Using existing systems to supplement small amounts of annotated grammatical relations training data [*]

Alexander Yeh
Mitre Corp.
202 Burlington Rd.
Bedford, MA 01730
USA
asy@mitre.org

Abstract

Grammatical relationships (GRs) form an important level of natural language processing, but different sets of GRs are useful for different purposes. Therefore, one may often only have time to obtain a small training corpus with the desired GR annotations. To boost the performance from using such a small training corpus on a transformation rule learner, we use existing systems that find related types of annotations.

1 Introduction

Grammatical relationships (GRs), which include arguments (e.g., subject and object) and modifiers, form an important level of natural language processing. Examples of GRs in the sentence

Today, my dog pushed the ball on the floor.

are *pushed* having the subject *my dog*, the object *the ball* and the time modifier *Today*, and *the ball* having the location modifier *on (the floor)*. The resulting annotation is

my dog −`subj`→ *pushed*
on −`mod-loc`→ *the ball*

etc. GRs are the objects of study in relational grammar (Perlmutter, 1983). In the SPARKLE project (Carroll et al., 1997), GRs form the top layer of a three layer syntax scheme. Many systems (e.g., the KERNEL system (Palmer et al., 1993)) use GRs as an intermediate form when determining the semantics of syntactically parsed text. GRs are often stored in structures similar to the F-structures of lexical-functional grammar (Kaplan, 1994).

A complication is that different sets of GRs are useful for different purposes. For example, Ferro et al. (1999) is interested in semantic interpretation, and needs to differentiate between time, location and other modifiers. The SPARKLE project (Carroll et al., 1997), on the other hand, does not differentiate between these types of modifiers. As has been mentioned by John Carroll (personal communication), combining modifier types together is fine for information retrieval. Also, having less differentiation of the modifiers can make it easier to find them (Ferro et al., 1999).

Furthermore, unless the desired set of GRs matches the set already annotated in some large training corpus,[1] one will have to either manually write rules to find the GRs, as done in Aït-Mokhtar and Chanod (1997), or annotate a new training corpus for the desired set. Manually writing rules is expensive, as is annotating a large corpus.

Often, one may only have the resources to produce a small annotated training set, and many of the less common features of the set's

[*] This paper reports on work performed at the MITRE Corporation under the support of the MITRE Sponsored Research Program. Marc Vilain provided the motivation to find GRs. Warren Greiff suggested using randomization-type techniques to determine statistical significance. Sabine Buchholz and John Carroll ran their GR finding systems over our data for the experiments. Jun Wu provided some helpful explanations. Christine Doran and John Henderson provided helpful editing. Three anonymous reviewers provided helpful suggestions.

[1] One example is a memory-based GR finder (Buchholz et al., 1999) that uses the GRs annotated in the Penn Treebank (Marcus et al., 1993).

domain may not appear at all in that set. In contrast are existing systems that perform well (probably due to a large annotated training set or a set of carefully hand-crafted rules) on related (but different) annotation standards. Such systems will cover many more domain features, but because the annotation standards are slightly different, some of those features will be annotated in a different way than in the small training and test set.

A way to try to combine the different advantages of these small training data sets and existing systems which produce related annotations is to use a sequence of two systems. We first use an existing annotation system which can handle many of the less common features, i.e., those which do not appear in the small training set. We then train a second system with that same small training set to take the output of the first system and correct for the differences in annotations. This approach was used by Palmer (1997) for word segmentation. Hwa (1999) describes a somewhat similar approach for finding parse brackets which combines a fully annotated related training data set and a large but incompletely annotated final training data set. Both these works deal with just one (word boundary) or two (start and end parse bracket) annotation label types and the same label types are used in both the existing annotation system/training set and the final (small) training set. In comparison, our work handles many annotation label types, and the translation from the types used in the existing annotation system to the types in the small training set tends to be both more complicated and most easily determined by empirical means. Also, the type of baseline score being improved upon is different. Our work adds an existing system to improve the rules learned, while Palmer (1997) adds rules to improve an existing system's performance.

