Randomized Rule Selection In Transformation-Based Learning: A Comparative Study[1]

Sandra Carberry*, Ken Samuel, K. Vijay-Shanker*, Andrew Wilson***

* Department of Computer Science
University of Delaware
Newark, Delaware USA 19716
Email: carberry@cis.udel.edu, vijay@cis.udel.edu, awilson@cis.udel.edu

** The Mitre Corporation
Reston, Virginia 22090
samuel@mitre.org

Abstract

Transformation-Based Learning (TBL) is a relatively new machine learning method that has achieved notable success on language problems. This paper presents a variant of TBL, called Randomized TBL, that overcomes the training time problems of standard TBL without sacrificing accuracy. It includes a set of experiments on part-of-speech tagging in which the size of the corpus and template set are varied. The results show that Randomized TBL can address problems that are intractable in terms of training time for standard TBL. In addition, for language problems such as dialogue act tagging where the most effective features have not been identified through linguistic studies, Randomized TBL allows the researcher to experiment with a large set of templates capturing many potentially useful features and feature interactions.

1 Introduction

Transformation-Based Learning (TBL) is a relatively new machine learning method that has sparked interest among language researchers[Bri95]. The many attractive characteristics of TBL[RM94, Bri95] and its success on part-of-speech tagging[Bri95] have prompted interest in applying TBL to a wider range of language learning problems such as BaseNP chunking[RM95], PP attachment[BR94], and parsing[SB96]. However, TBL's training time is heavily dependent on the number of potential rules that must be constructed and evaluated; this is a serious impediment for complex problems where the number of potential rules is high.

[1]This work was partially supported by NSF Grant GER-9354869.

In this paper, we investigate a variant of TBL, which we will call Randomized TBL, that considers only a random subset of the potential rules evaluated by standard TBL. Randomized TBL overcomes the training time problem of standard TBL and yet retains its advantages. We first developed and presented Randomized TBL in [Sam98, SCV98b], applying it to the task of recognizing dialogue acts[2] of utterances. Our preliminary experiments with discourse processing[Sam98, SCV98b, SCV98a] suggested that Randomized TBL could significantly decrease training time while maintaining accuracy of the learned model. However, this study was preliminary. In this paper, we present Randomized TBL in detail, conduct a more thorough investigation of the efficiency gains, and compare the nature of the rules and the accuracies of the respective learned models.

To make the comparison fair, we chose to use an existing efficient implementation of TBL, rather than implement it ourselves. Hence we used Eric Brill's implementation of a part-of-speech tagger based on TBL[Bri95]. Furthermore, the use of part-of-speech tagging allows us to conduct experiments on a significantly larger corpus than the one that was available for dialogue act tagging. Thus it allows us to evaluate the impact of large versus small corpus size on standard TBL and Randomized TBL. Also, by conducting our experiments on a different task than dialogue act tagging, we can evaluate whether the efficiencies noted in our preliminary experiments carry over to other domains.

Section 2 presents a brief description of TBL and notes some of its characteristics. Section 3 then presents the randomized version of TBL. Section 4 discusses some details of Brill's implementation of TBL that we believe significantly improve its training time, especially for tasks such as part-of-speech tagging, and then describes our implementation of Randomized TBL. Section 5 presents our experiments and experimental results, providing an in-depth comparison and analysis of Randomized TBL versus standard TBL using Brill's algorithm and the part-of-speech tagging problem. Our experiments include a variation in corpus size as well as the size of the set of rule templates. While the application task (part-of-speech tagging) remains fixed over this

[2]Dialogue act tagging attempts to classify utterances according to the abstract intention that represents the speaker's objective, such as *Inform*, *Clarify*, *Suggest*, or *Reject*.

investigation, we believe that the variations in the corpus size and set of rule templates represent different types of scenarios. Our results on part-of-speech tagging show that Randomized TBL achieves accuracy equivalent to standard TBL with significantly improved training time efficiency; since these results are consistent with our earlier work on dialogue act tagging[Sam98, SCV98b], we believe that Randomized TBL enables transformation-based learning to be used on a wider range of language problems. Our last experiment suggests that Randomized TBL offers promise as a means of enabling researchers to consider a large template set and identify a subset of useful templates.

We would like to emphasize that our main consideration in this paper is to contrast the requisite training time and the accuracy of the learned models using standard TBL and Randomized TBL. Hence, the exact accuracy figures of Randomized TBL are not of concern to us, but rather how the accuracy compares with that of the standard TBL method trained on the same corpus and with the same rule templates.

2 Transformation-Based Learning

Transformation-Based Learning is a symbolic error-driven machine learning method developed by Eric Brill for language learning[Bri95]. Below we give a description of TBL for classification tasks. However, TBL has been used for other purposes such as parsing[SB96].

The model that is produced by TBL to classify new data consists of a set of rules that are applied in sequence; the process is iterative, and an early rule may assign an intermediate tag to a new instance while later rules may alter that tag. During the training phase (Figure 1), TBL starts with an empty model and makes multiple passes through a training corpus where the instances have been given some initial (perhaps dummy) tags.

