What a Parser can Learn from a Semantic Role Labeler and *Vice Versa*

Stephen A. Boxwell, Dennis N. Mehay, Chris Brew
The Ohio State University
{boxwell, mehay, cbrew}@ling.ohio-state.edu

Abstract

In many NLP systems, there is a unidirectional flow of information in which a parser supplies input to a semantic role labeler. In this paper, we build a system that allows information to flow in both directions. We make use of semantic role predictions in choosing a single-best parse. This process relies on an averaged perceptron model to distinguish likely semantic roles from erroneous ones. Our system penalizes parses that give rise to low-scoring semantic roles. To explore the consequences of this we perform two experiments. First, we use a baseline generative model to produce n-best parses, which are then re-ordered by our semantic model. Second, we use a modified version of our semantic role labeler to predict semantic roles at parse time. The performance of this modified labeler is weaker than that of our best full SRL, because it is restricted to features that can be computed directly from the parser's packed chart. For both experiments, the resulting semantic predictions are then used to select parses. Finally, we feed the selected parses produced by each experiment to the full version of our semantic role labeler. We find that SRL performance can be improved over this baseline by selecting parses with likely semantic roles.

1 Introduction

In the semantic role labeling task, words or groups of words are described in terms of their relations to a predicate. For example, the sentence *Robin admires Leslie* has two semantic role-bearing words: *Robin* is the agent or experiencer of the *admire* predicate, and *Leslie* is the patient. These semantic relations are distinct from syntactic relations like subject and object – the proper nouns in the sentence *Leslie is admired by Robin* have the same semantic relationships as *Robin admires Leslie*, even though the syntax differs.

Although syntax and semantics do not always align with each other, they are correlated. Almost all automatic semantic role labeling systems take a syntactic representation of a sentence (taken from an automatic parser or a human annotator), and use the syntactic information to predict semantic roles. When a semantic role labeler predicts an incorrect role, it is often due to an error in the parse tree. Consider the erroneously annotated sentence from the Penn Treebank corpus shown in Figure 1. If a semantic role labeling system relies heavily upon syntactic attachment decisions, then it will likely predict that *in 1956* describes the time that asbestos was used, rather than when it ceased to be used.

Errors of this kind are common in treebanks and in automatic parses. It is telling, though, that while the hand-annotated Penn Treebank (Marcus et al., 1993), the Charniak parser (Charniak, 2001), and the C&C parser (Clark and Curran, 2004) all produce the erroneous parse from Figure 1, the hand-annotated Propbank corpus of verbal semantic roles (Palmer et al., 2005) correctly identifies *in 1956* as a temporal modifier of *stopped*, rather than *using*. This demonstrates that while syntactic attachment decisions like these are difficult for humans and for automatic parsers, a human reader has little difficulty identifying the correct semantic relationship between the temporal modifier and the verbs. This is likely due to the fact that the meaning suggested by the parse in Figure 1 is unlikely – the reader instinctively feels that a temporal modifier fits better with the verb *stop* than with the verb *use*.

In this paper, we will use the idea that semantic roles predicted by correct parses are more natural than semantic roles predicted by erroneous parses. By modifying a state-of-the-art CCG semantic role labeler to predict semantic roles at parse time, or by using it to select from an n-best list, we can prefer analyses that yield likely semantic roles. Syntactic analysis is treated not as an autonomous task, but rather as a contributor to the final goal of semantic role labeling.

2 Related Work

There has been a great deal of work in joint parsing and semantic role labeling in recent years. Two notable efforts have been the CoNLL 2008 and 2009 shared tasks

Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, pages 736–744,
MIT, Massachusetts, USA, 9-11 October 2010.

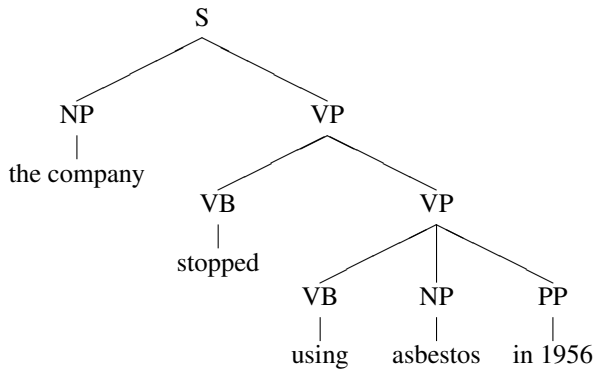

Figure 1: A parse tree based on the treebank parse of wsj_0003.3. Notice that the temporal adjunct is erroneously attached low. In a syntax-based SRL system, this will likely lead to a role prediction error.