We use this related system/small training set combination to improve the performance of the transformation-based error-driven learner described in Ferro et al. (1999). So far, this learner has started with a blank initial labeling of the GRs. This paper describes experiments where we replace this blank initial labeling with the output from an existing GR finder that is good at a somewhat different set of GR annotations. With each of the two existing GR finders that we use, we obtained improved results, with the improvement being more noticeable when the training set is smaller.

We also find that the existing GR finders are quite uneven on how they improve the results. They each tend to concentrate on improving the recovery of a few kinds of relations, leaving most of the other kinds alone.

We use this tendency to further boost the learner's performance by using a merger of these existing GR finders' output as the initial labeling.

2 The Experiment

We now improve the performance of the Ferro et al. (1999) transformation rule learner on a small annotated training set by using an existing system to provide initial GR annotations. This experiment is repeated on two different existing systems, which are reported in Buchholz et al. (1999) and Carroll et al. (1999), respectively.

Both of these systems find a somewhat different set of GR annotations than the one learned by the Ferro et al. (1999) system. For example, the Buchholz et al. (1999) system ignores verb complements of verbs and is designed to look for relationships to verbs and not GRs that exist between nouns, etc. This system also handles relative clauses differently. For example, in *"Miller, who organized ..."*, this system is trained to indicate that *"who"* is the subject of *"organized"*, while the Ferro et al. (1999) system is trained to indicate that *"Miller"* is the subject of *"organized"*. As for the Carroll et al. (1999) system, among other things, it does not distinguish between subtypes of modifiers such as time, location and possessive. Also, both systems handle copulas (usually using the verb "to be") differently than in Ferro et al. (1999).

2.1 Experiment Set-Up

As described in Ferro et al. (1999), the transformation rule learner starts with a p-o-s tagged corpus that has been "chunked" into noun chunks, etc. The starting state also includes imperfect estimates of pp-attachments and a blank set of initial GR annotations. In these experiments, this blank initial set is changed to be a translated version of the annotations produced by an existing system. This is how the existing system transmits what it found to the rule learner. The set-up for this experiment is shown in figure 1. The four components with + signs are taken out when one wants the transformation rule learner to start with a blank set of initial GR annotations.

The two arcs in that figure with a * indicate where the translations occur. These translations of the annotations produced by the existing system are basically just an attempt to map each type of annotation that it produces to the most likely type of corresponding annotation used in the Ferro et al. (1999) system. For example, in our experiments, the Buchholz et al. (1999) system uses the annotation `np-sbj` to indicate a subject, while the Ferro et al. (1999) system uses the annotation `subj`. We create the mapping by examining the training set to be given to the Ferro et al. (1999) system. For each type of relation e_i output by the existing system when given the training set text, we look at what relation types (which t_k's) co-occur with e_i in the training set. We look at the t_k's with the highest number of co-occurrences with that e_i. If that t_k is unique (no ties for the highest number of co-occurrences) and translating e_i to that t_k generates at least as many correct annotations in the training set as false alarms, then make that translation. Otherwise, translate e_i to no relation. This latter translation is not uncommon. For example, in one run of our experiments, 9% of the relation instances in the training set were so translated, in another run, 46% of the instances were so translated.

Some relations in the Carroll et al. (1999) system are between three or four elements. These relations are each first translated into a set of two element sub-relations before the examination process above is performed.

Even before applying the rules, the translations find many of the desired annotations. However, the rules can considerably improve what is found. For example, in two of our early experiments, the translations by themselves produced F-scores (explained below) of about 40% to 50%. After the learned rules were applied, those F-scores increased to about 70%.

An alternative to performing translations is to use the *un*translated initial annotations as an additional type of input to the rule system. This alternative, which we have yet to try, has the advantage of fitting into the transformation-based error-driven paradigm (Brill and Resnik, 1994) more cleanly than having a translation stage. However, this additional type of input will also further slowdown an already slow rule-learning module.

