The Effects of Semantic Annotations on Precision Parse Ranking

Andrew MacKinlay[♢♡], **Rebecca Dridan**[♠*], **Diana McCarthy**[♣†] **and Timothy Baldwin**[♢♡]
[♢] Dept. of Computing and Information Systems, University of Melbourne, Australia
[♡] NICTA Victoria Research Laboratories, University of Melbourne, Australia
[♠] Department of Informatics, University of Oslo, Norway
[♣] Computational Linguistics and Phonetics, Saarland University, Germany
amack@csse.unimelb.edu.au, rdridan@ifi.uio.no,
diana@dianamccarthy.co.uk, tb@ldwin.net

Abstract

We investigate the effects of adding semantic annotations including word sense hypernyms to the source text for use as an extra source of information in HPSG parse ranking for the English Resource Grammar. The semantic annotations are coarse semantic categories or entries from a distributional thesaurus, assigned either heuristically or by a pre-trained tagger. We test this using two test corpora in different domains with various sources of training data. The best reduces error rate in dependency F-score by 1% on average, while some methods produce substantial decreases in performance.

1 Introduction

Most start-of-the-art natural language parsers (Charniak, 2000; Clark and Curran, 2004; Collins, 1997) use lexicalised features for parse ranking. These are important to achieve optimal parsing accuracy, and yet these are also the features which by their nature suffer from data-sparseness problems in the training data. In the absence of reliable fine-grained statistics for a given token, various strategies are possible. There will often be statistics available for coarser categories, such as the POS of the particular token. However, it is possible that these coarser representations discard too much, missing out information which could be valuable to the parse ranking. An intermediate level of representation could provide valuable additional information here. For example, assume we wish to correctly attach the prepositional phrases in the following examples:

(1) *I saw a tree with my telescope*

(2) *I saw a tree with no leaves*

The most obvious interpretation in each case has the prepositional phrase headed by *with* attaching in different places: to the verb phrase in the first example, and to the noun *tree* in the second. Such distinctions are difficult for a parser to make when the training data is sparse, but imagine we had seen examples such as the following in the training corpus:

(3) *Kim saw a eucalypt with his binoculars*

(4) *Sandy observed a willow with plentiful foliage*

There are few lexical items in common, but in each case the prepositional phrase attachment follows the same pattern: in (3) it attaches to the verb, and in (4) to the noun. A conventional lexicalised parser would have no knowledge of the semantic similarity between *eucalypt* and *tree*, *willow* and *tree*, *binoculars* and *telescope*, or *foliage* and *leaves*, so would not be able to make any conclusions about the earlier examples on the basis of this training data. However if the parse ranker has also been supplied with information about synonyms or hypernyms of the lexemes in the training data, it could possibly have generalised, to learn that PPs containing nouns related to seeing instruments often modify verbs relating to observation (in preference to nouns denoting inanimate objects), while plant flora can often be modified by PPs relating to appendages of plants such as *leaves*. This is not necessarily applicable only to PP attachment, but may help in a range of other syntactic phenomena, such as distinguishing between complements and modifiers of verbs.

[*]This research was conducted while the second author was a postdoctoral researcher within NICTA VRL.

[†]The third author is a visiting scholar on the Erasmus Mundus Masters Program in 'Language and Communication Technologies' (LCT, 2007–0060).

*First Joint Conference on Lexical and Computational Semantics (*SEM)*, pages 228–236,
Montréal, Canada, June 7-8, 2012.

The synonyms or hypernyms could take the form of any grouping which relates word forms with semantic or syntactic commonality – such as a label from the WordNet (Miller, 1995) hierarchy, a subcategorisation frame (for verbs) or closely related terms from a distributional thesaurus (Lin, 1998).

We present work here on using various levels of semantic generalisation as an attempt to improve parse selection accuracy with the English Resource Grammar (ERG: Flickinger (2000)), a precision HPSG-based grammar of English.

2 Related Work

2.1 Parse Selection for Precision Grammars

The focus of this work is on parsing using handcrafted precision HPSG-based grammars, and in particular the ERG. While these grammars are carefully crafted to avoid overgeneration, the ambiguity of natural languages means that there will unavoidably be multiple candidate parses licensed by the grammar for any non-trivial sentence. For the ERG, the number of parses postulated for a given sentence can be anywhere from zero to tens of thousands. It is the job of the parse selection model to select the best parse from all of these candidates as accurately as possible, for some definition of 'best', as we discuss in Section 3.2.

