Treeblazing: Using External Treebanks to Filter Parse Forests for Parse Selection and Treebanking

Andrew MacKinlay[♢♡], **Rebecca Dridan**[♢♡], **Dan Flickinger**[♣], **Stephan Oepen**[♠] **and Timothy Baldwin**[♢♡]

[♢] Dept of Computer Science and Software Engineering, University of Melbourne, Australia
[♡] NICTA Victoria Research Laboratories, University of Melbourne, Australia
[♣] Center for Study of Language and Information, Stanford University, USA
[♠] Department of Informatics, Universitetet i Oslo, Norway
amack@csse.unimelb.edu.au, rdridan@csse.unimelb.edu.au,
danf@stanford.edu, oe@ifi.uio.no, tb@ldwin.net

Abstract

We describe "treeblazing", a method of using annotations from the GENIA treebank to constrain a parse forest from an HPSG parser. Combining this with self-training, we show significant dependency score improvements in a task of adaptation to the biomedical domain, reducing error rate by 9% compared to out-of-domain gold data and 6% compared to self-training. We also demonstrate improvements in treebanking efficiency, requiring 25% fewer decisions, and 17% less annotation time.

1 Introduction

Computational linguistic research is driven by the development of reference resources for specific tasks and languages. The advent of services such as Amazon's Mechanical Turk has driven down the cost of annotation considerably, assuming a given task can be broken down into piecemeal units which are intuitive and manageable for non-experts. This is not an option, however, for fine-grained tasks which require an expert understanding of a theory or domain, such as syntactic treebanking or discourse annotation.

Two main approaches have been adopted to efficiently create new resources: (1) domain adaptation, where a trained model from one domain is stochastically adapted to a new domain, using unlabelled data from the new domain (Daumé III and Marcu, 2006); and (2) annotation projection, where the labels in a pre-existing resource are semi-automatically translated into an independent formalism, e.g. in translating the PTB into the CCG formalism (Hockenmaier and Steedman, 2002). This paper looks at both of these approaches: domain adaptation from unannotated data in the form of self-training combined with resource translation over the GENIA treebank (Yuka et al., 2005), in the context of training an HPSG parse selection model for biomedical text, and using the GENIA treebank annotations and retrained parse selection model to accelerate treebanking.

Our contributions are: (1) we propose a series of methods for transferring annotation from a traditional phrase structure treebank to constrain the parse forest of a precision grammar; (2) we show that this constrained forest can be used to domain-adapt a parse selection model; (3) we demonstrate improvements in treebanking performance using the constrained forest; and (4) we develop a small-scale HPSG treebank for the biomedical domain.

2 Related Work

Domain adaptation is an active research area, triggered by the observation that parsers trained on one domain show decreased performance when used in other domains (Gildea, 2001). Much domain-adaptation work involves some small amount of in-domain data to tune a model, but McClosky et al. (2006) showed "self-training" using unannotated in-domain data could achieve significant improvements in parser accuracy.

In parsing-related research that has used annotated data, but in an incompatible format, we see two main use cases. The first uses an existing treebank to create a treebank for some completely different linguistic framework, generally to induce a grammar in that framework. Xia (1999) presents work on transforming Penn Treebank (PTB) trees into Lexicalized Tree Adjoining Grammar (LTAG) structures. The work of Hockenmaier and Steedman (2002) is roughly parallel, but targets Combinatory Categorial Grammar (CCG). The techniques include binarisation, adding an extra level

Proceedings of the 5th International Joint Conference on Natural Language Processing, pages 246–254,
Chiang Mai, Thailand, November 8 – 13, 2011.

of NP structure, and remapping node labels to CCG categories. An analog in the framework of Head-driven Phrase Structure Grammar (HPSG) is described by Miyao et al. (2004).

The second use case is to use the incompatible annotations to select the correct tree from the output of a compatible parser. Sometimes, a function over the original annotations produces a score of the new analysis candidates, and a single best analysis is selected (Wang et al., 1994; Niu et al., 2009). In other work, the original annotations do not uniquely disambiguate the parse forest, but the partial annotations can still be used. Riezler et al. (2002) used PTB annotations to partially disambiguate a parse forest built using a grammar in the Lexical-Functional Grammar (LFG) framework (Butt et al., 2002), and then built a model using the partially-disambiguated forest. Another use of partially disambiguated forests is described by Tanaka et al. (2005), who manually created a Japanese treebank (Bond et al., 2004) by selecting the best parses from the candidate parses offered as candidates from JaCy, an HPSG grammar of Japanese. The annotators reject or affirm *discriminants* to select the best tree, as is described in more detail in §3.1. Their data was already human-annotated with POS tags, which they used to constrain the parse forest, requiring on average 19.5% fewer decisions and 15% less time per tree.

