Issues in Synchronizing the English Treebank and PropBank

Olga Babko-Malaya[a], Ann Bies[a], Ann Taylor[b], Szuting Yi[a], Martha Palmer[c], Mitch Marcus[a], Seth Kulick[a] and Libin Shen[a]
[a]University of Pennsylvania, [b]University of York, [c]University of Colorado
{malayao,bies}@ldc.upenn.edu, {szuting,mitch,skulick,libin}@linc.cis.upenn.edu, at9@york.ac.uk, Martha.Palmer@colorado.edu

Abstract

The PropBank primarily adds semantic role labels to the syntactic constituents in the parsed trees of the Treebank. The goal is for automatic semantic role labeling to be able to use the domain of locality of a predicate in order to find its arguments. In principle, this is exactly what is wanted, but in practice the PropBank annotators often make choices that do not actually conform to the Treebank parses. As a result, the syntactic features extracted by automatic semantic role labeling systems are often inconsistent and contradictory. This paper discusses in detail the types of mismatches between the syntactic bracketing and the semantic role labeling that can be found, and our plans for reconciling them.

1 Introduction

The PropBank corpus annotates the entire Penn Treebank with predicate argument structures by adding semantic role labels to the syntactic constituents of the Penn Treebank. Theoretically, it is straightforward for PropBank annotators to locate possible arguments based on the syntactic structure given by a parse tree, and mark the located constituent with its argument label. We would expect a one-to-one mapping between syntactic constituents and semantic arguments. However, in practice, PropBank annotators often make choices that do not actually conform to the Penn Treebank parses.

The discrepancies between the PropBank and the Penn Treebank obstruct the study of the syntax and semantics interface and pose an immediate problem to an automatic semantic role labeling system. A semantic role labeling system is trained on many syntactic features extracted from the parse trees, and the discrepancies make the training data inconsistent and contradictory. In this paper we discuss in detail the types of mismatches between the syntactic bracketing and the semantic role labeling that can be found, and our plans for reconciling them. We also investigate the sources of the disagreements, which types of disagreements can be resolved automatically, which types require manual adjudication, and for which types an agreement between syntactic and semantic representations cannot be reached.

1.1 Treebank

The Penn Treebank annotates text for syntactic structure, including syntactic argument structure and rough semantic information. Treebank annotation involves two tasks: part-of-speech tagging and syntactic annotation.

The first task is to provide a part-of-speech tag for every token. Particularly relevant for PropBank work, verbs in any form (active, passive, gerund, infinitive, etc.) are marked with a verbal part of speech (VBP, VBN, VBG, VB, etc.). (Marcus, et al. 1993; Santorini 1990)

The syntactic annotation task consists of marking constituent boundaries, inserting empty categories (traces of movement, PRO, pro), showing the relationships between constituents (argument/adjunct structures), and specifying a particular subset of adverbial roles. (Marcus, et al. 1994; Bies, et al. 1995)

Constituent boundaries are shown through syntactic node labels in the trees. In the simplest case, a node will contain an entire constituent, complete with any associated arguments or modifiers. However, in structures involving syntactic movement, sub-constituents may be displaced. In these cases, Treebank annotation represents the original position with a trace and shows the relationship as co-indexing. In (1) below, for example, the direct object of *entail* is shown with the trace *T*, which is coindexed to the WHNP node of the question word *what*.

```
(1) (SBARQ (WHNP-1 (WP What ))
        (SQ (VBZ does )
            (NP-SBJ (JJ industrial )
                    (NN emigration ))
            (VP (VB entail)
                (NP *T*-1)))
        (. ?))
```

In (2), the relative clause modifying *a journalist* has been separated from that NP by the prepositional phrase *to al Riyadh*, which is an argument of the verb *sent*. The position where the relative clause originated or “belongs” is shown by the trace *ICH*, which is coindexed to the SBAR node containing the relative clause constituent.

```
(2)(S (NP-SBJ You)
   (VP sent
       (NP (NP a journalist)
           (SBAR *ICH*-2))
       (PP-DIR to
               (NP al Riyadh))
       (SBAR-2
         (WHNP-3 who)
         (S (NP-SBJ *T*-3)
            (VP served
               (NP (NP the name)
                   (PP of
                     (NP Lebanon)))
               (ADVP-MNR
                 magnificently))))))
```

Empty subjects which are not traces of movement, such as PRO and pro, are shown as * (see the null subject of the infinite clause in (4) below). These null subjects are coindexed with a governing NP if the syntax allows. The null subject of an infinitive clause complement to a noun is, however, *not* coindexed with another node in the tree in the syntax. This coindexing is shown as a semantic coindexing in the PropBank annotation.