(Surdeanu et al., 2008; Hajič et al., 2009). Many of these systems perform joint syntactic and semantic analysis by generating an n-best list of syntactic parses, labeling semantic roles on all of them, then re-ranking these parses by some means. Our approach differs from this strategy by abandoning the preliminary ranking and predicting semantic roles at parse time. By doing this, we effectively open semantic roles in the entire parse forest to examination by the ranking model, rather than restricting the model to an n-best list generated by a baseline parser. The spirit of this work more closely resembles that of Finkel and Manning (2009) , which improves both parsing and named entity recognition by combining the two tasks.

3 Why Predicting Semantic Roles in a Packed Chart is Difficult

Predicting semantic roles in the environment of a packed chart is difficult when using an atomic CFG. In order to achieve the polynomial efficiency appropriate for wide-coverage parsing, it is necessary to "pack" the chart – that is, to combine distinct analyses of a given span of words that produce the same category. The only other widely used option for wide-coverage parsing is to use beam search with a narrow beam, which runs the risk of search errors. On methodological grounds we prefer an exhaustive search, since systems that rely heavily on heuristics for their efficiency are difficult to understand, debug or improve. It is straightforward to read off the highest scoring parse from a packed chart, and similarly routine to generate an n-best list containing a highly-ranked subset of the parses. However, a packed chart built on an atomic CFG does not make available all of the features that are important to many CFG-based SRL systems. In particular, the very useful treepath feature, which lists the categories touched by walking the tree from the predicate to the target word, only makes sense when you have a complete tree, so cannot easily be computed from the chart (Figure 2). Chart edges can

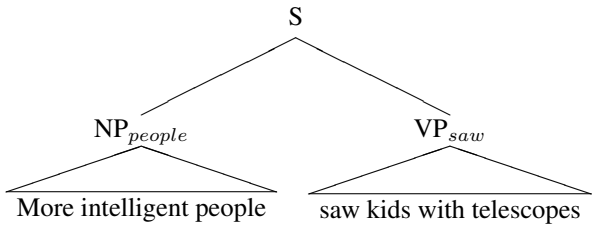

Figure 2: In the context of a packed chart, it is meaningless to speak of a treepath between *saw* and *people* because multiple analyses are "packed" under a single category.

be lexicalized with their headwords, and this information would be useful in role labeling – but even this misses vital subcategorization information that would be available in the complete parse. An ideal formalism for our purpose would condense into the category label a wide range of information about combinatory potential, heads, and syntactic dependencies. At the same time it should allow the creation of a packed chart, come with labeled training data, and have a high-quality parser and semantic role labeler already available. Fortunately, Combinatory Categorial Grammar offers these desiderata, so this is our formalism of choice.

4 Combinatory Categorial Grammar

Combinatory Categorial Grammar (Steedman, 2000) is a grammar formalism that describes words in terms of their combinatory potential. For example, determiners belong to the category np/n, or "the category of words that become noun phrases when combined with a noun to the right". The rightmost category indicates the argument that the category is seeking, the leftmost category indicates the result of combining this category with its argument, and the slash (/ or \) indicates the direction of combination. Categories can be nested within each other: a transitive verb like *devoured* belongs to the category

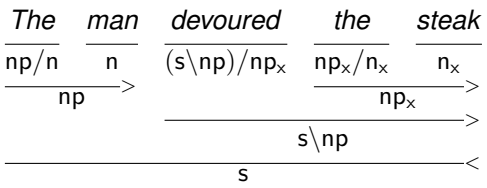

Figure 3: A simple CCG derivation.

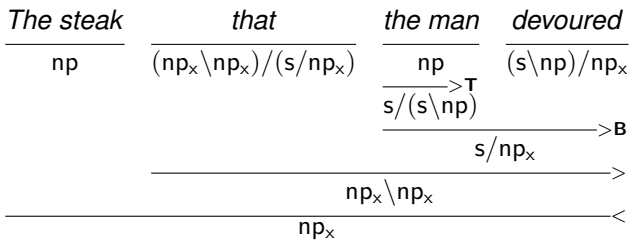

Figure 4: An example of CCG's treatment of relative clauses. The syntactic dependency between *devoured* and *steak* is the same as it was in figure 3. Co-indexations (the '*x*s') have been added here and above to aid the eye in following the relevant [*devoured-steak*] dependency.

