Using the Web as an Implicit Training Set: Application to Structural Ambiguity Resolution

Preslav Nakov and **Marti Hearst**
EECS and SIMS
University of California at Berkeley
Berkeley, CA 94720
nakov@cs.berkeley.edu, hearst@sims.berkeley.edu

Abstract

Recent work has shown that very large corpora can act as training data for NLP algorithms even without explicit labels. In this paper we show how the use of surface features and paraphrases in queries against search engines can be used to infer labels for structural ambiguity resolution tasks. Using unsupervised algorithms, we achieve 84% precision on PP-attachment and 80% on noun compound coordination.

1 Introduction

Resolution of structural ambiguity problems such as noun compound bracketing, prepositional phrase (PP) attachment, and noun phrase coordination requires using information about lexical items and their cooccurrences. This in turn leads to the data sparseness problem, since algorithms that rely on making decisions based on individual lexical items must have statistics about every word that may be encountered. Past approaches have dealt with the data sparseness problem by attempting to generalize from semantic classes, either manually built or automatically derived.

More recently, Banko and Brill (2001) have advocated for the creative use of very large text collections as an alternative to sophisticated algorithms and hand-built resources. They demonstrate the idea on a lexical disambiguation problem for which labeled examples are available "for free". The problem is to choose which of 2-3 commonly confused words (e.g., {*principle, principal*}) are appropriate for a given context. The labeled data comes "for free" by assuming that in most edited written text, the words are used correctly, so training can be done directly from the text. Banko and Brill (2001) show that even using a very simple algorithm, the results continue to improve log-linearly with more training data, even out to a billion words. A potential limitation of this approach is the question of how applicable it is for NLP problems more generally – how can we treat a large corpus as a labeled collection for a wide range of NLP tasks?

In a related strand of work, Lapata and Keller (2004) show that computing n-gram statistics over very large corpora yields results that are competitive with if not better than the best supervised and knowledge-based approaches on a wide range of NLP tasks. For example, they show that for the problem of noun compound bracketing, the performance of an n-gram based model computed using search engine statistics was not significantly different from the best supervised algorithm whose parameters were tuned and which used a taxonomy. They find however that these approaches generally fail to outperform supervised state-of-the-art models that are trained on smaller corpora, and so conclude that web-based n-gram statistics should be the baseline to beat.

We feel the potential of these ideas is not yet fully realized. We are interested in finding ways to further exploit the availability of enormous web corpora as implicit training data. This is especially important for structural ambiguity problems in which the decisions must be made on the basis of the behavior of individual lexical items. The trick is to figure out how to use information that is latent in the web as a corpus, and web search engines as query interfaces to that corpus.

In this paper we describe two techniques – *surface features* and *paraphrases* – that push the ideas of Banko and Brill (2001) and Lapata and Keller (2004) farther, enabling the use of statistics gathered from very large corpora in an unsupervised manner. In recent work (Nakov and Hearst, 2005) we showed that a variation of the techniques, when applied to the problem of noun compound bracketing, produces higher accuracy than Lapata and Keller (2004) and the best supervised results. In this paper we adapt the techniques to the structural disambiguation problems of prepositional phrase attachment and noun compound coordination.

2 Prepositional Phrase Attachment

A long-standing challenge for syntactic parsers is the attachment decision for prepositional phrases. In a configuration where a verb takes a noun complement that is followed by a PP, the problem arises of whether the PP attaches to the noun or to the verb. Consider the following contrastive pair of sentences:

(1) *Peter spent millions of dollars.* (noun)
(2) *Peter spent time with his family.* (verb)

In the first example, the PP *millions of dollars* attaches to the noun *millions*, while in the second the PP *with his family* attaches to the verb *spent*.

Past work on PP-attachment has often cast these associations as the quadruple (v, n_1, p, n_2), where v is the verb, n_1 is the head of the direct object, p is the preposition (the head of the PP) and n_2 is the head of the NP inside the PP. For example, the quadruple for (2) is (*spent*, *time*, *with*, *family*).

