Rule Writing or Annotation: Cost-efficient Resource Usage for Base Noun Phrase Chunking

Grace Ngai and **David Yarowsky**
Department of Computer Science
Johns Hopkins University
Baltimore, MD 21218, U.S.A.
Email:{gyn,yarowsky}@cs.jhu.edu

Abstract

This paper presents a comprehensive empirical comparison between two approaches for developing a base noun phrase chunker: human rule writing and active learning using interactive real-time human annotation. Several novel variations on active learning are investigated, and underlying cost models for cross-modal machine learning comparison are presented and explored. Results show that it is more efficient and more successful by several measures to train a system using active learning annotation rather than hand-crafted rule writing at a comparable level of human labor investment.

1 Introduction

One of the primary problems that NLP researchers who work in new languages or new domains encounter is a lack of available annotated data. Collection of data is neither easy nor cheap. The construction of the Penn Treebank significantly improved performance for English systems dealing in the "traditional" NLP domains (eg parsing, part-of-speech tagging, etc). However, for a new language, a similar investment of effort in time and money is most likely prohibitive, if not impossible.

Faced with the costs associated with data acquisition, rationalists may argue that it would be more cost effective to construct systems of hand-coded rule lists that capture the linguistic characteristics of the task at hand, rather than spending comparable effort annotating data and expecting the same knowledge to be acquired indirectly by a machine learning system. The question we are trying to address then is: for a given cost assumption, which approach would be the most effective.

Although learning curves showing performance relative to amount of training data are common in the machine learning literature, these are inadequate for comparing systems with different sources of training data or supervision. This is especially true when a human rule-based approach and empirical learning are evaluated relative to effort invested. Such a multi-factor cost analysis is long overdue.

This paper will conclude with a comprehensive cost model exposition and analysis, and an empirical study contrasting human rule-writing versus annotation-based learning approaches that are sensitive to these cost models.

2 Base Noun Phrase Chunking

The domain in which our experiments are performed is base noun phrase chunking. A significant amount of work has been done in this domain and many different methods have been applied: Church's PARTS (1988) program used a Markov model; Bourigault (1992) used heuristics along with a grammar; Voutilainen's NPTool (1993) used a lexicon combined with a constraint grammar; Juteson and Katz (1995) used repeated phrases; Veenstra (1998), Argamon, Dagan & Krymolowski (1998), Daelemans, van den Bosch & Zavrel (1999) and Tjong Kim Sang & Veenstra (1999) used memory-based systems; Ramshaw & Marcus (1999) and Cardie & Pierce (1998) used rule-based systems, Munoz et al. (1999) used a Winnow-based system, and the XTAG Research Group(1998) used a tree-adjoining grammar.

Of all the systems, Ramshaw & Marcus' transformation rule-based system had the best published performance (f-measure 92.0) for several years, and is regarded as the de facto standard for the domain. Although several systems have recently achieved slightly higher published results (Munoz et al.: 92.8, Tjong Kim Sang & Veenstra: 92.37, XTAG Research Group: 92.4), their algorithms are significantly more costly, or not feasible, to implement in an active learning framework. To facilitate contrastive studies, we have evaluated our active learning and cost model com-

parisons using Ramshaw & Marcus' system as the reference algorithm in these experiments.

3 Active Learning from Annotation

Supervised statistical machine learning systems have traditionally required large amounts of annotated data from which to extract linguistic properties of the task at hand. However, not all data is created equal. A random distribution of annotated data contains much redundant information. By intelligently choosing the training examples which get passed to the learner, it is possible to provide the necessary amount of information with less data.

Active learning attempts to perform this intelligent sampling of data to reduce annotation costs without damaging performance. In general, these methods calculate the usefulness of an example by first having the learner classify it, and then seeing how uncertain that classification was. The idea is that the more uncertain the example, the less well modeled this situation is, and therefore, the more useful it would be to have this example annotated.

3.1 Prior Work in Active Learning

Seung, Opper and Sompolinsky (1992) and Freund et al. (1997) proposed a theoretical *query-by-committee* approach. Such an approach uses multiple models (or a *committee*) to evaluate the data, and candidates for annotation (or *queries*) are drawn from the pool of examples in which the models disagree. Furthermore, Freund et al. prove that, under some situations, the generalization error decreases exponentially with the number of queries.

On the experimental side, active learning has been applied to several different problems. Lewis & Gale (1994), Lewis & Catlett (1994) and Liere & Tadepalli (1997) all applied it to text categorization; Engelson & Dagan (1996) applied it to part-of-speech tagging.