2.2 Overall Results

For our experiment, we use the same 1151 word (748 GR) test set used in Ferro et al. (1999), but for a training set, we use only a subset of the 3299 word training set used in Ferro et al. (1999). This subset contains 1391 (71%) of the 1963 GR instances in the original training set. The overall results for the test set are

Smaller Training Set, Overall Results				
	R	P	F	ER
IaC	*478 (63.9%)*	77.2%	*69.9%*	7.7%
IaB	*466 (62.3%)*	78.1%	*69.3%*	5.8%
NI	448 (59.9%)	77.1%	67.4%	

where row IaB is the result of using the rules learned when the Buchholz et al. (1999) system's translated GR annotations are used as the Initial Annotations, row IaC is the similar result with the Carroll et al. (1999) system, and row NI is the result of using the rules learned when No Initial GR annotations are used (the rule learner as run in Ferro et al. (1999)). R(ecall) is the number (and percentage) of the keys that are recalled. P(recision) is the number of cor-

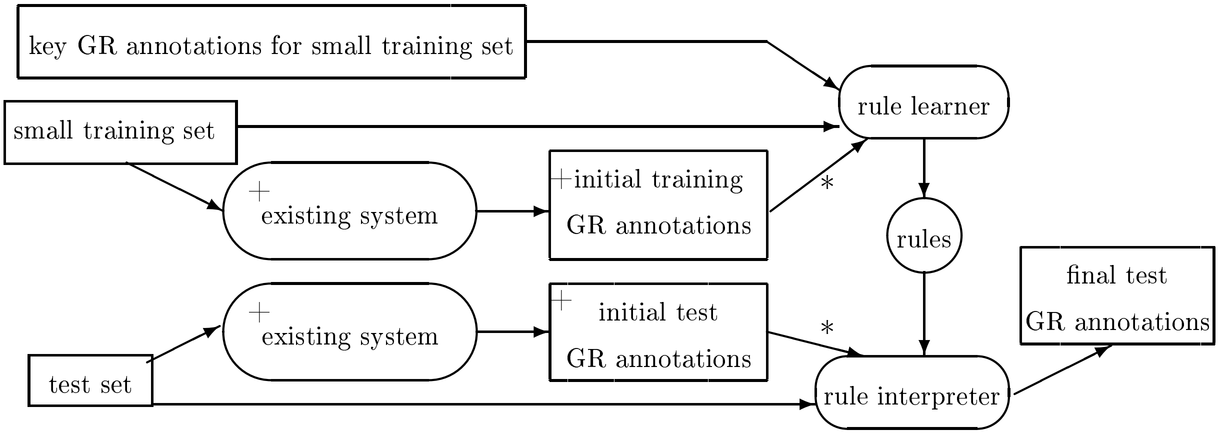
key GR annotations for small training set
small training set
rule learner
+ existing system
+initial training GR annotations
*
rules
final test GR annotations
+ existing system
+ initial test GR annotations
*
test set
rule interpreter

Figure 1: Set-up to use an existing system to improve performance

rectly recalled keys divided by the number of GRs the system claims to exist. F(-score) is the harmonic mean of recall (r) and precision (p) percentages. It equals $2pr/(p+r)$. ER stands for Error Reduction. It indicates how much adding the initial annotations reduced the missing F-score, where the missing F-score is 100%−F. ER= $100\% \times (\mathrm{F}_{IA} - \mathrm{F}_{NI})/(100\% - \mathrm{F}_{NI})$, where F_{NI} is the F-score for the NI row, and F_{IA} is the F-score for using the Initial Annotations of interest. Here, the differences in recall and F-score between NI and either IaB or IaC (but not between IaB and IaC) are statistically significant. The differences in precision is not.[2] In these results, most of the modest F-score gain came from increasing recall.

