Comparing two trainable grammatical relations finders *

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. On such a small training corpus, we compare two systems. They use different learning techniques, but we find that this difference by itself only has a minor effect. A larger factor is that in English, a different GR length measure appears better suited for finding simple argument GRs than for finding modifier GRs. We also find that partitioning the data may help memory-based learning.

1 Introduction

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

Yesterday, my cat ate the food in the bowl.

include *ate* having the subject *my cat*, the object *the food* and the time modifier *Yesterday*, and *the food* having the location modifier *in (the bowl)*.

However, 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), this is fine for information retrieval. Also, having less differentiation of the modifiers can make it easier to find them (Ferro et al., 1999).

Unless the desired set of GRs matches the set already annotated in some large training corpus (e.g., the Buchholz et al. (1999) GR finder used the GRs annotated in the Penn Treebank (Marcus et al., 1993)), one will have to either manually write rules to find the GRs or annotate a training corpus for the desired set. Manually writing rules is expensive, as is annotating a large corpus. We have performed experiments on learning to find GRs with just a small annotated training set. Our starting point is the work described in Ferro et al. (1999), which used a fairly small training set.

This paper reports on a comparison between the transformation-based error-driven learner described in Ferro et al. (1999) and the memory-based learner for GRs described in Buchholz et al. (1999) on finding GRs to verbs[1] by retraining the memory-based learner with the data used in Ferro et al. (1999). We find that the transformation versus memory-based difference only seems to cause a small difference in the results. Most of the result differences seem to instead be caused by differences in the representations and information used by the learners. An example is that different GR length measures are used. In English, one measure seems better for recovering simple argument GRs, while another measure seems better for modifier GRs. We also find that partitioning the data sometimes helps memory-based learn-

* This paper reports on work performed at the MITRE Corporation under the support of the MITRE Sponsored Research Program. Marc Vilain, Lynette Hirschman and Warren Greiff have helped make this work happen. Christine Doran and John Henderson provided helpful editing.

[1] That is, GRs that have a verb as the relation target. For example, in *Cats eat.*, there is a "subject" relation that has *eat* as the target and *Cats* as the source.

ing.

2 Differences Between the Two Systems

Ferro et al. (1999) and Buchholz et al. (1999) both describe learning systems to find GRs. The former (TR) uses transformation-based error-driven learning (Brill and Resnik, 1994) and the latter (MB) uses memory-based learning (Daelemans et al., 1999).

In addition, there are other differences. The TR system includes several types of information not used in the MB system (some because memory-based systems have a harder time handling set-valued attributes): possible syntactic (Comlex) and semantic (Wordnet) classes of a chunk headword, the stem(s) and named-entity category (e.g., person, location), if any, of a chunk headword, lexemes in a chunk besides the headword, pp-attachment estimate and certain verb chunk properties (e.g., passive, infinitive).

Some lexemes (e.g., coordinating conjunctions and punctuation) are usually outside of any chunk. The TR system will store these in an attribute of the nearest chunk to the left and to the right of such a lexeme. The MB system represents such lexemes as if they are one word chunks. The MB system cannot use the TR system method of storage because memory-based systems have difficulties with set-valued attributes (value is 0 or more lexemes).

The MB system (and not the TR system) also examines the number of commas and verb chunks crossed by a potential GR.

The space of possible GRs searched by the two systems is slightly different. The TR system searches for GRs of length three chunks or less. The MB system searches for GRs which cross at most either zero (target to the source's left) or one (to the right) verb chunks.

Also, slightly different are the chunks examined relative to a potential GR. Both systems will examine the target and source chunks, plus the source's immediate neighboring chunks. The MB system also examines the source's second neighbor to the left. The TR system instead also examines the target's immediate neighbors and all the chunks between the source and target.

The TR system has more data partitioning than the MB system. With the TR system, possible GRs that have a different source chunk type (e.g., noun versus verb), or a different relationship type (e.g., subject versus object) or direction or length (in chunks) are always considered separately and will be affected by different rules. The MB system will note such differences, but may decide to ignore some or all of them.

3 Comparing the Two Systems

3.1 Experiment Set-Up

One cannot directly compare the two systems from the descriptions given in Ferro et al. (1999) and Buchholz et al. (1999), as the results in the descriptions were based on different data sets and on different assumptions of what is known and what needs to be found.