Each approach has its own way of determining uncertainty in examples. Lewis & Gale used a probabilistic classifier and picked the examples e whose class-conditional *a posteriori* probability $P(C|e)$ is closest to 0.5 (for a 2-class problem). Engelson & Dagan implemented a committee of learners, and used vote entropy to pick examples which had the highest disagreement among the learners. In addition, Engelson & Dagan also investigate several different selection techniques in depth.

3.2 New Applications and Algorithmic Extensions in Active Learning

To our knowledge, this paper constitutes the first work to apply active learning to base noun phrase chunking, or to apply active learning to a transformation-learning paradigm (Brill, 1995) for any application. Since a transformation-based learner does not give a probabilistic output, we are not able to use Lewis & Gale's method for determining uncertainty. Our experimental framework thus uses the query by committee paradigm with batch selection:

1. Given a corpus C, arbitrarily pick t sentences for annotation.
2. Have these t sentences hand-annotated, delete them from C and put them into a training set, T.
3. Divide T into m non-identical, but not necessarily non-overlapping, subsets.
4. Use each subset as the training set for a model.
5. Evaluate each model on the remaining sentences in C
6. Using a measure of disagreement D, pick the x sentences in C with the highest D for annotation.
7. Delete the x sentences from C, have them annotated, and add them to T.
8. Repeat from 3.

In our experiments, the initial corpus C that we used consisted of sections 15-18 of the Wall Street Journal Treebank (Marcus et al., 1993), which is also the training set used by Ramshaw & Marcus (1999). The initial t sentences were the first 100 sentences of the training corpus, and $x = 50$ sentences were picked at each iteration. Sets of 50 sentences were selected because it takes approximately 15-30 minutes for humans to annotate them, a reasonable amount of work and time for the annotator to spend before taking a break while the machine selects the next set. The parameter m, which denotes the number of models to train, was set at 3, which could be expected to give us reasonable labelling variation over the samples, but also would not cause the processing phase to take a long time.

To divide the corpus into the different subsets in Step 3, we tried using two approaches: bagging and n-fold partitioning. In bagging, we randomly

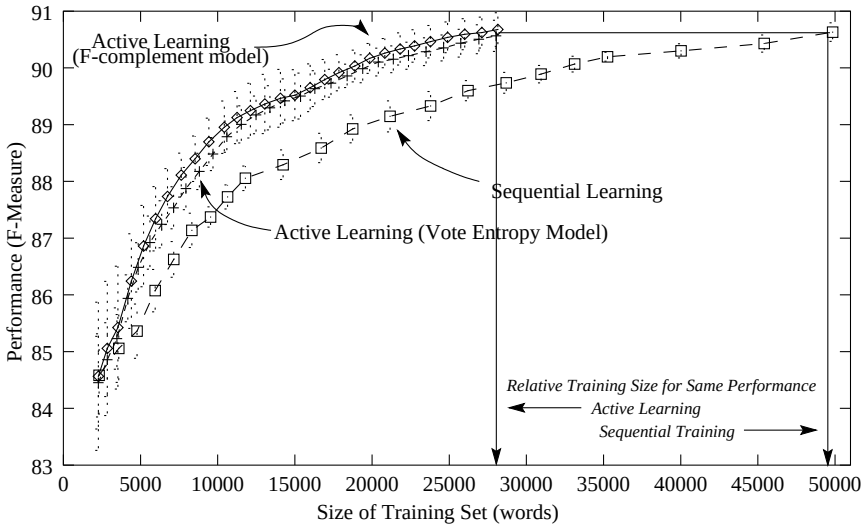

Figure 1: Performance vs. training set size: active learning and sequential annotation on Treebank data

pick (with replacement) $\frac{2}{3}$ of the total number of sentences in C to assign to each subset. With n-fold partitioning, we partitioned the data into 3 discrete partitions, and each model was then trained on 2 of the 3 partitions. We found no significant difference between the two methods.

3.2.1 Models of Disagreement for the Selection of New Data

The standard method for measuring disagreement for sample selection in active learning algorithms that use the query by committee is Engelson & Dagan's vote entropy measure. Given a tagged example e[1], the disagreement D for e is[2]:

$$D = -\frac{1}{\log k}\sum_{c}\frac{V(c,e)}{k}\log\frac{V(c,e)}{k}$$

where

k = Number of models in the committee.

