gains over a Mutual Information-based co-

Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing, pages 59–68,
Honolulu, October 2008.

occurrence model on two tasks: identifying objects of verbs in an unseen corpus and finding pronominal antecedents in coreference data.

2 Related Work

Most approaches to SPs generalize from observed predicate-argument pairs to semantically similar ones by modeling the semantic class of the argument, following Resnik (1996). For example, we might have a class *Mexican Food* and learn that the entire class is suitable for eating. Usually, the classes are from WordNet (Miller et al., 1990), although they can also be inferred from clustering (Rooth et al., 1999). Brockmann and Lapata (2003) compare a number of WordNet-based approaches, including Resnik (1996), Li and Abe (1998), and Clark and Weir (2002), and found that the more sophisticated class-based approaches do not always outperform simple frequency-based models.

Another line of research generalizes using similar words. Suppose we are calculating the probability of a particular noun, n, occurring as the object argument of a given verbal predicate, v. Let $\Pr(n|v)$ be the empirical maximum-likelihood estimate from observed text. Dagan et al. (1999) define the similarity-weighted probability, $\Pr_{\text{SIM}}$, to be:

$$\Pr_{\text{SIM}}(n|v) = \sum_{v' \in \text{SIMS}(v)} \textit{Sim}(v', v) \Pr(n|v') \quad (1)$$

where $\textit{Sim}(v', v)$ returns a real-valued similarity between two verbs v' and v (normalized over all pair similarities in the sum). In contrast, Erk (2007) generalizes by substituting similar *arguments*, while Wang et al. (2005) use the cross-product of similar pairs. One key issue is how to define the set of similar words, $\text{SIMS}(w)$. Erk (2007) compared a number of techniques for creating similar-word sets and found that both the Jaccard coefficient and Lin (1998a)'s information-theoretic metric work best. Similarity-smoothed models are simple to compute, potentially adaptable to new domains, and require no manually-compiled resources such as WordNet.

Selectional Preferences have also been a recent focus of researchers investigating the learning of paraphrases and inference rules (Pantel et al., 2007; Roberto et al., 2007). Inferences such as "[*X* wins *Y*] $\Rightarrow$ [*X* plays *Y*]" are only valid for certain arguments *X* and *Y*. We follow Pantel et al. (2007) in using automatically-extracted semantic classes to help characterize plausible arguments.

Discriminative techniques are widely used in NLP and have been applied to the related tasks of word prediction and language modeling. Even-Zohar and Roth (2000) use a classifier to predict the most likely word to fill a position in a sentence (in their experiments: a verb) from a set of candidates (sets of verbs), by inspecting the context of the target token (e.g., the presence or absence of a particular nearby word in the sentence). This approach can therefore learn which specific arguments occur with a particular predicate. In comparison, our features are second-order: we learn what *kinds* of arguments occur with a predicate by encoding features of the arguments. Recent distributed and latent-variable models also represent words with feature vectors (Bengio et al., 2003; Blitzer et al., 2005). Many of these approaches learn both the feature weights and the feature representation. Vectors must be kept low-dimensional for tractability, while learning and inference on larger scales is impractical. By partitioning our examples by predicate, we can efficiently use high-dimensional, sparse vectors.

Our technique of generating negative examples is similar to the approach of Okanohara and Tsujii (2007). They learn a classifier to disambiguate actual sentences from pseudo-negative examples sampled from an N-gram language model. Smith and Eisner (2005) also automatically generate negative examples. They perturb their input sequence (e.g. the sentence word order) to create a neighborhood of *implicit* negative evidence. We create negatives by substitution rather than perturbation, and use corpus-wide statistics to choose our negative instances.

3 Methodology

3.1 Creating Examples

To learn a discriminative model of selectional preference, we create positive and negative training examples automatically from raw text. To create the positives, we automatically parse a large corpus, and then extract the predicate-argument pairs that have a statistical association in this data. We measure this association using pointwise Mutual Information (MI) (Church and Hanks, 1990). The MI between a

verb predicate, v, and its object argument, n, is:

$$\text{MI}(v,n) = \log \frac{\Pr(v,n)}{\Pr(v)\Pr(n)} = \log \frac{\Pr(n|v)}{\Pr(n)} \quad (2)$$

If MI>0, the probability v and n occur together is greater than if they were independently distributed.

We create sets of positive and negative examples separately for each predicate, v. First, we extract all pairs where MI$(v,n)>\tau$ as positives. For each positive, we create pseudo-negative examples, (v,n'), by pairing v with a new argument, n', that either has MI below the threshold or did not occur with v in the corpus. We require each negative n' to have a similar frequency to its corresponding n. This prevents our learning algorithm from focusing on any accidental frequency-based bias. We mix in K negatives for each positive, sampling without replacement to create all the negatives for a particular predicate. For each v, $\frac{1}{K+1}$ of its examples will be positive. The threshold τ represents a trade-off between capturing a large number of positive pairs and ensuring these pairs have good association. Similarly, K is a trade-off between the number of examples and the computational efficiency. Ultimately, these parameters should be optimized for task performance.