On each pass, TBL examines each incorrectly tagged instance in the corpus and constructs every rule that would correct that tag. These rules are instantiations of an a priori determined set of rule templates. Each rule that is proposed during a pass is assigned an *improvement score*,

```
Input:  corpus: the training corpus
        dummy-tag: the initial tag assigned to instances
        threshold: the minimum improvement score for continued learning
Output: model: the set of rules for classifying new instances
Initialize model to ∅
For each instance in corpus
        label instance with dummy-tag
Repeat
        Begin
        Initialize best-score to 0
        For each instance in corpus where tag(instance) is incorrect
            For each rule that will correct tag(instance)
                Let score := evaluate(rule)
                If score > best-score
                    Then Let best-rule := rule
                         Let best-score := score
        If best-score < threshold
            Then output model and Exit
            Else add best-rule to model
                Apply best-rule to corpus
        End
```

Figure 1: Algorithmic Description of Training Phase for Standard TBL

which is the difference between the number of correctly tagged instances in the training corpus if the rule were to be applied to the *entire* corpus and the number of correctly tagged instances in the training corpus before the application of the rule. TBL evaluates this set of proposed rules, selects the rule with the best score, adds it to the sequence of rules comprising the model being developed, and applies the selected rule to the training corpus to update its tags. This process is repeated on the updated corpus until all suggested rules have an improvement score less than some threshold.

Since the number of incorrectly tagged instances is large during the initial stages of training, a rule selected in an early pass through the training data usually corrects many incorrectly tagged instances. In contrast, as the training phase proceeds, the number of incorrectly tagged instances on later passes is far fewer and hence the rules selected will tend to alter the tags of fewer instances. Thus, it is typical for the initial rules in the learned model to be very general and the later rules to be more specific.

As noted above, TBL uses a set of rule templates to constrain the rules considered by the system. An example of a template for the dialogue act tagging problem is :

IF	the current utterance u_i contains the word(s) $\mathbf{w}$
AND	the tag currently assigned to the preceding utterance u_{i-1} is $\mathbf{T}_a$
THEN	change u_i's tag to $\mathbf{T}_b$

This template can be instantiated with $\mathbf{w}$="No", $\mathbf{T}_a$=SUGGEST, and $\mathbf{T}_b$=REJECT to produce a sample TBL rule:

> If the current utterance contains the word "No" and the previous utterance is tagged SUGGEST, then assign the current utterance the tag REJECT.

The use of templates requires either that the problem be well enough understood so that restrictive templates can be formulated or that the templates be general enough to take into account all potential feature interactions.

2.1 Characteristics of TBL

TBL utilizes a set of templates that can encode linguistic information and take into account specific aspects of the problem being addressed. Moreover, its learned model consists of a sequence of intuitive rules that stress relevant features and feature interactions. A researcher can analyze these rules to determine what the system has learned and perhaps develop a working theory.

Ramshaw and Marcus analyzed TBL in a series of experiments[RM94]. They reported that TBL appears to be relatively resistant to overfitting and suggested that this characteristic could be explained by several factors, including the fact (also noted in [Bri95]) that each rule is evaluated on the entire training set. In addition, TBL's resistance to overfitting may be explained by observing how the rule sequence of the learned model tends to progress from general rules to specific rules. The early rules are based on many examples in the training set and so are likely to generalize effectively to new data. Although the last rules added to the learned model receive far less support from the training data, they tend to apply only to very specific instances and so they have little or no affect on new data. This is a property shared by other machine learning algorithms that first deal with generalities. However, if the threshold for terminating learning is set very low, then TBL can put such emphasis on the special cases in the training set that overfitting does occur.

TBL is also very flexible with respect to the features that it can utilize. In particular, it can learn set-valued features, where a set-valued feature is one that can take more than one value for a given instance. It can also utilize local contextual information captured by neighboring instances.

Unfortunately, despite these many characteristics that make TBL attractive for language problems, the intractability of TBL's training time is a serious impediment to its use. On each pass through the corpus during the training phase, TBL must construct and evaluate every rule (allowed by the templates) that would correct each incorrectly tagged instance. Not only are multiple rules constructed for each incorrectly tagged instance (one rule for each possible instantiation of a template), but also each constructed rule must be evaluated by hypothetically applying it to every instance in the training corpus. Thus training time is heavily influenced by the number of possible instantiations of the template set and by the corpus size. For language problems where the relevant features and feature interactions are not known from previous work, the templates must be very general to avoid overlooking significant feature interactions and therefore the number of potential rules for each incorrectly tagged instance will be very large. Thus we found that standard TBL was intractable for investigating dialogue act tagging. Others have encountered similar difficulties. For example, Bangalore was thwarted in his attempts to apply TBL to supertagging[Ban97]; as he notes, *(TBL) provides a great deal of flexibility to integrate domain specific and linguistic information into the model. However, a major drawback is that the training procedure is extremely slow...*

3 Randomized TBL

Recall that on each pass through the training data, standard TBL (outlined in Figure 1) will consider each incorrectly tagged instance and for each such instance it will construct every possible rule instantiation of the templates that would correct that tag assignment. Among all these proposed rules, one rule (the one with the greatest improvement score) is selected for inclusion in the learned model. The randomized version of TBL also examines each incorrectly tagged instance, but only R of the N possible template instantiations that would correct the tag assignment are randomly selected for consideration, where R is a predetermined constant. Thus, if at the start of a pass there are M incorrectly tagged instances, then at most $R * M$ rules (many of which could be duplicates) are proposed and evaluated. Only one of these rules (the one with the greatest improvement score) is then selected. In our current experiments, the value of R remains constant throughout training.

Our randomized version of TBL effectively eliminates the dependency between the number of possible template instantiations and training time efficiency. Since only R of the N possible template instantiations will be considered for each incorrectly tagged instance, training time will remain relatively constant as the number of templates increases, thereby maintaining tractability while still allowing the system to select from among a wide range of rules capturing very different feature interactions. Our experimental results show that high accuracy can still be achieved, even with very low values of R.