The distinction between syntactic arguments and adjuncts of the verb or verb phrase is made through the use of functional dashtags rather than with a structural difference. Both arguments and adjuncts are children of the VP node. No distinction is made between VP-level modification and S-level modification. All constituents that appear before the verb are children of S and sisters of VP; all constituents that appear after the verb are children of VP.

Syntactic arguments of the verb are NP-SBJ, NP (no dashtag), SBAR (either –NOM-SBJ or no dashtag), S (either –NOM-SBJ or no dashtag), -DTV, -CLR (closely/clearly related), -DIR with directional verbs.

Adjuncts or modifiers of the verb or sentence are any constituent with any other adverbial dashtag, PP (no dashtag), ADVP (no dashtag). Adverbial constituents are marked with a more specific functional dashtag if they belong to one of the more specific types in the annotation system (temporal –TMP, locative –LOC, manner –MNR, purpose –PRP, etc.).

Inside NPs, the argument/adjunct distinction is shown structurally. Argument constituents (S and SBAR only) are children of NP, sister to the head noun. Adjunct constituents are sister to the NP that contains the head noun, child of the NP that contains both:

```
(NP (NP head)
    (PP adjunct))
```

1.2 PropBank

PropBank is an annotation of predicate-argument structures on top of syntactically parsed, or Treebanked, structures. (Palmer, et al. 2005; Babko-Malaya, 2005). More specifically, PropBank annotation involves three tasks: argument labeling, annotation of modifiers, and creating co-reference chains for empty categories.

The first goal is to provide consistent argument labels across different syntactic realizations of the same verb, as in

(3) [ARG0 John] broke [ARG1 the window]
[ARG1 The window] broke.

As this example shows, semantic arguments are tagged with numbered argument labels, such as Arg0, Arg1, Arg2, where these labels are defined on a verb by verb basis.

The second task of the PropBank annotation involves assigning functional tags to all modifiers of the verb, such as MNR (manner), LOC (locative), TMP (temporal), DIS (discourse connectives), PRP (purpose) or DIR (direction) and others.

And, finally, PropBank annotation involves finding antecedents for ‘empty’ arguments of the verbs, as in (4). The subject of the verb *leave* in this example is represented as an empty category [*] in Treebank. In PropBank, all empty categories which could be co-referred with a NP within the same sentence are linked in ‘co-reference’ chains:

(4) I made a decision [*] to leave

Rel: leave,
Arg0: [*] -> I

As the following sections show, all three tasks of PropBank annotation result in structures which differ in certain respects from the corresponding Treebank structures. Section 2 presents our approach to reconciling the differences between Treebank and PropBank with respect to the third task, which links empty categories with their antecedents. Section 3 introduces mismatches between syntactic constituency in Treebank and PropBank. Mismatches between modifier labels are not addressed in this paper and are left for future work.

2 Coreference and syntactic chains

PropBank chains include all syntactic chains (represented in the Treebank) plus other cases of nominal semantic coreference, including those in which the coreferring NP is not a syntactic antecedent. For example, according to PropBank guidelines, if a trace is coindexed with a NP in Treebank, then the chain should be reconstructed:

(5) What-1 do you like [*T*-1]?

Original PropBank annotation:
Rel: like
Arg0: you
Arg1: [*T*] -> What

Such chains usually include traces of A and A' movement and PRO for subject and object control. On the other hand, not all instances of PROs have syntactic antecedents. As the following example illustrates, subjects of infinitival verbs and gerunds might have antecedents within the same sentence, which cannot be linked as a syntactic chain.

(6) On the issue of abortion , Marshall Coleman wants to take away your right [*] to choose and give it to the politicians .

ARG0:	[*] -> your
REL:	choose

Given that the goal of PropBank is to find all semantic arguments of the verbs, the links between empty categories and their coreferring NPs are important, independent of whether they are syntactically coindexed or not. In order to reconcile the differences between Treebank and PropBank annotations, we decided to revise PropBank annotation and view it as a 3 stage process.

First, PropBank annotators should not reconstruct syntactic chains, but rather tag empty categories as arguments. For example, under the new approach annotators would simply tag the trace as the Arg1 argument in (7):

(7) What-1 do you like [*T*-1]?

