On Parser Evaluation Metrics

First Author
Affiliation / Address line 1
email@domain

Second Author
Affiliation / Address line 1
email@domain

Abstract

This paper seeks to quantitatively evaluate the similarities and differences between a broad range of popular parser evaluation metrics.

1 Introduction

The question of how best to evaluate the performance of a parser has received considerable attention. Numerous metrics have been proposed, and their relative merits have been debated. In this paper, we seek to quantitatively evaluate the degree to which a number of popular metrics provide overlapping information to the parser designer.

The motivation for this study is twofold. First, we wanted to confirm our suspicion that parsing models that performed well under one metric were likely to perform well under other metrics, thereby validating the widespread practice of using just a single metric when conducting research on improving parser performance. Second, we had a more specialized interest in using these metrics as heuristic aids in feature engineering. Our results are cautiously optimistic on both fronts.

We use the problem of selecting the best performer from a large space of varied but related parse disambiguation models ("parsers" henceforth) as the setting for our study. The parsers are all conditional log-linear disambiguators coupled to the English Resource Grammar (ERG) (Flickinger, 2000). Analyses from the ERG consist of a syntax tree together with an MRS – an underspecified logical formula (Copestake et al., 2005).

The parsers differ from each other along two dimensions: the feature templates employed, and the degree of regularization used. There are 57 different sets of traditional and novel feature templates collecting a variety of syntactic and semantic data about candidate ERG analyses. For each set of feature templates, parsers were trained with 41 different regularization parameters, for a total of 2337 different parsers.

The WeScience Treebank of about 9100 sentence (Ytrestøl et al., 2009) was used both for training and testing the parsers, with 10-fold cross validation.

We break down the problem of selecting the best parser into two tasks. The first task is to identify the optimal regularization parameter for each set of feature templates. The second task is to compare the different sets of feature templates to each other, considering only the optimal regularization parameter for each, and select the overall best. We attack each task with each of 14 metrics, and discuss the results.

2 Prior Work

Comparisons of parser metrics have been undertaken in the past. Carroll et al (1998) describe a broad range of parser evaluation metrics, and comment on their advantages and disadvantages, but do not offer quantitative comparison. Clark and Curran (2007) explore the difficulty of parser comparison across different underlying formalisms.

Crouch et al (2002) compare two specific metrics in some detail on a single LFG-based parsing model, concluding that despite some differences in the metrics' strategies, they offer similar views on the problem of evaluating a parser.

We are unaware of prior research specifically seeking to quantitatively compare a broad range of metrics across a large array of "parsers."

3 Metrics

In our setup, the overall score a metric assigns to a parser is the average of the scores assigned to each item in the treebank (termed *micro-averaging*, in contrast to *macro-averaging* which is also common). In the case of ties in the parser, the actual metric score used for an item is averaged over all tied results. Fourteen metrics are considered:

- Exact Tree Match (ETM) (Toutanova et al., 2005) - 100% if the returned tree is identical to the gold tree, and 0% otherwise.
- Exact MRS Match (EMM) - 100% if the returned MRS is equivalent to the gold MRS, and 0% otherwise.
- Average Crossing Brackets (AXB) - the number of brackets (constituents) in the returned tree that overlap incompatibly with some bracket in the gold tree.
- Zero Crossing Brackets (ZXB) - 100% if the AXB score is 0, and 0% otherwise.
- Labeled PARSEVAL (LP) (Abney et al., 1991) - the harmonic mean (F_1) of the precision and recall for comparing the set of labeled brackets in the returned tree with the set of labeled brackets in the gold tree. Labels are rule names.
- Unlabeled PARSEVAL (UP) - identical to LP, except ignoring the labels on the brackets.
- Labeled Syntactic Dependencies (LSD) (Buchholz and Marsi, 2006) - the F_1 for comparing the sets of bilexical syntactic dependencies extracted from the returned and gold trees, labeled by rule name that joins the dependent to the dependee.
- Unlabeled Syntactic Dependencies (USD) - identical to LSD, except ignoring the labels.
- Labeled Elementary Dependencies (LED) - the F_1 for comparing the sets of elementary dependency triples (Oepen and Lønning, 2006) extracted from the returned and gold MRS. These annotations are similar in spirit to those used in the PARC 700 Dependency Bank (King et al., 2003) and other semantic dependency evaluation schemes.
- Unlabeled Elementary Dependencies (UED) - identical to LED, except ignoring all labeling information other than the input positions involved.
- Leaf Ancestor (LA) (Sampson and Babarczy, 2003) - the average of the edit distances between the paths through the returned and gold trees from root to each leaf.
- Lexeme Name Match (LNM) - the percentage of input words parsed with the gold lexeme.
- Part-of-Speech Match (POS) - the percentage of input words parsed with the gold part of speech.
- Node Count Match (NCM) - 100% if the gold and returned trees have exactly the same number of nodes, and 0% otherwise.