2.1 Related Work

Early work on PP-attachment ambiguity resolution relied on syntactic (e.g., “minimal attachment” and “right association”) and pragmatic considerations. Most recent work can be divided into supervised and unsupervised approaches. Supervised approaches tend to make use of semantic classes or thesauri in order to deal with data sparseness problems. Brill and Resnik (1994) used the supervised transformation-based learning method and lexical and conceptual classes derived from WordNet, achieving 82% precision on 500 randomly selected examples. Ratnaparkhi et al. (1994) created a benchmark dataset of 27,937 quadruples (v, n_1, p, n_2), extracted from the Wall Street Journal. They found the human performance on this task to be 88%[1]. Using this dataset, they trained a maximum entropy model and a binary hierarchy of word classes derived by mutual information, achieving 81.6% precision. Collins and Brooks (1995) used a supervised back-off model to achieve 84.5% precision on the Ratnaparkhi test set. Stetina and Makoto (1997) use a supervised method with a decision tree and WordNet classes to achieve 88.1% precision on the same test set. Toutanova et al. (2004) use a supervised method that makes use of morphological and syntactic analysis and WordNet synsets, yielding 87.5% accuracy.

In the unsupervised approaches, the attachment decision depends largely on co-occurrence statistics drawn from text collections. The pioneering work in this area was that of Hindle and Rooth (1993). Using a partially parsed corpus, they calculate and compare lexical associations over subsets of the tuple (v, n_1, p), ignoring n_2, and achieve 80% precision at 80% recall.

More recently, Ratnaparkhi (1998) developed an unsupervised method that collects statistics from text annotated with part-of-speech tags and morphological base forms. An extraction heuristic is used to identify unambiguous attachment decisions, for example, the algorithm can assume a noun attachment if there is no verb within k words to the left of the preposition in a given sentence, among other conditions. This extraction heuristic uncovered 910K unique tuples of the form (v, p, n_2) and (n, p, n_2), although the results are very noisy, suggesting the correct attachment only about 69% of the time. The tuples are used as training data for classifiers, the best of which achieves 81.9% precision on the Ratnaparkhi test set. Pantel and Lin (2000) describe an unsupervised method that uses a collocation database, a thesaurus, a dependency parser, and a large corpus (125M words), achieving 84.3% precision on the Ratnaparkhi test set. Using sim-

[1] When presented with a whole sentence, average humans score 93%.

ple combinations of web-based n-grams, Lapata and Keller (2005) achieve lower results, in the low 70's.

Using a different collection consisting of German PP-attachment decisions, Volk (2000) uses the web to obtain n-gram counts. He compared $\Pr(p|n_1)$ to $\Pr(p|v)$, where $\Pr(p|x) = \#(x,p)/\#(x)$. Here x can be n_1 or v. The bigram frequencies $\#(x,p)$ were obtained using the Altavista NEAR operator. The method was able to make a decision on 58% of the examples with a precision of 75% (baseline 63%). Volk (2001) then improved on these results by comparing $\Pr(p, n_2|n_1)$ to $\Pr(p, n_2|v)$. Using inflected forms, he achieved P=75% and R=85%.

Calvo and Gelbukh (2003) experimented with a variation of this, using exact phrases instead of the NEAR operator. For example, to disambiguate *Veo al gato con un telescopio*, they compared frequencies for phrases such as "ver con telescopio" and "gato con telescopio". They tested this idea on 181 randomly chosen Spanish disambiguation examples, labelling 89.5% recall with a precision of 91.97%.

2.2 Models and Features

2.2.1 n-gram Models

We computed two co-occurrence models;

(*i*) $\Pr(p|n_1)$ vs. $\Pr(p|v)$

(*ii*) $\Pr(p, n_2|n_1)$ vs. $\Pr(p, n_2|v)$.

Each of these was computed two different ways: using $\Pr$ (probabilities) and $\#$ (frequencies). We estimate the n-gram counts using exact phrase queries (with inflections, derived from WordNet 2.0) using the MSN Search Engine. We also allow for determiners, where appropriate, e.g., between the preposition and the noun when querying for $\#(p, n_2)$. We add up the frequencies for all possible variations. Web frequencies were reliable enough and did not need smoothing for (*i*), but for (*ii*), smoothing using the technique described in Hindle and Rooth (1993) led to better recall. We also tried back-off from (*ii*) to (*i*), as well as back-off plus smoothing, but did not find improvements over smoothing alone. We found n-gram counts to be unreliable when pronouns appear in the test set rather than nouns, and disabled them in these cases. Such examples can still be handled by paraphrases or surface features (see below).