Revised PropBank annotation:
Rel: like
Arg0: you
Arg1: [*T*]

As the second stage, syntactic chains will be reconstructed automatically, based on the coindexation provided by Treebank (note that the trace is coindexed with the NP *What* in (7)). And, finally, coreference annotation will be done on top of the resulting resource, with the goal of finding antecedents for the remaining empty categories, including empty subjects of infinitival verbs and gerunds.

One of the advantages of this approach is that it allows us to distinguish different types of chains, such as syntactic chains (i.e., chains which are derived as the result of syntactic movement, or control coreference), direct coreference chains (as illustrated by the example in (6)), and semantic type links for other 'indirect' types of links between an empty category and its antecedent.

Syntactic chains are annotated in Treebank, and are reconstructed automatically in PropBank. The annotation of direct coreference chains is done manually on top of Treebank, and is restricted to empty categories that are not coindexed with any NP in Treebank. And, finally, as we show next, a semantic type link is used for relative clauses and a coindex link for verbs of saying.

A semantic type link is used when the antecedent and the empty category do not refer to the same entity, but do have a certain kind of relationship. For example, consider the relative clause in (8):

(8) Answers that we'd like to have

Treebank annotation:
```
(NP (NP answers)
    (SBAR (WHNP-6 which)
          (S (NP-SBJ-3 we)
             (VP 'd
                (VP like
                  (S (NP-SBJ *-3)
                     (VP to
                       (VP have
                         (NP  *T*-6)
    ))))))))
```

In Treebank, the object of the verb *have* is a trace, which is coindexed with the relative pronoun. In the original PropBank annotation, a further link is provided, which specifies the relative pronoun as being of "semantic type" *answers*.

(9) *Original PropBank annotation:*

Arg1:	[NP *T*-6] -> which -> answers
rel:	have
Arg0:	[NP-SBJ *-3] -> we

This additional link between *which* and *answer*s is important for many applications that make use of preferences for semantic types of verb arguments, such as Word Sense Disambiguation (Chen & Palmer 2005). In the new annotation scheme, annotators will first label traces as arguments:

(10) *Revised PropBank annotation (stage 1):*

Rel:	have
Arg1:	[*T*-6]
Arg0:	[NP-SBJ *-3]

As the next stage, the trace [*T*-6] will be linked to the relative pronoun automatically (in addition to the chain *[NP-SBJ *-3] -> we* being automatically reconstructed). As the third stage, PropBank annotators will link *which* to *answers*. However, this chain will be labeled as a "semantic type" to distinguish it from direct coreference chains and to indicate that there is no identity relation between the coindexed elements.

Verbs of saying illustrate another case of links rather than coreference chains. In many sentences with direct speech, the clause which introduces a verb of saying is 'embedded' into the utterance. Syntactically this presents a problem for both Treebank and Propbank annotation. In Treebank, the original annotation style required a trace coindexed to the highest S node as the argument of the verb of saying, indicating syntactic movement.

(11) Among other things, they said [*T*-1] , Mr. Azoff would develop musical acts for a new record label .

Treebank annotation:
```
(S-1 (PP Among
        (NP other things))
    (PRN ,
      (S (NP-SBJ they)
         (VP said
            (SBAR 0
              (S *T*-1))))
         ,)
    (NP-SBJ Mr. Azoff)
    (VP would
      (VP develop
        (NP (NP musical acts)
            (PP for
               (NP a new record
                   label)))))
    .)
```
In PropBank, the different pieces of the utterance, including the trace under the verb *said*, were concatenated

(12) *Original PropBank annotation:*

ARG1: [Among other things] [Mr. Azoff] [would develop musical acts for a new record label] [[*T*-1]]
ARG0: they
rel: said

Under the new approach, in stage one, Treebank annotation will introduce not a trace of the S clause, but rather *?*, an empty category indicating ellipsis. In stage three, PropBank annotators will link this null element to the S node, but the resulting chain will not be viewed as 'direct' coreference. A special tag will be used for this link, in order to distinguish it from other types of chains.

(13) *Revised PropBank annotation:*

ARG1:	[*?*] (-> S)
ARG0:	they
rel:	said

3 Differences in syntactic constituency

3.1 Extractions of mismatches between PropBank and Treebank

In order to make the necessary changes to both the Treebank and the PropBank, we have to first find all instances of mismatches. We have used two methods to do this: 1) examining the argument locations; 2) examining the discontinuous arguments.