Note that the last three metrics are not commonly used in parser evaluation, and were included for variety – in a sense serving as controls.

4 Optimizing the Regularization Parameter

The first half of our problem is: given a set of feature templates T, determine the optimal regularization parameter λ. We interpret the word "optimal" relative to each of our 14 metrics. This is quite straightforward: to optimize relative to metric μ, we simply evaluate $\mu(M(T, \lambda))$ for each value of λ, where $M(T, \lambda)$ is a parser trained using feature templates T and regularization parameter λ, and declare the value of λ yielding the greatest value of μ the winner. Figure 1 shows values of the ETM as a function of the regularization parameter λ for $T =$ "pcfg baseline"; as can easily be seen, the optimal value is approximately $\hat{\lambda}_\mu = 2$.

We are interested in how $\hat{\lambda}_\mu$ varies with different choices of μ. Figure 2 shows all 14 metrics as functions of λ for the same $T =$ "pcfg baseline." The actual scores from the metrics vary broadly, so the

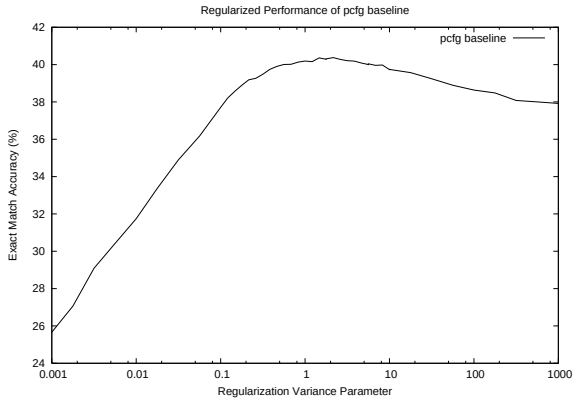

Figure 1: ETM for "pcfg baseline"

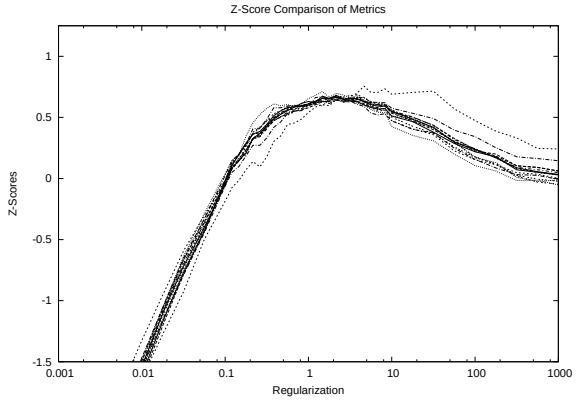

Figure 2: Z-scores for all metrics for "pcfg baseline"

vertical axes of the superimposed plots have been rescaled to allow for easier comparison.

A priori we might expect the optimal $\hat{\lambda}_\mu$ to be quite different for different μ, but this does not turn out to be the case. The curves for all of the metrics peak in roughly the same place, with one noticeable outlier (AXB). The actual peak[1] regularization parameters for the 14 metrics were all in the range $[1.8, 3.9]$ except the outlier AXB, which was 14.8.

Relative to the range under consideration, the optimal regularization parameters can be seen by inspection to depend very little on the metric. Near the optima, the graphs are all quite flat, and we calculate that by choosing the optimal regularization parameter according to any of the metrics (with the exception of the outlier AXB), the maximum increase in error rate visible through the other metrics was 1.6%. If we ignore LNM, POS and NCM (the non-standard metrics we included for variety) in addition to AXB, the maximum increase in error rate by using an alternate metric to optimize the regularization parameter drops to 0.41%.

