Discourse Parsing: Learning FOL Rules based on Rich Verb Semantic Representations to automatically label Rhetorical Relations

Rajen Subba
Computer Science
University of Illinois
Chicago, IL, USA
rsubba@cs.uic.edu

Barbara Di Eugenio
Computer Science
University of Illinois
Chicago, IL, USA
bdieugen@cs.uic.edu

Su Nam Kim
Department of CSSE
University of Melbourne
Carlton, VIC, Australia
snkim@csse.unimelb.edu.au

Abstract

We report on our work to build a discourse parser (SemDP) that uses semantic features of sentences. We use an Inductive Logic Programming (ILP) System to exploit rich verb semantics of clauses to induce rules for discourse parsing. We demonstrate that ILP can be used to learn from highly structured natural language data and that the performance of a discourse parsing model that only uses semantic information is comparable to that of the state of the art syntactic discourse parsers.

1 Introduction

The availability of corpora annotated with syntactic information have facilitated the use of probabilistic models on tasks such as syntactic parsing. Current state of the art syntactic parsers reach accuracies between 86% and 90%, as measured by different types of precision and recall (for more details see (Collins, 2003)). Recent semantic (Kingsbury and Palmer, 2002) and discourse (Carlson et al., 2003) annotation projects are paving the way for developments in semantic and discourse parsing as well. However unlike syntactic parsing, significant development in discourse parsing remains at large.

Previous work on discourse parsing ((Soricut and Marcu, 2003) and (Forbes et al., 2001)) have focused on syntactic and lexical features only. However, discourse relations connect clauses/sentences, hence, descriptions of events and states. It makes linguistic sense that the semantics of the two clauses —generally built around the semantics of the verbs, composed with that of their arguments— affects the discourse relation(s) connecting the clauses. This may be even more evident in our instructional domain, where relations derived from planning such as *Precondition-Act* may relate clauses.

Of course, since semantic information is hard to come by, it is not surprising that previous work on discourse parsing did not use it, or only used shallow word level ontological semantics as specified in WordNet (Polanyi et al., 2004). But when rich sentence level semantics is available, it makes sense to experiment with it for discourse parsing.

A second major difficulty with using such rich verb semantic information, is that it is represented using complex data structures. Traditional Machine Learning methods cannot handle highly structured data such as First Order Logic (FOL), a representation that is suitably used to represent sentence level semantics. Such FOL representations cannot be reduced to a vector of attribute/value pairs as the relations/interdependencies that exist among the predicates would be lost.

Inductive Logic Programming (ILP) can learn structured descriptions since it learns FOL descriptions. In this paper, we present our first steps using ILP to learn semantic descriptions of discourse relations. Also of relevance to the topic of this workshop, is that discourse structure is inherently highly structured, since discourse structure is generally described in hierarchical terms: basic units of analysis, generally clauses, are related by discourse relations, resulting in more complex units, which in turn can be related via discourse relations. At the moment, we do not yet address the problem of parsing at higher levels of discourse. We intend to build on the work we present in this paper to achieve that goal.

The task of discourse parsing can be divided into two disjoint sub-problems ((Soricut and Marcu, 2003) and (Polanyi et al., 2004)). The two sub-problems are automatic identification of segment boundaries and the labeling of rhetorical relations. Though we consider the problem of automatic segmentation to be an important part in discourse parsing, we have focused entirely on the latter problem of automatically labeling rhetorical

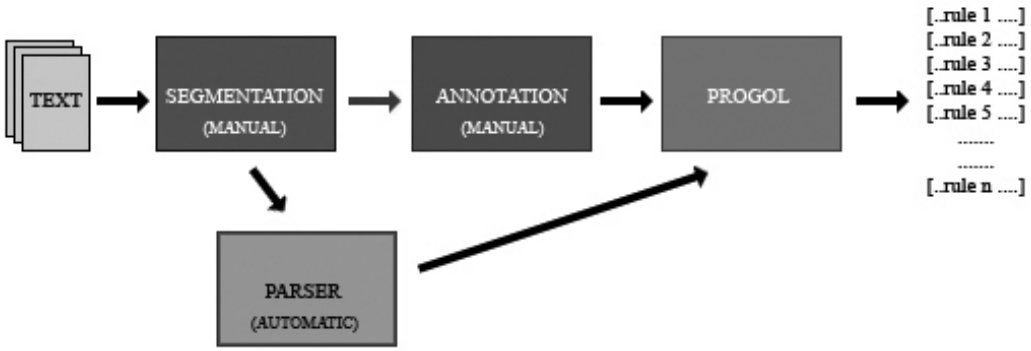

Figure 1: SemDP System Architecture (Discourse Parser)