Argument Locations In a parse tree which expresses the syntactic structure of a sentence, a semantic argument occupies specific syntactic locations: it appears in a subject position, a verb complement location or an adjunct location. Relative to the predicate, its argument is either a sister node, or a sister node of the predicate's ancestor. We extracted cases of PropBank arguments which do not attach to the predicate spine, and filtered out VP coordination cases. For example, the following case is a problematic one because the argument PP node is embedded too deeply in an NP node and hence it cannot find a connection with the main predicate verb *lifted*. This is an example of a PropBank annotation error.

```
(14) (VP (VBD[rel] lifted)
         (NP us) )
         (NP-EXT
            (NP a good 12-inches)
            (PP-LOC[ARGM-LOC] above
               (NP the water level))))
```

However, the following case is not problematic because we consider the ArgM PP to be a sister node of the predicate verb given the VP coordination structure:

```
(15) (VP (VP (VB[rel] buy)
           (NP the basket of … )
           (PP in whichever market …))
         (CC and)
         (VP (VBP sell)
           (NP them)
           (PP[ARGM] in the more
                 expensive market)))
```

Discontinuous Arguments happen when PropBank annotators need to concatenate several Treebank constituents to form an argument. Discontinuous arguments often represent different opinions between PropBank and Treebank annotators regarding the interpretations of the sentence structure.

For example, in the following case, the PropBank concatenates the NP and the PP to be the Arg1. In this case, the disagreement on PP attachment is simply a Treebank annotation error.

(16) The region lacks necessary mechanisms for handling the aid and accounting items.

Treebank annotation:
```
(VP lacks
    (NP necessary mechanisms)
    (PP for
        (NP handing the aid…)))
```

PropBank annotation:
REL: lacks
Arg1: [NP necessary mechanisms][PP for handling the aid and accounting items]

All of these examples have been classified into the following categories: (1) attachment ambiguities, (2) different policy decisions, and (3) cases where one-to-one mapping cannot be preserved.

3.2 Attachment ambiguities

Many cases of mismatches between Treebank and PropBank constituents are the result of ambiguous interpretations. The most common examples are cases of modifier attachment ambiguities, including PP attachment. In cases of ambiguous interpretations, we are trying to separate cases which can be resolved automatically from those which require manual adjudication.

PP-Attachment The most typical case of PP attachment annotation disagreement is shown in (17).

(17) *She wrote a letter for Mary.*

Treebank annotation:
```
(VP wrote
    (NP (NP a letter)
        (PP for
            (NP Mary))))
```

PropBank annotation:
REL: write
Arg1: a letter
Arg2: for Mary

In (17), the PP ‘for Mary’ is attached to the verb in PropBank and to the NP in Treebank. This disagreement may have been influenced by the set of roles of the verb ‘write’, which includes a beneficiary as its argument.

(18) Frameset write:	Arg0: writer
	Arg1: thing written
	Arg2: beneficiary

Examples of this type cannot be automatically resolved and require manual adjudication.

Adverb Attachment Some cases of modifier attachment ambiguities, on the other hand, could be automatically resolved. Many cases of mismatches are of the type shown in (19), where a directional adverbial follows the verb. In Treebank, this adverbial is analyzed as part of an ADVP which is the argument of the verb in question. However, in PropBank, it is annotated as a separate ArgM-DIR.

(19) Everything is going back to Korea or Japan.

Treebank annotation:
```
(S (NP-SBJ (NN Everything) )
   (VP (VBZ is)
       (VP (VBG[rel] going)
           (ADVP-DIR
                (RB[ARGM-DIR] back)
                (PP[ARG2] (TO to)
                    (NP (NNP Korea)
                        (CC and)
                        (NNP Japan)
       ))))) (. .))
```

Original PropBank annotation:
Rel: going
ArgM-DIR: back
Arg2: to Korea or Japan

For examples of this type, we have decided to automatically reconcile PropBank annotations to be consistent with Treebank, as shown in (20).

(20) *Revised PropBank annotation:*
Rel: going
Arg2: back to Korea or Japan

3.3 Sentential complements

Another area of significant mismatch between Treebank and PropBank annotation involves sentential complements, both infinitival clauses and small clauses. In general, Treebank annotation allows many more verbs to take sentential complements than PropBank annotation.

For example, the Treebank annotation of the sentence in (21) gives the verb *keep* a sentential complement which has *their markets active* under the S as the subject of the complement clause. PropBank annotation, on the other hand, does not mark the clause but rather labels each subconstituent as a separate argument.