2.2.2 Web-Derived Surface Features

Authors sometimes (consciously or not) disambiguate the words they write by using surface-level markers to suggest the correct meaning. We have found that exploiting these markers, when they occur, can prove to be very helpful for making disambiguation decisions. The enormous size of web search engine indexes facilitates finding such markers frequently enough to make them useful.

For example, *John opened the door with a key* is a difficult verb attachment example because doors, keys, and opening are all semantically related. To determine if this should be a verb or a noun attachment, we search for cues that indicate which of these terms tend to associate most closely. If we see parentheses used as follows:

"open the door (with a key)"

this suggests a verb attachment, since the parentheses signal that "with a key" acts as its own unit. Similarly, hyphens, colons, capitalization, and other punctuation can help signal disambiguation decisions. For *Jean ate spaghetti with sauce*, if we see

"eat: spaghetti with sauce"

this suggests a noun attachment.

Table 1 illustrates a wide variety of surface features, along with the attachment decisions they are assumed to suggest (events of frequency 1 have been ignored). The surface features for PP-attachment have low recall: most of the examples have no surface features extracted.

We gather the statistics needed by issuing queries to web search engines. Unfortunately, search engines usually ignore punctuation characters, thus preventing querying directly for terms containing hyphens, brackets, etc. We collect these numbers indirectly by issuing queries with exact phrases and then post-processing the top 1,000 resulting summaries[2], looking for the surface features of interest. We use Google for both the surface feature and paraphrase extractions (described below).

2.2.3 Paraphrases

The second way we extend the use of web counts is by paraphrasing the relation of interest and seeing if it can be found in its alternative form, which

[2] We often obtain more than 1,000 summaries per example because we usually issue multiple queries per surface pattern, by varying inflections and inclusion of determiners.

suggests the correct attachment decision. We use the following patterns along with their associated attachment predictions:

(1)	$v\ n_2\ n_1$	(noun)
(2)	$v\ p\ n_2\ n_1$	(verb)
(3)	$p\ n_2$ * $v\ n_1$	(verb)
(4)	$n_1\ p\ n_2\ v$	(noun)
(5)	v *pronoun* $p\ n_2$	(verb)
(6)	*be* $n_1\ p\ n_2$	(noun)

The idea behind Pattern (1) is to determine if "$n_1\ p\ n_2$" can be expressed as a noun compound; if this happens sufficiently often, we can predict a noun attachment. For example, *meet/v demands/n_1 from/p customers/n_2* becomes *meet/v the customers/n_2 demands/n_1*.

Note that the pattern could wrongly target ditransitive verbs: e.g., it could turn *gave/v an apple/n_1 to/p him/n_2* into *gave/v him/n_2 an apple/n_1*. To prevent this, we do not allow a determiner before n_1, but we do require one before n_2. In addition, we disallow the pattern if the preposition is *to* and we require both n_1 and n_2 to be nouns (as opposed to numbers, percents, pronouns, determiners etc.).

Pattern (2) predicts a verb attachment. It presupposes that "$p\ n_2$" is an indirect object of the verb v and tries to switch it with the direct object n_1, e.g., *had/v a program/n_1 in/p place/n_2* would be transformed into *had/v in/p place/n_2 a program/n_1*. We require n_1 to be preceded by a determiner (to prevent "$n_2\ n_1$" forming a noun compound).

Pattern (3) looks for appositions, where the PP has moved in front of the verb, e.g., *to/p him/n_2 I gave/v an apple/n_1*. The symbol * indicates a wildcard position where we allow up to three intervening words.

Pattern (4) looks for appositions, where the PP has moved in front of the verb together with n_1. It would transform *shaken/v confidence/n_1 in/p markets/n_2* into *confidence/n_1 in/p markets/n_2 shaken/v*.

Pattern (5) is motivated by the observation that if n_1 is a pronoun, this suggests a verb attachment (Hindle and Rooth, 1993). (A separate feature checks if n_1 is a pronoun.) The pattern substitutes n_1 with a dative pronoun (we allow *him* and *her*), e.g., it will convert *put/v a client/n_1 at/p odds/n_2* into *put/v him at/p odds/n_2*.