Here we test how well the systems perform using the same small annotated training set, the 3299 words of elementary school reading comprehension test bodies used in Ferro et al. (1999).[2] We are mainly interested in comparing the parts of the system that takes in syntax (noun, verb, etc.) chunks (also known as groups) and finds the GRs between those chunks. So for the experiment, we used the general TiMBL system (Daelemans et al., 1999) to just reconstruct the part of the MB system that takes in chunks and finds GRs. The input to both this reconstructed part and the TR system is data that has been manually annotated for syntax chunks and GRs, along with automatic lexeme and sentence segmentation and part-of-speech tagging. In addition, the TR system has manual named-entity annotation, and automatic estimations for verb properties and preposition and subordinate conjunction attachments (Ferro et al., 1999). Because the MB system was originally designed to handle GRs attached to verbs (and not noun to noun GRs, etc.), we ran the reconstructed part to only find GRs to verbs, and ignored other types of GRs when comparing the reconstructed part with the TR system. The test set is the 1151 word test set used in Ferro et al. (1999). Only GRs to verbs were examined, so the effective training set GR count fell from 1963 to 1298 and test set GR

[2]Note that if we had been trying to compare the two systems on a large annotated training set, the MB system would do better by default just because the TR system would take too long to process a large training set.

count from 748 to 500.

3.2 Initial Results

In looking at the test set results, it is useful to divide up the GRs into the following sub-types:

1. Simple arguments: subject, object, indirect object, copula subject and object, expletive subject (e.g., *"It"* in *"It rained today."*).
2. Modifiers: time, location and other modifiers.
3. Not so simple arguments: arguments that syntactically resemble modifiers. These are location objects, and also subjects, objects and indirect objects that are attached via a preposition.

Neither system produces a spurious response for type 3 GRs, but neither system recalls many of the test keys either. The reconstructed MB system recalls 6 of the 27 test key instances (22%), the TR system recalls 7 (26%). A possible explanation for these low performances is the lack of training data. Only 58 (3%) of the training data GR instances are of this type.

The type 2 GRs are another story. There are 103 instances of such GRs in the test set key. The results are

Type 2 GRs			
System	Recall	Precision	F-score
MB	47 (46%)	49%	47%
TR	25 (24%)	64%	35%

Recall is the number (and percentage) of the keys that are recalled. Precision is the number of correctly 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 $2pr/(p+r)$. Here, the differences in r, p and F-score are all statistically significant.[3] The MB system performs better as measured by the F-score. But a trade-off is involved. The MB system has both a higher recall and a lower precision.

The bulk (370 or 74%) of the 500 GR key instances in the test set are of type 1 and most of these are either subjects or objects. With type 1 GRs, the results are

Type 1 GRs			
System	Recall	Precision	F-score
MB	231 (62%)	66%	64%
TR	284 (77%)	82%	79%

With these GRs, the TR system performs considerably better both in terms of recall and precision. The differences in all three scores are statistically significant.

Because 74% of the GR test key instances are of type 1, where the TR system performs better, this system performs better when looking at the results for all the test GRs combined. Again, all three score differences are statistically significant:

Combined Results			
System	Recall	Precision	F-score
MB	284 (57%)	63%	60%
TR	316 (63%)	80%	71%

Later, we tried some extensions of the reconstructed MB system to try to improve its overall result. We could improve the overall result by a combination of using the *IB1* search algorithm (instead of *IGTREE*) in TiMBL, restricting the potential GRs to those that crossed no verb chunks, adding estimates on preposition and complement attachments (as was done in TR) and adding information on verb chunks about being passive, an infinitive or an unconjugated present participle. The overall F-score rose to 65% (63% recall, 67% precision). This is an improvement, but the TR system is still better. The differences between these scores and the other MB and TR combined scores are statistically significant.

3.3 Exploring the Result Differences

3.3.1 Type 2 GRs: modifiers

The reconstructed MB system performs better at type 2 GRs. How can we account for this result difference?

Letting the TR system find longer GRs (beyond 3 chunks in length) does not help much. It only finds one more type 2 GR in the test set (adds 1% to recall and 1% or less to precision).

Rerunning the TR system rule learning with an information organization closer to the MB system produces the same 47% F-score as the

[3] 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.

MB system (recall is lower, but precision is higher). Specifically, we got this result when the TR system was rerun with no information on pp-attachments, verb chunk properties (e.g., passive, infinitive), named-entity labels or headword stems. Also, the TR system now examines the chunks examined by the original MB system: target, source and source's neighbors. In addition, instead of 6 absolute length categories (target is 3 chunks to the left, 2 chunks, 1 chunk, and similarly for the right), the GRs considered now just fall into and are partitioned into 3 relative categories: target is the first verb chunk to the left, similarly to the right and target is the second verb chunk to the right. The MB system can distinguish between these same relative categories.

Redoing this TR system rerun *without* chunk headword syntactic or semantic classes produces a 46% F-score. If in addition, the pp-attachment, verb chunk property, named-entity label and headword stem information are added back in, the F-score actually drops to 43%. The differences between these 47%, 46% and 43% rerun scores are not statistically significant.

So with type 2 GRs, MB system's better performance seems to be mainly due its ability to differentiate the potential GRs by the feature of the number of verb chunks crossed by a GR. In particular, making this and a few other changes to the TR system increases its F-score to the MB system's F-score, and the other changes (removing certain information) does not have a significant effect. So using the right features can make a large difference.