One may note that the error reductions here are smaller than Palmer (1997)'s error reductions. Besides being for different tasks (word segmentation versus GRs), the reductions are also computed using a different type of baseline. In Palmer (1997), the baseline is how well an existing system performs before the rules are run. In this paper, the baseline is the performance of the rules learned without first using an existing system. If we were to use the same baseline as Palmer (1997), our baseline would be an F of 37.5% for IaB and 52.6% for IaC. This would result in a much higher ER of 51% and 36%, respectively.

We now repeat our experiment with the full 1963 GR instance training set. These results indicate that as a small training set gets larger, the overall results get better and the initial annotations help less in improving the overall results. So the initial annotations are more helpful with smaller training sets. The overall results on the test set are

Full Training Set, Overall Results				
	R	P	F	ER
IaC	*487 (65.1%)*	*79.7%*	*71.7%*	6.3%
IaB	486 (65.0%)	76.5%	70.3%	1.7%
NI	476 (63.6%)	77.3%	69.8%	

The differences in recall, etc. between IaB and NI are now small enough to be not statistically significant. The differences between IaC and NI are statistically significant,[3] but the difference in both the absolute F-score (1.9% versus 2.5% with the smaller training set) and ER (6.3% versus 7.7%) has decreased.

2.3 Results by Relation

The overall result of using an existing system is a modest increase in F-score. However, this increase is quite unevenly distributed, with a

[2] When comparing differences in this paper, the statistical significance of the higher score being better than the lower score is tested with a one-sided test. Differences deemed statistically significant are significant at the 5% level. Differences deemed non-statistically significant are not significant at the 10% level. For recall, we use a sign test for matched-pairs (Harnett, 1982, Sec. 15.5). For precision and F-score, a "matched-pairs" randomization test (Cohen, 1995, Sec. 5.3) is used.

[3] The recall difference is semi-significant, being significant at the 10% level.

few relation(s) having a large increase, and most relations not having much of a change. Different existing systems seem to have different relations where most of the increase occurs.

As an example, take the results of using the Buchholz et al. (1999) system on the 1391 GR instance training set. Many GRs, like *possessive modifier*, are not affected by the added initial annotations. Some GRs, like *location modifier*, do slightly better (as measured by the F-score) with the added initial annotations, but some, like *subject*, do better without. With GRs like *subject*, some differences between the initial and desired annotations may be too subtle for the Ferro et al. (1999) system to adjust for. Or those differences may be just due to chance, as the result differences in those GRs are not statistically significant. The GRs with statistically significant result differences are the *time* and "other"[4] *modifier*s, where adding the initial annotations helps. The *time modifier*[5] results are quite different:

Smaller Training Set, *Time Modifier*s				
	R	P	F	ER
IaB	*29 (64.4%)*	*80.6%*	*71.6%*	53%
NI	14 (31.1%)	56.0%	40.0%	

The difference in the number recalled (15) for this GR accounts for nearly the entire difference in the overall recall results (18). The recall, precision and F-score differences are all statistically significant.

Similarly, when using the Carroll et al. (1999) system on this training set, most GRs are not affected, while others do slightly better. The only GR with a statistically significant result difference is *object*, where again adding the initial annotations helps:

Smaller Training Set, *Object* Relations				
	R	P	F	ER
IaC	*198 (79.5%)*	79.5%	*79.5%*	17%
NI	179 (71.9%)	78.9%	75.2%	

The difference in the number recalled (19) for this GR again accounts for most of the difference in the overall recall results (30). The recall and F-score differences are statistically significant. The precision difference is not.