Of course, some negatives will actually be plausible arguments that were unobserved due to sparseness. Fortunately, modern discriminative methods like soft-margin SVMs can learn in the face of label error by allowing slack, subject to a tunable regularization penalty (Cortes and Vapnik, 1995).

If MI is a sparse and imperfect model of SP, what can DSP gain by training on MI's scores? We can regard DSP as learning a view of SP that is orthogonal to MI, in a co-training sense (Blum and Mitchell, 1998). MI labels the data based solely on co-occurrence; DSP uses these labels to identify other regularities – ones that extend beyond co-occurring words. For example, many instances of n where MI(*eat*, n)$>\tau$ also have MI(*buy*, n)$>\tau$ and MI(*cook*, n)$>\tau$. Also, compared to other nouns, a disproportionate number of *eat*-nouns are lower-case, single-token words, and they rarely contain digits, hyphens, or begin with a human first name like *Bob*. DSP encodes these interdependent properties as features in a linear classifier. This classifier can score any noun as a plausible argument of *eat* if indicative features are present; MI can only assign high plausibility to observed (*eat*,n) pairs. Similarity-smoothed models can make use of the regularities across similar verbs, but not the finer-grained string- and token-based features.

Our training examples are similar to the data created for *pseudodisambiguation*, the usual evaluation task for SP models (Erk, 2007; Keller and Lapata, 2003; Rooth et al., 1999). This data consists of triples (v,n,n') where v,n is a predicate-argument pair observed in the corpus and v,n' has not been observed. The models score correctly if they rank observed (and thus plausible) arguments above corresponding unobserved (and thus likely implausible) ones. We refer to this as *Pairwise Disambiguation*. Unlike this task, we classify each predicate-argument pair independently as plausible/implausible. We also use MI rather than frequency to define the positive pairs, ensuring that the positive pairs truly have a statistical association, and are not simply the result of parser error or noise.[1]

3.2 Partitioning for Efficient Training

After creating our positive and negative training pairs, we must select a feature representation for our examples. Let $\mathbf{\Phi}$ be a mapping from a predicate-argument pair (v,n) to a feature vector, $\mathbf{\Phi} : (v,n) \rightarrow \langle \phi_1 ... \phi_k \rangle$. Predictions are made based on a weighted combination of the features, $y = \boldsymbol{\lambda} \cdot \mathbf{\Phi}(v,n)$, where $\boldsymbol{\lambda}$ is our learned weight vector.

We can make training significantly more efficient by using a special form of attribute-value features. Let every feature ϕ_i be of the form $\phi_i(v,n) = \langle v = \hat{v} \wedge f(n) \rangle$. That is, every feature is an intersection of the occurrence of a particular predicate, $\hat{v}$, and some feature of the argument $f(n)$. For example, a feature for a verb-object pair might be, "the verb is *eat* and the object is lower-case." In this representation, features for one predicate will be completely independent from those for every other predicate. Thus rather than a single training procedure, we can actually partition the examples by predicate, and train a

[1] For a fixed verb, MI is proportional to Keller and Lapata (2003)'s conditional probability scores for pseudodisambiguation of (v,n,n') triples: $\Pr(v|n) = \Pr(v,n)/\Pr(n)$, which was shown to be a better measure of association than co-occurrence frequency $f(v,n)$. Normalizing by $\Pr(v)$ (yielding MI) allows us to use a constant threshold across all verbs. MI was also recently used for inference-rule SPs by Pantel et al. (2007).

classifier for each predicate independently. The prediction becomes $y^v = \boldsymbol{\lambda}^v \cdot \boldsymbol{\Phi}^v(n)$, where $\boldsymbol{\lambda}^v$ are the learned weights corresponding to predicate v and all features $\boldsymbol{\Phi}^v(n)$=$\mathbf{f}(n)$ depend on the argument only.

Some predicate partitions may have insufficient examples for training. Also, a predicate may occur in test data that was unseen during training. To handle these instances, we decided to cluster low-frequency predicates. In our experiments assigning SP to verb-object pairs, we cluster all verbs that have less than 250 positive examples, using clusters generated by the CBC algorithm (Pantel and Lin, 2002). For example, the low-frequency verbs *incarcerate*, *parole*, and *court-martial* are all mapped to the same partition, while more-frequent verbs like *arrest* and *execute* each have their own partition. About 5.5% of examples are clustered, corresponding to 30% of the 7367 total verbs. 40% of verbs (but only 0.6% of examples) were not in any CBC cluster; these were mapped to a single backoff partition.