"pcfg baseline" is just one of 57 sets of feature templates. However, the situation is essentially the same with each of the remaining 56. The average maximum error rate increase observed across all of the sets of feature templates when optimizing on any metric (including AXB, LNM, POS and NCM) was 2.54%; on the worst single set of feature templates it was 6.7%. Excluding AXB, the average maximum error rate increase was 1.7%. Additionally excluding LNM, POS and NCM it was 0.81%.

Given the size of the evaluation corpus we are using, the significance of an error rate increase of 0.81% is very marginal. We conclude that, at least in circumstances similar to ours, the choice of metric used to optimize regularization parameters is not important, provided we avoid AXB and the variety metrics LNM, POS and NCM.

5 Choosing a Set of Feature Templates

The second half of our problem is: given a collection $\mathcal{T}$ of different sets of feature templates, select the optimal performer. Again, we interpret the word "optimal" relative to each of our 14 metrics, and the selection is straightforward: given a metric μ, we first form a set of parsers $P = \{M(T, \arg\max_\lambda \mu(M(T, \lambda))) : T \in \mathcal{T}\}$ and then select $\arg\max_{p \in P} \mu(p)$. That is, we train parsers using the μ-optimal regularization parameter for each $T \in \mathcal{T}$, and then select the μ-optimal parser from that set.

In our experiments, all 14 of the metrics ranked the same set of feature templates as best.

It is also interesting to inspect the order that each metric imposes on P. There was some disagreement between the metrics about this order. We computed pairwise Spearman rank correlations coefficients[2] for the different metrics. As with the task

[1] Due to noisiness near the tops of the graphs, the reported optimum regularization parameters are actually the averages of the best 3 values.

[2] The Spearman rank correlation coefficient of two metrics is defined as the Pearson correlation coefficient of the ranks the metrics assign to the elements of P. It takes values between -1 and 1, with larger values indicating higher ranking agreement.

of choosing a regularization parameter, the metrics AXB, LNM, POS and NCM were outliers. The average pairwise Spearman rank correlation excluding these metrics was 0.859 and the minimum was 0.761.

An alternate method of quantifying the degree of agreement is described below.

5.1 Epsila

Consider two metrics $\mu : P \mapsto \mathbb{R}$ and $\rho : P \mapsto \mathbb{R}$. Assume for simplicity that for both μ and ρ, larger values are better and 100 is perfect. If $x, y \in P$ then the *error rate reduction* from y to x under μ is $\mu^*(x, y) = \frac{\mu(x)-\mu(y)}{100-\mu(y)}$. Let $\epsilon_{\mu,\rho}$ be the smallest number such that $\forall x, y \in P : \mu^*(x, y) > \epsilon_{\mu,\rho} \Rightarrow \rho^*(x, y) > 0$. Informally, this says for all pairs of parsers x and y, if x is at least $\epsilon_{\mu,\rho}$ better than y when evaluated under μ, then we are guaranteed that x is at least a tiny bit better than y when evaluated under ρ. For an unrestricted domain of parsers, we are not guaranteed that such epsila exist or are small enough to be interesting. However, since our P is finite, we can find an ϵ that will provide the required property at least within P.

$\epsilon_{\mu,\rho}$ serves as a measure of how similar μ and ρ are: if $\epsilon_{\mu,\rho}$ is small, then small improvements seen under μ will be visible as improvements under ρ, whereas if $\epsilon_{\mu,\rho}$ is large, then small improvements seen under μ may in fact be regressions when evaluating with ρ.

We computed pairwise epsila for our 14 metrics. A large portion of pairwise epsila were around 5%, with some being considerably smaller or larger.

5.2 Clustering

In order to make sense of the idea that these epsila provide a similarity measure, we applied Quality Threshold clustering (Heyer et al., 1999) to discover maximal clusters of metrics within which all pairwise epsila are smaller than a given threshold. Small thresholds produce many small clusters, while larger thresholds produce fewer, larger clusters.

At a 1% threshold, almost all of the metrics form singleton clusters; that is, a 1% error rate reduction on any given metric is generally not enough to guarantee that any other metrics will see any error reduction at all. The exceptions were that {ETM, EMM} formed a cluster, and {UED, LED} formed a cluster.