Pattern (6) is motivated by the observation that the verb *to be* is typically used with a noun attachment. (A separate feature checks if v is a form of the verb *to be*.) The pattern substitutes v with *is* and *are*, e.g. it will turn *eat/v spaghetti/n_1 with/p sauce/n_2* into *is spaghetti/n_1 with/p sauce/n_2*.

These patterns all allow for determiners where appropriate, unless explicitly stated otherwise. For a given example, a prediction is made if at least one instance of the pattern has been found.

2.3 Evaluation

For the evaluation, we used the test part (3,097 examples) of the benchmark dataset by Ratnaparkhi et al. (1994). We used all 3,097 test examples in order to make our results directly comparable.

Unfortunately, there are numerous errors in the test set[3]. There are 149 examples in which a bare determiner is labeled as n_1 or n_2 rather than the actual head noun. Supervised algorithms can compensate for this problem by learning from the training set that "the" can act as a noun in this collection, but unsupervised algorithms cannot.

In addition, there are also around 230 examples in which the nouns contain special symbols like: %, slash, &, ', which are lost when querying against a search engine. This poses a problem for our algorithm but is not a problem with the test set itself.

The results are shown in Table 2. Following Ratnaparkhi (1998), we predict a noun attachment if the preposition is *of* (a very reliable heuristic). The table shows the performance for each feature in isolation (excluding examples whose preposition is *of*). The surface features are represented by a single score in Table 2: for a given example, we sum up separately the number of noun- and verb-attachment pattern matches, and assign the attachment with the larger number of matches.

We combine the bold rows of Table 2 in a majority vote (assigning noun attachment to all *of* instances), obtaining P=85.01%, R=91.77%. To get 100% recall, we assign all undecided cases to *verb* (since the majority of the remaining non-*of* instances attach to the verb, yielding P=83.63%, R=100%. We show 0.95-level confidence intervals for the precision, computed by a general method based on constant chi-square boundaries (Fleiss, 1981).

A test for statistical significance reveals that our results are as strong as those of the leading unsuper-

[3] Ratnaparkhi (1998) notes that the test set contains errors, but does not correct them.

Example	Predicts	P(%)	R(%)
open Door with a key	noun	100.00	0.13
(open) door with a key	noun	66.67	0.28
open (door with a key)	noun	71.43	0.97
open - door with a key	noun	69.70	1.52
open / door with a key	noun	60.00	0.46
open, door with a key	noun	65.77	5.11
open: door with a key	noun	64.71	1.57
open; door with a key	noun	60.00	0.23
open. door with a key	noun	64.13	4.24
open? door with a key	noun	83.33	0.55
open! door with a key	noun	66.67	0.14
open door With a Key	verb	0.00	0.00
(open door) with a key	verb	50.00	0.09
open door (with a key)	verb	73.58	2.44
open door - with a key	verb	68.18	2.03
open door / with a key	verb	100.00	0.14
open door, with a key	verb	58.44	7.09
open door: with a key	verb	70.59	0.78
open door; with a key	verb	75.00	0.18
open door. with a key	verb	60.77	5.99
open door! with a key	verb	100.00	0.18

Table 1: **PP-attachment surface features.** Precision and recall shown are across all examples, not just the door example shown.

Model	**P**(%)	**R**(%)
Baseline (noun attach)	41.82	100.00
$\#(x, p)$	58.91	83.97
$\Pr(p \mid x)$	66.81	83.97
$\Pr(p \mid x)$ smoothed	**66.81**	83.97
$\#(x, p, n_2)$	65.78	81.02
$\Pr(p, n_2 \mid x)$	68.34	81.62
$\Pr(p, n_2 \mid x)$ smoothed	**68.46**	83.97
(1) “v n_2 n_1”	**59.29**	22.06
(2) “p n_2 v n_1”	**57.79**	71.58
(3) “n_1 * p n_2 v”	**65.78**	20.73
(4) “v p n_2 n_1”	**81.05**	8.75
(5) “v *pronoun* p n_2”	**75.30**	30.40
(6) “*be* n_1 p n_2”	**63.65**	30.54
n_1 is *pronoun*	**98.48**	3.04
v is *to be*	**79.23**	9.53
Surface features (summed)	**73.13**	9.26
Maj. vote, of → noun	85.01±1.21	91.77
Maj. vote, of → noun, N/A → verb	***83.63±1.30***	***100.00***

Table 2: **PP-attachment results, in percentages.**

vised approach on this collection (Pantel and Lin, 2000). Unlike that work, we do not require a collocation database, a thesaurus, a dependency parser, nor a large domain-dependent text corpus, which makes our approach easier to implement and to extend to other languages.