The parameters for each partition, $\boldsymbol{\lambda}^v$, can be trained with any supervised learning technique. We use SVM (Section 4.1) because it is effective in similar high-dimensional, sparse-vector settings, and has an efficient implementation (Joachims, 1999). In SVM, the sign of y^v gives the classification. We can also use the scalar y^v as our DSP score (i.e. the positive distance from the separating SVM hyperplane).

3.3 Features

This section details our argument features, $\mathbf{f}(n)$, for assigning verb-object selectional preference. For a verb predicate (or partition) v and object argument n, the form of our classifier is $y^v = \sum_i \lambda_i^v f_i(n)$.

3.3.1 Verb co-occurrence

We provide features for the empirical probability of the noun occurring as the object argument of other verbs, $\Pr(n|v')$. If we were to only use these features (indexing the feature weights by each verb v'), the form of our classifier would be:

$$y^v = \sum_{v'} \lambda_{v'}^v \Pr(n|v') \quad (3)$$

Note the similarity between Equation (3) and Equation (1). Now the feature weights, $\lambda_{v'}^v$, take the role of the similarity function, $Sim(v', v)$. Unlike Equation (1), however, these weights are not set by an external similarity algorithm, but are optimized to discriminate the positive and negative training examples. We need not restrict ourselves to a short list of similar verbs; we include $\Pr_{obj}(n|v')$ features for every verb that occurs more than 10 times in our corpus. $\lambda_{v'}^v$ may be positive or negative, depending on the relation between v' and v. We also include features for the probability of the noun occurring as the *subject* of other verbs, $\Pr_{subj}(n|v')$. For example, nouns that can be the object of *eat* will also occur as the subject of *taste* and *contain*. Other contexts, such as adjectival and nominal predicates, could also aid the prediction, but have not yet been investigated.

The advantage of tuning similarity to the application of interest has been shown previously by Weeds and Weir (2005). They optimize a few meta-parameters separately for the tasks of thesaurus generation and pseudodisambiguation. Our approach, on the other hand, discriminatively sets millions of individual similarity values. Like Weeds and Weir (2005), our similarity values are asymmetric.

3.3.2 String-based

We include several simple character-based features of the noun string: the number of tokens, the case, and whether it contains digits, hyphens, an apostrophe, or other punctuation. We also include a feature for the first and last token, and fire indicator features if any token in the noun occurs on in-house lists of given names, family names, cities, provinces, countries, corporations, languages, etc. We also fire a feature if a token is a corporate designation (like *inc.* or *ltd.*) or a human one (like *Mr.* or *Sheik*).

3.3.3 Semantic classes

Motivated by previous SP models that make use of semantic classes, we generated word clusters using CBC (Pantel and Lin, 2002) on a 10 GB corpus, giving 3620 clusters. If a noun belongs in a cluster, a corresponding feature fires. If a noun is in none of the clusters, a *no-class* feature fires.

As an example, CBC cluster 1891 contains:

> sidewalk, driveway, roadway, footpath, bridge, highway, road, runway, street, alley, path, Interstate, ...

In our training data, we have examples like *widen highway*, *widen road* and *widen motorway.* If we

see that we can widen a highway, we learn that we can also widen a sidewalk, bridge, runway, etc.

We also made use of the person-name/instance pairs automatically extracted by Fleischman et al. (2003).[2] This data provides counts for pairs such as "Edwin Moses, *hurdler*" and "William Farley, *industrialist*." We have features for all *concepts* and therefore learn their association with each verb.

4 Experiments and Results

4.1 Set up

We parsed the 3 GB AQUAINT corpus (Voorhees, 2002) using Minipar (Lin, 1998b), and collected verb-object and verb-subject frequencies, building an empirical MI model from this data. Verbs and nouns were converted to their (possibly multi-token) *root*, and string case was preserved. Passive subjects (*the car was bought*) were converted to objects (*bought car*). We set the MI-threshold, τ, to be 0, and the negative-to-positive ratio, K, to be 2.

Numerous previous pseudodisambiguation evaluations only include arguments that occur between 30 and 3000 times (Erk, 2007; Keller and Lapata, 2003; Rooth et al., 1999). Presumably the lower bound is to help ensure the negative argument is unobserved because it is unsuitable, not because of data sparseness. We wish to use our model on arguments of any frequency, including those that never occurred in the training corpus (and therefore have empty co-occurrence features (Section 3.3.1)). We proceed as follows: first, we exclude pairs whenever the noun occurs less than 3 times in our corpus, removing many misspellings and other noun noise. Next, we omit verb co-occurrence features for nouns that occur less than 10 times, and instead fire a low-count feature. When we move to a new corpus, previously-unseen nouns are treated like these low-count training nouns.