Tanaka et al. (2005) used only POS tags. Our work can be viewed as a syntactic extension of this. We investigate strategies for adapting the English Resource Grammar (ERG: Flickinger (2000)) to the biomedical domain using information contained in the GENIA treebank (GTB), a corpus of 1,999 abstracts from PubMed in the domain of human blood cells and transcription factors, annotated according to a slightly simplified version of the PTB II annotation guidelines.

3 Setup

We explore two branches of experimentation using a common core of tools, resources and methods. This section describes the necessary details of our treebanking process, the test data we use, and some peculiarities of parsing biomedical data that affected our experiments.

3.1 Treebanking

All our experiments are based on the Redwoods treebanking methodology (Oepen et al., 2004), where the treebank is constructed by selecting from a parse forest of candidate trees licensed by the grammar. All experiments reported in this paper make use of the ERG. We first parse an input, and then select the (up to) 500 top-ranked parse trees according to a parse selection model. This set of parse trees is then presented to the human treebanker in the form of *discriminants* (Carter, 1997; Oepen et al., 2004). The discriminants used here correspond to instantiations of the 200 lexical and syntactic rules of the ERG, as well as the lexical entries themselves, but only those that correspond to ambiguity in the parse forest and can thus discriminate between candidate parse trees.

During treebanking, the annotator confirms or rejects some subset of discriminants, and at each stage, the Redwoods machinery performs inference to automatically reject those discriminants that are incompatible with the current set of manually-selected and inferred discriminants. This means that each manual decision can directly or indirectly rule out a large number of trees, and the number of decisions required is on average proportional to the logarithm of the number of parses (Tanaka et al., 2005).

Treebanking gives us a large number of rejected trees, along with the single correct gold tree, which can be used to build a discriminative parse selection model, in our case using TADM (Malouf, 2002). This is applied to parsing unseen data, and also for the next iteration of treebanking.

3.2 Data: a new biomedical HPSG treebank

In order to evaluate the impact of the proposed method on parser accuracy over biomedical text, we require a gold-standard treebank in the target domain. We use a subset of the data used in the GTB, created by first removing those abstracts (approximately half) that overlap with the GENIA event corpus (GEC: Kim et al. (2008)), to hold out for future work. From this filtered set, our test corpus comes from the 993 sentences of the first 118 abstracts (PubMed IDs 1279685 to 2077396).

Our treebankers both have detailed knowledge of the ERG, but no domain-specific biomedical expertise. As a proxy for this, they used the original GTB syntactic annotations when a tie-breaker was needed for ambiguities such as PP-attachment or co-ordination. The annotators were instructed to only refer to GTB trees when the ambiguity was not resolvable on linguistic grounds. The first 200

sentences of the corpus were double-annotated in each round of treebanking (agreement figures for unseen data are shown in §6.3)

The first round of annotation of a 500-sentence subset of the corpus served to determine a suitable parser configuration and calibrate between annotators using the 200-sentence overlap. From this, we developed a set of annotation guidelines, which will be made available with the corpus. One key domain-specific guideline related to the treatment of noun compounds, which are never disambiguated in the flat GTB structure. In the biomedical domain, noun compounds are generally left-bracketed – 83% of three-word compounds according to Nakov and Hearst (2005) – so we stipulated that noun compounds should be left-bracketed and adjectives attached high in cases of doubt, as a tie-breaking strategy.

We also used this first-iteration treebank to build a domain-tuned parse-selection model (duplicating the data 10 times and combining it with a larger out-of-domain corpus, using the DUPLIC method of MacKinlay et al. (2011) to provide improvements for sparse in-domain data). The external corpus was the WeScience corpus (Ytrestøl et al., 2009), a selection of Wikipedia articles on NLP. We improved the parser's handling of named entities, as described in §3.3, and then reparsed the treebank with the new parsing configuration and parse selection model, giving 866 parseable sentences. After updating the treebank according to the new guidelines using this new parse forest, and checking inter-annotator agreement on the overlap, we annotated the remaining sentences. All accuracy figures we report are over the data set of 669 trees complete at the time of experimentation.