As one changes from the smaller 1391 GR instance training set to the larger 1963 GR instance training set, these F-score improvements become smaller. When using the Buchholz et al. (1999) system, the improvement in the "other" *modifier* is now no longer statistically significant. However, the *time modifier* F-score improvement stays statistically significant:

Full Training Set, *Time Modifier*s				
	R	P	F	ER
IaB	*29 (64.4%)*	*74.4%*	*69.0%*	46%
NI	15 (33.3%)	57.7%	42.3%	

When using the Carroll et al. (1999) system, the *object* F-score improvement stays statistically significant:

Full Training Set, *Object* Relations				
	R	P	F	ER
IaC	194 (77.9%)	*85.1%*	*81.3%*	16%
NI	188 (75.5%)	80.3%	77.8%	

2.4 Combining Sets of Initial Annotations

So the initial annotations from different existing systems tend to each concentrate on improving the performance of different GR types. From this observation, one may wonder about combining the annotations from these different systems in order to increase the performance on all the GR types affected by those different existing systems.

Various works (van Halteren et al., 1998; Henderson and Brill, 1999; Wilkes and Stevenson, 1998) on combining different systems exist. These works use one or both of two types of schemes. One is to have the different systems simply vote. However, this does not really make use of the fact that different systems are better at handling different GR types. The other approach uses a combiner that takes the systems' output as input and may perform such actions as determining which system to use under which circumstance. Unfortunately, this approach needs extra training data to train such a combiner. Such data may be more useful when

[4] Modifiers that do not fall into any of the subtypes used, such as time, location, possessive, etc. Examples of *unused* subtypes are purpose and modality.

[5] There are 45 instances in the test set key.

used instead as additional training data for the individual methods that one is considering to combine, especially when the systems being combined were originally given a small amount of training data.

To avoid the disadvantages of these existing schemes, we came up with a third method. We combine the existing related systems by taking a union of their translated annotations as the new initial GR annotation for our system. We rerun rule learning on the smaller (1391 GR instance) training set with a Union of the Buchholz et al. (1999) and Carroll et al. (1999) systems' translated GR annotations. The overall results for the test set are (shown in row IaU)

Smaller Training Set, Overall Results				
	R	P	F	ER
IaU	*496 (66.3%)*	76.4%	*71.0%*	11%
IaC	*478 (63.9%)*	77.2%	*69.9%*	7.7%
IaB	*466 (62.3%)*	78.1%	*69.3%*	5.8%
NI	448 (59.9%)	77.1%	67.4%	

where the other rows are as shown in Section 2.2. Compared to the F-score with using Carroll et al. (1999) (IaC), the IaU F-score is "borderline" statistically significantly better (11% significance level). The IaU F-score is statistically significantly better than the F-scores with either using Buchholz et al. (1999) (IaB) or not using any initial annotations (NI).

As expected, most (42 of 48) of the overall increase in recall going from NI to IaU comes from increasing the recall of the *object*, *time modifier* and other *modifier* relations, the relations that IaC and IaB concentrate on. The ER for *object* is 11% and for *time modifier* is 56%.

When this combining approach is repeated the full 1963 GR instance training set, the overall results for the test set are

Full Training Set, Overall Results				
	R	P	F	ER
IaU	*502 (67.1%)*	77.7%	*72.0%*	7.3%
IaC	*487 (65.1%)*	*79.7%*	*71.7%*	6.3%
IaB	486 (65.0%)	76.5%	70.3%	1.7%
NI	476 (63.6%)	77.3%	69.8%	

Compared to the smaller training set results, the difference between IaU and IaC here is smaller for both the absolute F-score (0.3% versus 1.1%) and ER (1.0% versus 3.3%). In fact, the F-score difference is small enough to not be statistically significant. Given the previous results for IaC and IaB as a small training set gets larger, this is not surprising.

3 Discussion

GRs are important, but different sets of GRs are useful for different purposes and different systems are better at finding certain types of GRs. Here, we have been looking at ways of improving automatic GR finders when one has only a small amount of data with the desired GR annotations. In this paper, we improve the performance of the Ferro et al. (1999) GR transformation rule learner by using existing systems to find related sets of GRs. The output of these systems is used to supply initial sets of annotations for the rule learner. We achieve modest gains with the existing systems tried. When one examines the results, one notices that the gains tend to be uneven, with a few GR types having large gains, and the rest not being affected much. The different systems concentrate on improving different GR types. We leverage this tendency to make a further modest improvement in the overall results by providing the rule learner with the merged output of these existing systems. We have yet to try other ways of combining the output of existing systems that do not require extra training data. One possibility is the example-based combiner in Brill and Wu (1998, Sec. 3.2).[6] Furthermore, finding additional existing systems to add to the combination may further improve the results.