Why should Randomized TBL result in a high quality learned model? On each pass through the training data, TBL is seeking to identify the best rule. This rule will have a large improvement score and thus will correct many incorrectly tagged instances. Consequently, Randomized TBL has many chances to find this rule — that is, there are many incorrectly tagged instances where this rule could be one of the R rules randomly selected for construction and evaluation. Moreover, there are often several rules that are equally effective. In addition, if Randomized TBL selects a suboptimal rule on one pass, it still has the opportunity on subsequent passes to select rules that compensate for this. Thus, although random sampling may fail to always identify the optimal rule on a particular pass, it is still very likely to construct a sequence of rules that produces optimal results.

Our work on dialogue act tagging supported these intuitions. We implemented standard TBL, trained it on the Verbmobil corpus of appointment scheduling dialogues[REKK96, RK97], and compared it with Randomized TBL trained on the same corpus[Sam98, SCV98b]. As the number of templates and potential rules was increased, standard TBL became intractable and could not complete the training phase in a reasonable amount of time. On the other hand, Randomized TBL showed only a small increase in training time. This small increase was primarily due to the utility of the added templates — when the system had access to more templates that captured useful feature interactions, it made more passes through the training data since new rules could be identified whose improvement score exceeded the threshold. Even for R=1, Randomized TBL achieved accuracy equivalent to standard TBL. Research by Lager produced similar results[Lag99]. He compared Prolog implementations of standard TBL and our Randomized TBL, applied them to the part-of-speech tagging problem, and demonstrated that Randomized TBL was also more efficient on this task than standard TBL, without compromising accuracy.

However, these experiments used a straightforward implementation of TBL. Upon careful examination of Brill's algorithm, we found that it contained optimizations that significantly improved its efficiency but which could not be applied to Randomized TBL. Thus we embarked on a series of experiments that employed Brill's optimized code and compared it to an optimized version of Randomized TBL. We do not claim that it is impossible to construct a more efficient version of standard TBL than Brill's implementation. However, Brill's implementation of standard TBL is the best available at this time and the optimizations it employs are quite innovative. We do not expect to see an implementation of standard TBL that is significantly more efficient.

Since the optimizations in Brill's implementation of standard TBL appear to be most useful if the templates generate only rules that refer to the tag of the current instance, we decided to apply the algorithms to the part-of-speech tagging problem on which Brill had demonstrated success instead of dialogue act tagging.[3] In addition, the availability of large corpora tagged by part-of-speech in the Penn treebank allowed us to evaluate the impact of large versus small corpus size on the algorithms. Although Brill's work used only 34 templates[4], our last experiments used a greatly expanded number of templates in order to evaluate the impact of the template set on both algorithms.

The results shown by our experiments on part-of-speech tagging (Section 5) are consistent with the results we previously obtained with dialogue act tagging, despite the fact that characteristics of the template set and the number of possible instantiations differ considerably between dialogue act tagging and part-of-speech tagging. However, the experiments described in this paper provide a more thorough study that utilizes Brill's optimized implementation of TBL, considers variations in corpus size, and applies TBL to a very different task.

[3]In dialogue act tagging, we found that useful rules did not always refer to the current tag of an incorrectly tagged instance.

[4]More precisely, Brill used a template set that could produce at most 34 different rules for a particular training instance.

4 Implementation of the Training Algorithms

4.1 Brill's Optimized Implementation of TBL

A straightforward implementation of standard TBL would examine each incorrectly tagged instance in the corpus, constructing every rule that would correct the tag assigned to that instance and scoring each rule by computing the affect of applying the rule to the entire corpus. Once rules have been constructed from each incorrectly tagged instance, the rule with the best score would be selected. This process would be repeated until a pass through the training data failed to produce a new rule whose improvement score exceeded a threshold.

Brill's implementation of standard TBL employs a different strategy that produces the same result but significantly optimizes the overall process. However, his implementation appears best-suited for problems where the rule templates force all rules to include the current tag for an instance as one of the conditions in the rule. Thus rules must be of the form

> *IF an instance's current tag is T_i and condition$_2$ and ... and condition$_n$ THEN assign tag T_j to the instance.*

Brill's system first examines all incorrectly tagged instances in the training corpus and computes a set of *confusion pairs*, (T_{guess}, T_{new}), consisting of each incorrect tag T_{guess} paired with the correct tag T_{new} for that instance. The confusion pairs are ordered according to how often they occur in the corpus — thus the ones that capture the largest number of needed corrections appear first in the list and represent the changes that are most likely to lead to the greatest improvement.

The system selects the first confusion pair (T_{guess_1}, T_{new_1}), with the goal of considering all rules that would replace T_{guess_1} with T_{new_1} for any incorrectly tagged instance. However, the construction of rules and their evaluation is done on a single pass through the corpus and represents a major optimization in Brill's algorithm. For each instance in the corpus that is currently tagged with T_{guess_1}, the algorithm determines the impact of assigning T_{new_1} as its tag. If the assignment would correct an incorrectly tagged instance or alter a tag that is already correct, then the templates are used to construct all rules that would produce this change. If the rule already appears in the hash table of rules under consideration (ie., it was constructed from an earlier instance in the corpus), then its improvement score is incremented or decremented depending on whether T_{new_1} is a correct or incorrect tag for this instance; otherwise the rule is entered into the hash table with an improvement score of 1 if T_{new_1} is the correct tag for this instance or -1 if assigning T_{new_1} to this instance alters a previously correct tag.[5] This approach minimizes the effort expended in constructing rules and evaluating them on the entire corpus and represents a major improvement over a straightforward implementation of TBL.