relations only. Our approach uses rich verb semantics[1] of elementary discourse units (EDUs)[2] based on VerbNet(Kipper et al., 2000) as background knowledge and manually annotated rhetorical relations as training examples. It is trained on a lot fewer examples than the state of the art syntax-based discourse parser (Soricut and Marcu, 2003). Nevertheless, it achieves a comparable level of performance with an F-Score of 60.24. Figure 1 shows a block diagram of SemDP's system architecture. Segmentation, annotation of rhetorical relations and parsing constitute the data collection phase of the system. Learning is accomplished using an ILP based system, Progol (Muggleton, 1995). As can be seen in Figure 1, Progol takes as input both rich verb semantic information of pairs of EDUs and the rhetorical relations between them. The goal was to learn rules using the semantic information from pairs of EDUs as in Example 1:

(1) EDU1: "Sometimes, you can add a liquid to the water
EDU2: "to hasten the process"
relation(EDU1,EDU2,"Act:goal").

to automatically label unseen examples with the correct rhetorical relation.

The rest of the paper is organized as follows. Section 2 describes our data collection methodology. In section 3, Progol, the ILP system that we used to induce rules for discourse parsing is detailed. Evaluation results are presented in section 4 followed by the conclusion in section 5.

2 Data Collection

The lack of corpora annotated with both rhetorical relations as well as sentence level semantic representation led us to create our own corpus. Resources such as (Kingsbury and Palmer, 2002) and (Carlson et al., 2003) have been developed manually. Since such efforts are time consuming and costly, we decided to semi-automatically build our annotated corpus. We used an existing corpus of instructional text that is about 9MB in size and is made up entirely of written English instructions. The two largest components are home repair manuals (5Mb) and cooking recipes (1.7Mb). [3]

Segmentation. The segmentation of the corpus was done manually by a human coder. Our segmentation rules are based on those defined in (Mann and Thompson, 1988). For example, (as shown in Example 2) we segment sentences in which a conjunction is used with a clause at the conjunction site.

(2) You can copy files (//) as well as cut messages.

(//) is the segmentation marker. Sentences are segmented into EDUs. Not all the segmentation

[1]The semantic information we used is composed of VerbNet semantic predicates that capture event semantics as well as thematic roles.

[2]EDUs are minimal discourse units produced as a result of discourse segmentation.

[3]It was collected opportunistically off the internet and from other sources, and originally assembled at the Information Technology Research Institute, University of Brighton.

rules from (Mann and Thompson, 1988) are imported into our coding scheme. For example, we do not segment relative clauses. In total, our segmentation resulted in 10,084 EDUs. The segmented EDUs were then annotated with rhetorical relations by the human coder[4] and also forwarded to the parser as they had to be annotated with semantic information.

2.1 Parsing of Verb Semantics

We integrated LCFLEX (Rosé and Lavie, 2000), a robust left-corner parser, with VerbNet (Kipper et al., 2000) and CoreLex (Buitelaar, 1998). Our interest in decompositional theories of lexical semantics led us to base our semantic representation on VerbNet.

VerbNet operationalizes Levin's work and accounts for 4962 distinct verbs classified into 237 main classes. Moreover, VerbNet's strong syntactic components allow it to be easily coupled with a parser in order to automatically generate a semantically annotated corpus.

To provide semantics for nouns, we use CoreLex (Buitelaar, 1998), in turn based on the generative lexicon(Pustejovsky, 1991). CoreLex defines basic types such as *art (artifact)* or *com (communication*). Nouns that share the same bundle of basic types are grouped in the same Systematic Polysemous Class (SPC). The resulting 126 SPCs cover about 40,000 nouns.

We modified and augmented LCFLEX's existing lexicon to incorporate VerbNet and CoreLex. The lexicon is based on COMLEX (Grishman et al., 1994). Verb and noun entries in the lexicon contain a link to a semantic type defined in the ontology. VerbNet classes (including subclasses and frames) and CoreLex SPCs are realized as types in the ontology. The deep syntactic roles are mapped to the thematic roles, which are defined as variables in the ontology types. For more details on the parser see (Terenzi and Di Eugenio, 2003).