(s\np)/np, or "the category that would become a sentence if it could combine with a noun phrase to the right and another noun phrase to the left". An example of how categories combine to make sentences is shown in Figure 3. CCG has many capabilities that go beyond that of a typical context-free grammar. First, it has a sophisticated internal system of managing syntactic heads and dependencies[1]. These dependencies are used to great effect in CCG-based semantic role labeling systems (Gildea and Hockenmaier, 2003; Boxwell et al., 2009), as they do not suffer the same data-sparsity effects encounted with treepath features in CFG-based SRL systems. Secondly, CCG permits these dependencies to be passed through intermediary categories in grammatical structures like relative clauses. In Figure 4, *the steak* is still in the object relation to *devoured*, even though the verb is inside a relative clause. Finally and most importantly, *these dependencies are represented directly on the CCG categories themselves*. This is what makes CCG resistant to the problem described in Section 3 – because the dependency is formed when the two heads combine, it is available to be used as a local feature by the semantic role labeler.

[1]A complete explanation of CCG predicate-argument dependencies can be found in the CCGbank user manual (Hockenmaier and Steedman, 2005)

5 Semantic Role Labeling

We use a modified version of the Brutus semantic role labeling system (Boxwell et al., 2009)[2]. The original version of this system takes complete CCG derivations as input, and predicts semantic roles over them. For our purposes, however, it is necessary to modify the system to make semantic predictions at parse time, inside a packed chart, before the complete derivation is available. For this reason, it is necessary to remove the global features from the system (that is, features that rely on the complete parse), leaving only local features (features that are known at the moment that the predicate is attached to the argument). Crucially, dependency features count as "local" features, even though they have the potential to connect words that are very far apart in the sentence.

Brutus is arranged in a two-stage pipeline. First, a maximum entropy classifier[3] predicts, for each predicate in turn, which words in the sentence are likely headwords of semantic roles. Then, a second maximum entropy classifier assigns role labels to each of these words. The features used in the identification model of the local-only version of Brutus are as follows:

- **Words.** A three-word window surrounding the candidate word. For example, if we were considering the word *steak* in Figure 3, the three features would be represented as word_-1=the, word_0=steak, and word_1=#, with the last feature representing an out-of-bounds index.

- **Predicate.** The predicate whose semantic roles the system is looking for. For example, the sentence in figure 3 contains one predicate: *devour*.

- **Syntactic Dependency.** As with a previous approach in CCG semantic role labeling (Gildea and Hockenmaier, 2003), this feature shows the exact nature of the syntactic dependency between the predicate and the word we are considering, if any such dependency exists. This feature is represented by the category of the predicate, the argument slot that this word fits into, and whether or not the predicate is the head of the resultant category, represented with a left or right arrow. In the example from figure 3, the relationship between *devoured* and *steak* would be represented as (s\np)/np.2.→.

The second maximum entropy classifier uses all of the features from the identifier, plus several more:

[2]Found at `http://www.ling.ohio-state.edu/~boxwell/software/brutus.html`

[3]Brutus uses Zhang Le's maxent toolkit, available at `http://homepages.inf.ed.ac.uk/s0450736/maxent_toolkit.html`.

Model	P	R	F
Local	89.8%	80.8%	85.1%
Global	89.8%	84.3%	87.0%

Table 1: SRL results for treebank parses, using the local model described in Section 5 and the full global model.

- **Before / After.** A binary indicator feature indicating whether the candidate word is before or after the predicate.
- **Result Category Detail.** This indicates the feature on the result category of the predicate. Possible values include dcl (for declarative sentences), pss (for passive sentences), ng (for present-progressive phrases like "running the race"), etc. These are read trivially off of the verbal category.
- **Argument Mapping.** An argument mapping is a prediction of a likely set of semantic roles for a given CCG predicate category. For example, a likely argument mapping for devoured:(s[dcl]\np)/np is [Arg0,Arg1]. These are predicted from string-level features, and are useful for bringing together otherwise independent classification decisions for individual roles. Boxwell et al. (2009) describe this feature in detail.