This processing results in a set of 6.8 million pairs, divided into 2318 partitions (192 of which are verb clusters (Section 3.2)). For each partition, we take 95% of the examples for training, 2.5% for development and 2.5% for a final unseen test set. We provide full results for two models: DSP$_{cooc}$ which only uses the verb co-occurrence features, and DSP$_{all}$ which uses all the features mentioned in Section 3.3. Feature values are normalized within each feature type. We train our (linear kernel) discriminative models using SVM^{light} (Joachims, 1999) on each partition, but set meta-parameters C (regularization) and j (cost of positive vs. negative misclassifications: max at j=2) on the macro-averaged score across all development partitions. Note that we can not use the development set to optimize τ and K because the development examples are obtained *after* setting these values.

[2] Available at http://www.mit.edu/~mbf/instances.txt.gz

4.2 Feature weights

It is interesting to inspect the feature weights returned by our system. In particular, the weights on the verb co-occurrence features (Section 3.3.1) provide a high-quality, argument-specific similarity-ranking of other verb contexts. The DSP parameters for *eat*, for example, place high weight on features like Pr(n|*braise*), Pr(n|*ration*), and Pr(n|*garnish*). Lin (1998a)'s similar word list for *eat* misses these but includes *sleep* (ranked 6) and *sit* (ranked 14), because these have similar *subjects* to *eat*. Discriminative, context-specific training seems to yield a better set of similar predicates, e.g. the highest-ranked contexts for DSP$_{cooc}$ on the verb *join*,[3]

> lead 1.42, rejoin 1.39, form 1.34, belong to 1.31, found 1.31, quit 1.29, guide 1.19, induct 1.19, launch (*subj*) 1.18, work at 1.14

give a better SIMS(*join*) for Equation (1) than the top similarities returned by (Lin, 1998a):

> participate 0.164, lead 0.150, return to 0.148, say 0.143, rejoin 0.142, sign 0.142, meet 0.142, include 0.141, leave 0.140, work 0.137

Other features are also weighted intuitively. Note that case is a strong indicator for some arguments, for example the weight on being lower-case is high for *become* (0.972) and *eat* (0.505), but highly negative for *accuse* (-0.675) and *embroil* (-0.573) which often take names of people and organizations.

4.3 Pseudodisambiguation

We first evaluate DSP on disambiguating positives from pseudo-negatives, comparing to recently-

[3] Which all correspond to nouns occurring in the object position of the verb (e.g. Pr$_{obj}$(n|*lead*)), except "launch (*subj*)" which corresponds to Pr$_{subj}$(n|*launch*).

System	*MacroAvg*			*MicroAvg*			*Pairwise*	
	P	R	F	P	R	F	Acc	Cov
Dagan et al. (1999)	0.36	**0.90**	0.51	0.68	**0.92**	0.78	0.58	0.98
Erk (2007)	0.49	0.66	0.56	0.70	0.82	0.76	0.72	0.83
Keller and Lapata (2003)	**0.72**	0.34	0.46	**0.80**	0.50	0.62	0.80	0.57
DSP$_{cooc}$	0.53	0.72	0.61	0.73	0.94	**0.82**	0.77	**1.00**
DSP$_{all}$	0.60	0.71	**0.65**	0.77	0.90	**0.83**	**0.81**	**1.00**

Table 1: Pseudodisambiguation results averaged across each example (*MacroAvg*), weighted by word frequency (*MicroAvg*), plus coverage and accuracy of pairwise competition (*Pairwise*).

proposed systems that also require no manually-compiled resources like WordNet. We convert Dagan et al. (1999)'s similarity-smoothed probability to MI by replacing the empirical $\Pr(n|v)$ in Equation (2) with the smoothed $\Pr_{\text{SIM}}$ from Equation (1). We also test an MI model inspired by Erk (2007):

$$\text{MI}_{\text{SIM}}(n, v) = \log \sum_{n' \in \text{SIMS}(n)} \textit{Sim}(n', n) \frac{\Pr(v, n')}{\Pr(v)\Pr(n')}$$

We gather similar words using Lin (1998a), mining similar verbs from a comparable-sized parsed corpus, and collecting similar nouns from a broader 10 GB corpus of English text.[4]

We also use Keller and Lapata (2003)'s approach to obtaining web-counts. Rather than mining parse trees, this technique retrieves counts for the pattern "V Det N" in raw online text, where V is any inflection of the verb, Det is *the*, *a*, or the empty string, and N is the singular or plural form of the noun. We compute a web-based MI by collecting $\Pr(n, v)$, $\Pr(n)$, and $\Pr(v)$ using all inflections, except we only use the root form of the noun. Rather than using a search engine, we obtain counts from the Google Web 5-gram Corpus.[5]

All systems are thresholded at zero to make a classification. Unlike DSP, the comparison systems may not be able to provide a score for each example. The similarity-smoothed examples will be undefined if SIMS(w) is empty. Also, the Keller and Lapata (2003) approach will be undefined if the pair is unobserved on the web. As a reasonable default for these cases, we assign them a negative decision.