Each of the 10,084 EDUs was parsed using the parser. The parser generates both a syntactic tree and the associated semantic representation – for the purpose of this paper, we only focus on the latter. Figure 2 shows the semantic representation generated for EDU1 from Example 1, "sometimes, you can add a liquid to the water".

The semantic representation in Figure 2 is part

[4]Double annotation and segmentation is currently being done to assess inter-annotator agreement using kappa.

```
(*SEM*
 ((AGENT YOU)
  (VERBCLASS ((VNCLASS MIX-22.1-2))) (EVENT +)
  (EVENT0
   ((END
    ((ARG1 (LIQUID))
     (FRAME *TOGETHER) (ARG0 PHYSICAL)
     (ARG2 (WATER)))))))
  (EVENTSEM
   ((FRAME *CAUSE) (ARG1 E) (ARG0 (YOU)))))
  (PATIENT1 LIQUID)
  (PATIENT2 WATER)
 (ROOT-VERB ADD))
```

Figure 2: Parser Output (Semantic Information)

of the F-Structure produced by the parser. The verb *add* is parsed for a transitive frame with a PP modifier that belongs to the verb class *'MIX-22.1-2'*. The sentence contains two *PATIENT*s, namely *liquid* and *water*. *you* is identified as the *AGENT* by the parser. **TOGETHER* and **CAUSE* are the primitive semantic predicates used by VerbNet. Verb Semantics in VerbNet are defined as events that are decomposed into stages, namely start, end, during and result. The semantic representation in Figure 2 states that there is an event *EVENT0* in which the two *PATIENT*s are together at the end.

An independent evaluation on a set of 200 sentences from our instructional corpus was conducted. [5] It was able to generate complete parses for 72.2% and partial parses for 10.9% of the verb frames that we expected it to parse, given the resources. The parser cannot parse those sentences (or EDUs) that contain a verb that is not covered by VerbNet. This coverage issue, coupled with parser errors, exacerbates the problem of data sparseness. This is further worsened by the fact that we require both the EDUs in a relation set to be parsed for the Machine Learning part of our work. Addressing data sparseness is an issue left for future work.

2.2 Annotation of Rhetorical Relations

The annotation of rhetorical relations was done manually by a human coder. Our coding scheme builds on Relational Discourse Analysis (RDA) (Moser and Moore, 1995), to which we made mi-

[5]The parser evaluation was not based on EDUs but rather on unsegmented sentences. A sentence contained one or more EDUs.

nor modifications; in turn, as far as discourse relations are concerned, RDA was inspired by Rhetorical Structure Theory (RST) (Mann and Thompson, 1988).

Rhetorical relations were categorized as *informational*, *elaborational*, *temporal* and *others*. *Informational* relations describe how contents in two relata are related in the domain. These relations are further subdivided into two groups; *causality* and *similarity*. The former group consists of relations between an action and other actions or between actions and their conditions or effects. Relations like '*act:goal*', '*criterion:act*' fall under this group. The latter group consists of relations between two EDUs according to some notion of similarity such as '*restatement*' and '*contrast1:contrast2*'. *Elaborational* relations are interpropositional relations in which a proposition(s) provides detail relating to some aspect of another proposition (Mann and Thompson, 1988). Relations like '*general:specific*' and '*circumstance:situation*' belong to this category. Temporal relations like '*before:after*' capture time differences between two EDUs. Lastly, the category *others* includes relations not covered by the previous three categories such as '*joint*' and '*indeterminate*'.

Based on the modified coding scheme manual, we segmented and annotated our instructional corpus using the augmented RST tool from (Marcu et al., 1999). The RST tool was modified to incorporate our relation set. Since we were only interested in rhetorical relations that spanned between two adjacent EDUs [6], we obtained 3115 sets of potential relations from the set of all relations that we could use as training and testing data.

The parser was able to provide complete parses for both EDUs in 908 of the 3115 relation sets. These constitute the training and test set for Progol.

The semantic representation for the EDUs along with the manually annotated rhetorical relations were further processed (as shown in Figure 4) and used by Progol as input.

3 The Inductive Logic Programming Framework

We chose to use Progol, an Inductive Logic Programming system (ILP), to learn rules based on the data we collected. ILP is an area of research at the intersection of Machine Learning (ML) and Logic Programming. The general problem specification in ILP is given by the following property:

$$B \wedge H \models E \tag{3}$$

Given the background knowledge *B* and the examples *E*, ILP systems find the simplest consistent hypothesis *H*, such that *B* and *H* entails *E*.