Parse selection is usually performed by training discriminative parse selection models, which 'discriminate' between the set of all candidate parses. A widely-used method to achieve this is outlined in Velldal (2007). We feed both correct and incorrect parses licensed by the grammar to the TADM toolkit (Malouf, 2002), and learn a maximum entropy model. This method is used by Zhang et al. (2007) and MacKinlay et al. (2011) *inter alia*. One important implementation detail is that rather than exhaustively ranking all candidates out of possibly many thousands of trees, Zhang et al. (2007) showed that it was possible to use 'selective unpacking', which means that the exhaustive parse forest can be represented compactly as a 'packed forest', and the top-ranked trees can be successively reconstructed, enabling faster parsing using less memory.

2.2 Semantic Generalisation for parse ranking

Above, we outlined a number of reasons why semantic generalisation of lexemes could enable parsers to make more efficient use of training data, and indeed, there has been some prior work investigating this possibility. Agirre et al. (2008) applied two state-of-the-art treebank parsers to the sense-tagged subset of the Brown corpus version of the Penn Treebank (Marcus et al., 1993), and added sense annotation to the training data to evaluate their impact on parse selection and specifically on PP-attachment. The annotations they used were oracle sense annotations, automatic sense recognition and the first sense heuristic, and it was this last method which was the best performer in general. The sense annotations were either the WordNet synset ID or the coarse *semantic file*, which we explain in more detail below, and replaced the original tokens in the training data. The largest improvement in parsing F-score was a 6.9% reduction in error rate for the Bikel parser (Bikel, 2002), boosting the F-score from 0.841 to 0.852, using the noun supersense only. More recently, Agirre et al. (2011) largely reproduced these results with a dependency parser.

Fujita et al. (2007) add sense information to improve parse ranking with JaCy (Siegel and Bender, 2002), an HPSG-based grammar which uses similar machinery to the ERG. They use baseline syntactic features, and also add semantic features based on dependency triples extracted from the semantic representations of the sentence trees output by the parser. The dataset they use has human-assigned sense tags from a Japanese lexical hierarchy, which they use as a source of annotations. The dependency triples are modified in each feature set by replacing elements of the semantic triples with corresponding senses or hypernyms. In the best-performing configuration, they use both syntactic and semantic features with multiple levels of the the semantic hierarchy from combined feature sets. They achieve a 5.6% improvement in exact match parsing accuracy.

3 Methodology

We performed experiments in HPSG parse ranking using the ERG, evaluating the impact on parse selection of semantic annotations such as coarse sense labels or synonyms from a distributional the-

	WESCIENCE	LOGON
Total Sentences	9632	9410
Parseable Sentences	9249	8799
Validated Sentences	7631	8550
Train/Test Sentences	6149/1482	6823/1727
Tokens/sentence	15.0	13.6
Training Tokens	92.5k	92.8k

Table 1: Corpora used in our experiments, with total sentences, how many of those can be parsed, how many of the parseable sentences have a single gold parse (and are used in these experiments), and average sentence length

saurus. Our work here differs from the aforementioned work of Fujita et al. (2007) in a number of ways. Firstly, we use purely syntactic parse selection features based on the derivation tree of the sentence (see Section 3.4.3), rather than ranking using dependency triples, meaning that our method is in principle able to be integrated into a parser more easily, where the final set of dependencies would not be known in advance. Secondly, we do not use human-created sense annotations, instead relying on heuristics or trained sense-taggers, which is closer to the reality of real-world parsing tasks.

3.1 Corpora

Following MacKinlay et al. (2011), we use two primary training corpora. First, we use the LOGON corpus (Oepen et al., 2004), a collection of English translations of Norwegian hiking texts. The LOGON corpus contains 8550 sentences with exactly one gold parse, which we partitioned randomly by sentence into 10 approximately equal sections, reserving two sections as test data, and using the remainder as our training corpus. These sentences were randomly divided into training and development data. Secondly, we use the WeScience (Ytrestøl et al., 2009) corpus, a collection of Wikipedia articles related to computational linguistics. The corpus contains 11558 sentences, from which we randomly chose 9632, preserving the remainder for future work. This left 7631 sentences with a single gold tree, which we divided into a training set and a development set in the same way. The corpora are summarised in Table 1.