3.3 Biomedical parsing setup

We parsed sentences using the ERG with the PET parser (Callmeier, 2000), which uses POS tags to constrain unknown words. Following Velldal et al. (2010), we primarily use the biomedically trained GENIA tagger (Tsuruoka et al., 2005), but defer to TnT (Brants, 2000) for tagging nominal elements, because it makes a useful distinction between common and proper nouns.

Biomedical text poses a unique set of challenges, mostly relating to named entities, such as proteins, DNA and cell lines. To address this, we used the GENIA tagger as a named-entity (NE) recogniser, treating named entities as atomic lexical items. However, the NE tagging is often overzealous and discards internal structure, misleading the parser. To overcome this, we supply multi-token NEs as both a single atomic NE token and the individual words, thus giving PET a lattice as input. The increased parse coverage and better parse quality made this a worthwhile strategy, with the downside of increased ambiguity, making parse selection more difficult.

4 Blazing

In §2, we reviewed work that uses linguistic information from superficially incompatible formalisms for treebanking or parse selection. Our experiments here use syntactic information from the GTB to partially disambiguate the parse forest produced by the ERG. We do this by disallowing certain candidate ERG trees on the basis of GTB-derived information, and we follow Tanaka et al. (2005) in denoting this process "blazing".[1]

As detailed below, we can use this partially disambiguated forest: (1) to train parse selection models; and (2) to reduce treebanking effort, abstractly similarly to Tanaka et al. (2005). The goal is not to apply all constraints from the GTB to the ERG parse trees; rather, we want to apply the *minimal* amount of constraints possible, while still sufficiently restricting the parse forest for our target application. We call the set of trees remaining after blazing *silver* trees, to represent the fact that they are not gold standard, but are generally of better quality than the discarded analyses.

For an iteration of blazing, we parse each GTB sentence, obtaining the top-500 trees according to the parse selection model. Each discriminant (as discussed in §3.1) which corresponds to a meaningful difference between the candidate trees (derivations) is supplied to the blazing module.

A given discriminant can be ruled out, ignored or asserted to be true, but we never make use of the latter, since we can just rule out incompatible discriminants, which are easier to identify. This process happens with all discriminants for a sentence simultaneously, so it is possible to rule out all parse trees. This may indicate that none of the candidate parses are desirable, or that the imperfect blazing process is not completely successful.

The blazing module is given the GTB XML source for the tree, and a set of discriminants, each of which includes the name of the rule or lexical

[1] Which is a term in forestry: marking trees for removal.

entry, as well as the corresponding character span in the source tree. It applies some pre-configured transformations to the GTB tree, and examines each discriminant for whether it should be ruled out, by comparing to the corresponding GTB constituents overlapping with the supplied character span. Primarily, these decisions depend on whether a discriminant is a *crossing-bracket discriminant*, i.e. corresponds to phrase structures in the ERG derivation trees which would have crossing brackets with any overlapping constituents (ignoring punctuation). As discussed below, in some configurations we can also use the rule name or lexical type to rule out particular discriminants.

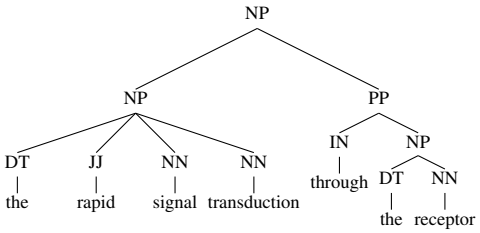
5 Parse Selection

Our first set of experiments was designed to evaluate the impact of blazing on parse selection, specifically in a domain-adaptation scenario. As mentioned in §3.1, parse selection is the process of selecting the top *n* parses, using a discriminative statistical model trained using the correct and incorrect trees from the treebanking process. However, as discussed in §2, statistical models are highly sensitive to differences in domain, and ideally, one would domain-tune off in-domain treebank data. Self-training (e.g. McClosky et al. (2006)) bypasses this need for in-domain annotations, by parsing the new domain with an out-of-domain model, treating the top-ranked parse as gold, and training a new model accordingly. In this work, we extend that idea by using blazing to transfer annotations from the GTB, hopefully filtering out incorrect trees in the process, and arrive at better-quality top-ranked parses.

