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.

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-adjointing 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 re-implement in an active learning frame-
work. To facilitate contrastive studies, we have evaluated our active learning and cost model comparisons 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 normal 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 do such intelligent sampling of data, to reduce annotation costs without damaging performance. In general, these methods calculate the usefulness of an example by first getting the learner to classify it, and then seeing how uncertain that classification was. The idea is that the more uncertain the example, the "harder" it 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 are 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 50 sentences were picked at each iteration. The parameter of 50 sentences was picked because it takes approximately 15-30 minutes to annotate them — a reasonable amount of work for the annotator before waiting for the machine to select the next sentences. 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 pick (with replacement) \( \frac{5}{6} \) of the total number of
Figure 1: Performance vs. training set size: active learning and sequential annotation on Treebank data

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 using one method over another.

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 $e^i$, 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

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

The variable $\beta$ allows precision and recall to be weighed differently. In all our experiments, $\beta$ 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:

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

where $K$ is the committee of models, $M_i,M_j$ are individual models in $K$, and $F_1$ is $M_i$’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.

Since 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-flung 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 are 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 are translated into Perl regular expressions and evaluated on the corpus. New
rules are appended onto the end of the list and each rule is 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.

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 is 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.

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

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 con-
structured 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, that experimental conditions across groups were kept as equal as possible, with known potential biases favoring the rules-writing group. First, observe that both groups began with the identical 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 rulewriters 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 (further experiments will help to solidify significance conclusions), 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, are given in Table 2. These measures of agreement with the gold standard effectively constitute an upper bound on the performance of any system trained on their data.

Thus to further put annotation-trained system performance in perspective, Figure 5 shows 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.

Table 2: Annotation Performance on 100 Test Set Sentences

<table>
<thead>
<tr>
<th>Annotator</th>
<th>F-Measure Performance on 100 held-out sentences</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>92.92</td>
</tr>
<tr>
<td>2</td>
<td>92.54</td>
</tr>
<tr>
<td>3</td>
<td>91.27</td>
</tr>
<tr>
<td>4</td>
<td>90.20</td>
</tr>
<tr>
<td>5</td>
<td>88.17</td>
</tr>
<tr>
<td>6</td>
<td>86.14</td>
</tr>
<tr>
<td>7</td>
<td>83.86</td>
</tr>
</tbody>
</table>

Figure 5: Performance of Ramshaw & Marcus’ system trained on annotations by Annotator 1, 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 approach is inefficient, however, for contrasting data-trained with rule-writing approaches where time and labor cost are the primary variables. Here we present a cost model allowing these disparate approaches to be compared fairly.

There are two possible ways to view the cost of a system. The first method is the simple time-based cost, which is constant, and takes into account only the construction time. The second method is the money-based cost, which takes into account all possible construction cost factors and is heavily dependent on individual circumstances.

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 Figure 4 indicates, some annotators tagged the total pool of sentences faster, and while the final performance from tagging the full 500-sentence data set tends to be lower for the faster annotators (probably due to less quality results), the accuracy achieved by a learning system on the noisier but faster-tagged data tends to yield an overall higher performance at a given level of time invested (although this difference may not extrapolate well).

6.2 Monetary Cost

It is also useful to measure resource investment in terms of the common denominator of monetary cost. Table 3 details the parameters applicable to the current studies. Given these parameters, one possible approximation of this cost function given variable time investment $T$ and learning method $M$ is:

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

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 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 ensure 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 learning 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 people. 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 shrinks 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 current learning algorithm performance. Since annotated data can be used by other current or future machine learning algorithms, 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.

In contrast, 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. Yet under foreseeable situations where machine analysis of text is even pursued, and assuming public domain access to our annotation and active learning toolkits, 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