$V(c,e)$ = Number of models assigning c to e

However, here we propose a novel disagreement measure that is both more applicable and achieves slightly improved performance. We base our measure on the f-measure metric, which is defined as:

$$F_\beta = \frac{(\beta^2+1)\times \text{Precision}\times \text{Recall}}{\beta^2\times \text{Precision} + \text{Recall}}$$

where

$$\text{Precision} = \frac{\#\text{ of correct proposed labellings}}{\#\text{ of proposed labellings}}$$

$$\text{Recall} = \frac{\#\text{ of correct proposed labellings}}{\#\text{ of correct labellings}}$$

The variable β allows precision and recall to be weighed differently. In all our experiments, β is set to 1, giving an equal weight to both precision and recall.

For our disagreement measure D, we use the *f-complement*, which is calculated as:

$$D = \frac{1}{2}\sum_{M_i,M_j\in\mathcal{K}}(1 - F_1(M_i(e), M_j(e)))$$

where $\mathcal{K}$ is the committee of models, M_i, M_j are individual models in $\mathcal{K}$, and $F_1(M_i(e), M_j(e))$ is F_1 of M_i's labelling of e relative to M_j's evaluation of e.[3]

Figure 1 shows the test set performance against the number of words in the training corpus for sequential annotation and active learning, using vote entropy and f-complement as the measures of disagreement. As can be seen from the graphs, f-complement gives a small empirical boost in performance. More importantly, f-complement can be used in applications where implementation of vote entropy is difficult, for example, parsing. The comparison between systems trained on annotated

[1] Vote entropy calculates the disagreement on a per tagged unit basis. In domains such as part-of-speech tagging or base noun phrase chunking, each tagged unit is a word. We prefer to select entire sentences as candidates for annotation. In situations like these, the disagreement over the entire sentence is simply the mean disagreement over the words in the sentence.

[2] Dividing by $\log k$ normalizes for the number of models

[3] $\beta = 1$ makes the F-measure symmetrical.

sentences selected by active learning and annotated sentences selected sequentially shows that active learning reduces the amount of data needed to reach a given level of performance by approximately a factor of two.

3.3 Active Learning with Real Time Human Supervision

Most of the published work on active learning are simulations of an idealized situation. One has a large annotated corpus, and the new tags for the "newly annotated" sentences are simply drawn from what was observed in the annotated corpus, as if the gold standard annotator was producing this feedback in real time, while the test set itself is, of course, not used for this feedback. This is an idealized situation, since it assumes that a true active learning situation would have access to someone who could annotate with perfect consistency to the gold standard corpus annotation conventions.

Because our goal is to investigate the relative costs of rule writing versus annotation, it is essential that we use a realistic model of annotation. Therefore, we decided to do a fully-fledged active learning annotation experiment, with real time human supervision, rather than assume the simulated feedback of actual Treebank annotators.

We developed an annotation tool that is modeled on MITRE's Alembic Workbench software (Day et al., 1997), but written in Java for platform-independence. To enable data storage and the active learning sample selection to take place on the more powerful machines in our lab rather than the user's home machine, the tool was designed with network support so that it could communicate with our servers over the internet.

Our real-time active learning experiment subjects were seven graduate students in computer science. Five of them are native English speakers, but none had any formal linguistics training. The initial training set T is the first 100 sentences of Ramshaw & Marcus' training set. To acquaint the subjects with the Treebank conventions, they were first asked to spend some time in a feedback phase, where they would annotate up to 50 sentences (they were allowed to stop at any time) drawn from the initial 100 sentences in T. The sentences were annotated one at a time, and the Treebank annotation was shown to them after every sentence. On average, the annotators spent around 15 minutes on this feedback phase before deciding that they were comfortable enough with the convention.

The active learning phase follows the feedback phase. The f-complement disagreement measure was used to select 50 sentences from the rest of Ramshaw & Marcus' training set and the annotator was instructed to annotate them. The annotated sentences were then sent back to the server. The system chose the next 50 sentences. The experiment consists of 10 iterations, during which the annotators were allowed to make use of the original 100 sentences as a reference corpus. After completing all 10 iterations, they were asked to annotate a further 100 consecutive sentences drawn randomly from the test set. The purpose of this final annotation was to judge how well annotators tag sentences drawn with the true distribution from the test corpus, as we shall see in section 5.