Once the best rule has been selected for one confusion pair, this process must be repeated for the other entries in the list of confusion pairs until the improvement score for the best rule found so far exceeds the number of occurrences of the next confusion pair. At that point, the best rule has already been found since the improvement score for a generated rule cannot exceed the number of occurrences of the confusion pair that spawned it. Thus the use of confusion pairs represents a second major optimization in Brill's implementation since it eliminates many rules from consideration.

4.2 Our Implementation of Randomized TBL

Randomized TBL only constructs R of the possible rules for each incorrectly tagged instance. Consequently, a rule can be constructed for a particular instance that would also apply to an earlier instance, but which need not have been constructed when considering the earlier instance. Thus Brill's strategy of constructing and evaluating rules in parallel cannot be used in Randomized TBL.

However, Brill's strategy of using confusion pairs can be extended for Randomized TBL to optimize the evaluation process. Our implementation associates with each confusion pair 1) a list of the locations in the corpus of training instances that are currently tagged with T_{guess} but whose correct tag is T_{new}, along with the number C_{pos} of such instances, and 2) a list of the locations in the corpus for instances that are correctly tagged with T_{guess} (and thus should not be altered),

[5] This approach may construct some rules whose only impact on the training corpus is to replace correct tags with incorrect ones, but such rules will have negative improvement scores and thus will not pass the threshold test for inclusion in the learned model.

along with the number C_{neg} of such instances. The confusion pairs are sorted according to C_{pos}. The system selects the first confusion pair (T_{guess_1}, T_{new_1}) and constructs R rules for each instance whose tag should be changed from T_{guess_1} to T_{new_1}. Each rule is then evaluated by 1) examining each instance that should be changed from T_{guess_1} to T_{new_1} and incrementing the score for the rule if it applies to this instance, and 2) examining each instance that is correctly tagged with T_{guess_1} and decrementing the score for the rule if it applies to the instance. Note that at any point during the computation of a rule's improvement score, the upper bound on the rule's score is the sum of its current score and the number of instances still to be examined under step 1). Thus the evaluation of a rule is terminated if the upper bound on its score is less than the score of the best rule found so far. Similar to Brill's algorithm, rules are constructed from other entries in the list of confusion pairs only if the next confusion pair could conceivably spawn a better rule — that is, only if the score for the best rule so far does not exceed the number of occurrences of the next confusion pair.

4.3 Comparison of the Implementations

Since Brill's implementation computes improvement scores in parallel with construction of the rules, it not only must construct rules for incorrectly tagged instances (ones where T_{guess} needs to be changed to T_{new}), but it also has to construct rules for instances where T_{guess} is already the correct tag. This is because these latter rules might be proposed later to correct an incorrectly tagged instance and their improvement scores must have already been decremented to account for these earlier instances. Consequently Brill's approach constructs many more rules than need to be examined; in fact, since the number of instances correctly tagged with T_{guess} increases as training progresses while the number of instances incorrectly tagged with T_{guess} decreases, the number of rules proposed for a confusion pair remains relatively constant over the course of training rather than decreasing with the number of incorrectly tagged instances. However, this large number of extra constructed rules is offset with the savings achieved by evaluating the rules in parallel as they are constructed. This parallelism cannot be accomplished with Randomized TBL since only a subset of the possible rules are constructed when an instance is considered. On the other hand, Randomized TBL can terminate evaluation of a rule if the upper bound on its score is less than the score for the best rule so far. Thus each implementation contains a significant optimization.

Theoretically one might expect Randomized TBL to propose fewer rules on a pass through the training data near the end of training, since at that point relatively few instances are incorrectly tagged. However, in both Brill's implementation and the implementation of Randomized TBL, the evaluation of potential rules terminates when the number of occurrences of the next best confusion pair indicates that it cannot spawn a better rule than the best one found so far. Since many locations will be incorrectly tagged early in the training phase, the rules spawned by the first confusion pair are likely to have high improvement scores which will eliminate the need to consider most of the other confusion pairs. Consequently, for both Brill's implementation of TBL and for Randomized TBL, we find that more rules are proposed later in the training phase than at earlier stages.

5 Experimental Comparison and Analysis

To evaluate Randomized TBL and compare it to Brill's standard TBL, we performed a series of experiments with part-of-speech tagging. Our first experiments compare Randomized TBL with standard TBL on a single large corpus using the same templates employed by Brill[Bri95]. We then investigate the impact of different corpus sizes on the two methods. Our last set of experiments significantly increases the size of the template set in order to investigate the ability of the two methods to handle a large number of templates. These experiments were intended to explore different kinds of situations that different applications might encounter.

