Parser-Based Retraining for Domain Adaptation of Probabilistic Generators

Deirdre Hogan, Jennifer Foster, Joachim Wagner and Josef van Genabith
National Centre for Language Technology
School of Computing
Dublin City University
Ireland
{dhogan, jfoster, jwagner, josef}@computing.dcu.ie

Abstract

While the effect of domain variation on Penn-treebank-trained probabilistic parsers has been investigated in previous work, we study its effect on a Penn-Treebank-trained probabilistic generator. We show that applying the generator to data from the British National Corpus results in a performance drop (from a BLEU score of 0.66 on the standard WSJ test set to a BLEU score of 0.54 on our BNC test set). We develop a generator retraining method where the domain-specific training data is automatically produced using state-of-the-art parser output. The retraining method recovers a substantial portion of the performance drop, resulting in a generator which achieves a BLEU score of 0.61 on our BNC test data.

1 Introduction

Grammars extracted from the Wall Street Journal (WSJ) section of the Penn Treebank have been successfully applied to natural language parsing, and more recently, to natural language generation. It is clear that high-quality grammars can be extracted for the WSJ domain but it is not so clear how these grammars scale to other text genres. Gildea (2001), for example, has shown that WSJ-trained parsers suffer a drop in performance when applied to the more varied sentences of the Brown Corpus. We investigate the effect of domain variation in treebank-grammar-based generation by applying a WSJ-trained generator to sentences from the British National Corpus (BNC).

As with probabilistic parsing, probabilistic generation aims to produce the most likely output(s) given the input. We can distinguish three types of probabilistic generators, based on the type of probability model used to select the most likely sentence. The first type uses an n-gram language model, e.g. (Langkilde, 2000), the second type uses a probability model defined over trees or feature-structure-annotated trees, e.g. (Cahill and van Genabith, 2006), and the third type is a mixture of the first and second type, employing n-gram and grammar-based features, e.g. (Velldal and Oepen, 2005). The generator used in our experiments is an instance of the second type, using a probability model defined over Lexical Functional Grammar c-structure and f-structure annotations (Cahill and van Genabith, 2006; Hogan et al., 2007).

In an initial evaluation, we apply our probabilistic WSJ-trained generator to BNC material, and show that the generator suffers a substantial performance degradation, with a drop in BLEU score from 0.66 to 0.54. We then turn our attention to the problem of adapting the generator so that it can more accurately generate the 1,000 sentences in our BNC test set. The problem of adapting any NLP system to a domain different from the domain upon which it has been trained and for which no gold standard training material is available is a very real one, and one which has been the focus of much recent research in parsing. Some success has been achieved by training a parser, not on gold standard hand-corrected trees, but on parser output trees. These parser output trees can by produced by a second parser in a *co-training* scenario (Steedman et al., 2003), or by the same parser with a reranking component in a type of *self-training* scenario (McClosky et al., 2006). We tackle the problem of domain adaptation in generation in a similar way, by training the generator on domain specific *parser output* trees instead of manually corrected gold standard trees. This experiment achieves promising results, with an increase in BLEU score from 0.54 to 0.61. The method is generic and can be applied to other probabilistic generators (for which suitable training material can be automatically produced).

2 Background

The natural language generator used in our experiments is the WSJ-trained system described in Cahill and van Genabith (2006) and Hogan et al. (2007). Sentences are generated from Lexical Functional Grammar (LFG) f-structures (Kaplan and Bresnan, 1982). The f-structures are created automatically by annotating nodes in the gold standard WSJ trees with LFG functional equations and then passing these equations through a constraint solver (Cahill et al., 2004). The generation algorithm is a chart-based one which works by finding the most probable tree associated with the input f-structure. The yield of the most probable tree is the output sentence. An annotated PCFG, in which the non-terminal symbols are decorated with functional information, is used to generate the most probable tree from an f-structure. Cahill and van Genabith (2006) attain 98.2% coverage and a BLEU score of 0.6652 on the standard WSJ test set (Section 23). Hogan et al. (2007) describe an extension to the system which replaces the annotated PCFG selection model with a more sophisticated history-based probabilistic model. Instead of conditioning the righthand side of a rule on the lefthand non-terminal and its associated functional information alone, the new model includes non-local conditioning information in the form of functional information associated with ancestor nodes of the lefthand side category. This system achieves a BLEU score of 0.6724 and 99.9% coverage.

Other WSJ-trained generation systems include Nakanishi et al. (2005) and White et al. (2007). Nakanishi et al. (2005) describe a generator trained on a HPSG grammar derived from the WSJ Section of the Penn Treebank. On sentences of ≤ 20 words in length, their system attains coverage of 90.75% and a BLEU score of 0.7733. White et al. (2007) describe a CCG-based realisation system which has been trained on logical forms derived from CCG-Bank (Hockenmaier and Steedman, 2005), achieving 94.3% coverage and a BLEU score of 0.5768 on WSJ23 for all sentence lengths. The input structures upon which these systems are trained vary in form and specificity, but what the systems have in common is that their various input structures are derived from Penn Treebank trees.