On average, the annotators took 17 minutes to annotate each set of 50 sentences, ranging from 8 to 30 minutes. The average amount of time the server took to run the active learning algorithm and select the next batch of sentences was approximately 3 minutes, a rest break for the annotators.

The analysis of the results is presented in section 5.

4 Learning by Rules

In previous work, Brill & Ngai (1999) showed that under certain circumstances, it is possible for humans writing rules to perform as well as a state-of-the-art machine learning system for base noun phrase chunking. What that study did not address, however, was the cost of the human labor and/or machine cycles involved to construct such a system, nor the relative cost of obtaining the training data for the machine learning system. This paper will estimate and contrast these costs relative to performance.

To investigate the costs of a human rule-writing system, we used a similar framework to that of Brill & Ngai. The system was written as a cgi script which could be accessed across the web from a browser such as Netscape or Internet Explorer. Like Brill & Ngai's 1999 approach, our rules were based on Perl regular expressions. However, instead of explicitly defining rule actions and having different kinds of rules, our rules implicitly define their actions by using different symbols to denote the placement of the base noun phrase-enclosing parentheses prior to and after the application of the rule. Table 1 presents a comparison of our rule format against that of Brill & Ngai's. The rules presented here may be considered less cumbersome and more intuitive.

In a way that is similar to Brill & Ngai's system, our rules were translated into Perl regular expressions and evaluated on the corpus. New

	Example Task		
	Inserting New Brackets	Splitting A Noun Phrase	Moving A Bracket
Brill & Ngai 1999 Rule Format	**TY:** A **LC:** null **TAR:** ({1} t=DT) (* t=JJ[RS]?) \ (+ t=NNP?S?) **RC:** null	**TY:** S **LC:** null **TAR1:** (* t=\w+) (+ t=NNP?S?) **MC:** null **TAR2:** (* t=JJ[RS]?) ({1} w=\w+day) **RC:** null	**TY:** T **LC:** <<< ({1} w=about) **TAR:** ({1} t=$) (+ t=CD) **RC:** null
New Rule Format	{ _DT ADJ* NOUN+ }	[{ ANYWORD* NOUN+ } { ADJ* TIMEDAY }]	{ about_ [_$ NUM+] }
Effect of Rule Application	$\text{The}_{DT}\ \text{man}_{NN}\ \text{ran}_{VBD}\ .$ ⇓ $(\ \text{The}_{DT}\ \text{man}_{NN}\ )\ \text{ran}_{VBD}\ .$	$(\ \text{New}_{NNP}\ \text{York}_{NNP}\ \text{Friday}_{NNP}\ )$ ⇓ $(\ \text{New}_{NNP}\ \text{York}_{NNP}\ )\ (\ \text{Friday}_{NNP}\ )$	$\text{about}_{IN}\ (\ \$_{\$}\ 5_{CD}\ )$ ⇓ $(\ \text{about}_{IN}\ \$_{\$}\ 5_{CD}\ )$

Table 1: Comparison of our current rule format with Brill & Ngai (1999)

rules are appended onto the end of the list and each rule applied in order, in the paradigm of a transformation-based rule list.

4.1 Rule-Writing Experiments

The rule-writing experiments were conducted by a group of 17 advanced computer science students, using the identical test set as in the annotation experiments and the same initial 100 gold standard sentences for both initial bracketing standards guidance and rule-quality feedback throughout their work.

```
# Grab-all rule
{ _RB::? ADJ* ANOUN* ADJ* ANOUN+ }
# (blah blah last Fri)->(blah blah) (last Fri)
{ [ ANYTHING* ] { _JJ TIME_W ] }
{ [ NOT_ADJ+ ] { TIME_W ] }
# about $8 (an ounce) -> (about $8 an ounce)
{ (Only|only|About|about)_::? _(\$|#)::?  \
   _CD::+ [ ANYTHING+ ] }
{ _RBR::* _(PDT|JJ)::? _(DT|PRP\$|POS) ADJ* \
   _RB::? VERB? [ ANYTHING+ ] }
# ( boy ) -> ( that boy )
[ (That|that)__DT { ANYTHING+ } ]
# ( about 4 1/2 )
{ (only|about)_::? (\$|#)_::? _CD::+ }
{ [ ANYTHING+ [? ANYTHING* ]? ] _-LRB-  \
   [ ANYTHING+ ]  _-RRB- [ ANYTHING+ ] }
# Pronouns are usually baseNPs
{ _DT::? _PRP }
# ``and'' usually isn't in a baseNP
{ [ _\S+::+ ] (and|&)_ [ _\S+::+ ] }
# more singleton baseNPs
{ _(DT|EX|WP|WDT) } VERB
# some numbers are singleton baseNPs
{ [ ANYTHING ] [ _CD ] }
# ( much/most ) of
{ _(DT|RB)::? (much|most)_ } _IN
```

Figure 2: An Example Rule List. Lines beginning with hash marks (#) are comments.