While most of the work in NLP that involves learning has used more traditional ML paradigms like decision-tree algorithms and SVMs, we did not find them suitable for our data which is represented as Horn clauses. The requirement of using a ML system that could handle first order logic data led us to explore ILP based systems of which we found Progol most appropriate.

Progol combines Inverse Entailment with general-to-specific search through a refinement graph. A most specific clause is derived using mode declarations along with Inverse Entailment. All clauses that subsume the most specific clause form the hypothesis space. An A*-like search is used to search for the most probable theory through the hypothesis space. Progol allows arbitrary programs as background knowledge and arbitrary definite clauses as examples.

3.1 Learning from positive data only

One of the features we found appealing about Progol, besides being able to handle first order logic data, is that it can learn from positive examples alone.

Learning in natural language is a universal human process based on positive data only. However, the usual traditional learning models do not work well without negative examples. On the other hand, negative examples are not easy to obtain. Moreover, we found learning from positive data only to be a natural way to model the task of discourse parsing.

To make the learning from positive data only feasible, Progol uses a Bayesian framework. Progol learns logic programs with an arbitrarily low expected error using only positive data. Of course, we could have synthetically labeled examples of relation sets (pairs of EDUs), that did not belong to a particular relation, as negative examples. We plan to explore this approach in the future.

A key issue in learning from positive data only using a Bayesian framework is the ability to learn complex logic programs. Without any

[6] At the moment, we are concerned with learning relations between two EDUs at the base level of a Discourse Parse Tree (DPT) and not at higher levels of the hierarchy.

negative examples, the simplest rule or logic program, which in our case would be a single definite clause, would be assigned the highest score as it captures the most number of examples. In order to handle this problem, Progol's scoring function exercises a trade-off between the size of the function and the generality of the hypothesis. The score for a given hypothesis is calculated according to formula 4.

$$ln\, p(H \mid E) = m\, ln\left(\frac{1}{g(H)}\right) - sz(H) + d_m \quad (4)$$

$sz(H)$ and $g(H)$ computes the size of the hypothesis and the its generality respectively. The size of a hypothesis is measured as the number of atoms in the hypothesis whereas generality is measured by the number of positive examples the hypothesis covers. m is the number of examples covered by the hypothesis and d_m is a normalizing constant. The function $ln\ p(H|E)$ decreases with increases in $sz(H)$ and $g(H)$. As the number of examples covered (m) grow, the requirements on $g(H)$ become even stricter. This property facilitates the ability to learn more complex rules as they are supported by more positive examples. For more information on Progol and the computation of Bayes' posterior estimation, please refer to (Muggleton, 1995).

3.2 Discourse Parsing with Progol

We model the problem of assigning the correct rhetorical relation as a classification task within the ILP framework. The rich verb semantic representation of pairs of EDUs, as shown in Figure 3 [7], form the background knowledge and the manually annotated rhetorical relations between the pairs of EDUs, as shown in Figure 4, serve as the positive examples in our learning framework. The numbers in the definite clauses are *ids* used to identify the EDUs.

Progol constructs logic programs based on the background knowledge and the examples in Figures 3 and 4. Mode declarations in the Progol input file determines which clause to be used as the head (i.e. modeh) and which ones to be used in the body (i.e. modeb) of the hypotheses. Figure 5 shows an abridged set of our mode declarations.

[7] The output from the parser was further processed into definite clauses.

```
...
agent(97,you).
together(97,event0,end,physical,liquid,water).
cause(97,you,e).
patient1(97,liquid).
patient2(97,water).

theme(98,process).
rushed(98,event0,during,process).
cause(98,AGENT98,e).
...
```

Figure 3: Background Knowledge for Example 1

```
...
relation(18,19,'Act:goal').
relation(97,98,'Act:goal').
relation(1279,1280,'Step1:step2').
relation(1300,1301,'Step1:step2').
relation(1310,1311,'Step1:step2').
relation(412,413,'Before:after').
relation(441,442,'Before:after').
...
```

Figure 4: Positive Examples

Our mode declarations dictate that the predicate relation be used as the head and the other predicates (has_possession, transfer and visible) form the body of the hypotheses. '*' indicates that the number of hypotheses to learn for a given relation is unlimited. '+' and '-' signs indicate variables within the predicates of which the former is an input variable and the latter an output variable. '#' is used to denote a constant. Each argument of the predicate is a type, whether a constant or a variable. Types are defined as a single definite clause.