With these corpora, we are able to investigate in-domain and cross-domain effects, by testing on a different corpus to the training corpus, so we can examine whether sense-tagging alleviates the cross-domain performance penalty noted in MacKinlay et al. (2011). We can also use a subset of each training corpus to simulate the common situation of sparse training data, so we can investigate whether sense-tagging enables the learner to make better use of a limited quantity of training data.

3.2 Evaluation

Our primary evaluation metric is Elementary Dependency Match (Dridan and Oepen, 2011). This converts the semantic output of the ERG into a set of dependency-like triples, and scores these triples using precision, recall and F-score as is conventional for other dependency evaluation. Following MacKinlay et al. (2011), we use the EDM_{NA} mode of evaluation, which provides a good level of comparability while still reflecting most the semantically salient information from the grammar.

Other work on the ERG and related grammars has tended to focus on exact tree match, but the granular EDM metric is a better fit for our needs here – among other reasons, it is more sensitive in terms of error rate reduction to changes in parse selection models (MacKinlay et al., 2011). Additionally, it is desirable to be able to choose between two different parses which do not match the gold standard exactly but when one of the parses is a closer match than the other; this is not possible with exact match accuracy.

3.3 Reranking for parse selection

The features we are adding to the parse selection procedure could all in principle be applied by the parser during the selective unpacking stage, since they all depend on information which can be pre-computed. However, we wish to avoid the need for multiple expensive parsing runs, and more importantly the need to modify the relatively complex internals of the parse ranking machinery in the PET parser (Callmeier, 2000). So instead of performing the parse ranking in conjunction with parsing, as is the usual practice, we use a pre-parsed forest of the top-500 trees for each corpus, and rerank the forest afterwards for each configuration shown.

The pre-parsed forests use the same models which were used in treebanking. Using reranking means that the set of candidate trees is held constant, which

means that parse selection models never get the chance to introduce a new tree which was not in the original parse forest from which the gold tree was annotated, which may provide a very small performance boost (although when the parse selection models are similar as is the case for most of the models here, this effect is likely to be very small).

3.4 Word Sense Annotations

3.4.1 Using the WordNet Hierarchy

Most experiments we report on here make some use of the WordNet sense inventory. Obviously we need to determine the best sense and corresponding WordNet synset for a given token. We return to this in Section 3.4.2, but for now assume that the sense disambiguation is done.

As we are concerned primarily with making commonalities between lemmas with different base forms apparent to the parse selection model, the fine-grained synset ID will do relatively little to provide a coarser identifier for the token – indeed, if two tokens with identical forms were assigned different synset IDs, we would be obscuring the similarity.[1]

We can of course make use of the WordNet hierarchy, and use hypernyms from the hierarchy to tag each candidate token, but there are a large number of ways this can be achieved, particularly when it is possibly to assign multiple labels per token as is the case here (which we discuss in Section 3.4.3). We apply two relatively simple strategies. We noted in Section 2.2 that Agirre et al. (2008) found that the semantic file was useful. This is the coarse lexicographic category label, elsewhere denoted *supersense* (Ciaramita and Altun, 2006), which is the terminology we use. Nouns are divided into 26 coarse categories such as 'animal', 'quantity' or 'phenomenon', and verbs into 15 categories such as 'perception' or 'consumption'. In some configurations, denoted SS, we tag each open-class token with one of the supersense labels.

Another configuration attempts to avoid making assumptions about which level of the hierarchy will be most useful for parse disambiguation, instead leaving it the MaxEnt parse ranker to pick those labels from the hierarchy which are most useful. Each open class token is labelled with multiple synsets, starting with the assigned leaf synset and travelling as high as possible up the hierarchy, with no distinction made between the different levels in the hierarchy. Configurations using this are designated HP, for 'hypernym path'.

[1] This could be useful for verbs since senses interact strongly subcategorisation frames, but that is not our focus here.

3.4.2 Disambiguating senses

We return now to the question of determination of the synset for a given token. One frequently-used and robust strategy is to lemmatise and POS-tag each token, and assign it the first-listed sense from WordNet (which may or may not be based on actual frequency counts). We POS-tag using TnT (Brants, 2000) and lemmatise using WordNet's native lemmatiser. This yields a leaf-level synset, making it suitable as a source of annotations for both SS and HP. We denote this 'WNF' for 'WordNet First' (shown in parentheses after SS or HP).