We evaluate disambiguation using precision (P), recall (R), and their harmonic mean, F-Score (F). Table 1 gives the results of our comparison. In the *MacroAvg* results, we weight each example equally. For *MicroAvg*, we weight each example by the frequency of the noun. To more directly compare with previous work, we also reproduced *Pairwise Disambiguation* by randomly pairing each positive with one of the negatives and then evaluating each system by the percentage it ranks correctly (Acc). For the comparison approaches, if one score is undefined, we choose the other one. If both are undefined, we abstain from a decision. Coverage (Cov) is the percent of pairs where a decision was made.[6]

Our simple system with only verb co-occurrence features, DSP$_{cooc}$, outperforms all comparison approaches. Using the richer feature set in DSP$_{all}$ results in a statistically significant gain in performance, up to an F-Score of 0.65 and a pairwise disambiguation accuracy of 0.81.[7] DSP$_{all}$ has both broader coverage and better accuracy than all competing approaches. In the remainder of the experiments, we use DSP$_{all}$ and refer to it simply as DSP.

Some errors are because of plausible but unseen arguments being used as test-set pseudo-negatives. For example, for the verb *damage*, DSP's three most high-scoring false positives are the nouns *jetliner*, *carpet*, and *gear*. While none occur with *damage* in

[4]For both the similar-noun and similar-verb smoothing, we only smooth over similar pairs *that occurred in the corpus*. While averaging over all similar pairs tends to underestimate the probability, averaging over only the observed pairs tends to overestimate it. We tested both and adopt the latter because it resulted in better performance on our development set.

[5]Available from the LDC as LDC2006T13. This collection was generated from approximately 1 trillion tokens of online text. Unfortunately, tokens appearing less than 200 times have been mapped to the ⟨UNK⟩ symbol, and only N-grams appearing more than 40 times are included. Unlike results from search engines, however, experiments with this corpus are replicable.

[6]I.e. we use the "half coverage" condition from Erk (2007).

[7]The differences between DSP$_{all}$ and all comparison systems are statistically significant (McNemar's test, $p<0.01$).

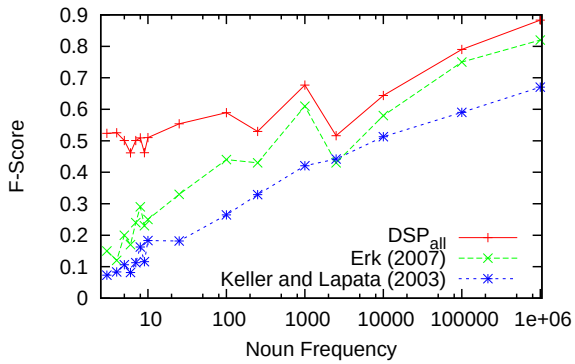

Figure 1: Disambiguation results by noun frequency.

our corpus, all intuitively satisfy the verb's SPs.

MacroAvg performance is worse than *MicroAvg* because all systems perform better on frequent nouns. When we plot F-Score by noun frequency (Figure 1), we see that DSP outperforms comparison approaches across all frequencies, but achieves its biggest gains on the low-frequency nouns. A richer feature set allows DSP to make correct inferences on examples that provide minimal co-occurrence data. These are also the examples for which we would expect co-occurrence models like MI to fail.

As a further experiment, we re-trained DSP but with only the string-based features removed. Overall macro-averaged F-score dropped from 0.65 to 0.64 (a statistically significant reduction in performance). The system scored nearly identically to DSP on the high-frequency nouns, but performed roughly 15% worse on the nouns that occurred less than ten times. This shows that the string-based features are important for selectional preference, and particularly helpful for low-frequency nouns.

4.4 Human Plausibility

Table 2 compares some of our systems on data used by Resnik (1996) (also Appendix 2 in Holmes et al. (1989)). The plausibility of these pairs was initially judged based on the experimenters' intuitions, and later confirmed in a human experiment. We include the scores of Resnik's system, and note that its errors were attributed to sense ambiguity and other limitations of class-based approaches (Resnik, 1996).[8]

[8]For example, *warn-engine* scores highly because engines are in the class *entity*, and physical entities (e.g. people) are often objects of *warn*. Unlike DSP, Resnik's approach cannot learn that for *warn*, "the property of being a person is more important than the property of being an entity" (Resnik, 1996).

Seen Criteria	Unseen Verb-Object Freq.				
	All	$= 1$	$= 2$	$= 3$	> 3
MI > 0	0.44	0.33	0.57	0.70	0.82
Freq. > 0	0.57	0.45	0.76	0.89	0.96
DSP > 0	0.73	0.69	0.80	0.85	0.88

Table 3: Recall on identification of Verb-Object pairs from an unseen corpus (divided by pair frequency).