3 Coordination

Coordinating conjunctions (*and*, *or*, *but*, etc.) pose major challenges to parsers and their proper handling is essential for the understanding of the sentence. Consider the following “cooked” example:

The Department of Chronic Diseases ***and*** *Health Promotion leads* ***and*** *strengthens global efforts to prevent* ***and*** *control chronic diseases* ***or*** *disabilities* ***and*** *to promote health* ***and*** *quality of life.*

Conjunctions can link two words, two constituents (e.g., NPs), two clauses or even two sentences. Thus, the first challenge is to identify the boundaries of the conjuncts of each coordination. The next problem comes from the interaction of the coordinations with other constituents that attach to its conjuncts (most often prepositional phrases). In the example above we need to decide between *[health and [quality of life]]* and *[[health and quality] of life]*. From a semantic point of view, we need to determine whether the *or* in *chronic diseases or disabilities* really means *or* or is used as an *and* (Agarwal and Boggess, 1992). Finally, we need to choose between a *non-elided* and an *elided* reading: *[[chronic diseases] or disabilities]* vs. *[chronic [diseases or disabilities]]*.

Below we focus on a special case of the latter problem: noun compound (NC) coordination. Consider the NC *car and truck production*. Its real meaning is *car production and truck production*. However, due to the principle of economy of expression, the first instance of *production* has been compressed out by means of ellipsis. By contrast, in *president and chief executive*, *president* is simply linked to *chief executive*. There is also an all-way coordination, where the conjunct is part of the whole, as in *Securities and Exchange Commission*.

More formally, we consider configurations of the kind n_1 c n_2 h, where n_1 and n_2 are nouns, c is a coordination (*and* or *or*) and h is the head noun[4]. The task is to decide whether there is an ellipsis or not, independently of the local context. Syntactically, this can be expressed by the following bracketings: [[n_1 c n_2] h] versus [n_1 c [n_2 h]]. (Collins' parser (Collins, 1997) always predicts a flat NP for such configurations.) In order to make the task more

[4] The configurations of the kind n h_1 c h_2 (e.g., *company/n cars/h_1 and/c trucks/h_2*) can be handled in a similar way.

realistic (from a parser's perspective), we ignore the option of all-way coordination and try to predict the bracketing in Penn Treebank (Marcus et al., 1994) for configurations of this kind. The Penn Treebank brackets NCs with ellipsis as, e.g.,
(NP car/NN and/CC truck/NN production/NN).

and without ellipsis as
(NP (NP president/NN) and/CC (NP chief/NN executive/NN))
The NPs with ellipsis are flat, while the others contain internal NPs. The all-way coordinations can appear bracketed either way and make the task harder.

3.1 Related Work

Coordination ambiguity is under-explored, despite being one of the three major sources of structural ambiguity (together with prepositional phrase attachment and noun compound bracketing), and belonging to the class of ambiguities for which the number of analyses is the number of binary trees over the corresponding nodes (Church and Patil, 1982), and despite the fact that conjunctions are among the most frequent words.

Rus et al. (2002) present a deterministic rule-based approach for bracketing *in context* of coordinated NCs of the kind $n_1\ c\ n_2\ h$, as a necessary step towards logical form derivation. Their algorithm uses POS tagging, syntactic parses, semantic senses of the nouns (manually annotated), lookups in a semantic network (WordNet) and the type of the coordination conjunction to make a 3-way classification: ellipsis, no ellipsis and all-way coordination. Using a back-off sequence of 3 different heuristics, they achieve 83.52% precision (baseline 61.52%) on a set of 298 examples. When 3 additional context-dependent heuristics and 224 additional examples with local contexts are added, the precision jumps to 87.42% (baseline 52.35%), with 71.05% recall.