3 The BNC Test Data

The new English test set consists of 1,000 sentences taken from the British National Corpus (Burnard, 2000). The BNC is a one hundred million word balanced corpus of British English from the late twentieth century. Ninety per cent of it is written text, and the remaining 10% consists of transcribed spontaneous and scripted spoken language. The BNC sentences in the test set are not chosen completely at random. Each sentence in the test set has the property of containing a word which appears as a verb in the BNC but not in the usual training sections of the Wall Street Journal section of the Penn Treebank (WSJ02-21). Sentences were chosen in this way so that the resulting test set would be a difficult one for WSJ-trained systems. In order to produce input f-structures for the generator, the test sentences were manually parsed by one annotator, using as references the Penn Treebank trees themselves and the Penn Treebank bracketing guidelines (Bies et al., 1995). When the two references did not agree, the guidelines took precedence over the Penn Treebank trees. Difficult parsing decisions were documented. Due to time constraints, the annotator did not mark functional tags or traces. The context-free gold standard parse trees were transformed into f-structures using the automatic procedure of Cahill et al. (2004).

4 Experiments

Experimental Setup In our first experiment, we apply the original WSJ-trained generator to our BNC test set. The gold standard trees for our BNC test set differ from the gold standard Wall Street Journal trees, in that they do not contain Penn-II traces or functional tags. The process which produces f-structures from trees makes use of trace and functional tag information, if available. Thus, to ensure that the training and test input f-structures are created in the same way, we use a version of the generator which is trained using gold standard WSJ trees *without* functional tag or trace information. When we test this system on the WSJ23 f-structures (produced in the same way as the WSJ training material), the BLEU score decreases slightly from 0.67 to 0.66. This is our baseline system.

In a further experiment, we attempt to adapt the generator to BNC data by using BNC trees as training material. Because we lack gold standard BNC trees (apart from those in our test set), we try instead to use parse trees produced by an accurate parser. We choose the Charniak and Johnson reranking parser because it is freely available and achieves state-of-the-art accuracy (a Parseval f-score of 91.3%) on the WSJ domain (Charniak and Johnson, 2005). It is, however, affected by domain variation — Foster et al. (2007) report that its f-score drops by approximately 8 percentage points when applied to the BNC domain. Our training size is 500,000 sentences. We conduct two experiments: the first, in which 500,000 sentences are extracted randomly from the BNC (minus the test set sentences), and the second in which only shorter sentences, of length ≤ 20 words, are chosen as training material. The rationale behind the second experiment is that shorter sentences are less likely to contain parser errors.

We use the BLEU evaluation metric for our experiments. We measure both coverage and full coverage. Coverage measures the number of cases for which the generator produced some kind of output. Full coverage measures the number of cases for which the generator produced a tree spanning all of the words in the input.

Results The results of our experiments are shown in Fig. 1. The first row shows the results we obtain when the baseline system is applied to the f-structures derived from the 1,000 BNC gold standard parse trees. The second row shows the results on the same test set for a system trained on Charniak and Johnson parser output trees for 500,000 BNC sentences. The results in the final row are obtained by training the generator on Charniak and Johnson parser output trees for 500,000 BNC sentences of length ≤ 20 words in length.

Discussion As expected, the performance of the baseline system degrades when faced with out-of-domain test data. The BLEU score drops from a 0.66 score for WSJ test data to a 0.54 score for the BNC test data, and full coverage drops from 85.97% to 68.77%. There is a substantial improvement, however, when the generator is trained on BNC data. The BLEU score jumps from 0.5358 to 0.6135. There are at least two possible reasons why a BLEU score of 0.66 is not obtained: The first is that the quality of the f-structure-annotated trees upon which the generator has been trained has degraded. For the baseline system, the generator is trained on f-structure-annotated trees derived from *gold* trees. The new system is trained on f-structure-annotated parser output trees, and the performance of Charniak and Johnson's parser degrades when applied to BNC data (Foster et al., 2007). The second reason has been suggested by Gildea (2001): WSJ data is easier to learn than the more varied data in the Brown Corpus or BNC. Perhaps even if gold standard BNC parse trees were available for training, the system would not behave as well as it does for WSJ material.

It is interesting to note that training on 500,000 shorter sentences does not appear to help. We hypothesized that it would improve results because shorter sentences are less likely to contain parser errors. The drop in full coverage from 86.69% to 79.58% suggests that the number of short sentences needs to be increased so that the size of the training material stays constant.