The other comparison approaches also make a number of mistakes, which can often be traced to a misguided choice of similar word to smooth with.

We also compare to our empirical MI model, trained on our parsed corpus. Although Resnik (1996) reported that 10 of the 16 plausible pairs did not occur in his training corpus, all of them occurred in ours and hence MI gives very reasonable scores on the plausible objects. It has no statistics, however, for many of the implausible ones. DSP can make finer decisions than MI, recognizing that "warning an engine" is more absurd than "judging a climate."

4.5 Unseen Verb-Object Identification

We next compare MI and DSP on a much larger set of plausible examples, and also test how well the models generalize across data sets. We took the MI and DSP systems trained on AQUAINT and asked them to rate observed (and thus likely plausible) verb-object pairs taken from an unseen corpus. We extracted the pairs by parsing the San Jose Mercury News (SJM) section of the TIPSTER corpus (Harman, 1992). Each unique verb-object pair is a single instance in this evaluation.

Table 3 gives recall across all pairs **(All)** and grouped by pair-frequency in the unseen corpus (1, 2, 3, >3). DSP accepts far more pairs than MI (73% vs. 44%), even far more than a system that accepts any previously observed verb-object combination as plausible (57%). Recall is higher on more frequent verb-object pairs, but 70% of the pairs occurred only once in the corpus. Even if we smooth MI by smoothing $\Pr(n|v)$ in Equation 2 using modified KN-smoothing (Chen and Goodman, 1998), the recall of MI>0 on SJM only increases from 44.1% to 44.9%, still far below DSP. Frequency-based models have fundamentally low coverage. As fur-

Verb	Plaus./Implaus.	Resnik	Dagan et al.	Erk	MI	DSP
see	friend/method	5.79/-0.01	*0.20/1.40**	0.46/-0.07	1.11/-0.57	0.98/0.02
read	article/fashion	6.80/-0.20	3.00/0.11	3.80/1.90	4.00/—	2.12/-0.65
find	label/fever	1.10/0.22	*1.50/2.20**	0.59/0.01	0.42/0.07	1.61/0.81
hear	story/issue	*1.89/1.89**	*0.66/1.50**	*2.00/2.60**	2.99/-1.03	1.66/0.67
write	letter/market	7.26/0.00	2.50/-0.43	3.60/-0.24	5.06/-4.12	3.08/-1.31
urge	daughter/contrast	*1.14/1.86**	*0.14/1.60**	*1.10/3.60**	-0.95/—	-0.34/-0.62
warn	driver/engine	4.73/3.61	1.20/0.05	2.30/0.62	2.87/—	2.00/-0.99
judge	contest/climate	1.30/0.28	*1.50/1.90**	*1.70/1.70**	3.90/—	1.00/0.51
teach	language/distance	*1.87/1.86*	2.50/1.30	3.60/2.70	3.53/—	1.86/0.19
show	sample/travel	1.44/0.41	1.60/0.14	0.40/-0.82	0.53/-0.49	1.00/-0.83
expect	visit/mouth	*0.59/5.93**	*1.40/1.50**	1.40/0.37	1.05/-0.65	1.44/-0.15
answer	request/tragedy	4.49/3.88	2.70/1.50	3.10/-0.64	2.93/—	1.00/0.01
recognize	author/pocket	*0.50/0.50**	*0.03/0.37**	*0.77/1.30**	0.48/—	1.00/0.00
repeat	comment/journal	*1.23/1.23**	2.30/1.40	2.90/—	2.59/—	1.00/-0.48
understand	concept/session	*1.52/1.51*	2.70/0.25	2.00/-0.28	3.96/—	2.23/-0.46
remember	reply/smoke	1.31/0.20	2.10/1.20	*0.54/2.60**	1.13/-0.06	1.00/-0.42

Table 2: Selectional ratings for plausible/implausible direct objects (Holmes et al., 1989). Mistakes are marked with an asterisk (*), undefined scores are marked with a dash (—). Only DSP is completely defined and completely correct.

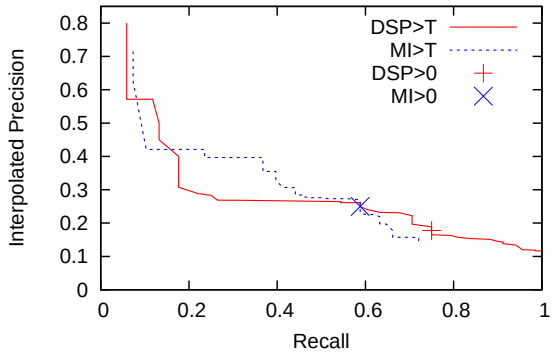

Figure 2: Pronoun resolution precision-recall on MUC.

ther evidence, if we build a model of MI on the SJM corpus and use it in our pseudodisambiguation experiment (Section 4.3), MI>0 gets a *MacroAvg* precision of 86% but a *MacroAvg* recall of only 12%.[9]

4.6 Pronoun Resolution

Finally, we evaluate DSP on a common application of selectional preferences: choosing the correct antecedent for pronouns in text (Dagan and Itai, 1990; Kehler et al., 2004). We study the cases where a pronoun is the direct object of a verb predicate, v. A pronoun's antecedent must obey v's selectional preferences. If we have a better model of SP, we should be able to better select pronoun antecedents.

