Use of a Genetic Algorithm in Brill's Transformation-Based Part-of-Speech Tagger

Garnett Wilson and Malcolm Heywood
Faculty of Computer Science
Dalhousie University, Halifax
NS, Canada B3H 1W5
19024010869

gwilson@cs.dal.ca, mheywood@cs.dal.ca

ABSTRACT
The tagging problem in natural language processing is to find a way to label every word in a text as a particular part of speech, e.g., proper noun. An effective way of solving this problem with high accuracy is the transformation-based or "Brill" tagger. In Brill's system, a number of transformation templates are specified *a priori* that are instantiated and ranked during a greedy search-based algorithm. This paper describes a variant of Brill's implementation that instead uses a genetic algorithm to generate the instantiated rules and provide an adaptive ranking. Based on tagging accuracy, the new system provides a better hybrid evolutionary computation solution to the part-of-speech (POS) problem than the previous attempt. Although not able to make up for the use of *a priori* knowledge utilized by Brill, the method appears to point the way for an improved solution to the tagging problem.

Categories and Subject Descriptors
I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search – *Heuristic methods.* I.2.7 [**Artificial Intelligence**]: Natural Language Processing – *Text Analysis.*

General Terms
Algorithms, Experimentation, Languages.

Keywords
Brill tagger, Genetic Algorithm, Natural Language Processing.

1. INTRODUCTION
The labeling (or tagging) of the words in a corpus as parts of speech (POS) such as a noun, verb, adjective, and so on has important applications involving internet security and intelligence, including document filtering, message understanding, data extraction, and information retrieval. There are a number of alternative means of tagging text, with one of the more sophisticated implementations being the transformation-based learning (TBL) of tags, or the "Brill Tagger," named for its designer Eric Brill. The Brill tagger conditions its tagging decisions on a more complex set of events than the entirely probabilistic models like Hidden Markov Models, or HMMs, by examining tags both to the left and the right of a given word [3]. The tagger can also examine not only surrounding tags, but surrounding words and the morphology in the neighborhood of a particular location in the corpus. The way these aspects of the context are examined is through what are called "transformation rules." The transformation rules look for particular situations of inter-related tags, words, or morphology and change a particular target tag if the situation pertains. A learning algorithm takes these transformation rules, which are devised *a priori*, and instantiates and ranks them. The solution is a ranked set of rules for tagging a corpus. The result of the use of the Brill tagger is impressive: its performance on the Wall Street Journal (WSJ) data set (97.0%) is at the high end of accuracy numbers currently reported for tagging (95% to 97%) [7].

The idea of having a set of instantiated rules that can be applied in an attempt to tag a corpus can be seen as an optimization problem: the goal is to minimize the error (or number of different tags) between the correctly tagged corpus and the corpus that has been tagged as a result of the transformation rules. What is desired is the best set of rules, in the right order, to achieve a minimum error. As an optimization problem it is an ideal candidate for improvement using evolutionary algorithms such as a GA, where the GA is used to minimize the cost or error function between the correct tagging and the rule-generated tag. In this implementation GA is used to automatically generate the rules within the Brill tagger and provide an adapted ranking of the rules through the natural process of re-ordering done by the crossover operator.

In Section 2, the current literature related to the use of evolutionary computation for the problem of POS tagging and its application to the Brill tagger is surveyed. Section 3 outlines the original tagger as proposed by Brill. The GA Brill Tagger that is the subject of this work, and the two implementations of it that were created for its theoretical and experimental analysis, are described in Section 4. Section 5 includes the results of the experiments performed using the GA Brill tagger. Conclusions and future work follow in Section 6.

2. EVOLUTIONARY POS TAGGERS
While transformation-based part-of-speech tagging can be seen as an optimization problem, few implementations have used evolutionary methods. There are too many rule-based and

statistical methods of lesser accuracy than the Brill tagger to cover in this work, and their performance is well documented elsewhere. We therefore provide a survey of EC attempts to solve the POS problem in particular, where four attempts related to the use of EC for POS tagging have been described in the literature.