The Maximum-Entropy models were trained to 500 iterations. To prevent overfitting, we used Gaussian priors with global variances of 1 and 5 for the identifier and the labeler, respectively. Table 1 shows SRL performance for the local model described above, and the full global CCG-system described by Boxwell et al. (2009). We use the method for calculating the accuracy of Propbank verbal semantic roles described in the CoNLL-2008 shared task on semantic role labeling (Surdeanu et al., 2008). Because the Brutus SRL system is not designed to accommodate Nombank roles (Meyers et al., 2004), we restrict ourselves to predicting Propbank roles in the present work.

The local system has the same precision as the global one, but trails it on recall and F-measure. Note that this performance is achieved with gold standard parses.

6 Performing Semantic Role Predictions at Parse Time

Recall that the reasoning for using a substantially pared down version of the Brutus SRL system is to allow it to predict semantic roles in the context of a packed chart. Because we predict semantic roles for each constituent immediately after the constituent is formed and before it is added to the chart, we can use semantic roles to inform parsing. We use a CKY parsing algorithm, though this approach could be easily adapted to other parse strategies.

Whenever two constituents are combined, the SRL system checks to see if either of the constituents contains predicates. The system then attempts to identify semantic roles in the other constituent related to this predicate. This process repeats at every step, creating a combined syntax-semantics parse forest. Crucially, this allows us to use features derived from the semantic roles to rank parses inside the packed chart. This could result in an improvement over ranking completed parses, because re-ranking completed parses requires first generating an n-best list of parse candidates, potentially preventing the re-ranker from examining high value parses falling outside the n-best list.

In order to train our parse model, it is necessary to first employ a baseline parse model over the training set. The baseline model is a PCFG model, where the products of the probabilities of individual rule applications are used to rank candidate parses. We use a cross-fold validation technique to parse the training set (train on sections 02-20 to parse section 21, train on sections 02-19 and 21 to parse section 20, and so on). As we parse these sentences, we use the local SRL model described in Section 5 to predict semantic roles inside the packed chart. We then iterate over the packed chart and extract features based on the semantic roles in it, effectively learning from every possible semantic role in the parse forest. Notice that this does not require enumerating every parse in the forest (which would be prohibitively expensive) – the roles are labeled at parse time and can therefore be read directly from the packed chart. For each role in the packed chart, we label it as a "good" semantic role if it appears in the human-judged Propbank annotation for that sentence, and a "bad" semantic role if it does not.

The features extracted from the packed chart are as follows:

- **Role**. The semantic role itself, concatenated with the predicate. For example, play.Arg1. This will represent the intuition described in Section 1 that certain roles are more semantically appealing than others.
- **Role and Headword**. The semantic role concatenated with the predicate and the headword of the semantic role. This reflects the idea that certain words fit with particular roles better than others.

These features are used to train an averaged perceptron model to distinguish between likely and unlikely semantic roles. We incorporate the perceptron directly with the parser using a packed feature forest implementation, following an approach used by the current state-of-the-art CCG parser (Clark and Curran, 2004). By prefer-

ring sentences with good semantic roles, we hope to produce parses that give better overall semantic role predictions. The parser prefers spans with better semantic roles, and breaks ties that would have arisen using the baseline model alone. Similarly the baseline model can break ties between equivalent semantic roles; this has the added benefit of encouraging normal-form derivations in cases of spurious ambiguity. The result is a single-best complete parse with semantic roles already predicted. Once the single-best parse is selected, we allow the global SRL model to predict any additional roles over the parse, to catch those roles that are difficult to predict from local features alone.

7 Experiment 1: Choosing a Single-Best Derivation from an N-best List

Our first experiment demonstrates our model's performance in a ranking task. In this task, a list of candidate parses are generated by our baseline model. This baseline model treats rule applications as a PCFG – each rule application (say, np + s\np = s) is given a probability in the standard way. The rule probabilities are unsmoothed maximum likelihood estimates derived from rule counts in the training portion of CCGbank. After n-best derivations are produced by the baseline model, we use the Brutus semantic role labeler to assign roles to each candidate derivation. We vary the size of the n-best list from 1 to 10 (note that an n-best list of size 1 is equivalent to the single-best baseline parse). We then use the semantic model to re-rank the candidate parses and produce a single-best parse. The outcomes are shown in Table 2.

n	P	R	F
1	85.1	71.7	77.8
2	85.9	74.8	79.9
5	**84.5**	**76.8**	**80.5**
10	83.7	76.8	80.1
C&C	83.6	76.8	80.0

Table 2: SRL performance on the development set (section 00) for various values of n. The final row indicates SRL performance on section 00 parses from the Clark and Curran CCG parser.