Our goal is to learn rules where the LHS of the rule contains the relation that we wish to learn and

```
:- modeh(*,relation(+edu,+edu,#relationtype))?

:- modeb(*,has_possession(+edu,#event,
          #eventstage,+verbarg,+verbarg))?
:- modeb(*,has_possession(+edu,#event,
          #eventstage,+verbarg,-verbarg))?
:- modeb(*,transfer(+edu,#event,#eventstage,-verbarg))?
:- modeb(*,visible(+edu,#event,#eventstage,+verbarg))?
:- modeb(*,together(+edu,#event,
          #eventstage,+verbarg,+verbarg,+verbarg))?
:- modeb(*,rushed(+edu,#event,#eventstage,+verbarg))?
```

Figure 5: Mode Declarations

```
RULE1:
 relation(EDU1,EDU2,'Act:goal') :-
   degradation_material_integrity(EDU1,event0,result,C),
   allow(EDU2,event0,during,C,D).
RULE2:
 relation(EDU1,EDU2,'Act:goal') :-
   cause(EDU1,C,D),
   together(EDU1,event0,end,E,F,G),
   cause(EDU2,C,D).
RULE3:
 relation(EDU1,EDU2,'Step1:step2') :-
   together(EDU2,event0,end,C,D,E),
   has_possession(EDU1,event0,during,C,F).
RULE4:
 relation(EDU1,EDU2,'Before:after') :-
   motion(EDU1,event0,during,C),
   location(EDU2,event0,start,C,D).
RULE6:
 relation(EDU1,EDU2,'Act:goal') :-
   motion(EDU1,event0,during,C).
```

Figure 6: Rules Learned

the RHS is a CNF of the semantic predicates defined in VerbNet with their arguments. Given the amount of training data we have, the nature of the data itself and the Bayesian framework used, Progol learns simple rules that contain just one or two clauses on the RHS. 6 of the 68 rules that Progol manages to learn are shown in Figure 6. RULE4 states that there is a theme in motion during the event in EDU A (which is the first EDU) and that the theme is located in location D at the start of the event in EDU B (the second EDU). RULE2 is learned from pairs of EDUs such as in Example 1. The simple rules in Figure 6 may not readily appeal to our intuitive notion of what such rules should include. It is not clear at this point as to how elaborate these rules should be, in order to correctly identify the relation in question. One of the reasons why more complex rules are not learned by Progol is that there aren't enough training examples. As we add more training data in the future, we will see if rules that are more elaborate than the ones in Figure 6 are learned .

4 Evaluation of the Discourse Parser

Table 1 shows the sets of relations for which we managed to obtain semantic representations (i.e. for both the EDUs).

Relations like *Preparation:act* did not yield any examples that could potentially be used. For a number of relations, the total number of examples we could use were less than 50. For the time being, we decided to use only those relation sets that had more than 50 examples. In addition, we chose not to use *Joint* and *General:specific* relations. They will be included in the future. Hence, our training and testing data consisted of the following four relations: *Goal:act*, *Step1:step2*, *Criterion:act* and *Before:after*. The total number of examples we used was 508 of which 423 were used for training and 85 were used for testing.

Relation	Total	Train Set	Test Set
Step1:step2:	**232**	**188**	**44**
Joint:	190		
Goal:act:	**170**	**147**	**23**
General:specific:	77		
Criterion:act:	**53**	**46**	**7**
Before:after:	**53**	**42**	**11**
Act:side-effect:	38		
Co-temp1:co-temp2:	22		
Cause:effect:	19		
Prescribe-act:wrong-act:	14		
Obstacle:situation:	11		
Reason:act:	9		
Restatement:	6		
Contrast1:contrast2:	6		
Circumstance:situation:	3		
Act:constraint:	2		
Criterion:wrong-act:	2		
Set:member:	1		
Act:justification:	0		
Comparison:	0		
Preparation:act:	0		
Object:attribute:	0		
Part:whole:	0		
Same-unit:	0		
Indeterminate:	0		
	908	423	85

Table 1: Relation Set Count (Total Counts include examples that yielded semantic representations for both EDUs)

Table 2, Table 3 and Table 4 show the results from running the system on our test data. A total of 85 positive examples were used for testing the system.