Curran and Wong [4] have actually suggested the use of evolved transformations in the Brill Tagger. In 2000, they published a set-based formal analysis of tagging as the basis for a generalization of Brill's tagger. As a result of this analysis, they claim that an evolutionary strategy would improve the learning time of the algorithm without reduction of accuracy. This could be achieved by changing the size, shape, and number of the templates. Learning would begin with only short and simple templates, and it would progress to longer and more complex templates. The change from simple to more complex templates is what constitutes evolution of the templates according to Curran and Wong; they do not actually suggest the use of evolutionary algorithms or genetic algorithm to automatically produce the initial transformations to be used in Brill's tagger or use the natural action of crossover to provide an adaptive ranking.

Araujo [1, 2] has developed a couple of GA implementations that yield impressive accuracies of up to approximately 95% [1] and a mean of up 96.3% [2] on his chosen corpus (Brown corpus). His first implementation [1] did use a GA for POS tagging, but it differed from the Brill GA hybrid proposed here in a number of respects. The structure of the individual was markedly different: each gene in an individual was simply a tag with probabilities of different associated contexts attached to it. The fitness function for the implementation is rather complex, but the fitness function and individuals' structure amounts to a tag assignment mechanism that is not rule-based but statistical. In our implementation, the structure of a gene amounts to a transformation-based rule. In addition, Araujo's implementation does not attempt to apply rules in a ranked order as in Brill's tagger, but searches for the most probable tag for a word expressed as a gene. Araujo's second implementation [2] simply improves performance by using a more flexible representation of individuals in a messy GA.

Losee [6] describes an implementation called LUST (Linguistics Using Sexual Techniques) that also uses GAs to generate an entire grammar for a test corpus. The system randomly generates both syntactic rules and generates its own POS tags. The grammatical rules of the system are reproduced and mutated. The result is an empirically based grammar that is optimized for the particular documents on which it was evolved. Fitness of individual solutions was measured using an equation and related metrics that reflected the usefulness of an entire grammar for filtering and retrieval purposes. The main goal of Losee was to use his system to improve filtering and retrieval of document components, and thus he did not explicitly measure POS performance.

The Brill/GA hybrid system described in this paper is quite different than LUST. The goal of this work is to provide improved tagging using pre-existing Penn Treebank POS tags, only allowing the generation of associated rules and their ranking through the genetic algorithm. Furthermore, the rules are also transformations in the style of those used in Brill's tagger, with the goal being to provide better labeling of words in a corpus. In contrast, Losee generated his own POS tags and rule forms.

Reiser and Riddle [8] use an evolutionary algorithm (EA) to accomplish POS tagging. Their attempt is actually the only other EC hybrid system found in the literature: a set of inductive logic programs (ILPs) written in Prolog are subjected to evolutionary processes with a suitable crossover operator and mutation replaced by an inductive logic algorithm. Thus, the tagging method is a logic program instead of a ranked set of transformation rules as in these experiments. Their results do show an improvement over traditional ILPs, but the actual accuracy of the hybrid system is approximately 76%, with traditional ILPs boasting only over 73.8% accuracy using Penn Treebank's Wall Street Journal [8]. The traditional Brill tagger solution reports much better results (97.0%) on the same corpus, as does the GA Brill tagger implemented in this paper (up to 89.8%, see Sections 5 and 6).