Secondly, to evaluate whether a more informed approach to sense-tagging helps beyond the naive WNF method, in the 'SST' method, we use the outputs of SuperSense Tagger (Ciaramita and Altun, 2006), which is optimised for assigning the supersenses described above, and can outperform a WNF-style baseline on at least some datasets. Since this only gives us coarse supersense labels, it can only provide SS annotations, as we do not get the leaf synsets needed for HP. The input we feed in is POS-tagged with TnT as above, for comparability with the WNF method, and to ensure that it is compatible with the configuration in which the corpora were parsed – specifically, the unknown-word handling uses a version of the sentences tagged with TnT. We ignore multi-token named entity outputs from SuperSense Tagger, as these would introduce a confounding factor in our experiments and also reduce comparability of the results with the WNF method.

3.4.3 A distributional thesaurus method

A final configuration attempts to avoid the need for curated resources such as WordNet, instead using an automatically-constructed distributional thesaurus (Lin, 1998). We use the thesaurus from McCarthy et al. (2004), constructed along these lines using the grammatical relations from RASP (Briscoe and Carroll, 2002) applied to 90 millions words of text from the British National Corpus.

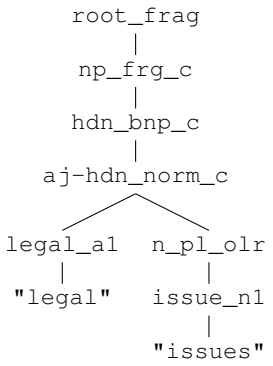


Figure 1: ERG derivation tree for the phrase *Legal issues*

```
[n_-_c_le "issues"]
[n_pl_olr n_-_c_le "issues"]
[aj-hdn_norm_c n_pl_olr n_-_c_le "issues"]
```

(a) Original features

```
[n_-_c_le noun.cognition]
[n_pl_olr n_-_c_le noun.cognition]
[aj-hdn_norm_c n_pl_olr n_-_c_le noun.cognition]
```

(b) Additional features in leaf mode, which augment the original features

```
[noun.cognition "issues"]
[n_pl_olr noun.cognition "issues"]
[aj-hdn_norm_c n_pl_olr noun.cognition "issues"]
```

(c) Additional features in leaf-parent ('P') mode, which augment the original features

Figure 2: Examples of features extracted from for `"issues"` node in Figure 1 with grandparenting level of 2 or less

To apply the mapping, we POS-tag the text with TnT as usual, and for each noun, verb and adjective we lemmatise the token (with WordNet again, falling back to the surface form if this fails), and look up the corresponding entry in the thesaurus. If there is a match, we select the top five most similar entries (or fewer if there are less than five), and use these new entries to create additional features, as well as adding a feature for the lemma itself in all cases. This method is denoted LDT for 'Lin Distributional Thesaurus'. We note that many other methods could be used to select these, such as different numbers of synonyms, or dynamically changing the number of synonyms based on a threshold against the top similarity score, but this is not something we evaluate in this preliminary investigation.

Adding Word Sense to Parse Selection Models

We noted above that parse selection using the methodology established by Velldal (2007) uses human-annotated incorrect and correct derivation trees to train a maximum entropy parse selection model. More specifically, the model is trained using features extracted from the candidate HPSG derivation trees, using the labels of each node (which are the rule names from the grammar) and those of a limited number of ancestor nodes.

As an example, we examine the noun phrase *Legal issues* from the WESCIENCE corpus, for which the correct ERG derivation tree is shown in Figure 1. Features are created by examining each node in the tree and at least its parent, with the feature name set to the concatenation of the node labels. We also generally make used of *grandparenting* features, where we examine earlier ancestors in the derivation tree. A grandparenting level of one means we would also use the label of the grandparent (i.e. the parent's parent) of the node, a level of two means we would add in the great-grandparent label, and so on. Our experiments here use a maximum grandparenting level of three. There is also an additional transformation applied to the tree – the immediate parent of each leaf is, which is usually a lexeme, is replaced with the corresponding *lexical type*, which is a broader parent category from the type hierarchy of the grammar, although the details of this are not relevant here.

For the node labelled `"issues"` in Figure 1 with grandparenting levels from zero to two, we would extract the features as shown in Figure 2(a) (where the parent node `issue_n1` has already been replaced with its lexical type `n_-_c_le`).