5.1 Experimental Setup

Brill divided the problem into two parts: an initial state tagger and a contextual rule learner. In the first experiment, we randomly split the Penn treebank training corpus into a 350,000 word subset for training Brill's initial state tagger and a 600,000 word subset for training the contextual rule learner.[6] We used Brill's initial state tagger to assign initial tags to words. Our experiments then focused on the contextual rule learner. We used Brill's implementation of standard TBL and our implementation of Randomized TBL to generate learned models. In both cases, we used the 34 templates provided by Brill. These templates place constraints on the current tag of the word being examined, the word itself, the word one or two instances preceding or following the word being examined, and/or the tag currently assigned to any of the two preceding or following words. The following is an example of a template:

IF	the tag currently assigned to the current word w_i is $\mathbf{T}_a$
AND	one of the two preceding words, w_{i-1} or w_{i-2}, is $\mathbf{w}$
THEN	change w_i's tag to $\mathbf{T}_b$

We also used the threshold values for continuing learning that were suggested by Brill for the particular corpus size. Training was performed using different sized subsets of the 600,000 word corpus, and testing was performed on the standard Penn treebank test corpus of 56,684 words.[7] In the experiments described in Section 5.3, we keep the test corpus and the training corpus used for the initial state tagger the same as in the previous experiment and vary the size of the training corpus for the contextual rule learner. In the last experiment, we also increase the size of the template set. All experiments run Brill's implementation of standard TBL and our Randomized TBL on the same data and the same machines.

5.2 Accuracy and Efficiency on a Large Corpus

We used both Brill's implementation of standard TBL and Randomized TBL with R=1 to produce learned models on the 600,000 word corpus. Since Randomized TBL randomly selects rules for consideration, we performed five runs with Randomized TBL and averaged the results. Table 1 displays the results. Randomized TBL required substantially less training time yet achieved the same accuracy on the test set as Brill's implementation of TBL, even with an R value of only 1.

[6] We used the same proportions of the Penn treebank training corpus as did Brill for training the initial state tagger and the contextual rule learner, on the assumption that Brill had identified the appropriate division of the training corpus. Since we were unable to determine the split of the Penn treebank used by Brill in his reported work, our split of the corpus may not be the same as Brill's.

[7] This is Section))) of the Penn treebank training corpus which is traditionally used by language researchers for testing and comparison of results.

	Processor Time	Accuracy	Rules in Learned Model
Randomized TBL Run-1	7.66 hours	96.11%	172
Randomized TBL Run-2	7.47 hours	96.09%	173
Randomized TBL Run-3	7.32 hours	96.08%	171
Randomized TBL Run-4	7.42 hours	96.10%	172
Randomized TBL Run-5	7.44 hours	96.09%	172
Randomized TBL Average	7.46 hours	96.09%	172
Standard TBL	21.92 hours	96.09%	176

Table 1: Randomized TBL Versus Standard TBL on 600K Corpus

It is interesting to compare the rules comprising the two learned models. Early in the training phase, the rules learned by the two systems are identical. Intuitively, this is what we would expect. For example, the first rule learned by both systems is

> If the current tag of an instance is NOUN and the tag of the preceding word is TO, then change the tag of that instance to VERB

This rule has an improvement score of 1233. This means that the rule was applicable to at least 1233 instances in the corpus and could have been generated at any of these places. Since for a given instance, Randomized TBL randomly selects $R = 1$ of 34 possible rules for any instance, the probability of generating this particular rule is at least $(1 - (\frac{33}{34})^{1233})$ which exceeds 99.99%. As the improvement score of the best rule becomes smaller, the chance that Randomized TBL will learn different rules from standard TBL increases. In our experimental runs, we tend to start seeing differences when the improvement score drops below 40; at this point the probability of generating the best rule falls to about 70%.[8] However, even if Randomized TBL selects a suboptimal rule for inclusion in the learned model on one iteration, it can still learn a compensating rule on a later iteration. In fact, examination of the rule set shows that the two systems occasionally learn rules in a different order, but that these rules generally apply to very different instances and thus do not interact. When the best rules that remain have improvement scores very close to the termination threshold, Randomized TBL will probably not learn all of them. This is because, due to the random

[8]The probability of generating the best rule remains above 50% until the improvement score falls below 23.

nature of its rule construction, Randomized TBL will probably have a pass through the training data where it constructs only rules whose improvement scores are below the termination threshold before it manages to learn all of the better low-scoring rules. This accounts for the learned model produced by Randomized TBL containing fewer rules than the learned model produced by standard TBL. However, the last few learned rules are likely to be very specific and thus have little impact on the accuracy of the learned model applied to the test set, which is supported by our experimental results. On the other hand, if the template set is very large, one could consider increasing the value of R as training progresses in order to increase the opportunity for a useful rule to be found.

How does the time required to identify a rule for inclusion in the learned model differ for the two systems over the course of the training phase? Figure 2 shows the training time per rule for Randomized TBL and for Brill's standard TBL. Both systems learn earlier rules more quickly than later rules. For Randomized TBL, this increase appears to be linear with a small slope. Although Brill's TBL learns the first rule much faster than does Randomized TBL, his system slows down much more rapidly. Thus we see that Brill's optimizations are much more effective early in the training phase (when there are more incorrectly tagged instances and when better rules are applicable on more instances), and that the time advantages of Randomized TBL become appreciable soon thereafter.

5.3 Varying the Corpus Size and Templates

For the part-of-speech tagging task, the available training corpus is quite large. Furthermore, considerable effort has been devoted to carefully narrowing down the set of templates to a relatively small set. This situation is in contrast to our work on dialogue act tagging where the number of instances in the training corpus is much smaller but the number of templates (and possible rule instantiations) is significantly higher. As the proportion of templates to corpus size increases, we would expect the training time gains of Randomized TBL to be even more pronounced. However, in such situations, one might wonder (especially with low values of the random rule sampling parameter R) whether many good rules proposed by standard TBL are even considered by Randomized