The time that the students spent on the task varied widely, from a minimum of 1.5 hours to a maximum of 9 hours, with an average of 5 hours. Because we captured and saved every change the students made to their rule list and logged every mouse click they made while doing the experiment, it was possible for us to trace the performance of the system as a function of time. Figure 2 shows the rule list constructed by one of the subjects. The quantitative results of the rule-writing experiments are presented in the next section.

5 Experiment Results — Rule Writing vs. Annotation

This section will analyze and compare the performance of systems constructed with hand-built rules with systems that were trained from data selected during real-time active learning.

The performance of Ramshaw & Marcus' system trained on the annotations of each subject in the real-time active learning experiments, and the performance achieved by the manually constructed systems of the top 6 rule writers are shown in Figures 3 and 4, depicting the performance achieved by each individual system. The x-axes show the time spent by each human subject (either annotating or writing rules) in minutes; the y-axes show the f-measure performance achieved by the systems built using the given level of supervision.

5.1 Analysis of Comparative Experimental Data

It is important to note that when comparing the curves in Figure 4, experimental conditions across groups were kept as equal as possible, with any known potential biases favoring the rules-writing group. First, both groups began with the identical

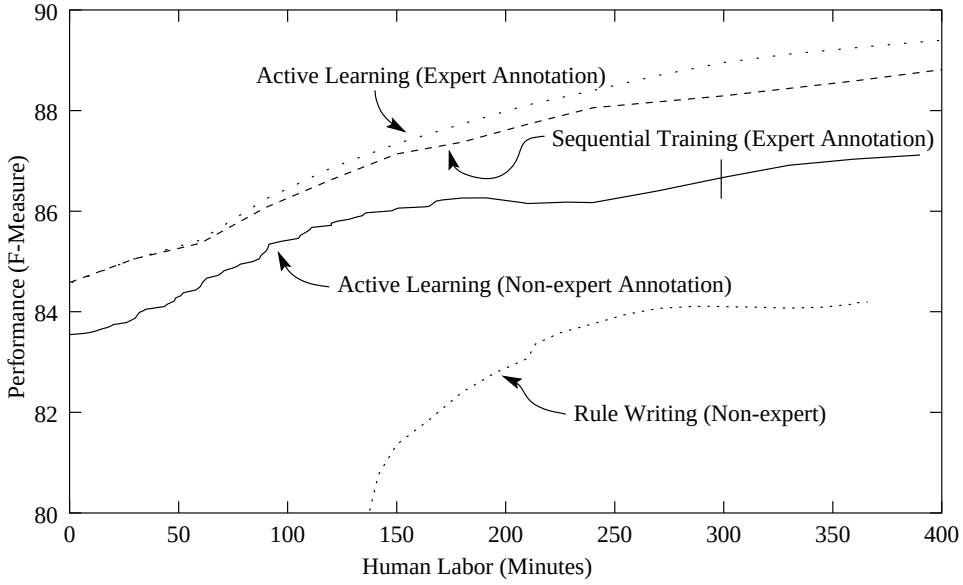

Figure 3: Consensus Performance for Systems constructed via Rule Writing, non-expert Annotation and Treebank Annotation (Individual curves are in Figure 4)

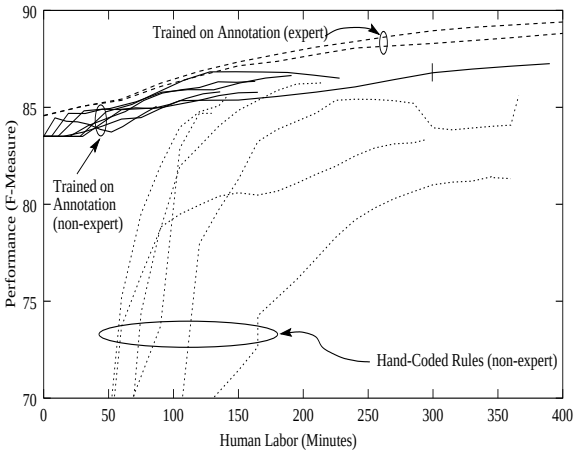

Figure 4: Annotation versus Rule Writing: Performance detailed by individual participant.