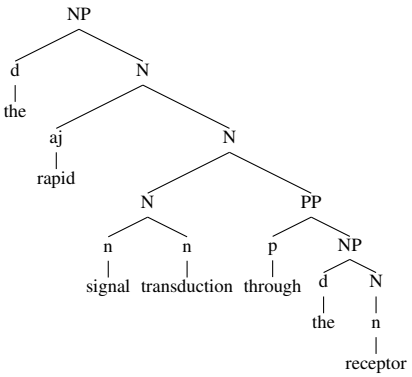
5.1 Blazing configurations

Blazing depends on the fact that the ERG and the GTB both have a theoretical linguistic underpinning, and so we expect they would share many assumptions about phrase structure, particularly for phenomena such as PP-attachment and co-ordination. However there are also disparities, even between the unlabelled bracketing of the GTB and ERG trees.

One pervasive difference is the attachment of specifiers and pre- and post-modifiers to NPs. The GTB attaches pre-modifiers and specifiers as low as possible, before attaching post-modifying PPs at a higher level, while the ERG makes the opposite decision and disallows this order of attachment. The other important difference is phrase branching – the GTB allows arbitrarily many children per node, while the ERG allows at most two. We show a sample NP exemplifying the differences in Figure 1.



(a) A sample native GTB subtree



(b) A simplified ERG subtree

Figure 1: Sample trees

One strategy for handling this is to make as few assumptions as possible while avoiding spurious conflicts. Denoted **IEP** (*ignore equal parent*), it involves ignoring GTB nodes with the same label as the parent when looking for crossing-bracket constituents. From the GTB tree in Figure 1, the blazing module would ignore the boundaries of the second-level NP when looking for crossing-bracket discriminants. This ensures that we never rule out the corresponding good subtree shown in the figure in favour of some invalid bracketing from the ERG that by chance has no conflicts with the GTB tree; meanwhile the PP would still be considered. Note that for a flat NP with no post-modifiers, no changes are necessary as the external boundaries still correspond with the edges of the top-level NP in the ERG, and the extra internal boundaries in the ERG have no effect since they cannot cause any crossing brackets.

Alternatively, to avoid discarding possibly valid syntactic information, we can attempt to account for the systematic differences by mapping the GTB as closely as possible to the structures we would expect in the ERG before looking

for crossing-bracket discriminants. Firstly, the phrases are binarised in a similar way to much previous work (Miyao et al., 2004; Hockenmaier and Steedman, 2002). We heuristically determine the head of each phrase using a simple category match between the phrase category and the POS, then recursively split multiple-branched nodes. This binarisation applies to all phrasal categories, not just NPs. We then systematically alter the attachment positions of determiners and pre-nominal modifiers, forcing them to attach as high as possible, but preserving the binary branching. As a lightweight but imperfect attempt to avoid creating invalid structures for appositions and conjunctions, this NP rearrangement is abandoned if any tokens are parentheses, commas or conjunctions. The transformation is labelled **RP** (*raise premods*).

5.2 Experimental Configuration

We create a parse forest by parsing 10747 sentences from a GTB subset not overlapping with the test corpus, using the WeScience model to determine the top 500 parses. The best-performing method we found to create parse selection models from this parse forest was to apply the blazing configurations to determine the *silver* trees, and then select the top-ranked parse from that set, according to the WeScience model. We call this top parse our *pseudo-gold* analysis. Remaining silver trees are ignored, while incorrect trees are used as negative data as usual. To show how much effect blazing has, we also used two other methods to select the pseudo-gold parse: random selection from that same top 500, to give us a baseline, and 'plain' self-training using the top-ranked parse without the blazing-based filtering, in each case using other trees from the forest as negative data. We then trained parse selection models using the gold standard out-of-domain (WeScience) data plus the in-domain pseudo-gold analyses from each configuration, and evaluated by parsing our test corpus.

5.3 Evaluation

We use two different styles of evaluation metric. In keeping with previous work using the ERG, we report exact match figures, denoted Acc_N, representing the percentage of sentences for which the exact gold tree was in the top *N* parses. Here, as in Zhang et al. (2007), we use Acc_1 and Acc_{10}. However, exact match can be very blunt for the fine-grained ERG analyses, giving no indication of how 'right' or 'wrong' the top analysis is. To supplement Acc_N, we use Elementary Dependency Match (EDM: Dridan and Oepen (2011)). This is based on triples extracted from the semantic output of the parser, providing a more granular measure of the quality of the analyses. We use the EDM_{NA} configuration that is arguably the most compatible with other dependency-based parser evaluation, although we make no claims of direct comparability.