The availability of even two candidate parses yields a 2.1% boost to the balanced F-measure. This is because the semantic role labeler is very sensitive to syntactic attachment decisions, and in many cases the set of rule applications used in the derivation are very similar or even the same. Consider the simplified version of a phenomenon found in wsj_0001.1 shown in Figures 5 and 6. The only difference in rule applications in these derivations is whether the temporal adjunct attaches to s[b]\np or s[dcl]\np. Because the s[dcl]\np case is slightly more

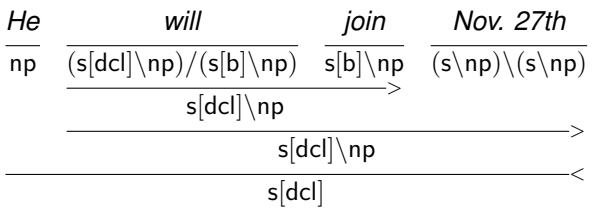

Figure 5: The single-best analysis for *He will join Nov 27th* according to the baseline model. Notice that the temporal adjunct is attached high, leading the semantic role labeler to fail to identify ArgM-TMP.

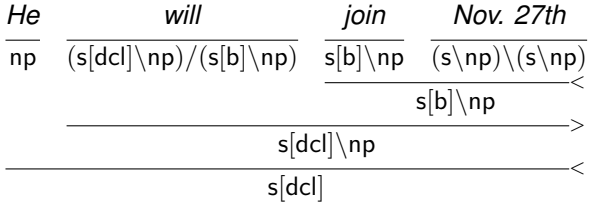

Figure 6: The second-best analysis of *He will join Nov 27th*. This analysis correctly predicts *Nov 27th* as the ArgM-TMP of *join*, and the semantic model correctly re-ranks this analysis to the single-best position.

common in the treebank, the baseline model identifies it as the single-best parse, and identifies the derivation in figure 6 as the second-best parse. The semantic model, however, correctly recognizes that the semantic roles predicted by the derivation in Figure 6 are superior to those predicted by the derivation in figure 5. This demonstrates how a second or third-best parse according to the baseline model can be greatly superior to the single-best in terms of semantics.

8 Experiment 2: Choosing a Single-Best Derivation Directly from the Packed Chart

One potential weakness with the n-best list approach described in Section 7 is choosing the size of the n-best list. As the length of the sentence grows, the number of candidate analyses grows. Because sentences in the treebank and in real-world applications are of varying length and complexity, restricting ourselves to an n-best list of a particular size opens us to considering some badly mangled derivations on short, simple sentences, and not enough derivations on long, complicated ones. One possible solution to this is to simply choose a single best derivation directly from the packed chart using the semantic model, eschewing the baseline model entirely except for breaking ties. In this approach, we use the local SRL model described in section 6 to predict semantic roles at parse time, inside the packed chart. This frees us from the

need to have a complete derivation (as in the n-best list approach in Section 7). We use the semantic model to choose a single-best parse from the packed chart, then we pass this complete parse through the global SRL model to give it all the benefits afforded to the parses in the n-best approach. The results for the semantic model compared to the baseline model are shown in table 3. Interestingly,

Model	P	R	F
Baseline	85.1	71.7	77.8
Semantic	82.7	70.5	76.1

Table 3: A comparison of the performance of the baseline model and the semantic model on semantic role labeling. The semantic model, when unrestrained by the baseline model, performs substantially worse.

the semantic model performs considerably worse than the baseline model. To understand why, it is necessary to remember that the semantic model uses only semantic features – probabilities of rule applications are not considered. Therefore, the semantic model is perfectly happy to predict derivations with sequences of highly unlikely rule applications so long as they predict a role that the model has been trained to prefer.