100 sentence gold standard set, for initial inspection and performance feedback throughout the rule-writing process. The higher starting point for the annotation-driven learning curves was due to the fact that the machine learning algorithm could do initial training immediately on this data. The rule-writing learners also received immediate feedback on their first rules using this data, but were slower to incorporate this feedback into their new rules. The six rule-writers used for comparative purposes were all native speakers, while the annotation group included 2 non-native speakers. Also, to further minimize the potential for any unknown biases in sample selection in favor of annotation, the rule-writers who were evaluated and illustrated in these graphs were the 6 strongest performers out of the pool of 17; while all 7 annotation results are compared. Despite this favorable treatment, rule-writing still underperforms annotation-based learning with statistical significance of $P < 0.02$ for 100 minutes of investment, and with significance of $P < 0.05$ for times up to at least 2.5 hours. The high variance in the rule-writer pool complicates a finding of significance beyond this point, but at all quantities of human labor invested, mean annotation-based F-measure outperformed rule-writing and these trends appear to extrapolate.

5.2 Analysis of Human Performance on the Annotation Task

It appears that a major limiting factor to higher annotation-based learning is the accuracy of the annotators themselves relative to the evaluation gold standard (the Treebank in this case). To study this factor, at the end of their active-learning experiments annotators were asked to annotate a further 100 sentences from the same test data used to evaluate the learning algorithms. Their F-measure performance on this data, as if they were a competing annotation system, is given in Table 2. These measures of agreement with the gold standard effectively constitutes an upper bound on the performance of any system trained on their data.

	F-Measure Performance on 100 held-out sentences
Annotator 1	92.92
Annotator 2	92.54
Annotator 3	91.27
Annotator 4	90.20
Annotator 5	88.17
Annotator 6	86.14
Annotator 7	83.86

Table 2: Annotation Performance on 100 Test Set Sentences

Thus to further put annotation-trained system performance in perspective, Figure 5 shows the performance of individual trained systems relative to the highest achieved performance of the annotator on which that system was trained. In each case, the ratio is close to 1, indicating that the machine learning model achieves performance close to that of the annotator whose data it was trained on.

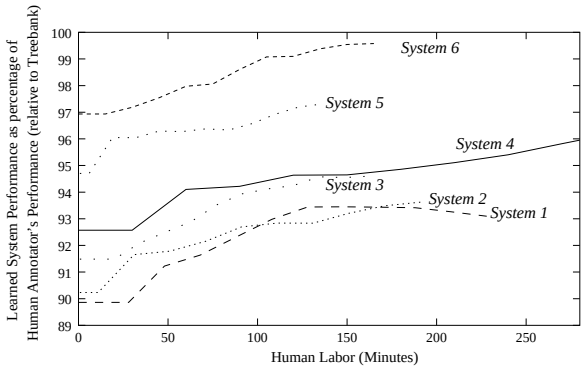

Figure 5: Performance of Ramshaw & Marcus' system trained on annotations by Annotator$_i$, as a percentage of that person's own performance measured against the Treebank (an effective upper bound).

6 Cost Models for Cross-Modal Learning Comparison

Traditionally, evaluation models in the machine learning literature measure performance relative to variable quantities of training data. This measure is inappropiate, however, for contrasting data-trained with rule-writing approaches where time and labor cost are the primary variables. Here we present cost models allowing these different approaches to be compared directly and objectively.

Below are two ways to evaluate the relative cost of a learning algorithm. The first measure contrasts performance relative to a common standard of invested human labor. The second measure considers a fuller range of potential development costs in a monetary-based common denominator.

6.1 Time-Based Cost

For the purposes of the above experiments, we have chosen total human effort (in time) as the variable resource, which successfully maps machine-learning and rule-based learning on a common measure of training resource investment.

Time-based evaluation is also useful for comparing systems trained on different annotators. As shown in Figure 4, annotators varied in the time they used to tag the full 500-sentence data set. Because performance tended to be lower when trained on the faster annotators (perhaps due to less careful work), a performance-per-sentences-tagged measure would tend to rank these individuals lower than their slower (but more careful) colleagues. However, when measured by the learning accuracy achieved per amount of annotator time invested, the faster but noiser annotators performed much more competitively (although the relative benefits of higher volumes of noiser data may not extrapolate well). Annotator-time-based performance measures provide a useful standard for evaluating this volume-noise tradeoff.