5 Conclusion

We have investigated the effect of domain variation on a LFG-based WSJ-trained generation system by testing the system's performance on 1,000 sentences from the British National Corpus. Performance drops from a BLEU score of 0.66 on WSJ test data to 0.54 on the BNC test set. Encouragingly, we have also shown that domain-specific training material produced by a parser can be used to claw back a significant portion of this performance degradation. Our method is general and could be applied to other WSJ-trained generators (e.g. (Nakanishi et

Train	BLEU	Coverage	Full Coverage
WSJ02-21	0.5358	99.1	68.77
BNC(500k)	0.6135	99.1	86.69
BNC(500k) $\leq$ *20 words*	0.5834	99.1	79.58

Figure 1: Results for 1,000 BNC Sentences

al., 2005; White et al., 2007)). We intend to continue this research by training our generator on parse trees produced by a BNC-self-trained version of the Charniak and Johnson reranking parser (Foster et al., 2007). We also hope to extend the evaluation beyond the BLEU metric by carrying out a human judgement evaluation.

Acknowledgments

This research has been supported by the Enterprise Ireland Commercialisation Fund (CFTD/2007/229), Science Foundation Ireland (04/IN/I527) and the IRCSET Embark Initative (P/04/232). We thank the Irish Centre for High End Computing for providing computing facilities.

References

Ann Bies, Mark Ferguson, Karen Katz, and Robert MacIntyre. 1995. Bracketing guidelines for treebank II style, Penn Treebank project. Technical Report Tech Report MS-CIS-95-06, University of Pennsylvania, Philadelphia, PA.

Lou Burnard. 2000. User reference guide for the British National Corpus. Technical report, Oxford University Computing Services.

Aoife Cahill and Josef van Genabith. 2006. Robust PCFG-based generation using automatically acquired lfg approximations. In *Proceedings of the 21st COLING and the 44th Annual Meeting of the ACL*, pages 1033–1040, Sydney.

Aoife Cahill, Michael Burke, Ruth O'Donovan, Josef van Genabith, and Andy Way. 2004. Long-Distance Dependency Resolution in Automatically Acquired Wide-Coverage PCFG-Based LFG Approximations. In *Proceedings of the 42nd Meeting of the ACL*, pages 320–327, Barcelona.

Eugene Charniak and Mark Johnson. 2005. Course-to-fine n-best-parsing and maxent discriminative reranking. In *Proceedings of the 43rd Annual Meeting of the ACL*, pages 173–180, Ann Arbor.

Jennifer Foster, Joachim Wagner, Djamé Seddah, and Josef van Genabith. 2007. Adapting WSJ-trained parsers to the British National Corpus using in-domain self-training. In *Proceedings of the Tenth IWPT*, pages 33–35, Prague.

Daniel Gildea. 2001. Corpus variation and parser performance. In *Proceedings of EMNLP*, Pittsburgh.

Julia Hockenmaier and Mark Steedman. 2005. Ccgbank: Users' manual. Technical report, Computer and Information Science, University of Pennsylvania.

Deirdre Hogan, Conor Cafferkey, Aoife Cahill, and Josef van Genabith. 2007. Exploiting multi-word units in history-based probabilistic generation. In *Proceedings of the joint EMNLP/CoNLL*, pages 267–276, Prague.

Ron Kaplan and Joan Bresnan. 1982. Lexical Functional Grammar: a Formal System for Grammatical Representation. In Joan Bresnan, editor, *The Mental Representation of Grammatical Relations*, pages 173–281. MIT Press.

Irene Langkilde. 2000. Forest-based statistical sentence generation. In *Proceedings of NAACL*, Seattle.

David McClosky, Eugene Charniak, and Mark Johnson. 2006. Effective self-training for parsing. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 152–159, New York City.

Hiroko Nakanishi, Yusuke Miyao, and Jun'ichi Tsujii. 2005. Probabilistic methods for disambiguation of an HPSG-based chart generator. In *Proceedings of the Ninth IWPT*, pages 93–102, Vancouver.

Mark Steedman, Miles Osbourne, Anoop Sarkar, Stephen Clark, Rebecca Hwa, Julia Hockenmaier, Paul Ruhlen, Steven Baker, and Jeremiah Crim. 2003. Bootstrapping statistical parsers from small datasets. In *Proceedings of EACL*, pages 331–338, Budapest.

Erik Velldal and Stephan Oepen. 2005. Maximum entropy models for realization ranking. In *Proceedings of the MT-Summit*, Phuket.

Michael White, Rajakrishnan Rajkumar, and Scott Martin. 2007. Towards broad coverage surface realization with CCG. In *Proceedings of the Workshop on Using Corpora for NLG: Language Generation and Machine Translation (UCNLG+MT)*, pages 267–276, Copenhagen.