Config	Gold Added	Acc A_1 / A_{10}	EDM_{NA} P / R / F
(WeSc only)	WeSc	12.3 / 39.2	82.4 / 79.2 / 80.7
Random	WeSc	6.1 / 20.0	70.7 / 70.2 / 70.5
Self-train	WeSc	12.9 / 39.2	82.4 / 80.3 / 81.3 *
IEP + S-T	WeSc	12.9 / 39.2	83.5 / 80.9 / 82.2 *** ††
RP + S-T	WeSc	13.3 / 40.1	83.8 / 81.2 / 82.5 *** †††

Table 1: Results over the test corpus. "WeSc only" shows parsing using a pure WeScience model. Other configurations used models trained from the same training sentence parse forest, setting a pseudo-gold tree either randomly, self-trained (best from a WeScience model), or blazing (highest-ranked of the silver trees, other silver trees discarded). The gold WeScience data is also used for training. Significance figures are against "WeSc only", (*: $p < 0.05$; ***:$p < 0.001$), and "Self-train", (††: $p < 0.01$; †††: $p < 0.001$)

5.4 Results

We present our results in Table 1, including the best-performing blazing configurations, the self-training results and the weak baseline trained on a random tree from the same GTB parse forest as used in blazing.

We also show the parsing accuracy results obtained using only out-of-domain data, designated "WeSc only", as a strong baseline. We see some evidence that self-training can be a useful domain-adaptation strategy, giving a weakly significant F-score improvement over using WeScience only. This echoes previously mentioned work, although has not been evaluated for this parser or grammar before. More importantly, our blazing strategy yields strongly significant F-score improvements over both the strong baseline out-of-domain model and the standard self-training.

5.5 Discussion

There is strong evidence that these blazing methods can help create a parse selection model to give

	IEP	RP
Discrims/Sent	144.2	144.2
Rejected/Sent	40.8	42.8
Unblazed Sents	3.9%	3.4%
Overblazed Sents	14.2%	15.3%
Usably Blazed Sents	81.9%	81.3%
Trees/Sent (overall)	423.3	423.3
Silver Trees/Sent (blazed)	98.4	88.5
Silver Trees/Sent (usable)	120.1	108.8

Table 2: Blazing Statistics, over all 10747 parseable training sentences. The first block shows discriminants available per sentence, and how many were rejected by the blazing (removing ≥ 1 tree). The second block shows percentage of unblazed sentences (no discriminants rejected), overblazed sentences (all trees removed) and usably-blazed sentences (≥ 1 removed and ≥ 1 silver tree remaining). The third block shows how many parses were produced initially, the average number of trees remaining over blazed sentences (inc. overblazed with 0 trees) and the average number remaining over usably blazed.

a significant boost in dependency score without needing additional human annotation. The exact match results are inconclusive – the best improvement is not significant, although the granular EDM metric may provide a better reflection of performance for downstream applications in any case.

In our initial investigations, a range of configurations failed to provide improvements over the baseline. If we don't augment the training data with human-annotated WeScience data, the performance drops. Also, if we don't use the self-training/blazing combination as described but instead treated all silver trees as pseudo-gold (i.e treat all remaining post-blazing parse trees as if they were manually marked as good), the model performs poorly. Table 2 provides some explanation for this. Over sentences which provide usable discriminative training data (at least one incorrect and one silver tree), on average more than 100 silver trees remain, so it is failing to disambiguate sufficiently between the ERG analyses. This is probably due to an imperfect transfer process and shallower, less precise GTB analyses.

6 Reducing treebanking labour

Blazing is designed to reduce the size of the parse forest, so it seems natural to evaluate its impact on the treebanking process, and whether we can reduce the amount of time and number of decisions required to enable more efficient treebanking.

6.1 Mapping between treebanks

In addition to the transformation strategies mentioned in §5.1, we used a number of additional strategies (most of which we had already tried initially, for parse selection, but rejected). One rule concerns the internals of noun compounds, which are flat in the GTB; we may wish to add some structure to them. As discussed in §3.2, biomedical noun compounds are predominantly left-bracketed, and left-bracketing was also our tie-breaking policy for annotating the test set. In the **BNC** strategy (*bracket noun compounds*), we added bracketing to noun compounds to have noun sequences maximally left bracketed, and adjectives attaching as high as possible. This makes assumptions which are not explicitly licensed by the data (and arguably overfits to our data set), so this transformation is only applied where no useful distinctions are made by less restrictive approaches.