Table 2 evaluates our SemDP system against a baseline. Our baseline is the majority function, which performs at a 51.7 F-Score. SemDP outperforms the baseline by almost 10 percentage points

Discourse Parser	Precision	Recall	F-Score
SemDP	**61.7**	**58.8**	**60.24**
Baseline*	51.7	51.7	51.7

Table 2: Evaluation vs Baseline (* *our baseline is the majority function*)

Relation	Precision	Recall	F-Score
Goal:act	31.57	26.08	28.57
Step1:step2	**75**	**75**	**75**
Before:after	54.5	54.5	54.5
Criterion:act	71.4	71.4	71.4
Total	**61.7**	**58.8**	**60.24**

Table 3: Test Results for SemDP

with an F-Score of 60.24. To the best of our knowledge, we are also not aware of any work that uses rich semantic information for discourse parsing. (Polanyi et al., 2004) do not provide any evaluation results at all. (Soricut and Marcu, 2003) report that their SynDP parser achieved up to 63.8 F-Score on human-segmented test data. Our result of 60.24 F-Score shows that a Discourse Parser based purely on semantics can perform as well. However, since the corpus, the size of training data and the set of rhetorical relations we have used differ from (Soricut and Marcu, 2003), a direct comparison cannot be made.

Table 3 breaks down the results in detail for each of the four rhetorical relations we tested on. Since we are learning from positive data only and the rules we learn depend heavily on the amount of training data we have, we expected the system to be more accurate with the relations that have more training examples. As expected, SemDP did very well in labeling *Step1:step2* relations. Surprisingly though, it did not perform as well with *Goal:act*, even though it had the second highest number of training examples (147 in total). In fact, SemDP misclassified more positive test examples for *Goal:act* than *Before:after* or *Criterion:act*, relations which had almost one third the number of training examples. Overall SemDP achieved a precision of 61.7 and a Recall of 58.8.

In order to find out how the positive test examples were misclassified, we investigated the distribution of the relations classified by SemDP. Table 4 is the confusion matrix that highlights this issue. A majority of the actual *Goal:act* relations are incorrectly classified as *Step1:step1* and *Before:after*. Likewise, most of the misclassification of actual *Step1:step1* seems to labeled as *Goal:act* or *Before:after*. Such misclassification occurs because the simple rules learned by SemDP are not able to accurately distinguish cases where positive examples of two different relations share similar semantic predicates. Moreover, since we are learning using positive examples only, it is possible that a positive example may satisfy two or more rules for different relations. In such cases, the rule that has the highest score (as calculated by formula 4) is used to label the unseen example.

5 Conclusions and Future Work

We have shown that it is possible to learn First Order Logic rules from complex semantic data using an ILP based methodology. These rules can be used to automatically label rhetorical relations. Moreover, our results show that a Discourse Parser that uses only semantic information can perform as well as the state of the art Discourse Parsers based on syntactic and lexical information.

Future work will involve the use of syntactic information as well. We also plan to run a more thorough evaluation on the complete set of relations that we have used in our coding scheme. It is also important that the manual segmentation and annotation of rhetorical relations be subject to inter-annotator agreement. A second human annotator is currently annotating a sample of the annotated corpus. Upon completion, the annotated corpus will be checked for reliability.

Data sparseness is a well known problem in Machine Learning. Like most paradigms, our learning model is also affected by it. We also plan to explore techniques to deal with this issue.

Relation	Goal:act	Step1:step2	Before:after	Criterion:act
Goal:act	6	**8**	5	0
Step1:step2	6	**33**	5	0
Before:after	0	4	**6**	1
Criterion:act	0	0	2	**5**

Table 4: Confusion Matrix for SemDP Test Result

Lastly, we have not tackled the problem of discourse parsing at higher levels of the DPT and segmentation in this paper. Our ultimate goal is to build a Discourse Parser that will automatically segment a full text as well as annotate it with rhetorical relations at every level of the DPT using semantic as well as syntactic information. Much work needs to be done but we are excited to see what the aforesaid future work will yield.

Acknowledgments

This work is supported by award 0133123 from the National Science Foundation. Thanks to C.P. Rosé for LCFLEX, M. Palmer and K. Kipper for VerbNet, C. Buitelaar for CoreLex, and Stephen Muggleton for Progol.