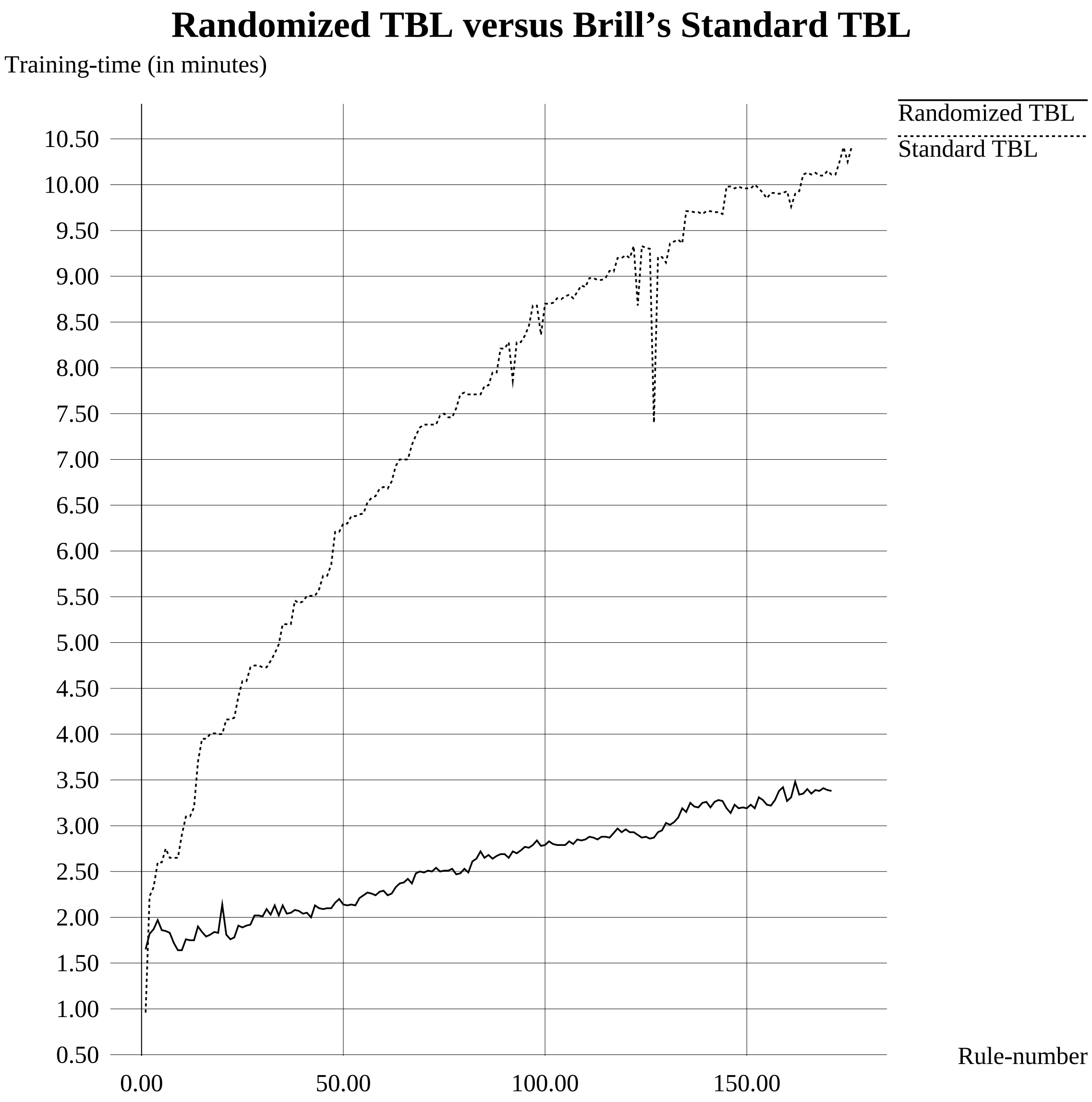

Figure 2: Comparison of Training Time per Rule

	Processor Time	Accuracy	Rules in Learned Model
Randomized TBL Run-1	2.70 hours	96.09%	175
Randomized TBL Run-2	2.74 hours	96.10%	177
Randomized TBL Run-3	2.80 hours	96.11%	180
Randomized TBL Run-4	2.71 hours	96.09%	176
Randomized TBL Run-5	2.72 hours	96.08%	176
Randomized TBL Average	2.73 hours	96.09%	177
Standard TBL	10.74 hours	96.09%	180

Table 2: Randomized TBL Versus Standard TBL on 300K Corpus

TBL due to its random rule selection aspect, and hence whether the difference in accuracy can become significant.

The next two sets of experiments investigate these factors. We vary the corpus size and the number of templates individually in order to thoroughly test and analyze the impact of each variation. For example, with a significantly smaller corpus, the proposed rules will affect the tags of fewer instances even early in the training phase. Thus the chance that Randomized TBL will select an optimal rule decreases. On the other hand, increasing the number of templates increases the number of possible rules, and perhaps there might be several equally effective rules that could be selected.

In the first set of experiments, we retain the templates used by Brill for part-of-speech tagging but change the corpus size. In the next set of experiments, we increase the number of templates significantly and attempt to train on the different corpus sizes used in the first set of experiments.

5.3.1 Accuracy and Efficiency on Smaller Corpora

We randomly split the 600K corpus into corpora of 300K, 150K, 75K and 37.5K words. Tables 2, 3, 4, and 5 present the results of our experiments on these different sized training corpora.