3. BRILL'S TBL (TRANSFORMATION-BASED LEARNING) TAGGER

The Brill tagging system consists of two main components: a list of error-correcting transformations and a learning algorithm [3]. Transformations in the Brill tagger are of the form *Replace tag T1 by tag T2 when tag T3 is found in a given position*. The tag T1 is located in a rewrite site (the current position where the tag is potentially to be rewritten). Tag T3 is sought in one or more positions ahead or behind the rewrite site, a schema which is called the "triggering environment." If the conditions of the triggering environment are met (tag T3 is in an appropriate position relative to tag T1), then a transformation is triggered and tag T2 replaces tag T1. The instantiated triggering environment (when the variables T1, T2, and T3 are instantiated) is known as a "rewrite rule." An example of a rewrite rule corresponding to the triggering environment above (using Penn Treebank tags) is *Replace tag NN by VB if previous tag is TO,* where that triggering environment is actually the first of Brill's *a priori* transformations. What this transformation rule dictates is that a noun should be changed to a verb if the previous tag is TO.

Transformations are applied left to right to the input, and transformations have a delayed affect in Brill's implementation. This means that applications of the same transformation cannot influence each other. For instance, given the transformation *B replaces A if the preceding tag is A* transforms AAAA into ABBB rather than ABAB. All experiments in this paper implement delayed effect transformations like Brill's original system.

The input data to the original Brill tagging system, and the input used for the experiments in this work, was the Penn Treebank Wall Street Journal corpus, a dictionary indicating each word in the training document(s), and the tag most commonly associated with each word. Using the dictionary, the system first tags each word in the training corpus with its most frequent tag. The learning algorithm then constructs a ranked list of transformations that changes the initial tagging into one closer to the correct one. For every rewrite rule, the algorithm keeps track of how many good and bad transformations it is responsible for. The goodness of the rewrite rule is the number of good transformations it performed, minus the number of bad transformations.

Good rules are appended to an ongoing list, resulting in a list of rewrite rules ranked in descending order of goodness. Every rule good enough to be appended to the list also gets applied to the training corpus before it is stored. The ultimate result of executing this algorithm is a ranked list of rewrite rules that can be applied to a new corpus. The learning algorithm itself is a brute-force, greedy search for the optimal rewrite rules in an optimal order: Every possible instantiation of all transformation templates (triggering environments) is tried on every possible tag position in the training corpus. The ranked list of transformations that results from the learning algorithm can then be used to tag new text.

Brill's results [3] were impressive, clearly outperforming the previous standard of the probabilistic Markov model of Weischedel *et al*. [9]. Brill's tagger achieved an accuracy of 97.0% when training on 600,000 words, and it generated 378 rules in total. For a lower accuracy of 96.7%, the Markov model needed to be trained on a million words and it used 10,000 context probabilities. It is worth noting that when Brill allowed the use of lexicalized rules (allowing explicit use of keywords as part of the triggering environment), the accuracy only improved to 97.2% (an improvement of 0.2%).

Since the interest of this paper is comparing basic performance/accuracy of the original Brill tagger with one that evolves lists of transformations using genetic algorithms, only nonlexicalized transformations are used. Since Brill found the use of lexicalized rules to only result in a minimum increase in performance, it appears safe to pass up the additional complication of lexicalized rules. The implementation to be described in this work was designed to compete with Brill's results in the section of his paper [3] where he uses a closed vocabulary assumption (no unknown words):

- Evolution of the solution (analogous to training in Brill's system) was done using the Penn Treebank Wall Street Journal Corpus.
- The results for the experiment were based on using only the tag-based transformations (or nonlexicalized transformations).
- Evolution was performed over a training set of 600,000 words.

4. THE GENETIC ALGORITHM (GA) BRILL TAGGER

In the problem of POS tagging, one ought not to expect a solution to be present simply due to random generation of the initial population of rules. The GA uses a combination of selection and search operators to incrementally improve the average fitness of the population. The selection operator employed here takes the form of a steady state tournament, thus greater selection pressure is anticipated. In the case of the search operators, the probability of mutation and crossover occurring in children is set to be reasonably high (0.5 and 0.9, respectively), in order to ensure continued introduction of new material into the population (a steady state tournament is elitist). Fitness of an individual is the accuracy resulting from the rules that make up the individual in transforming the initial tagging of the text to match the actual human tagging of the corpus, and the crossover operator used is standard two-point crossover.