References

Paul Buitelaar. 1998. *CoreLex: Systematic Polysemy and Underspecification*. Ph.D. thesis, Computer Science, Brandeis University, February.

Lynn Carlson, Daniel Marcu, and Mary Ellen Okurowski. 2003. Building a discourse-tagged corpus in the framework of Rhetorical Structure Theory. *In Current Directions in Discourse and Dialogue*, pp. 85-112, Jan van Kuppevelt and Ronnie Smith eds., Kluwer Academic Publishers.

Michael Collins. 2003. Head-driven statistical methods for natural language parsing. *Computational Linguistics*, 29.

Katherine Forbes, Eleni Miltsakaki, Rashmi Prasad, Anoop Sarkar, Aravind Joshi and Bonnie Webber. 2001. D-LTAG System - Discourse Parsing with a Lexicalized Tree Adjoining Grammar. *Information Stucture, Discourse Structure and Discourse Semantics*, ESSLLI, 2001.

Ralph Grishman, Catherine Macleod, and Adam Meyers. 1994. COMLEX syntax: Building a computational lexicon. In *COLING 94, Proceedings of the 15th International Conference on Computational Linguistics*, pages 472–477, Kyoto, Japan, August.

Paul Kingsbury and Martha Palmer. 2000. From Treebank to Propbank. In *Third International Conference on Language Resources and Evaluation, LREC-02*, Las Palmas, Canary Islands, Spain, May 28 - June 3, 2002.

Karin Kipper, Hoa Trang Dang, and Martha Palmer. 2000. Class-based construction of a verb lexicon. In *AAAI-2000, Proceedings of the Seventeenth National Conference on Artificial Intelligence*, Austin, TX.

Beth Levin and Malka Rappaport Hovav. 1992. Wiping the slate clean: a lexical semantic exploration. In Beth Levin and Steven Pinker, editors, *Lexical and Conceptual Semantics, Special Issue of Cognition: International Journal of Cognitive Science*. Blackwell Publishers.

William C. Mann and Sandra Thompson. 1988. Rhetorical Structure Theory: toward a Functional Theory of Text Organization. *Text*, 8(3):243–281.

Daniel Marcu and Abdessamad Echihabi. 2002. An unsupervised approach to recognizing discourse relations. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL-2002)*, Philadelphia, PA, July.

Daniel Marcu, Magdalena Romera and Estibaliz Amorrortu. 1999. Experiments in Constructing a Corpus of Discourse Trees: Problems, Annotation Choices, Issues. In *The Workshop on Levels of Representation in Discourse, pages 71-78*, Edinburgh, Scotland, July.

M. G. Moser, and J. D. Moore. 1995. Using Discourse Analysis and Automatic Text Generation to Study Discourse Cue Usage. In *AAAI Spring Symposium on Empirical Methods in Discourse Interpretation and Generation*, 1995.

Stephen H. Muggleton. 1995. Inverse Entailment and Progol. In *New Generation Computing Journal*, Vol. 13, pp. 245-286, 1995.

Martha Palmer, Daniel Gildea and, Paul Kingsbury. 2005. The Proposition Bank: An Annotated Corpus of Semantic Roles. *Computational Linguistics*, 31(1):71–105.

Livia Polanyi, Christopher Culy, Martin H. van den Berg, Gian Lorenzo Thione, and David Ahn. 2004. Sentential Structure and Discourse Parsing. *Proceedings of the ACL2004 Workshop on Discourse Annotation*, Barcelona, Spain, July 25, 2004.

James Pustejovsky. 1991. The generative lexicon. *Computational Linguistics*, 17(4):409–441.

Carolyn Penstein Rosé and Alon Lavie. 2000. Balancing robustness and efficiency in unification-augmented context-free parsers for large practical applications. In Jean-Clause Junqua and Gertjan van Noord, editors, *Robustness in Language and Speech Technology*. Kluwer Academic Press.

Radu Soricut and Daniel Marcu. 2003. Sentence Level Discourse Parsing using Syntactic and Lexical Information. In *Proceedings of the Human Language Technology and North American Assiciation for Computational Linguistics Conference (HLT/NAACL-2003)*, Edmonton, Canada, May-June.

Elena Terenzi and Barbara Di Eugenio. 2003. Building lexical semantic representations for natural language instructions. In *HLT-NAACL03, 2003 Human Language Technology Conference*, pages 100–102, Edmonton, Canada, May. (Short Paper).