Resnik (1999) disambiguates two kinds of patterns: $n_1\ and\ n_2\ n_3$ and $n_1\ n_2\ and\ n_3\ n_4$ (e.g., *[food/n_1 [handling/n_2 and/c storage/n_3] procedures/n_4]*). While there are two options for the former (all-way coordinations are not allowed), there are 5 valid bracketings for the latter. Following Kurohashi and Nagao (1992), Resnik makes decisions based on similarity of form (i.e., number agreement: P=53%, R=90.6%), similarity of meaning (P=66%, R=71.2%) and conceptual association (P=75.0%, R=69.3%). Using a decision tree to combine the three information sources, he achieves 80% precision (baseline 66%) at 100% recall for the 3-noun coordinations. For the 4-noun coordinations the precision is 81.6% (baseline 44.9%), 85.4% recall.

Example	Predicts	P(%)	R(%)
(buy) and sell orders	NO ellipsis	33.33	1.40
buy (and sell orders)	NO ellipsis	70.00	4.67
buy: and sell orders	NO ellipsis	0.00	0.00
buy; and sell orders	NO ellipsis	66.67	2.80
buy. and sell orders	NO ellipsis	68.57	8.18
buy[...] and sell orders	NO ellipsis	49.00	46.73
buy- and sell orders	ellipsis	77.27	5.14
buy and sell / orders	ellipsis	50.54	21.73
(buy and sell) orders	ellipsis	92.31	3.04
buy and sell (orders)	ellipsis	90.91	2.57
buy and sell, orders	ellipsis	92.86	13.08
buy and sell: orders	ellipsis	93.75	3.74
buy and sell; orders	ellipsis	100.00	1.87
buy and sell. orders	ellipsis	93.33	7.01
buy and sell[...] orders	ellipsis	85.19	18.93

Table 3: **Coordination surface features.** Precision and recall shown are across all examples, not just the *buy and sell orders* shown.

Chantree et al. (2005) cover a large set of ambiguities, not limited to nouns. They allow the head word to be a noun, a verb or an adjective, and the modifier to be an adjective, a preposition, an adverb, etc. They extract distributional information from the British National Corpus and distributional similarities between words, similarly to (Resnik, 1999). In two different experiments they achieve P=88.2%, R=38.5% and P=80.8%, R=53.8% (baseline P=75%).

Goldberg (1999) resolves the *attachment of ambiguous coordinate phrases* of the kind $n_1\ p\ n_2\ c\ n_3$, e.g., *box/n_1 of/p chocolates/n_2 and/c roses/n_3*. Using an adaptation of the algorithm proposed by Ratnaparkhi (1998) for PP-attachment, she achieves P=72% (baseline P=64%), R=100.00%.

Agarwal and Boggess (1992) focus on the *identification of the conjuncts of coordinate conjunctions*. Using POS and case labels in a deterministic algorithm, they achieve P=81.6%. Kurohashi and Nagao (1992) work on the same problem for Japanese. Their algorithm looks for similar word sequences among with sentence simplification, and achieves a precision of 81.3%.

3.2 Models and Features

3.2.1 n-gram Models

We use the following n-gram models:

(*i*) $\#(n_1, h)$ vs. $\#(n_2, h)$

(*ii*) $\#(n_1, h)$ vs. $\#(n_1, c, n_2)$

Model (*i*) compares how likely it is that n_1 modifies h, as opposed to n_2 modifying h. Model (*ii*) checks which association is stronger: between n_1 and h, or between n_1 and n_2. Regardless of whether the coordination is *or* or *and*, we query for both and we add up the corresponding counts.

3.2.2 Web-Derived Surface Features

The set of surface features is similar to the one we used for PP-attachment. These are brackets, slash, comma, colon, semicolon, dot, question mark, exclamation mark, and any character. There are two additional ellipsis-predicting features: a dash after n_1 and a slash after n_2, see Table 3.