There are a number of readily apparent advantages to using a genetic algorithm in the Brill tagging system. The triggering environments in the original Brill tagger were chosen by Brill *a priori*, and the context of the triggering environments is limited by only looking for one type of tag in three positions prior to and following the target position. In contrast, the rules generated by GA are derived randomly (hence not chosen *a priori*) and are also formed through evolutionary processes. The way the bits are interpreted is flexible, and the parsing of the bits into a rewrite rule can allow for different tags to be sought in the triggering environment in potentially any number of positions prior to or after the target position. In this implementation, the transformation rules are more powerful in that they involve a search for particular tags in *each* (rather than *any* of the select positions) of the three previous and following positions. The rules are thus highly adaptive and can structure themselves to reflect individual corpus language structures. Mutation serves to generate new rules to be tried in the algorithm.

In the Brill tagger, the rewrite rules that are considered to be beneficial are applied to the training corpus during the learning process. The order in which the best rules are applied to the training corpus can affect the outcome and ranking of the rules. The order in which the transformation environments are examined seems arbitrary: why should one schema be sought before another? There is no need for the arbitrary ordering of the triggering environments (and thus an arbitrary ordering of the application of beneficial rules) in the GA Brill tagger. An individual in the population is a list of instantiated triggering environments (rewrite rules), and random choice determines the initial order in which the rules are applied. Following that, the order of the rules within individuals is altered with the crossover operator. Since there are a number of individuals in the population, and a considerable number of them undergo evolutionary change through crossover, new orderings of the rules frequently arise and can be tried on the corpus.

The final advantage is that the brute-force, greedy search embodied by the Brill learning algorithm can be avoided. There is a tournament with a number of generations, and in each generation a number of possible solutions to the tagging problem are tried. The tournament can end after some fitness criterion is met, e.g., a number of tags are correct, or the tournament can end after a given number of generations. In either case there is the possibility of finding a solution with less effort than a greedy search.

4.1 Interpretation of Bits and Individuals

A fixed length representation is employed with each individual composed from 378 rules (the number of rules that are presented in a solution to Brill's experiment with a result of 97.0% accuracy). Each rule is a bit sequence of length 48 representing an instantiated triggering environment (rewrite rule). The collection of rules represents an entire solution to the POS tagging problem for the Wall Street Journal Corpus—it is a ranked list of rules that could potentially solve the tagging problem with a high level of accuracy. Each bit in the string has an equal chance of being a 0 or a 1 upon initialization. The way each bit string in the list is interpreted is shown below in Figure 1.

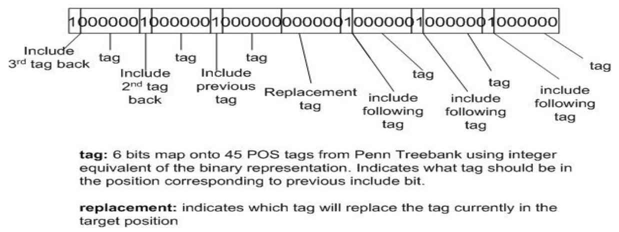
1000000100000010000000000000100000010000001000000
Include 3rd tag back
tag
Include 2nd tag back
tag
Include previous tag
tag
Replacement tag
include following tag
tag
include following tag
tag
include following tag
tag
tag: 6 bits map onto 45 POS tags from Penn Treebank using integer equivalent of the binary representation. Indicates what tag should be in the position corresponding to previous include bit.
replacement: indicates which tag will replace the tag currently in the target position

Figure 1. Interpretation of a bit string of length 48 in an individual used in the GA Brill Tagger.

The leftmost bit in the string indicates whether the triggering environment involves examining the 3rd tag back from the target tag position. The next six bits indicate what tag should be sought in that position by using the integer equivalent of the binary representation. The six bits map onto 45 POS tags from the Penn Treebank, and if the integer equivalent of the six bits exceeds 45, the tag is considered to be the integer equivalent mod 45. The next bit indicates whether to examine the 2nd tag back, and what tag to look for there is indicated by the next 6 bits, similarly for the position previous to the target position. What tag is to replace that of the target position is represented by the next six bits. The tags sought (and whether or not they should be sought) in the 3 positions following the target position are given by the following 21 bits. The interpretation is the same as that described for the positions previous to the target position.