(21) …keep their markets active

Treebank annotation:
```
(VP keep
    (S (NP-SBJ their markets)
       (ADJP-PRD active)))
```

PropBank annotation:
REL: keep
Arg1: their markets
Arg2: active

In Propbank, an important criterion for deciding whether a verb takes an S argument, or decomposes it into two arguments (usually tagged as Arg1 and Arg2) is based on the semantic interpretation of the argument, e.g. whether the argument can be interpreted as an event or proposition.

For example, causative verbs (e.g. *make, get*), verbs of perception (*see, hear*), and intensional verbs (*want, need, believe*), among others, are analyzed as taking an S clause, which is interpreted as an event in the case of causative verbs and verbs of perception, and as a proposition in the case of intensional verbs. On the other hand, 'label' verbs (*name, call, entitle, label*, etc.), do not select for an event or proposition and are analyzed as having 3 arguments: Arg0, Arg1, and Arg2.

Treebank criteria for distinguishing arguments, on the other hand, were based on syntactic considerations, which did not always match with Propbank. For example, in Treebank, evidence of the syntactic category of argument that a verb can take is used as part of the decision process about whether to allow the verb to take a small clause. Verbs that take finite or non-finite (verbal) clausal arguments, are also treated as taking small clauses. The verb *find* takes a finite clausal complement as in *We found that the book was important* and also a non-finite clausal complement as in *We found the book to be important*. Therefore, *find* is also treated as taking a small clause complement as in *We found the book important*.

(22)
```
(S (NP-SBJ We)
   (VP found
       (S (NP-SBJ the book)
          (ADJP-PRD important))))
```

The obligatory nature of the secondary predicate in this construction also informed the decision to use a small clause with a verb like *find*. In (22), for example, *important* is an obligatory part of the sentence, and removing it makes the sentence ungrammatical with this sense of *find* ("We found the book" can only be grammatical with a different sense of *find*, essentially "We located the book").

With verbs that take infinitival clausal complements, however, the distinction between a single S argument and an NP object together with an S argument is more difficult to make. The original Treebank policy was to follow the criteria and the list of verbs taking both an NP object and an infinitival S argument given in Quirk, et al. (1985).

Resultative constructions are frequently a source of mismatch between Treebank annotation as a small clause and PropBank annotation with Arg1 and Arg2. Treebank treated a number of resultative as small clauses, although certain verbs received resultative structure annotation, such as the one in (23).

```
(23) (S (NP-SBJ They)
       (VP painted
           (NP-1 the apartment)
           (S-CLR (NP-SBJ *-1)
                  (ADJP-PRD orange))))
```

In all the mismatches in the area of sentential complementation, Treebank policy tends to overgeneralize S-clauses, whereas Propbank leans toward breaking down clauses into separate arguments.

This type of mismatch is being resolved on a verb-by-verb basis. Propbank will reanalyze some of the verbs (like *consider* and *find*), which have been analyzed as having 3 arguments, as taking an S argument. Treebank, on the other hand, will change the analysis of *label* verbs like *call*, from a small clause analysis to a structure with two complements.

Our proposed structure for *label* verbs, for example, is in (24).

```
(24) (S (NP-SBJ[Arg0] his parents)
       (VP (VBD called)
           (NP-1[Arg1] him)
           (S-CLR[Arg2]
                (NP-SBJ *-1)
                (NP-PRD John))))
```

This structure will accommodate both Treebank and PropBank requirements for *label* verbs.

4 Where Syntax and Semantics do not match

Finally, there are some examples where the differences seem to be impossible to resolve without sacrificing some important features of PropBank or Treebank annotation.

4.1 Phrasal verbs

PropBank has around 550 phrasal verbs like *keep up, touch on, used to* and others, which are analyzed as separate predicates in PropBank. These verbs have their own set of semantic roles, which is different from the set of roles of the corresponding 'non-phrasal' verbs, and therefore they require a separate PropBank entry. In Treebank, on the other hand, phrasal verbs are not distinguished. If the second part of the phrasal verb is labeled as a verb+particle combination in the Treebank, the PropBank annotators concatenate it with the verb as the REL. If Treebank labels the second part of the 'phrasal verb' as part of a prepositional phrase, there is no way to resolve the inconsistency.

(25) But Japanese institutional investors are used to quarterly or semiannual payments on their investments, so …

Treebank annotation:

```
(VBN used)
(PP (TO to)
    (NP quarterly or …
        on their investments))
```

PropBank annotation:

Arg1: quarterly or … on their investments
Rel: used to ('used to' is a separate predicate in PropBank)

4.2 Conjunction

In PropBank, conjoined NPs and clauses are usually analyzed as one argument, parallel to Treebank. For example, in *John and Mary came*, the NP *John and Mary* is a constituent in Treebank and it is also marked as Arg0 in PropBank. However, there are a few cases where one of the conjuncts is modified, and PropBank policy is to mark these modifiers as ArgMs. For example, in the following NP, the temporal ArgM *now* modifies a verb, but it only applies to the second conjunct.