As we had hypothesized, Randomized TBL becomes more efficient with respect to standard TBL as the corpus size decreases. On the 600K corpus, the training time for Randomized TBL is

	Processor Time	Accuracy	Rules in Learned Model
Randomized TBL Run-1	.81 hours	95.97%	157
Randomized TBL Run-2	.82 hours	95.96%	155
Randomized TBL Run-3	.85 hours	95.91%	159
Randomized TBL Run-4	.82 hours	95.98%	158
Randomized TBL Run-5	.82 hours	95.97%	158
Randomized TBL Average	.82 hours	95.96%	157
Standard TBL	4.48 hours	95.96%	165

Table 3: Randomized TBL Versus Standard TBL on 150K Corpus

	Processor Time	Accuracy	Rules in Learned Model
Randomized TBL Run-1	.28 hours	95.94%	152
Randomized TBL Run-2	.27 hours	95.96%	149
Randomized TBL Run-3	.27 hours	95.88%	149
Randomized TBL Run-4	.27 hours	95.93%	146
Randomized TBL Run-5	.29 hours	95.95%	155
Randomized TBL Average	.28 hours	95.93%	150
Standard TBL	1.94 hours	95.95%	159

Table 4: Randomized TBL Versus Standard TBL on 75K Corpus

	Processor Time	Accuracy	Rules in Learned Model
Randomized TBL Run-1	.09 hours	95.41%	121
Randomized TBL Run-2	.09 hours	95.58%	128
Randomized TBL Run-3	.09 hours	95.60%	125
Randomized TBL Run-4	.10 hours	95.48%	132
Randomized TBL Run-5	.08 hours	95.49%	118
Randomized TBL Average	.09 hours	95.51%	125
Standard TBL	.77 hours	95.54%	139

Table 5: Randomized TBL Versus Standard TBL on 37.5K Corpus

34% of the training time for Brill's implementation of standard TBL while it is only 12% for the 37.5K corpus. This is not surprising since Randomized TBL had more impact for later rules in the 600K corpus, and the smaller number of incorrectly tagged instances late in the training phase equates to some extent with a smaller corpus size.

In terms of accuracy we note that the average accuracy of Randomized TBL is still the same as Brill's on the 300K and 150K corpora, but drops very slightly on the 75K and 37.5K corpora. As the corpus size decreases, the number of opportunities to generate a particular rule also decreases and thus the probability that Randomized TBL will add the best rule to the learned model is smaller. Nonetheless, even on a corpus that is $\frac{1}{16}$ the size of the full corpus, Randomized TBL requires substantially less training time than Brill's standard TBL yet produces a learned model of comparable quality. This is despite the fact that we have chosen the lowest possible setting for the rule sampling parameter R, i.e., $R = 1$.

However, as the corpus size decreases, training time, accuracy, and the number of rules comprising the learned model vary more over a set of training runs for Randomized TBL. Although these training time variations have little impact, the variations in accuracy among the different runs can be more of a concern since one cannot know the quality of the particular learned model generated by one execution of Randomized TBL. This would suggest then that for very small corpora, several learned models might be developed and a separate tagged validation set used to choose among the models. Note that the training time would still be only a fraction of that required by Brill's standard TBL. However, this may hamper learning since it decreases the amount of data available for training. Alternatively, as suggested earlier, one might increase the value of R near the end of training when the corpus size is small in order to allow more rules to be considered.

5.3.2 Efficiency and Size of Template Set

In our dialogue act tagging experiments, the number of possible template instantiations for a given incorrectly tagged instance was very large compared to the size of the corpus whereas for the part-of-speech tagging experiments the opposite is true. Experiments in dialogue act

	Processor Time	Accuracy	Rules in Learned Model
Randomized TBL Run-1	.77 hours	95.94%	153
Randomized TBL Run-2	.81 hours	95.95%	159
Randomized TBL Run-3	.88 hours	95.94%	169
Randomized TBL Average	.82 hours	95.94%	160
Standard TBL	unfinished after 33.16 hours		

Table 6: Extended Templates: Randomized TBL Versus Standard TBL on 150K Corpus

	Processor Time	Accuracy	Rules in Learned Model
Randomized TBL Run-1	.30 hours	95.85%	140
Randomized TBL Run-2	.22 hours	95.88%	126
Randomized TBL Run-3	.26 hours	95.85%	139
Randomized TBL Average	.26 hours	95.86%	135
Standard TBL	25.67 hours	95.88%	183

Table 7: Extended Templates: Randomized TBL Versus Standard TBL on 75K Corpus

tagging[Sam98, SCV98b] suggested that Randomized TBL was tractable for large template sets while standard TBL was not. We wanted to more thoroughly investigate the impact of the template set using Brill's implementation of standard TBL.

Once again, we used the Penn treebank and the part-of-speech tagging problem. To experiment with a very large template set, we paired the conditions in each of the 34 original templates; once duplicates had been removed[9], the set consisted of 482 distinct templates.[10] The results from training Brill's TBL implementation and Randomized TBL on 150K, 75K, and 37.5K word corpora[11] are shown in Tables 6, 7 and 8. When the number of possible template instantiations increases, standard TBL must consider many more rules and thus quickly becomes intractable; for example, after 33.16 hours of processing time, standard TBL had produced only 82 rules for the 150,000 word corpus and training was terminated. The situation was actually much worse, since

[9]Some templates in Brill's set represent combinations of individual templates.

[10]Despite the larger set of templates which cover more possible feature interactions, we did not expect to improve on the accuracy obtained by using Brill's templates. Throughout this work, our focus is on a comparative study of the training time and accuracy between standard TBL and Randomized TBL under the same circumstances.

[11]We did not experiment with the 300K or 600K corpora since standard TBL was already intractable on the 150K corpora.