3.2.3 Paraphrases

We use the following paraphrase patterns:

(1)	$n_2\ c\ n_1\ h$	(ellipsis)
(2)	$n_2\ h\ c\ n_1$	(NO ellipsis)
(3)	$n_1\ h\ c\ n_2\ h$	(ellipsis)
(4)	$n_2\ h\ c\ n_1\ h$	(ellipsis)

If matched frequently enough, each of these patterns predicts the coordination decision indicated in parentheses. If found only infrequently or not found at all, the opposite decision is made. Pattern (1) switches the places of n_1 and n_2 in the coordinated NC. For example, *bar and pie graph* can easily become *pie and bar graph*, which favors ellipsis. Pattern (2) moves n_2 and h together to the left of the coordination conjunction, and places n_1 to the right. If this happens frequently enough, there is no ellipsis. Pattern (3) inserts the elided head h after n_1 with the hope that if there is ellipsis, we will find the full phrase elsewhere in the data. Pattern (4) combines pattern (1) and pattern (3); it not only inserts h after n_1 but also switches the places of n_1 and n_2.

As shown in Table 4, we included four of the heuristics by Rus et al. (2002). Heuristic 1 predicts no coordination when n_1 and n_2 are the same, e.g., *milk and milk products*. Heuristics 2 and 3 perform a lookup in WordNet and we did not use them. Heuristics 4, 5 and 6 exploit the local context, namely the adjectives modifying n_1 and/or n_2. Heuristic 4 predicts no ellipsis if both n_1 and n_2 are modified by adjectives. Heuristic 5 predicts ellipsis if the coordination is *or* and n_1 is modified by an adjective, but n_2 is not. Heuristic 6 predicts no ellipsis if n_1 is not modified by an adjective, but n_2 is. We used versions of heuristics 4, 5 and 6 that check for determiners rather than adjectives.

Model	**P**(%)	**R**(%)
Baseline: ellipsis	56.54	100.00
(n_1, h) vs. (n_2, h)	**80.33**	28.50
(n_1, h) vs. (n_1, c, n_2)	61.14	45.09
(n_2, c, n_1, h)	**88.33**	14.02
(n_2, h, c, n_1)	**76.60**	21.96
(n_1, h, c, n_2, h)	**75.00**	6.54
(n_2, h, c, n_1, h)	**78.67**	17.52
Heuristic 1	**75.00**	0.93
Heuristic 4	64.29	6.54
Heuristic 5	61.54	12.15
Heuristic 6	**87.09**	7.24
Number agreement	**72.22**	46.26
Surface sum	**82.80**	21.73
Majority vote	83.82	80.84
Majority vote, N/A → no ellipsis	***80.61***	***100.00***

Table 4: **Coordination results, in percentages.**

Finally, we included the number agreement feature (Resnik, 1993): (a) if n_1 and n_2 match in number, but n_1 and h do not, predict ellipsis; (b) if n_1 and n_2 do not match in number, but n_1 and h do, predict no ellipsis; (c) otherwise leave undecided.

3.3 Evaluation

We evaluated the algorithms on a collection of 428 examples extracted from the Penn Treebank. On extraction, determiners and non-noun modifiers were allowed, but the program was only presented with the quadruple (n_1, c, n_2, h). As Table 4 shows, our overall performance of 80.61 is on par with other approaches, whose best scores fall into the low 80's for precision. (Direct comparison is not possible, as the tasks and datasets all differ.)

As Table 4 shows, n-gram model (*i*) performs well, but n-gram model (*ii*) performs poorly, probably because the (n_1, c, n_2) contains three words, as opposed to two for the alternative (n_1, h), and thus a priori is less likely to be observed.

The surface features are less effective for resolving coordinations. As Table 3 shows, they are very good predictors of ellipsis, but are less reliable when predicting NO ellipsis. We combine the bold rows of Table 4 in a majority vote, obtaining P=83.82%, R=80.84%. We assign all undecided cases to no ellipsis, yielding P=80.61%, R=100%.

4 Conclusions and Future Work

We have shown that simple unsupervised algorithms that make use of bigrams, surface features and paraphrases extracted from a very large corpus are effective for several structural ambiguity resolutions tasks, yielding results competitive with the best unsupervised results, and close to supervised results. The method does not require labeled training data, nor lexicons nor ontologies. We think this is a promising direction for a wide range of NLP tasks. In future work we intend to explore better-motivated evidence combination algorithms and to apply the approach to other NLP problems.

Acknowledgements. This research was supported by NSF DBI-0317510 and a gift from Genentech.

References

Rajeev Agarwal and Lois Boggess. 1992. A simple but useful approach to conjunct identification. In *Proceedings of ACL*.