We also use a mapping strategy which does not make changes to the tree structure but which use the POS labels to rule out trees, denoted **MP** for *map POS*. It uses the prefixes of the lexical types – e.g. a simple transitive verb would have the lexical type *v_np_le*, where the prefix 'v' indicates 'verb'. We used a mapping constructed by manual inspection of a correspondence matrix between the POS tags produced by TnT (Brants, 2000) and the lexical type prefixes from the gold-standard ERG parse of the same sentences over a WeScience subset. This gave us the matching ERG type prefixes for 20 PTB/GTB POS tags, which are mostly what we would expect for the open classes – e.g. *VB** verb tags map to the 'v' prefix.

During mapping, given a pairing of a GENIA tree and a set of ERG discriminants, for each POS tag or inner node in the GENIA tree, we find all lexical discriminants with the same character span. If there are multiple discriminants with different matching labels, and there is at least one allowed and one disallowed by the mapping, then we reject all disallowed discriminants. This is less sophisticated than the mapping technique of Tanaka et al. (2005) for various reasons.

6.2 Selecting a blazing strategy

There are a range of blazing strategies and combinations thereof, with varying levels of validity and restrictiveness. Ideally, during treebanking we would start with a more restrictive blazing strat-

		Standard		Blazed	
Ann 1	Decisions	6.25	7	3.51	4
	Time (sec)	150	144	113	107
Ann 2	Decisions	6.42	7	4.68	4
	Time (sec)	105	101	96	80

Table 3: Number of decisions and treebanking time (mean then median) using the fallback blazing configuration (80 sentences for each column)

	Ann. 1			
	Std	Blz		
Agreed Sentences	42.5	45.0	Std	Ann. 2
Agreed, excl rej	32.4	33.3		
Rejection F-score	80.0	82.4		
Constituent F-score	88.7	87.6		
Agreed Sentences	42.5	57.5	Blz	
Agreed, excl rej	39.5	45.2		
Rejection F-score	44.4	78.3		
Constituent F-score	86.2	84.8		

Table 4: Agreement figures for different combinations of blazed and unblazed overlap between annotators 1 and 2, with 40 sentences per cell. 'Agreed' is the percentage of those with an identical tree selected, or all trees rejected; 'excl rej' ignores sentences rejected by either annotator. 'Constituent F-score' (also excludes rejections) is the harmonic mean of the labelled per-constituent precision. 'Rejection F-score' is the harmonic mean of the precision of rejection decisions.

egy, and dynamically fall back to a less restrictive strategy, but this capability is not yet present in the Redwoods machinery. Our approach is based on the number of decisions being logarithmic in the number of trees. If we can get roughly 40 silver trees, the remaining treebanking is very tractable and fast, only requiring a few decisions chosen from a handful of remaining discriminants. However further restriction below this saves relatively little time, and increases the chance of the blazing removing the correct tree, which we wish to avoid (the machinery does not allow the treebanker to 'undo' either manual or blazed decisions, except by clearing and restarting).

The parse forest for treebanking was created by having a list of candidate blazing strategies using various combinations of **IEP** (least restrictive), **RP**, **BNC** and **MP** – with the combination **RP +BNC +MP** as the most restrictive. For each sentence, we select the least restrictive strategy which still gives us fewer remaining trees than a threshold of 40. If no strategies do so, we use the strategy which gives us the fewest trees for the given sentence. Using another subset of the GENIA treebank containing 864 parseable sentences, 48% of sentences came below the threshold of 40, while 36% were above and 16% were not disambiguated at all. Most sentences (41%) used the least restrictive configuration using **IEP** alone.

6.3 Blazed Treebanking Results

For this strategy to be useful for treebanking, it should be both more efficient, in terms of fewer decisions and less annotation time, and valid, in terms of not introducing a bias when compared to conventional unblazed treebanking. To evaluate these questions, we selected 160 sentences at random from the previously described parse forest of 864 sentences. These sentences were divided randomly into four equal-sized groups: blazed for both annotators, standard for both annotators, and two groups blazed for one annotator only, so we could compare data about timing and decisions between the standard and blazed sentences for each annotator, and inter-annotator agreement for each possible combination of blazed and standard treebanking. The divisions took no account of whether we were able to usably blaze the sentences, reflecting the real-world scenario, so some sentences in the blazed configuration had no restrictions applied. The items were presented to the annotators so they could not tell whether the other annotator was treebanking in standard or blazed configuration, to prevent subconscious biases affecting inter-annotator agreement. The experiments were conducted after both annotators had already familiarised themselves with the treebanking environment as well as the characteristics of the domain and the annotation guidelines.