6.2 Monetary Cost

Annotator labor doesn't capture the complete relative cost of a given approach, however. It is useful, therefore, to measure resource investment in terms of the common denominator of monetary cost over a fuller set of potential cost variables. Table 3 details a set of other monetary parameters considered in the current studies. Given these parameters, one possible approximation of this cost function given variable time investment T and learning method M is:

$$MonetaryCost(M,T) = IDC_M + (S_0 + AC_{TB}) + (T * (LC_M + MC_A))$$

Although we assume equal labor cost rates LC_M for annotation and rule writing, these may substantially differ in some environments, and certainly will be higher for professional-quality annotation or rule-based development. And while the estimates of the machine cycle cost necessary to support this work on Linux-based PC's vary somewhat, they are relatively dwarfed by the labor costs. We have assumed that the infrastructure development costs for the tagging and rule-writing

Cost Model Parameter	Annotation	Rule-writing
IDC_M = Infrastructure Development Cost (for tagging/RW environment)	Shared	Shared
S_O = Number of initial gold standard sentences for training	100	100
AC_{TB} = gold standard (Treebank) Annotation Cost (per sentence)	x	x
LC_M = Labor Cost for Annotation or RW (per hour)	\$12.00/hour	\$12.00/hour
MC_A = Cost of Machine Cycles for Annotation/RW Support	\$0.24/hour	\$0.12/hour
T = Variable time investment		

Table 3: Example Monetary-Based Cost Parameters for Model Comparison

environments, while initially variable across methods, have already been borne and to the extent that both interface systems port to new languages and domains with relative ease, the incremental development costs for new trials are likely to be relatively low and comparable. Finally, this cost model takes into account the cost of developing or acquiring the S_0 gold standard tagged data (e.g. from the Treebank) to provide initial and/or incremental training feedback to the annotator or rule writer to help force consistency with the gold standard. We have found that both learning modes can benefit from this high quality feedback, but the cost x of developing such a high-quality resource for new languages or domains is unknown, but likely to be higher than the non-expert labor costs employed here.

7 Rules vs. Annotation-based Learning — Advantages and Disadvantages

In the previous sections, we investigated the performance differences and resource costs involved for using humans to write rules vs. using them for annotations. In this section, we will further compare these system development paradigms.

Annotation-based human participation has a number of significant practical advantages relative to developing a system by manual rule-writing:

- Annotation-based learning can continue indefinitely, over weeks and months, with relatively self-contained annotation decisions at each point. In contrast, rule-writers must remain cognizant of potential previous rule interdependencies when adding or revising rules, ultimately bounding continued rule-system growth by cognitive load factors.

- Annotation-based learning can more effectively combine the efforts of multiple individuals. The tagged sentences from different data sets can be simply concatenated to form a larger data set with broader coverage. In contrast, it is much more difficult, if not impossible, for a rule writer to resume where another one left off. Furthermore, combining rule lists is very difficult because of the tight and complex interaction between successive rules. Combination of rule writing systems is therefore limited to voting or similar classifier techniques which can be applied to annotation systems as well.

- Rule-based learning requires a larger skill set, including not only the linguistic knowledge needed for annotation, but also competence in regular expressions and an ability to grasp the complex interactions within a rule list. These added skill requirements naturally shrink the pool of viable participants and increases their likely cost.

- Based on empirical observation, the performance of rule writers tend to exhibit considerably more variance, while systems trained on annotation tend to yield much more consistent results.

- Finally, the current performance of annotation-based training is only a lower bound based on the performance of current learning algorithms. Since annotated data can be used by other current or future machine learning techniques, subsequent algorithmic improvements may yield performance improvements without any change in the data. In contrast, the performance achieved by a set of rules is effectively final without additional human revision.

The potential disadvantages of annotation-based system development for applications such as base NP chunking are limited. Given the cost models presented in Section 6, one potential negative scenario would be an environment where the machine cost significantly outweighed human labor costs, or where access to active learning and annotation infrastructure was unavailable or costly. However, under normal circumstances where machine analysis of text is pursued, and

public domain access to our annotation and active learning toolkits is assumed, such a scenario is unlikely.