References

S. Aït-Mokhtar and J.-P. Chanod. 1997. Subject and object dependency extraction using finite-state transducers. In *Proc. ACL workshop on automatic information extraction and building*

[6] Based on the paper, we were unsure if extra training data is needed for this combiner. One of the authors, Wu, has told us that extra data is not needed.

of lexical semantic resources for NLP applications, Madrid.

E. Brill and P. Resnik. 1994. A rule-based approach to prepositional phrase attachment disambiguation. In *15th International Conf. on Computational Linguistics (COLING)*.

E. Brill and J. Wu. 1998. Classifier combination for improved lexical disambiguation. In *COLING-ACL'98*, pages 191–195, Montréal, Canada.

S. Buchholz, J. Veenstra, and W. Daelemans. 1999. Cascaded grammatical relation assignment. In *Joint SIGDAT Conference on Empirical Methods in NLP and Very Large Corpora (EMNLP/VLC'99)*. cs.CL/9906004.

J. Carroll, T. Briscoe, N. Calzolari, S. Federici, S. Montemagni, V. Pirrelli, G. Grefenstette, A. Sanfilippo, G. Carroll, and M. Rooth. 1997. Sparkle work package 1, specification of phrasal parsing, final report. Available at http://www.ilc.pi.cnr.it/-sparkle/sparkle.htm, November.

J. Carroll, G. Minnen, and T. Briscoe. 1999. Corpus annotation for parser evaluation. In *EACL99 workshop on Linguistically Interpreted Corpora (LINC'99)*. cs.CL/9907013.

P. Cohen. 1995. *Empirical Methods for Artificial Intelligence*. MIT Press, Cambridge, MA, USA.

L. Ferro, M. Vilain, and A. Yeh. 1999. Learning transformation rules to find grammatical relations. In *Computational natural language learning (CoNLL-99)*, pages 43–52. EACL'99 workshop, cs.CL/9906015.

D. Harnett. 1982. *Statistical Methods*. Addison-Wesley Publishing Co., Reading, MA, USA, third edition.

J. Henderson and E. Brill. 1999. Exploiting diversity in natural language processing: combining parsers. In *Joint SIGDAT Conference on Empirical Methods in NLP and Very Large Corpora (EMNLP/VLC'99)*.

R. Hwa. 1999. Supervised grammar induction using training data with limited constituent information. In *ACL'99*. cs.CL/9905001.

R. Kaplan. 1994. The formal architecture of lexical-functional grammar. In M. Dalrymple, R. Kaplan, J. Maxwell III, and A. Zaenen, editors, *Formal issues in lexical-functional grammar*. Stanford University.

M. Marcus, B. Santorini, and M. Marcinkiewicz. 1993. Building a large annotated corpus of english: the penn treebank. *Computational Linguistics*, 19(2).

M. Palmer, R. Passonneau, C. Weir, and T. Finin. 1993. The kernel text understanding system. *Artificial Intelligence*, 63:17–68.

D. Palmer. 1997. A trainable rule-based algorithm for word segmentation. In *Proceedings of ACL/EACL97*.

D. Perlmutter. 1983. *Studies in Relational Grammar 1*. U. Chicago Press.

H. van Halteren, J. Zavrel, and W. Daelemans. 1998. Improving data driven wordclass tagging by system combination. In *COLING-ACL'98*, pages 491–497, Montréal, Canada.

Y. Wilkes and M. Stevenson. 1998. Word sense disambiguation using optimized combinations of knowledge sources. In *COLING-ACL'98*, pages 1398–1402, Montréal, Canada.