(26)
```
(NP (NNP Richard)
    (NNP Thornburgh) )
(, ,)
(SBAR
  (WHNP-164 (WP who))
  (S
    (NP-SBJ-1 (-NONE- *T*-164))
    (VP
      (VBD went)
      (PRT (RP on) )
      (S
        (NP-SBJ (-NONE- *-1))
        (VP (TO to)
          (VP (VB[rel] become)
            (NP-PRD
              (NP[ARG2]
                (NP (NN governor))
                (PP (IN of)
                 (NP
                  (NNP
                   Pennsylvania))))
(CC and)
(PRN (, ,)
  (ADVP-TMP (RB now))
     (, ,) )
(NP[ARG2] (NNP U.S.)
       (NNP Attorney)
       (NNP General))
)))))))
```

In PropBank, cases like this can be decomposed into two propositions:

(27) Prop1: rel: become
Arg1: attorney general
Arg0: [-NONE- *-1]

Prop2: rel: become
ArgM-TMP: now
Arg0: [-NONE- *-1]
Arg1: a governor

In Treebank, the conjoined NP is necessarily analyzed as one constituent. In order to maintain the one-to-one mapping between PropBank and Treebank, PropBank annotation would have to be revised in order to allow the sentence to have one proposition with a conjoined phrase as an argument. Fortunately, these types of cases do not occur frequently in the corpus.

4.3 Gapping

Another place where the one-to-one mapping is difficult to preserve is with gapping constructions. Treebank annotation does not annotate the gap, given that gaps might correspond to different syntactic categories or may not even be a constituent. The policy of Treebank, therefore, is simply to provide a coindexation link between the corresponding constituents:

(28) Mary-1 likes chocolates-2 and
Jane=1 – flowers=2

This policy obviously presents a problem for one-to-one mapping, since Propbank annotators tag *Jane* and *flowers* as the arguments of an implied second *likes* relation, which is not present in the sentence.

5 Summary

In this paper we have considered several types of mismatches between the annotations of the English Treebank and the PropBank: coreference and syntactic chains, differences in syntactic constituency, and cases in which syntax and semantics do not match. We have found that for the most part, such mismatches arise because Treebank decisions are based primarily on syntactic considerations while PropBank decisions give more weight to semantic representation..

In order to reconcile these differences we have revised the annotation policies of both the PropBank and Treebank in appropriate ways. A fourth source of mismatches is simply annotation error in either the Treebank or PropBank. Looking at the mismatches in general has allowed us to find these errors, and will facilitate their correction.

References

Olga Babko-Malaya. 2005. *PropBank Annotation Guidelines.* http://www.cis.upenn.edu/~mpalmer/project_pages/PBguidelines.pdf

Ann Bies, Mark Ferguson, Karen Katz, Robert MacIntyre. 1995. *Bracketing Guidelines for Treebank II Style*. Penn Treebank Project, University of Pennsylvania, Department of Computer and Information Science Technical Report MS-CIS-95-06.

Jinying Chen and Martha Palmer. 2005. Towards Robust High Performance Word Sense Disambiguation of English Verbs Using Rich Linguistic Features. In *Proceedings of the 2nd International Joint Conference on Natural Language Processing, IJCNLP2005*, pp. 933-944. Oct. 11-13, Jeju Island, Republic of Korea.

M. Marcus, G. Kim, M. Marcinkiewicz, R. MacIntyre, A. Bies, M. Ferguson, K. Katz & B. Schasberger, 1994. The Penn Treebank: Annotating predicate argument structure. *Proceedings of the Human Language Technology Workshop*, San Francisco.

M. Marcus, B. Santorini and M.A. Marcinkiewicz, 1993. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics.*

Martha Palmer, Dan Gildea, and Paul Kingsbury. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics,* 31(1).

R. Quirk, S. Greenbaum, G. Leech and J. Svartvik. 1985. *A Comprehensive Grammar of the English Language*. Longman, London.

B. Santorini. 1990. *Part-of-speech tagging guidelines for the Penn Treebank Project.* University of Pennsylvania, Department of Computer and Information Science Technical Report MS-CIS-90-47.