In this work here, we create variants of these features. A preprocessing script runs over the training or test data, and for each sentence lists variants of each token using standoff markup indexed by character span, which are created from the set of additional semantic tags assigned to each token by the word sense configuration (from those described in Section 3.4) which is currently in use. These sets of semantic tags for a given word could be a single supersense tag, as in SS, a set of synset IDs as in HP or a set of replacement lemmas in LDT. In all cases, the set of semantic tags could also be empty – if either the word has a part of speech which we are not

Test	Train				SS (WNF)				SS_p(WNF)			
		P/	R/	F	P/	R/	F	ΔF	P/	R/	F	ΔF
LOG	WESC (23k)	85.02/82.22/83.60			85.09/82.33/83.69			+0.09	84.81/82.20/83.48			−0.11
	WESC (92k)	86.56/83.58/85.05			86.83/84.04/85.41			+0.36	87.03/83.96/85.47			+0.42
	LOG (23k)	88.60/87.23/87.91			88.72/87.20/87.95			+0.04	88.43/87.00/87.71			−0.21
	LOG (92k)	91.74/90.15/90.94			91.82/90.07/90.94			−0.00	91.90/90.13/91.01			+0.07
WESC	WESC (23k)	86.80/84.43/85.60			87.12/84.44/85.76			+0.16	87.18/84.50/85.82			+0.22
	WESC (92k)	89.34/86.81/88.06			89.54/86.76/88.13			+0.07	89.43/87.23/88.32			+0.26
	LOG (23k)	83.74/81.41/82.56			84.02/81.43/82.71			+0.15	84.10/81.67/82.86			+0.31
	LOG (92k)	85.98/82.93/84.43			86.02/82.69/84.32			−0.11	85.89/82.76/84.30			−0.13

Table 2: Results for SS (WNF) (supersense from first WordNet sense), evaluated on 23k tokens (approx 1500 sentences) of either WESCIENCE or LOGON, and trained on various sizes of in-domain and cross-domain training data. Subscript '$_p$' indicates mappings were applied to leaf parents rather than leaves.

Test	Train				SS (SST)				SS_p(SST)			
		P/	R/	F	P/	R/	F	ΔF	P/	R/	F	ΔF
LOG	WESC (23k)	85.02/82.22/83.60			84.97/82.38/83.65			+0.06	85.32/82.66/83.97			+0.37
	WESC (92k)	86.56/83.58/85.05			87.05/84.47/85.74			+0.70	86.98/83.87/85.40			+0.35
	LOG (23k)	88.60/87.23/87.91			88.93/87.50/88.21			+0.29	88.84/87.40/88.11			+0.20
	LOG (92k)	91.74/90.15/90.94			91.67/90.02/90.83			−0.10	91.47/89.96/90.71			−0.23
WESC	WESC (23k)	86.80/84.43/85.60			86.88/84.29/85.56			−0.04	87.32/84.48/85.88			+0.27
	WESC (92k)	89.34/86.81/88.06			89.53/86.54/88.01			−0.05	89.50/86.56/88.00			−0.05
	LOG (23k)	83.74/81.41/82.56			84.06/81.30/82.66			+0.10	83.96/81.64/82.78			+0.23
	LOG (92k)	85.98/82.93/84.43			86.13/82.96/84.51			+0.08	85.76/82.84/84.28			−0.16

Table 3: Results for SS (SST) (supersense from SuperSense Tagger)

Test	Train				HPWNF				HP_p(WNF)			
		P/	R/	F	P/	R/	F	ΔF	P/	R/	F	ΔF
LOG	WESC (23k)	85.02/82.22/83.60			84.56/82.03/83.28			−0.32	84.74/82.20/83.45			−0.15
	WESC (92k)	86.56/83.58/85.05			86.65/84.22/85.42			+0.37	86.41/83.65/85.01			−0.04
	LOG (23k)	88.60/87.23/87.91			88.58/87.26/87.92			+0.00	88.58/87.35/87.96			+0.05
	LOG (92k)	91.74/90.15/90.94			91.68/90.19/90.93			−0.01	91.66/89.85/90.75			−0.19
WESC	WESC (23k)	86.80/84.43/85.60			86.89/84.19/85.52			−0.08	87.18/84.43/85.78			+0.18
	WESC (92k)	89.34/86.81/88.06			89.74/86.96/88.33			+0.27	89.23/86.88/88.04			−0.01
	LOG (23k)	83.74/81.41/82.56			83.87/81.20/82.51			−0.04	83.47/81.00/82.22			−0.33
	LOG (92k)	85.98/82.93/84.43			85.89/82.38/84.10			−0.33	85.75/83.03/84.37			−0.06