Michele Banko and Eric Brill. 2001. Scaling to very very large corpora for natural language disambiguation. In *Proceedings of ACL*.

Eric Brill and Philip Resnik. 1994. A rule-based approach to prepositional phrase attachment disambiguation. In *Proceedings of COLING*.

Hiram Calvo and Alexander Gelbukh. 2003. Improving prepositional phrase attachment disambiguation using the web as corpus. In *Progress in Pattern Recognition, Speech and Image Analysis: 8th Iberoamerican Congress on Pattern Recognition, CIARP 2003*.

Francis Chantree, Adam Kilgarriff, Anne De Roeck, and Alistair Willis. 2005. Using a distributional thesaurus to resolve coordination ambiguities. In *Technical Report 2005/02*. The Open University, UK.

Kenneth Church and Ramesh Patil. 1982. Coping with syntactic ambiguity or how to put the block in the box on the table. *Amer. J. of Computational Linguistics*, 8(3-4):139–149.

Michael Collins and James Brooks. 1995. Prepositional phrase attachment through a backed-off model. In *Proceedings of EMNLP*, pages 27–38.

M. Collins. 1997. Three generative, lexicalised models for statistical parsing. In *Proceedings of ACL*, pages 16–23.

Joseph Fleiss. 1981. *Statistical Methods for Rates and Proportions (2nd Ed.)*. John Wiley & Sons, New York.

Miriam Goldberg. 1999. An unsupervised model for statistically determining coordinate phrase attachment. In *Proceedings of ACL*.

Donald Hindle and Mats Rooth. 1993. Structural ambiguity and lexical relations. *Computational Linguistics*, 19(1):103–120.

Sadao Kurohashi and Makoto Nagao. 1992. Dynamic programming method for analyzing conjunctive structures in japanese. In *Proceedings of COLING*, volume 1.

Mirella Lapata and Frank Keller. 2004. The Web as a baseline: Evaluating the performance of unsupervised Web-based models for a range of NLP tasks. In *Proceedings of HLT-NAACL*, pages 121–128, Boston.

Mirella Lapata and Frank Keller. 2005. Web-based models for natural language processing. *ACM Transactions on Speech and Language Processing*, 2:1–31.

Mitchell Marcus, Beatrice Santorini, and Mary Marcinkiewicz. 1994. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Preslav Nakov and Marti Hearst. 2005. Search engine statistics beyond the n-gram: Application to noun compound bracketing. In *Proceedings of CoNLL 2005*.

Patrick Pantel and Dekang Lin. 2000. An unsupervised approach to prepositional phrase attachment using contextually similar words. In *Proceedings of ACL*.

Adwait Ratnaparkhi, Jeff Reynar, and Salim Roukos. 1994. A maximum entropy model for prepositional phrase attachment. In *Proceedings of the ARPA Workshop on Human Language Technology.*, pages 250–255.

Adwait Ratnaparkhi. 1998. Statistical models for unsupervised prepositional phrase attachment. In *Proceedings of COLING-ACL*, volume 2, pages 1079–1085.

Philip Resnik. 1993. *Selection and information: a class-based approach to lexical relationships*. Ph.D. thesis, University of Pennsylvania, UMI Order No. GAX94-13894.

Philip Resnik. 1999. Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language. *JAIR*, 11:95–130.

Vasile Rus, Dan Moldovan, and Orest Bolohan. 2002. Bracketing compound nouns for logic form derivation. In Susan M. Haller and Gene Simmons, editors, *FLAIRS Conference*. AAAI Press.

Jiri Stetina and Makoto. 1997. Corpus based PP attachment ambiguity resolution with a semantic dictionary. In *Proceedings of WVLC*, pages 66–80.

Kristina Toutanova, Christopher D. Manning, and Andrew Y. Ng. 2004. Learning random walk models for inducing word dependency distributions. In *Proceedings of ICML*.

Martin Volk. 2000. Scaling up. using the WWW to resolve PP attachment ambiguities. In *Proceedings of Konvens-2000. Sprachkommunikation*.

Martin Volk. 2001. Exploiting the WWW as a corpus to resolve PP attachment ambiguities. In *Proc. of Corpus Linguistics*.