Apparently, the reckless pursuit of appealing semantic roles can ultimately harm semantic role labeling accuracy as well as parse accuracy. Consider the analysis shown in Figure 7. Because the averaged perceptron semantic model is not sensitive to the relationships between different semantic roles, and because Arg1 of *name* is a "good" semantic role, the semantic model predicts as many of them as it can. The very common np-appositive construction is particularly vulnerable to this kind of error, as it can be easily mistaken for a three-way coordination (like *carrots, peas and watermelon*). Many of the precision errors generated by the local model are of this nature, and the global model is unlikely to remove them, given the presence of strong dependencies between each of the "subjects" and the predicate.

Coordination errors are also common when dealing with relative clause attachment. Consider the analysis in Figure 8. To a PCFG model, there is little difference between attaching the relative clause to *the researchers* or *Lorillard nor the researchers*. The semantic model, however, would rather predict two semantic roles than just one (because study:Arg0 is a highly appealing semantic role). Once again, the pursuit of appealing semantic roles has led the system astray.

We have shown in Section 7 that the semantic model can improve SRL performance when it is constrained to the most likely PCFG derivations, but enumerating n-best lists is costly and cumbersome. We can, however, combine the semantic model with the baseline PCFG. Our method for doing this is designed to avoid the kinds of error described above. We first identify the highest-scoring parse according to the PCFG model. This parse will be used in later processing unless we are able to identify another parse that satisfies the following criteria:

1. It must be closely related to the parse that has the best score according to the semantic model. To identify such parses, we ask the chart unpacking algorithm to generate all the parses that can be reached by making up to five attachment changes to this semantically preferred parse – no more.

2. It must have a PCFG score that is not much less than that of the single-best PCFG parse. We do this by requiring that it has a score that is within a factor of α of the best available. That is, the single-best parse from the semantic model must satisfy

$$\log P(\text{sem}) > \log P(\text{baseline}) + \log(\alpha)$$

where the α value is tuned on the development set.

If no semantically preferred parse meets the above criteria, the single-best PCFG parse is used. We find that the PCFG-preferred parse is used about 35% of the time and an alternative used instead about 65% of the time. The SRL performance for this regime, using a range of cut-off factors, is shown in table 4. On this basis we select a cut-off of 0.5 as suitable for use for final testing. On the development set this method gives the best precision in extracting dependencies, but is slightly inferior to the method using a 2-best list on recall and balanced F-measure.

Factor (α)	P	R	F
0.5	86.3	71.9	78.5
0.1	85.4	72.0	78.1
0.05	85.2	72.0	78.0
0.005	84.3	71.3	77.3

Table 4: SRL accuracy when the semantic model is constrained by the baseline model

9 Results and Discussion

We use the method for calculating SRL performance described in the CoNNL 2008 and 2009 shared tasks. However, because the semantic role labeler we use was not designed to work with Nombank (and it is difficult to separate Nombank and Propbank predicates from the publicly released shared task output), it is not feasible to compare results with the candidate systems described there. We can, however, compare our two experimental models with our baseline parser and the current state-of-the-art CCG

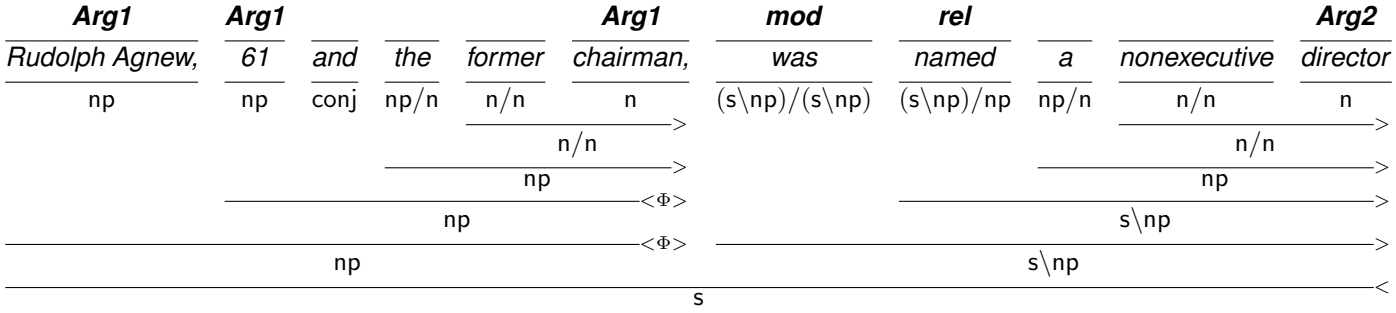

Figure 7: A parse produced by the unrestricted semantic model. Notice that *Rudolph Agnew, 61 and the former chairman* is erroneously treated as a three-way conjunction, assigning semantic roles to all three heads.