4.2 The GA Brill Tagger Learning Algorithm

This implementation attempts to use the parameters of Brill's experiments (described in Section 3) involving nonlexicalized rules as closely as possible. As described in Section 4.1, each individual can be composed of a maximum of 378 instructions, and the tags located 3 positions previous to the target position and 3 positions following the target position are examined. The bit strings, as seen in Figure 1, only represent nonlexicalized transformations. Thus, there is considerable similarity between the original Brill and GA Brill with respect to both the size of the ranked list and the actual rewrite rules of which the list is composed.

Within the constraints of using a genetic algorithm solution to the POS tagging problem, Brill's other experimental parameters are matched. Training (more properly called "evolution" in our experiments) is done on 600,000 word/tag pairs using the Penn Treebank Wall Street Journal (WSJ) Corpus. In the GA algorithm, the number of individuals/possible solutions must be specified, as well as the criteria for stopping the tournament selection process. This results in two basic approaches.

The first approach requires that at least one individual in the population of solutions should be evolved on all 600,000 words. In order to establish when this had occurred, the number of words that each individual was evolved on was counted. An individual was identified by a numerical id, and the id was kept regardless of whether the individual was replaced or the individual underwent mutation or crossover. For this interpretation, a population of 20 and 100 individuals was created and the tournament was run until *some* individual had been evolved on 600,000 words where each file contains an average of 49 591 words. The algorithm for this interpretation is shown below in Figure 2.

The second approach required that all individuals be evolved on the same 600,000 words. Thus, each individual is evolved on 600,000 words from no files that are alike. This actually provides a more fair comparison to Brill's results, since his solution involves the closed vocabulary assumption. The algorithm from Figure 2 does not adhere closely to the closed vocabulary assumption: evolution on randomly chosen files up to 600,000 words will always leave never before seen phrase situations to be tagged in other files. The reason for this is that there are 1 239 781 words in all 25 WSJ files, so no matter what files an individual is evolved on, they will still encounter a very large number of unknown words and phrases.

To allow this implementation to better meet the closed vocabulary assumption, each individual is evolved on the first 13 files in the WSJ corpus to provide evolution on the same 628 841 words. To evaluate this algorithm fairly, the accuracy of the resulting list of rewrite rules ought to be evaluated on the first 13 files of the corpus. Since each individual is now evolved on 600,000 words in each generation of the tournament, the tournament cannot be stopped based on the criteria of number of words the individual has been evolved on. Thus, it is stipulated that the tournament must end after a given number of generations. This number was chosen to be 25 simply to allow a reasonable run time given all the text processing necessary for fitness evaluation to take place. The algorithm for this interpretation of evolution is given in Figure 3.

```
Initialize corpus words with their most
common tags
Create 20 randomly generated individuals
composed of 378 rules in bit sequences
  While (!(some individual has processed >=
  600,000 word/tag pairs) or error != 0)
    1) randomly select 4 individuals and
    rank them on randomly chosen (0 to 25)
    WSJ texts
    2) replace the chromosomes of the 2
    individuals last in the ranking with the
    chromosomes of the first 2 individuals
    and apply mutation (with probability
    0.5) and crossover (with probability
    0.9)
    3) return all individuals to population

  Fitness function: accuracy of the
  individual's 378 rules applied to the WSJ
  file chosen in tournament step 1
```

Figure 2. GA Brill Tagger algorithm where some individual is to be evolved on 600,000 words.

```
Initialize corpus words with their most
common tags
Create 20 randomly generated individuals
composed of 378 rules in bit sequences
For (generation = 0; generation <= 25;
generation++)
    1) randomly select 4 individuals and
    rank them on WSJ texts 0 - 12
    2) replace the chromosome of the 2
    individuals last in the ranking with the
    chromosome of the first 2 individuals
    and apply mutation (with probability
    0.5) and crossover (with probability
    0.9)
    3) return all individuals to population

  Fitness function: accuracy of the
  individuals' 378 rules applied to WSJ
  files 0-12
```

Figure 3. GA Brill Tagger algorithm where all individuals are evolved on the same 600,000 words.