Table 4: Results for HPWNF (hypernym path from first WordNet sense)

Test	Train				$LDT_p(5)$			
		P/	R/	F	P/	R/	F	ΔF
LOG	WESC (23k)	85.02/82.22/83.60			84.48/82.18/83.31			−0.28
	WESC (92k)	86.56/83.58/85.05			86.36/84.14/85.23			+0.19
	LOG (23k)	88.60/87.23/87.91			88.28/86.99/87.63			−0.28
	LOG (92k)	91.74/90.15/90.94			91.01/89.25/90.12			−0.82
WESC	WESC (23k)	86.80/84.43/85.60			86.17/83.51/84.82			−0.78
	WESC (92k)	89.34/86.81/88.06			88.31/85.61/86.94			−1.12
	LOG (23k)	83.74/81.41/82.56			83.60/81.18/82.37			−0.19
	LOG (92k)	85.98/82.93/84.43			85.74/82.96/84.33			−0.11

Table 5: Results for LDT (5) (Lin-style distributional thesaurus, expanding each term with the top-5 most similar)

attempting to tag semantically, or if our method has no knowledge of the particular word.

The mapping is applied at the point of feature extraction from the set of derivation trees – at model construction time for the training set and at reranking time for the development set. If a given leaf token has some set of corresponding semantic tags, we add a set of variant features for each semantic tag, duplicated and modified from the matching "core" features described above. There are two ways these mappings can be applied, since it is not immediately apparent where the extra lexical generalisation would be most useful. The 'leaf' variant applies to the leaf node itself, so that in each feature involving the leaf node, add a variant where the leaf node surface string has been replaced with the new semantic tag. The 'parent' variant, which has a subscript 'P' (e.g. SS_p(WNF)) applies the mapping to the immediate parent of the leaf node, leaving the leaf itself unchanged, but creating variant features with the parent nodes replaced with the tag.

For our example here, we assume that we have an SS mapping for Figure 2(a), and that this has mapped the token for `"issues"` to the WordNet supersense `noun.cognition`. For the leaf variant, the extra features that would be added (either for considering inclusion in the model, or for scoring a sentence when reranking) are shown in Figure 2(b), while those for the parent variant are in Figure 2(c).

3.4.4 Evaluating the contribution of sense annotations

We wish to evaluate whether adding sense annotations improve parser accuracy against the baseline of training a model in the conventional way using only syntactic features. As noted above, we suspect that this semantic generalisation may help in cases where appropriate training data is sparse – that is, where the training data is from a different domain or only a small amount exists. So to evaluate the various methods in these conditions, we train models from small (23k token) training sets and large (96k token) training sets created from subsets of each corpus (WESCIENCE and LOGON). For the baseline, we train these models without modification. For each of the various methods of adding semantic tags, we then re-use each of these training sets to create new models after adding the appropriate additional features as described above, to evaluate whether these additional features improve parsing accuracy

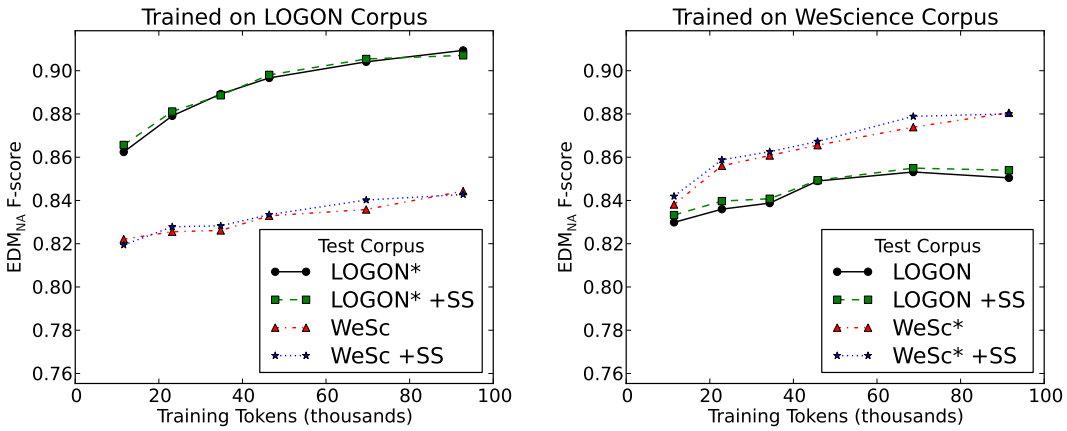
4 Results

We present an extensive summary of the results obtained using the various methods in Tables 2, 3, 4 and 5. In each case we show results for applying to the leaf and to the parent. Aggregating the results for each method, the differences range between substantially negative and modestly positive, with a large number of fluctuations due to statistical noise.