	Processor Time	Accuracy	Rules in Learned Model
Randomized TBL Run-1	.08 hours	95.54%	126
Randomized TBL Run-2	.08 hours	95.40%	124
Randomized TBL Run-3	.06 hours	95.34%	110
Randomized TBL Average	.07 hours	95.43%	120
Standard TBL	8.77 hours	95.46%	166

Table 8: Extended Templates: Randomized TBL Versus Standard TBL on 37.5K Corpus

Brill's standard TBL exhausted the system's memory (192MB). Consequently, even though the process was running on a single-user machine, operating system overhead for memory management was extensive, making the resident time on the machine much longer than 33 hours. On the other hand, Randomized TBL's training time remains relatively constant. Note also that Randomized TBL still produces models with accuracy comparable to standard TBL even though the number of templates has dramatically increased, thereby decreasing the probability of selecting the optimal rule at each stage of training. This further supports the hypothesis that if a suboptimal rule is selected during one iteration of training, a compensating rule can be added during a subsequent iteration.

For well-understood problems such as part-of-speech tagging, linguistic studies can be used to limit the template set. However, for problems that have undergone less study, it is much more difficult to identify an appropriate set of templates a priori that do not omit useful features or feature interactions. In such cases, Randomized TBL offers promise as a means of enabling the researcher to consider a large template set and identify those templates that are most useful for the particular problem. Subsequent runs with this select set of templates can then be performed with either Randomized TBL or standard TBL depending on the corpus size and the resources that are available for training.

As a preliminary test of this proposed strategy, we trained Randomized TBL on the 600K corpus with the extended set of 482 templates. We then extracted the new templates that were most often used to form rules appearing in the learned model and added them to the set of 34 templates used by Brill, resulting in an additional 32 possible template instantiations. We then ran Brill's standard TBL using this larger template set. The resulting learned model achieved an accuracy of 96.14% on the test set, which is .05% higher than was achieved with the 34 templates identified by Brill. This suggests that Randomized TBL can be a helpful tool for identifying useful templates.

6 Summary

This paper has presented a randomized version of Transformation-Based Learning that randomly samples the possible rules that might be considered for correcting an incorrectly tagged instance. It discussed the advantages and disadvantages of Randomized TBL and presented an experimental comparison with Brill's implementation of standard TBL. Although our experiments focused on the part-of-speech tagging task, we believe our conclusions will be found to be more generally applicable (beyond part-of-speech tagging). By comparing the two methods on different size corpora and number of templates, we have covered situations that might arise in different application domains.

Our experiments demonstrate that Randomized TBL can train significantly faster (in some cases, over a hundred times faster) than standard TBL , even using Brill's very efficient implementation. Even with the lowest possible setting for the rule sampling parameter (i.e., with $R = 1$), there is little or no degradation in accuracy. Therefore, we conclude that Randomized TBL can address problems that are intractable in terms of training time for standard TBL, and that it allows the researcher to experiment with a large set of templates capturing many features and feature interactions. Given the many attractive features of Transformation-Based-Learning for language problems, Randomized TBL offers much promise as a useful language learning method.

References

[Ban97] S. Bangalore. *Complexity of Lexical Description and Its Relevance to Partial Parsing.*

PhD thesis, University of Pennsylvania, Philadelphia, Pennsylvania, 1997.

[BR94] Eric Brill and Phillip Resnik. A transformation-based approach to prepositional phrase attachment disambiguation. In *Proceedings of the 15th International Conference on Computational Linguistics.*, 1994.

[Bri95] Eric Brill. Transformation-based error-driven learning and natural language processing: A case study in part-of-speech tagging. *Computational Linguistics*, 21(4):543–566, 1995.

[Lag99] Torbjörn Lager. The μ-tbl system: Logic programming tools for transformation-based learning. In *Proceedings of the Third Computational Natural Language Learning Workshop (CoNLL-99)*, pages 33–42, 1999.

[REKK96] Norbert Reithinger, Ralf Engel, Michael Kipp, and Martin Klesen. Predicting dialogue acts for a speech-to-speech translation system. Technical Report Verbmobil-Report 151, DFKI GmbH Saarbrücken, 1996.

[RK97] Norbert Reithinger and Martin Klesen. Dialogue act classification using language models. In *Proceedings of EuroSpeech-97*, pages 2235–2238, 1997.

[RM94] Lance A. Ramshaw and Mitchell P. Marcus. Exploring the statistical derivation of transformation rule sequences for part-of-speech tagging. In *Proceedings of the Balancing Act Workshop at the 32nd Annual Meeting of the Association for Computational Linguistics*, pages 86–95, 1994.

[RM95] Lance A. Ramshaw and Mitchell P. Marcus. Text chunking using transformation-based learning. In *Proceedings of the Third ACL Workshop on Very large Corpora*, 1995.

[Sam98] Ken Samuel. Lazy transformation-based learning. In *Proceedings of the Eleventh International Florida Artificial Intelligence Research Symposium Conference*, pages 235–239, 1998.

[SB96] Giorgio Sattaand and Eric Brill. Efficient transformation-based parsing. In *Proceedings of the 34th Annual Meeting of the Association For Computational Linguistics.*, pages 255–262, 1996.

[SCV98a] Ken Samuel, Sandra Carberry, and K. Vijayashanker. Dialogue act tagging with transformation-based learning. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics*, pages 1150–1156, 1998.

[SCV98b] Ken Samuel, Sandra Carberry, and K. Vijayashanker. An investigation of transformation-based learning in discourse. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 497–505, 1998.