We parsed the MUC-7 (1997) coreference corpus and extracted all pronouns in a direct object relation. For each pronoun, p, modified by a verb, v, we extracted all preceding nouns within the current or previous sentence. Thirty-nine anaphoric pronouns had an antecedent in this window and are used in the evaluation. For each p, let $N(p)^+$ by the set of preceding nouns coreferent with p, and let $N(p)^-$ be the remaining non-coreferent nouns. We take all (v, n^+) where $n^+ \in N(p)^+$ as positive, and all other pairs (v, n^-), $n^- \in N(p)^-$ as negative.

We compare MI and DSP on this set, classifying every (v, n) with MI>T (or DSP>T) as positive. By varying T, we get a precision-recall curve (Figure 2). Precision is low because, of course, there are many nouns that satisfy the predicate's SPs that are not coreferent. DSP>0 has both a higher recall and higher precision than accepting every pair previously seen in text (the right-most point on MI>T). The DSP>T system achieves higher precision than MI>T for points where recall is greater than 60% (where MI<0). Interestingly, the recall of MI>0 is

[9] Recall that even the Keller and Lapata (2003) system, built on the world's largest corpus, achieves only 34% recall (Table 1) (with only 48% of positives and 27% of all pairs previously observed, but see Footnote 5).

System	Acc
Most-Recent Noun	17.9%
Maximum MI	28.2%
Maximum DSP	38.5%

Table 4: Pronoun resolution accuracy on nouns in current or previous sentence in MUC.

higher here than it is for general verb-objects (Section 4.5). On the subset of pairs with strong empirical association (MI>0), MI generally outperforms DSP at equivalent recall values.

We next compare MI and DSP as stand-alone pronoun resolution systems (Table 4). As a standard baseline, for each pronoun, we choose the most recent noun in text as the pronoun's antecedent, achieving 17.9% resolution accuracy. This baseline is quite low because many of the most-recent nouns are subjects of the pronoun's verb phrase, and therefore resolution violates syntactic coreference constraints. If instead we choose the previous noun with the highest MI as antecedent, we get an accuracy of 28.2%, while choosing the previous noun with the highest DSP achieves 38.5%. DSP resolves 37% more pronouns correctly than MI. We leave as future work a full-scale pronoun resolution system that incorporates both MI and DSP as backed-off, interpolated, or separate semantic features.

5 Conclusions and Future Work

We have presented a simple, effective model of selectional preference based on discriminative training. Supervised techniques typically achieve higher performance than unsupervised models, and we duplicate these gains with DSP. Here, however, these gains come at no additional labeling cost, as training examples are generated automatically from unlabeled text. DSP allows an arbitrary combination of features, including verb co-occurrence features that yield high-quality similar-word lists as latent output. This work only scratches the surface of possible feature mining; information from WordNet relations, Wikipedia categories, or parallel corpora could also provide valuable clues to SP. Also, if any other system were to exceed DSP's performance, it could also be included as one of DSP's features.

It would be interesting to expand our co-occurrence features, including co-occurrence counts across more grammatical relations and using counts from external, unparsed corpora like the world wide web. We could also reverse the role of noun and verb in our training, having verb-specific features and discriminating separately for each argument noun. The latent information would then be lists of similar nouns.

Finally, note that while we focused on word-word co-occurrences, sense-sense SPs can also be learned with our algorithm. If our training corpus was sense-labeled, we could run our algorithm over the senses rather than the words. The resulting model would then require sense-tagged input if it were to be used within an application like parsing or coreference resolution. Also, like other models of SP, our technique can also be used for sense disambiguations: the weightings on our semantic class features indicate, for a particular noun, which of its senses (classes) is most compatible with each verb.

Acknowledgments

We gratefully acknowledge support from the Natural Sciences and Engineering Research Council of Canada, the Alberta Ingenuity Fund, and the Alberta Informatics Circle of Research Excellence.

References

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155.

Daniel M. Bikel. 2004. Intricacies of Collins' parsing model. *Computational Linguistics*, 30(4):479–511.

John Blitzer, Amir Globerson, and Fernando Pereira. 2005. Distributed latent variable models of lexical co-occurrences. In *AISTATS*.