Annotators worked in a distraction-free environment so we could get accurate timing figures. The treebanking machinery records how many decisions were made as well as annotation time, both important factors in annotation efficiency. The results for efficiency are shown in Table 3 where we see a 43% reduction in the mean decisions required for annotator 1, and 27% reduction for annotator 2. Annotator 1 also shows substantial 25% reduction in mean annotation time, but the time decrease for annotator 2 is only 8%. In 30% of successfully-blazed sentences, the annotators cleared all blazed decisions, suggesting it is sometimes too zealous.

For agreement, we show results for the strictest

possible criterion of exact tree match. For a less blunt metric that still roughly reflects agreement, we also follow Tanaka et al. (2005) in reporting the (micro-averaged) harmonic mean of precision across labelled constituents indexed by character span, where constituents selected by both annotators are treated as gold (inaccurately denoted 'F-score' for brevity). Annotators should also agree on rejected trees, where no parses are valid. In Table 4, we show exact match agreement accuracy (identical trees and matching rejections both count as correct), as well as the same figure ignoring sentences rejected by either, and the harmonic mean of precision of both labelled constituents and tree rejections. The figures are similar between cells, with notable exceptions being higher exact match when both annotators had blazed forests, and a surprising dip in the rejection "F-score" in the bottom left cell. The latter is partially because the rejection scores are based on small numbers of trees (5–10, the union of the sets of rejected trees), so are sensitive to small numbers of disagreements. In this particular case, of 7 trees rejected by either annotator, 2 were rejected by both.

6.4 Blazed Treebanking Discussion

The reductions in mean numbers of decisions strongly support the efficacy of this technique, although the discrepancies between the annotators suggest that the different treebanking techniques may be more or less amenable to speed-up using these tools. The timing figures are somewhat more equivocal, although still a substantial 25% for annotator 1. This is partially to be expected, since some of the treebanking will be taken up with unavoidable tasks such as evaluating whether the final tree is acceptable that blazing cannot avoid. However, the 8% reduction in mean annotation time for annotator 2 is still fairly modest. This could be affected by annotator 2's more extensive treebanking experience leading to a lower baseline time, with less room for improvement, but as we still see a 21% reduction in median parsing time this could be due to a few outlier sentences inflating the mean for the blazed configuration.

For agreement, we are primarily concerned here with whether blazing here introduces a bias that is distinguishable from what we see when annotators are working under standard non-blazed conditions – which may be manifested in decreased agreement between configurations where only one annotator has blazed data, and when both have non-blazed data. Thus the fact that we see quite similar agreement figures between the half-blazed and standard configurations is very encouraging (apart from the low F-score for rejections in one cell). This small amount of data suggests that any changes in the resultant trees introduced by blazing are hard to distinguish from the inevitable "background noise". Given this, the fact that we see a noticeably higher exact match score when both annotators have blazed sentences suggests we may be justified in using blazing to improve inter-annotator agreement, although the lower constituent score may indicate we have insufficient data to reach that conclusion.

7 Conclusion and Future Work

We have presented a procedure for blazing – using annotations from an external phrase structure treebank to constrain the parse forest produced by a precision HPSG grammar. Our work used the GENIA treebank and the ERG as the target grammar, although it would in principle be applicable to any similar phrase structure treebank and other grammars or even frameworks. The GENIA trees were mapped onto corresponding ERG parse forests and used to exclude incompatible trees. In conjunction with self-training, we used this to create a parse selection model for the ERG adapted to the biomedical domain. We also used it as a prefilter to the treebanking process to improve treebanking efficiency, and created an HPSG treebank of biomedical text.

For future work, we would investigate whether this training data can be useful to augment a small in-domain human-annotated treebank, and whether the methods do indeed generalise to other corpora and grammars.

Acknowledgements

We thank Phil Blunsom and Jonathon Read for their helpful advice. 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. The first author's conference travel is funded by a Google Travel Award. Large-scale experimentation was in part made possible though access to the TITAN HPC facilities at the University of Oslo.

References

Francis Bond, Sanae Fujita, Chikara Hashimoto, Kaname Kasahara, Shigeko Nariyama, Eric Nichols, Akira Ohtani, Takaaki Tanaka, and Shigeaki Amano. 2004. The Hinoki treebank: A treebank for text understanding. In *In Proc. of the First IJCNLP, Lecture Notes in Computer Science*, pages 554–559. Springer Verlag.

Thorsten 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. Association for Computational Linguistics.

Miriam Butt, Helge Dyvik, Tracy Holloway King, Hiroshi Masuichi, and Christian Rohrer. 2002. The parallel grammar project. In *Proceedings of COLING-2002 Workshop on Grammar Engineering and Evaluation*, pages 1–7.