8 Conclusion

This paper has illustrated that there are potentially compelling practical and performance advantages to pursuing active-learning based annotation rather than rule-writing to develop base noun phrase chunkers. The relative balance depends ultimately on one's cost model, but given the goal of minimizing total human labor cost, it appears to be consistently more efficient and effective to invest these human resources in system-development via annotation rather than rule writing.

9 Acknowledgements

The authors would like to thank Jan Hajic, Eric Brill, Radu Florian, and various members of the Natural Language Processing Lab at Johns Hopkins for their valuable feedback regarding this work.

References

S. Argamon, I. Dagan, and Y. Krymolowski. 1998. A memory-based approach to learning shallow language patterns. In *Proceedings of the 17th International Conference on Computational Linguistics*, pages 67–73. COLING-ACL.

D. Bourigault. 1992. Surface grammatical analysis for the extraction of terminological noun phrases. In *Proceedings of the 30th Annual Meeting of the Association of Computational Linguistics*, pages 977–981. Association of Computational Linguistics.

E. Brill and G. Ngai. 1999. Man vs. machine: A case study in base noun phrase chunking. In *Proceedings of ACL 1999*. Association for Computational Linguistics.

E. Brill. 1995. Transformation-based error-driven learning and natural language: A case study in part of speech tagging. *Computational Linguistics*, 21(4):543–565.

C. Cardie and D. Pierce. 1998. Error-driven pruning of treebank gramars for base noun phrase identification. In *Proceedings of the 36th Annual Meeting of the Association of Computational Linguistics*, pages 218–224. Association of Computational Linguistics.

K. Church. 1988. A stochastic parts program and noun phrase parser for unrestricted text. In *Proceedings of the Second Conference on Applied Natural Language Processing*, pages 136–143. Association of Computational Linguistics.

W. Daelemans, A. van den Bosch, and J. Zavrel. 1999. Forgetting exceptions is harmful in language learning. In *Machine Learning, special issue on natural language learning*, volume 11, pages 11–43. to appear.

D. Day, J. Aberdeen, L. Hirschman, R. Kozierok, P. Robinson, and M. Vilain. 1997. Mixed-initiative development of language processing systems. In *Fifth Conference on Applied Natural Language Processing*, pages 348–355. Association for Computational Linguistics, March.

S. Engelson and I. Dagan. 1996. Minimizing manual annotation cost in supervised training from corpora. In *Proceedings of ACL 1996*. Association for Computational Linguistics.

Y. Freund, H. S. Seung, E. Shamir, and N. Tishby. 1997. Selective sampling using the query by committee algorithm. *Machine Learning*, 28:133–168.

J. Juteson and S. Katz. 1995. Technical terminology: Some linguistic properties and an algorithm for identification in text. *Natural Language Engineering*, 1:9–27.

D. Lewis and J. Catlett. 1994. Heterogeneous uncertainty sampling for supervised learning. In *Proceedings of the 11th International Conference on Machine Learning*.

D. Lewis and W. Gale. 1994. A sequential algorithm for training text classifiers. In *Proceedings of ACM-SIGIR 1994*. ACM-SIGIR.

R. Liere and P. Tadepalli. 1997. Active learning with committees for text categorization. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence*, pages 591–596. AAAI.

M. Marcus, M. Marcinkiewicz, and B. Santorini. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

M. Munoz, V. Punyakanok, D. Roth, and D. Zimak. 1999. A learning approach to shallow parsing. In *Proceedings of EMNLP-WVLC'99*. Association for Computational Linguistics.

L. Ramshaw and M. Marcus. 1999. Text chunking using transformation-based learning. In *Natural Language Processing Using Very Large Corpora*. Kluwer.

H. S. Seung, M. Opper, and H. Sompolinsky. 1992. Query by committee. In *Proceedings of the Fifth Annual ACM Workshop on Computational Learning Theory*, pages 287–294. ACM.

The XTAG Research Group. 1998. A lexicalized tree adjoining grammar for english. Technical Report IRCS Tech Report 98-18, University of Pennsylvania.

E. Tjong Kim Sang and J. Veenstra. 1999. Representing text chunks. In *Proceedings of EACL'99*. ACL.

J. Veenstra. 1998. Fast NP chunking using memory-based learning techniques. In *BENELEARN-98: Proceedings of the Eighth Belgian-Dutch Conference on Machine Learning*, Wageningen, the Netherlands.

A. Voutilainen. 1993. *NPTool*, a detector of English noun phrases. In *Proceedings of the Workshop on Very Large Corpora*, pages 48–57. Association for Computational Linguistics.