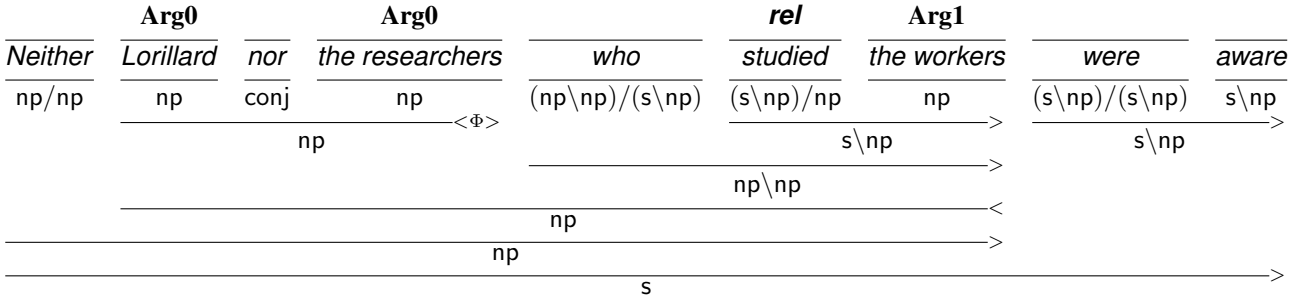

Figure 8: Relative clause attachment poses problems when preceded by a conjunction – the system generally prefers attaching relative clauses high. In this case, the relative clause should be attached low.

parser (Clark and Curran, 2004). The results on the test set (WSJ Section 23, <40 words) are shown in Table 5.

There are many areas for potential improvement for the system. The test set scores of both of our experimental models are lower than their development set scores,where the n-best model outperforms even the Clark and Curran parser in the SRL task. This may be due to vocabulary issues (we are of course unable to evaluate if the vocabulary of the training set more closely resembles the development set or the test set). If there are vocabulary issues, they could be alleviated by experimenting with POS based lexical features, or perhaps even generalizing a latent semantics over heads of semantic roles (essentially identifying broad categories of words that appear with particular semantic roles, rather than counting on having encountered that particular word in training). Alternately, this drop in performance could be caused by a mismatch in the average length of sentences, which would cause our α factor and the size of our n-best lists (which were tuned on the development set) to be suboptimal. We anticipate the opportunity to further explore better ways of determining n-best list size. We also anticipate the possibility of integrating the semantic model with a state-of-the-art CCG parser, potentially freeing the ranker from the limitations of a simple PCFG baseline.

It is also worth noting that the chart-based model seems heavily skewed towards precision. Because the parser can dig deeply into the chart, it is capable of choosing a parse that predicts only semantic roles that it is highly confident about. By choosing these parses (and not parses with less attractive semantic roles), the model can maximize the average score of the semantic roles it predicts. This tendency towards identifying only the most certain roles is consistent with high-precision low-recall results. The n-best parser has a much more restricted set of semantic roles from parses more closely resembling the single-best parse, and therefore is less likely to be presented with the opportunity to choose parses that do away with less likely (but still reasonable) roles.

10 Conclusions and Future Work

In this paper, we discuss the procedure for identifying semantic roles at parse time, and using these roles to guide the parse. We demonstrate that using semantic roles to guide parsing can improve overall SRL performance, but that these same benefits can be realized by re-ranking an n-best list with the same model. Regardless, there are several reasons why it is useful to have the ability to predict semantic roles inside the chart.

Predicting semantic roles inside the chart could be used to perform SRL on very long or unstructured passages.

	SRL			Labeled Deps		
Model	P	R	F	P	R	F
Baseline	84.7	70.7	77.0	80.0	79.8	79.9
Rank n=5	82.0	73.7	77.7	80.1	80.0	80.0
Chart	90.0	68.4	77.7	82.3	80.2	81.2
C&C	83.3	77.6	80.4	84.9	84.6	84.7
Char	77.1	75.5	76.5	-	-	-

Table 5: The full system results on the test set of the WSJ corpus (Section 23). Included are the baseline parser, the n-best reranking model from Section 7, the single-best chart-unpacking model from Section 8, and the state-of-the-art C&C parser. The final row shows the SRL performance obtained by Punyakanok et al. (2008) using the Charniak parser. Unfortunately, their results are evaluated based on spans of words (rather than headword labels), which interferes with direct comparison. The Charniak parser is a CFG-style parser, making labeled dependency non-applicable.