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

David Carter. 1997. The TreeBanker. A tool for supervised training of parsed corpora. In *Proceedings of the Workshop on Computational Environments for Grammar Development and Linguistic Engineering*, pages 9–15, Madrid, Spain.

Hal Daumé III and Daniel Marcu. 2006. Domain adaptation for statistical classifiers. *Journal of Artificial Intelligence Research*, 26(1):101–126.

Rebecca Dridan and Stephan Oepen. 2011. Parser evaluation using elementary dependency matching (to appear). In *Proceedings of the 12th International Conference on Parsing Technologies*.

Dan Flickinger. 2000. On building a more efficient grammar by exploiting types. *Natural Language Engineering*, 6 (1):15–28.

Daniel Gildea. 2001. Corpus variation and parser performance. In *Proceedings of EMNLP 2001*, pages 167–202, Pittsburgh, USA.

Julia Hockenmaier and Mark Steedman. 2002. Acquiring compact lexicalized grammars from a cleaner treebank. In *Proceedings of Third International Conference on Language Resources and Evaluation*, Las Palmas.

Jin-Dong Kim, Tomoko Ohta, and Jun'ichi Tsujii. 2008. Corpus annotation for mining biomedical events from literature. *BMC Bioinformatics*, 9:10.

Andrew MacKinlay, Rebecca Dridan, Dan Flickinger, and Timothy Baldwin. 2011. Cross-domain effects on parse selection for precision grammars (to appear). *Research on Language and Computation*.

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.

David McClosky, Eugene Charniak, and Mark Johnson. 2006. Reranking and self-training for parser adaptation. In *Proceedings of the COLING/ACL 2006*, pages 337–344, Sydney, Australia.

Yusuke Miyao, Takashi Ninomiya, and Jun'ichi Tsujii. 2004. Corpus-oriented grammar development for acquiring a head-driven phrase structure grammar from the penn treebank. In *Proceedings of IJCNLP-04*.

Preslav Nakov and Marti Hearst. 2005. Search engine statistics beyond the n-gram: application to noun compound bracketing. In *Proceedings of the Ninth Conference on Computational Natural Language Learning*, CONLL '05, pages 17–24, Stroudsburg, PA, USA. Association for Computational Linguistics.

Zheng-Yu Niu, Haifeng Wang, and Hua Wu. 2009. Exploiting heterogeneous treebanks for parsing. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 46–54, Suntec, Singapore, August. Association for Computational Linguistics.

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.

Stefan Riezler, Tracy H. King, Ronald M. Kaplan, Richard Crouch, John T. Maxwell, III, and Mark Johnson. 2002. Parsing the wall street journal using a lexical-functional grammar and discriminative estimation techniques. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 271–278, Stroudsburg, PA, USA. Association for Computational Linguistics.

Takaaki Tanaka, Francis Bond, Stephan Oepen, and Sanae Fujita. 2005. High precision treebanking—blazing useful trees using POS information. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 330–337, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Yoshimasa Tsuruoka, Yuka Tateishi, Jin-Dong Kim, Tomoko Ohta, John McNaught, Sophia Ananiadou, and Jun'ichi Tsujii. 2005. Developing a robust part-of-speech tagger for biomedical text. *Advances in Informatics*, pages 382–392.

Erik Velldal, Lilja Øvrelid, and Stephan Oepen. 2010. Resolving speculation: MaxEnt cue classification and dependency-based scope rules. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, pages 48–55, Uppsala, Sweden, July. Association for Computational Linguistics.

Jong-Nae Wang, Jing-Shin Chang, and Keh-Yih Su. 1994. An automatic treebank conversion algorithm for corpus sharing. In *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, pages 248–254, Las Cruces, New Mexico, USA, June. Association for Computational Linguistics.

F. Xia. 1999. Extracting tree adjoining grammars from bracketed corpora. In *Proceedings of the 5th Natural Language Processing Pacific Rim Symposium (NLPRS-99)*, pages 398–403.

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

Tateisi Yuka, Akane Yakushiji, Tomoko Ohta, and Jun'ichi Tsujii. 2005. Syntax annotation for the GENIA corpus. In *Proceedings of the IJCNLP 2005*, pages 222–227, Jeju Island, Korea.

Yi Zhang, Stephan Oepen, and John Carroll. 2007. Efficiency in unification-based N-best parsing. In *Proceedings of IWPT 2007*, pages 48–59, Prague, Czech Republic.