Avrim Blum and Tom Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *Proceedings of COLT*, pages 92–100.

Carsten Brockmann and Mirella Lapata. 2003. Evaluating and combining approaches to selectional preference acquisition. In *EACL*, pages 27–34.

Stanley F. Chen and Joshua Goodman. 1998. An empirical study of smoothing techniques for language modeling. TR-10-98, Harvard University.

Kenneth Ward Church and Patrick Hanks. 1990. Word association norms, mutual information, and lexicography. *Computational Linguistics*, 16(1):22–29.

Stephen Clark and David Weir. 2002. Class-based probability estimation using a semantic hierarchy. *Computational Linguistics*, 28(2):187–206.

Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine Learning*, 20(3):273–297.

Ido Dagan and Alan Itai. 1990. Automatic processing of large corpora for the resolution of anaphora references. In *COLING*, volume 3, pages 330–332.

Ido Dagan, Lillian Lee, and Fernando C. N. Pereira. 1999. Similarity-based models of word cooccurrence probabilities. *Machine Learning*, 34(1-3):43–69.

Katrin Erk. 2007. A simple, similarity-based model for selectional preference. In *ACL*, pages 216–223.

Yair Even-Zohar and Dan Roth. 2000. A classification approach to word prediction. In *NAACL*, pages 124–131.

Michael Fleischman, Eduard Hovy, and Abdessamad Echihabi. 2003. Offline strategies for online question answering: answering questions before they are asked. In *ACL*, pages 1–7.

Donna Harman. 1992. The DARPA TIPSTER project. *ACM SIGIR Forum*, 26(2):26–28.

Virginia M. Holmes, Laurie Stowe, and Linda Cupples. 1989. Lexical expectations in parsing complement-verb sentences. *Journal of Memory and Language*, 28:668–689.

Thorsten Joachims. 1999. Making large-scale Support Vector Machine learning practical. In B. Schölkopf and C. Burges, editors, *Advances in Kernel Methods: Support Vector Machines*, pages 169–184. MIT-Press.

Andrew Kehler, Douglas Appelt, Lara Taylor, and Aleksandr Simma. 2004. The (non)utility of predicate-argument frequencies for pronoun interpretation. In *HLT/NAACL*, pages 289–296.

Frank Keller and Mirella Lapata. 2003. Using the web to obtain frequencies for unseen bigrams. *Computational Linguistics*, 29(3):459–484.

Hang Li and Naoki Abe. 1998. Generalizing case frames using a thesaurus and the MDL principle. *Computational Linguistics*, 24(2):217–244.

Dekang Lin. 1998a. Automatic retrieval and clustering of similar words. In *COLING-ACL*, pages 768–773.

Dekang Lin. 1998b. Dependency-based evaluation of MINIPAR. In *LREC Workshop on the Evaluation of Parsing Systems*.

George A. Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine J. Miller. 1990. Introduction to WordNet: an on-line lexical database. *International Journal of Lexicography*, 3(4):235–244.

MUC-7. 1997. Coreference task definition (v3.0, 13 Jul 97). In *Proceedings of the Seventh Message Understanding Conference (MUC-7)*.

Daisuke Okanohara and Jun'ichi Tsujii. 2007. A discriminative language model with pseudo-negative samples. In *ACL*, pages 73–80.

Patrick Pantel and Dekang Lin. 2002. Discovering word senses from text. In *KDD*, pages 613–619.

Patrick Pantel, Rahul Bhagat, Bonaventura Coppola, Timothy Chklovski, and Eduard Hovy. 2007. ISP: Learning inferential selectional preferences. In *NAACL-HLT*, pages 564–571.

Philip Resnik. 1996. Selectional constraints: An information-theoretic model and its computational realization. *Cognition*, 61:127–159.

Basili Roberto, Diego De Cao, Paolo Marocco, and Marco Pennacchiotti. 2007. Learning selectional preferences for entailment or paraphrasing rules. In *RANLP*.

Mats Rooth, Stefan Riezler, Detlef Prescher, Glenn Carroll, and Franz Beil. 1999. Inducing a semantically annotated lexicon via EM-based clustering. In *ACL*, pages 104–111.

Noah A. Smith and Jason Eisner. 2005. Contrastive estimation: training log-linear models on unlabeled data. In *ACL*, pages 354–362.

Ellen Voorhees. 2002. Overview of the TREC 2002 question answering track. In *Proceedings of the Eleventh Text REtrieval Conference (TREC)*.

Qin Iris Wang, Dale Schuurmans, and Dekang Lin. 2005. Strictly lexical dependency parsing. In *International Workshop on Parsing Technologies*, pages 152–159.

Julie Weeds and David Weir. 2005. Co-occurrence retrieval: a flexible framework for lexical distributional similarity. *Computational Linguistics*, 31(4):439–475.