Most parsing research on the Penn Treebank (the present work included) focuses on sentences of 40 words or less, because parsing longer sentences requires an unacceptably large amount of computing resources. In practice, however, semantic roles are rarely very distant from their predicates – generally they are only a few words away; often they are adjacent. In long sentences, the prediction of an entire parse may be unnecessary for the purposes of SRL.

The CKY parsing algorithm works by first predicting all constituents spanning two words, then all constituents spanning three words, then four, and so on until it predicts constituents covering the whole sentence. By setting a maximum constituent size (say, ten or fifteen), we could abandon the goal of completing a spanning analysis in favor of identifying semantic roles in the neighborhood of their predicates, eliminating the need to unpack the chart at all. This could be used to efficiently perform SRL on poorly structured text or even spoken language transcriptions that are not organized into discrete sentences. Doing so would also eliminate the potentially noisy step of automatically separating out individual sentences in a larger text. Alternately, roles predicted in the chart could even be incorporated into a low-precision-high-recall information retrieval system seeking a particular semantic relationship by scanning the chart for a particular semantic role.

Another use for the packed forest of semantic roles could be to predict complete sets of roles for a given sentence using a constraint based method like integer linear programming. Integer linear programming takes a large number of candidate results (like semantic roles), and applies a set of constraints over them (like "roles may not overlap" or "no more than one of each role is allowed in each sentence") to find the optimal set. Doing so could eliminate the need to unpack the chart at all, effectively producing semantic roles without committing to a single syntactic analysis.

11 Acknowledgements

We would like to thank Mike White, William Schuler, Eric Fosler-Lussier, and Matthew Honnibal for their helpful feedback.

References

Stephen A. Boxwell, Dennis N. Mehay, and Chris Brew. 2009. Brutus: A semantic role labeling system incorporating CCG, CFG, and Dependency features. In *Proc. ACL-09*.

E. Charniak. 2001. Immediate-head parsing for language models. In *Proc. ACL-01*, volume 39, pages 116–123.

Stephen Clark and James R. Curran. 2004. Parsing the WSJ using CCG and Log-Linear Models. In *Proc. ACL-04*.

J.R. Finkel and C.D. Manning. 2009. Joint parsing and named entity recognition. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 326–334. Association for Computational Linguistics.

Daniel Gildea and Julia Hockenmaier. 2003. Identifying semantic roles using Combinatory Categorial Grammar. In *Proc. EMNLP-03*.

J. Hajič, M. Ciaramita, R. Johansson, D. Kawahara, M.A. Martí, L. Màrquez, A. Meyers, J. Nivre, S. Padó, J. Štěpánek, et al. 2009. The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning: Shared Task*, pages 1–18. Association for Computational Linguistics.

J. Hockenmaier and M. Steedman. 2005. CCGbank manual. Technical report, MS-CIS-05-09, University of Pennsylvania.

M. P. Marcus, B. Santorini, and M. A. Marcinkiewicz. 1993. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

A. Meyers, R. Reeves, C. Macleod, R. Szekely, V. Zielinska, B. Young, and R. Grishman. 2004. The nombank project: An interim report. In A. Meyers, editor, *HLT-NAACL 2004 Workshop: Frontiers in Corpus Annotation*, pages 24–31, Boston, Massachusetts, USA, May 2 - May 7. Association for Computational Linguistics.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The Proposition Bank: An Annotated Corpus of Semantic Roles. *Computational Linguistics*, 31(1):71–106.

Vasin Punyakanok, Dan Roth, and Wen tau Yih. 2008. The Importance of Syntactic Parsing and Inference in Semantic Role Labeling. *Computational Linguistics*, 34(2):257–287.

Mark Steedman. 2000. *The Syntactic Process*. MIT Press.

M. Surdeanu, R. Johansson, A. Meyers, L. Màrquez, and J. Nivre. 2008. The CoNLL-2008 shared task on joint parsing of syntactic and semantic dependencies. In *Proceedings*

of the Twelfth Conference on Computational Natural Language Learning, pages 159–177. Association for Computational Linguistics.