Increasing the threshold to 3%, a new cluster {USD, LSD} forms (indicating that a 3% error rate reduction in USD always is visible as some level of error rate reduction in LSD, and vice versa), and ZXB joins the {ETM, EMM} cluster.

By the time we reach a 5% threshold, the majority (7 out of 11) of the "standard" parser evaluation metrics have merged into a single cluster, consisting of {ETM, EMM, ZXB, LA, LSD, UED, LED}. The PARSEVAL metrics form a cluster of their own {UP, LP}.

Increasing the threshold even more to 10% causes 10 out of 11 "standard" evaluation metrics to cluster together; the only holdout is AXB (average number of crossing brackets), which does not join the cluster even at a 20% threshold.

6 Conclusions

From both subtasks, we saw that the Average Crossing Brackets metric (AXB) is a serious outlier. We cannot say whether it provides complementary information or actually misleading information.

We can say with confidence that for the subtask of optimizing a regularization parameter, there is very little difference between the popular metrics {ETM, EMM, ZXB, LA, LP, UP, LSD, USD, LED, UED}.

For the subtask of choosing the optimal set of feature templates, there was even greater agreement: all 14 metrics arrived at the same result. Although they did not impose the exact same rankings, the rankings were similar.

Clustering based on epsila at the 5% and 10% thresholds showed interesting insights as well. We demonstrated that a 5% error rate reduction as seen on any of {ETM, EMM, ZXB, LA, LSD, UED, LED} is also visible from the others (although the popular PARSEVAL metrics were outliers at this threshold). This has the encouraging implication that a decision made on the basis of strong evidence from just one metric is not likely to be contradicted by evaluations by other metrics.

We must point out that these results have only been validated for our particular dataset. It is possible that these generalizations fail completely when comparing different sorts of parsers, and at the least we can expect that the thresholds may need to be adjusted.

References

S. Abney, D. Flickinger, C. Gdaniec, C. Grishman, P. Harrison, D. Hindle, R. Ingria, F. Jelinek, J. Klavans, M. Liberman, et al. 1991. Procedure for quantitatively comparing the syntactic coverage of English grammars. In *Proceedings of the workshop on Speech and Natural Language*, pages 306–311. Association for Computational Linguistics.

Sabine Buchholz and Erwin Marsi. 2006. Conll-x shared task on multilingual dependency parsing. In *Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL-X)*, pages 149–164, New York City, June. Association for Computational Linguistics.

J. Carroll, T. Briscoe, and A. Sanfilippo. 1998. Parser evaluation: a survey and a new proposal. In *Proceedings of the 1st International Conference on Language Resources and Evaluation*, pages 447–454. Citeseer.

S. Clark and J. Curran. 2007. Formalism-independent parser evaluation with CCG and DepBank. In *Annual Meeting-Association for Computational Linguistics*, volume 45, page 248.

A. Copestake, D. Flickinger, C. Pollard, and I.A. Sag. 2005. Minimal recursion semantics: An introduction. *Research on Language & Computation*, 3(4):281–332.

R. Crouch, R.M. Kaplan, T.H. King, and S. Riezler. 2002. A comparison of evaluation metrics for a broad-coverage stochastic parser. In *Beyond PARSEVAL workshop at 3rd Int. Conference on Language Resources an Evaluation (LREC02)*. Citeseer.

Dan Flickinger. 2000. On building a more efficient grammar by exploiting types. *Natural Language Engineering*, 6(01):15–28.

L.J. Heyer, S. Kruglyak, and S. Yooseph. 1999. Exploring expression data: identification and analysis of coexpressed genes. *Genome research*, 9(11):1106.

T.H. King, R. Crouch, S. Riezler, M. Dalrymple, and R. Kaplan. 2003. The PARC 700 dependency bank. In *Proceedings of the EACL03: 4th International Workshop on Linguistically Interpreted Corpora (LINC-03)*, pages 1–8.

S. Oepen and J.T. Lønning. 2006. Discriminant-based MRS banking. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC 2006)*.

G. Sampson and A. Babarczy. 2003. A test of the leaf-ancestor metric for parse accuracy. *Natural Language Engineering*, 9(04):365–380.

K. Toutanova, C.D. Manning, D. Flickinger, and S. Oepen. 2005. Stochastic HPSG parse disambiguation using the Redwoods corpus. *Research on Language & Computation*, 3(1):83–105.

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*, Groningen, The Netherlands.