LDT is the least promising performer, with only one very modest improvement, and the largest decreases in performance, of around 1%. The HPWNF and HP_p(WNF) methods make changes in either direction – on average, over all four training/test combinations, there are very small drops in F-score of 0.02% for HPWNF, and 0.06% for HP_p(WNF), which indicates that neither of the methods is likely to be useful in reliably improving parser performance.

The SS methods are more promising. SS (WNF) and SS_p(WNF) methods yield an average improvement of 0.10% each, while SS (SST) and SS_p(SST) give average improvements of 0.12% and 0.13% respectively (representing an error rate reduction of around 1%). Interestingly, the increase in tagging accuracy we might expect using SuperSense Tagger only translates to a modest (and probably not significant) increase in parser performance, possibly because the tagger is not optimised for the domains in question. Amongst the statistical noise it is hard to discern overall trends; surprisingly, it seems that the size of the training corpus has relatively little to do with the success of adding these supersense annotations, and that the corpus being from an unmatched domain doesn't necessarily mean that sense-tagging will improve accuracy either. There may be a slight trend for sense annotations to be more useful when WESCIENCE is the training corpus (either in the small or the large size).

To gain a better insight into how the effects change as the size of the training corpus changes for the different domains, we created learning curves for the best-performing method, SS_p(SST) (although as noted above, all SS methods give similar levels of improvement), shown in Figure 3. Overall, these



(a) LOGON (b) WESCIENCE

Figure 3: EDM_{NA} learning curves for SS (SST) (supersense from SuperSense Tagger). '*' denotes in-domain training corpus.

graphs support the same conclusions as the tables – the gains we see are very modest and there is a slight tendency for WESCIENCE models to benefit more from the semantic generalisation, but no strong tendencies for this to work better for cross-domain training data or small training sets.

5 Conclusion

We have presented an initial study evaluating whether a fairly simple approach to using automatically-created coarse semantic annotations can improve HPSG parse selection accuracy using the English Resource Grammar. We have provided some weak evidence that adding features based on semantic annotations, and in particular word supersense, can provide modest improvements in parse selection performance in terms of dependency F-score, with the best-performing method SS_p(SST) providing an average reduction in error rate over 4 training/test corpus combinations of 1%. Other approaches were less promising. In all configurations, there were instances of F-score decreases, sometimes substantial.

It is somewhat surprising that we did not achieve reliable performance gains which were seen in the related work described above. One possible explanation is that the model training parameters were suboptimal for this data set since the characteristics of the data are somewhat different than without sense annotations. The failure to improve somewhat mirrors the results of Clark (2001), who was attempting to improve the parse ranking performance of the unification-based based probabilistic parser of Carroll and Briscoe (1996). Clark (2001) used dependencies to rank parses, and WordNet-based techniques to generalise this model and learn selectional preferences, but failed to improve performance over the structural (i.e. non-dependency) ranking in the original parser. Additionally, perhaps the changes we applied in this work to the parse ranking could possibly have been more effective with features based on semantic dependences as used by Fujita et al. (2007), although we outlined reasons why we wished to avoid this approach.

This work is preliminary and there is room for more exploration in this space. There is scope for much more feature engineering on the semantic annotations, such as using different levels of the semantic hierarchy, or replacing the purely lexical features instead of augmenting them. Additionally, more error analysis would reveal whether this approach was more useful for avoiding certain kinds of parser errors (such as PP-attachment).

Acknowledgements

NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program.

References

E. Agirre, T. Baldwin, and D. Martinez. 2008. Improving parsing and PP attachment performance with sense information. In *Proceedings of ACL-08: HLT*, pages 317–325, Columbus, Ohio, June.

Eneko Agirre, Kepa Bengoetxea, Koldo Gojenola, and Joakim Nivre. 2011. Improving dependency parsing with semantic classes. In *Proceedings of the 49th Annual Meeting of the Association of Computational Linguistics, ACL-HLT 2011 Short Paper, Portland, Oregon"*.