For these type 2 GRs (modifiers) in English, it does seem that the number of verb chunks crossed is a better way to group possible modifiers than the absolute chunk length. An example is comparing *I fly on Tuesday.* and *I fly home from here on Tuesday.* In both sentences, *on Tuesday* is a time modifier of *fly* and *on* crosses no verbs to reach *fly* (*on* attaches to the first verb to its left). But in the first sentence *on* is next to *fly*, while in the second sentence, there are three chunks separating *on* and *fly*.

3.3.2 Type 1 GRs: simple arguments

For type 1 GRs, the TR system performs better. How can we account for this?

Much of the extra information the TR system examines (compared to the MB system) does not seem to have much of an effect. When the TR system was rerun with no information on headword syntactic or semantic classes, named-entity labels or headword stems, the F-score increased from 79% to 80%. Another rerun that in addition had no information on pp-attachment estimates or any of the non-headwords in the chunks also had an F-score of 80%. A third rerun that furthermore had no information on verb chunk properties (e.g., passive, infinitive) had an F-score of 78%. In this set of F-scores, only the differences between the 80% scores and the 78% score are statistically significant.

Some MB system reruns showed factors that seemed to matter more. In the first rerun, we partitioned the data by potential GR source chunk type (e.g, noun versus verb) and ran a separate memory-based training and test for each partition. The combined F-score increased from 64% to 69%. Afterwards, we made a rerun that resembled the TR system run with the 78% F-score (except that memory-based learning was used): only GRs of length 3 chunks or less were considered, the data was partitioned (in addition to source chunk type) by GR length and direction (e.g., target is two chunks to the left) and also by relation type (separate runs for each type), the comma and verb chunk crossing counts were not considered, and the chunks normally examined by the TR system were examined. This further increased the F-score to 75%. In this set of F-scores, all the differences are statistically significant and all the F-scores in this set are statistically significantly different from the TR system runs with the 78% and 79% F-scores.

From the statistically significant score differences, it seems that partitioning data by potential GR source chunk type helps (increase from 64% to 69%), as does the rest of the partitioning performed and making some slight changes in what is examined (increase to 75%), using transformation-based learning instead of memory-based learning (increase to 78%) and using verb chunk property information (increase to 80%).

In the original MB system run, the source chunk type and the potential GR length and direction were already determined by the memory-based learner to be the most important

attributes examined. So why would partitioning the data and runs by the values of these attributes be of extra help? A possible answer is that for different values, the relative order of importance of the other attributes (as determined by the memory-based learner) changes. For example, when the source chunk type is a noun, the second most important attribute is the source chunk's headword when the target is one to the right, but is the source chunk's right neighbor's headword when the target is one to the left. Partitioning the data and runs lets these different relative orders be used. Having one combined data set and run means that only one relative order is used. Note that while this partitioning may not be the standard way of using memory-based learning, it is consistent with the central idea in memory-based learning of storing all the training instances and trying to find the "nearest" training instance to a test case.

Another question is why using transformation-based (rule) learning seems to be slightly better than memory-based learning for these type 1 GRs. Memory-based learning keeps all of the training instances and does not try to find generalizations such as rules (Daelemans et al., 1999, Ch. 4). However, with type 1 GRs, a few simple generalizations can account for many of the instances. In the manner of Stevenson (1998), we wrote a set of six simple rules that when run on the test set type 1 GRs produces an F-score of 77%. This is better than what our reconstructed MB system originally achieved and is close to the TR system's original results (close enough not to be statistically significantly different). An example of these six rules: IF (1) the center chunk is a verb chunk and (2) is not considered as possibly passive and (3) its headword is not some form of *to be* and (4) the right neighbor is a noun or verb chunk, THEN consider that chunk to the right as being an object of the center chunk.

4 Discussion

GRs are important, but different sets of GRs are useful for different purposes. 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 compared a transformation rule-based system with a memory-based system on a small training corpus. We found that on GRs that point to verbs, most of the result differences can be accounted for by differences in the representations and information used. The type of GR determines which information is more important. The rule versus memory-based difference itself only seems to produce a small result difference. We also find that partitioning the data may help memory-based learning.

References

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

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.

W. Daelemans, J. Zavrel, K. van der Sloot, and A. van den Bosch. 1999. Timbl: Tilburg memory based learner, version 2.0, reference guide. ILK Technical Report ILK 99-01. Available from `http://ilk.kub.nl/~ilk/-papers/ilk9901.ps.gz`.

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`.

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

M. Stevenson. 1998. Extracting syntactic relations using heuristics. In I. Kruijff-Korbayová, editor, *Proc. of the 3rd ESSLLI Student Session*. Chapter 19.