D. M. Bikel. 2002. Design of a multi-lingual, parallel-processing statistical parsing engine. In *Proceedings of the second international conference on Human Language Technology Research*, pages 178–182, San Francisco, CA, USA.

T. Brants. 2000. Tnt – a statistical part-of-speech tagger. In *Proceedings of the Sixth Conference on Applied Natural Language Processing*, pages 224–231, Seattle, Washington, USA, April.

T. Briscoe and J. Carroll. 2002. Robust accurate statistical annotation of general text. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation*, pages 1499–1504.

U. Callmeier. 2000. Pet – a platform for experimentation with efficient HPSG processing techniques. *Nat. Lang. Eng.*, 6(1):99–107.

J. Carroll and E. Briscoe. 1996. Apportioning development effort in a probabilistic lr pars- ing system through evaluation. In *Proceedings of the SIGDAT Conference on Empirical Methods in Natural Language Processing*, pages 92–100, Philadelphia, PA.

E. Charniak. 2000. A maximum-entropy-inspired parser. In *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference*, pages 132–139.

M. Ciaramita and Y. Altun. 2006. Broad-coverage sense disambiguation and information extraction with a supersense sequence tagger. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 594–602, Sydney, Australia, July.

S. Clark and J.R. Curran. 2004. Parsing the WSJ using CCG and log-linear models. In *Proceedings of the 42nd Meeting of the ACL*, pages 104–111.

S. Clark. 2001. *Class-based Statistical Models for Lexical Knowledge Acquisition*. Ph.D. thesis, University of Sussex.

M. Collins. 1997. Three generative, lexicalised models for statistical parsing. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics*, pages 16–23, Madrid, Spain, July.

R. Dridan and S. Oepen. 2011. Parser evaluation using elementary dependency matching. In *Proceedings of the 12th International Conference on Parsing Technologies*, pages 225–230, Dublin, Ireland, October.

Dan Flickinger. 2000. On building a more efficient grammar by exploiting types. *Nat. Lang. Eng.*, 6 (1):15–28.

S. Fujita, F. Bond, S. Oepen, and T. Tanaka. 2007. Exploiting semantic information for HPSG parse selection. In *ACL 2007 Workshop on Deep Linguistic Processing*, pages 25–32, Prague, Czech Republic, June.

D. Lin. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of the 17th international conference on Computational linguistics-Volume 2*, pages 768–774.

A. MacKinlay, R. Dridan, D. Flickinger, and T. Baldwin. 2011. Cross-domain effects on parse selection for precision grammars. *Research on Language & Computation*, 8(4):299–340.

R. Malouf. 2002. A comparison of algorithms for maximum entropy parameter estimation. In *Proceedings of the Sixth Conference on Natural Language Learning (CoNLL-2002)*, pages 49–55.

M. P. Marcus, M. A. Marcinkiewicz, and B. Santorini. 1993. Building a large annotated corpus of english: the penn treebank. *Comput. Linguist.*, 19(2):313–330.

D. McCarthy, R. Koeling, J. Weeds, and J. Carroll. 2004. Finding predominant word senses in untagged text. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, pages 279–es.

G.A. Miller. 1995. WordNet: a lexical database for English. *Communications of the ACM*, 38(11):39–41.

S. Oepen, D. Flickinger, K. Toutanova, and C.D. Manning. 2004. LinGO Redwoods: A rich and dynamic treebank for HPSG. *Research on Language & Computation*, 2(4):575–596.

M. Siegel and E.M. Bender. 2002. Efficient deep processing of japanese. In *Proceedings of the 3rd workshop on Asian language resources and international standardization-Volume 12*, pages 1–8.

E. Velldal. 2007. *Empirical Realization Ranking*. Ph.D. thesis, University of Oslo Department of Informatics.

G. Ytrestøl, D. Flickinger, and S. Oepen. 2009. Extracting and annotating Wikipedia sub-domains – towards a new eScience community resourc. In *Proceedings of the Seventh International Workshop on Treebanks and Linguistic Theories*, Groeningen, The Netherlands, January.

Y. Zhang, S. Oepen, and J. Carroll. 2007. Efficiency in unification-based n-best parsing. In *IWPT '07: Proceedings of the 10th International Conference on Parsing Technologies*, pages 48–59, Morristown, NJ, USA.