5. RESULTS

Results were established by considering different types (and thus degrees) of evolution. Furthermore, the results were done in the same manner as Brill's paper: the training (in this case evolution) and test were done once per parameterization of the algorithm. Experiments were run using Java 2 on a 1.0 GHz PC.

Before entering into the results of this implementation, it is useful to consider the accuracy of the initial tagging step. The initialization is the same for this implementation as it is for the original Brill tagger. Each word in the WSJ texts is tagged with the tag that is most commonly associated with all the files of the WSJ corpus. The accuracy of the initial tagging can give some idea of the degree of improvement the GA Brill tagger adds to the POS tagging of this corpus. The tagging accuracy just as a result of the initial tagging is shown below in Figure 4. The overall accuracy of the initial tagging on all WSJ files is 0.803.

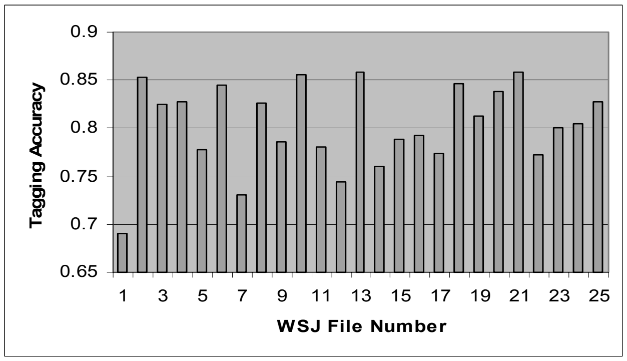

Figure 4. Accuracy as a result of initial tagging in the Brill Tagger.

5.1 Some individual evolved on 600,000 words

The highest fitness individual in each of the tournament generations for the first interpretation of evolution on 600,000 words (where some individual must be evolved on 600,000 words) given a population of 20 is shown below in Figure 5. Note that the end criterion of some individual being evolved on 600,000 words was met after 106 generations of the tournament had taken place.

The highest accuracy reached by an individual was 0.931. The result of the application of all the rules in this best individual on each file in Wall Street Journal corpus is shown below in Figure 6. The tagging accuracy of the implementation over all files in the WSJ test corpus was 0.898, and an additional experiment with the same evolving algorithm was conducted with a higher initial population (100) in order to improve that result. The overall accuracy for that experiment declined to 0.869, reinforcing the hypothesis that the solution is better generated using a smaller search space and letting the operators of mutation and crossover do the exploratory work than spreading the work of the operators over a larger population and hoping for fruitful initial chromosomes.

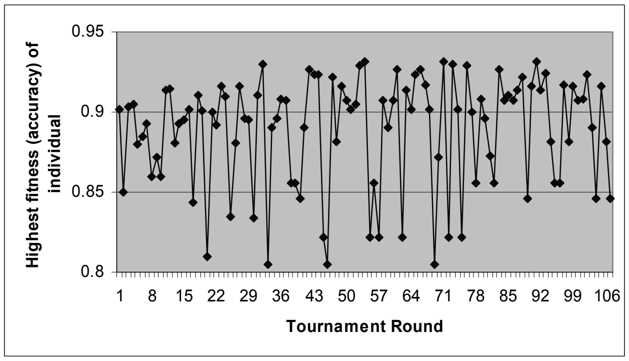

Figure 5. Highest fitness of an individual per generation in the implementation where some individual in a population of 20 is evolved on 600,000 words.

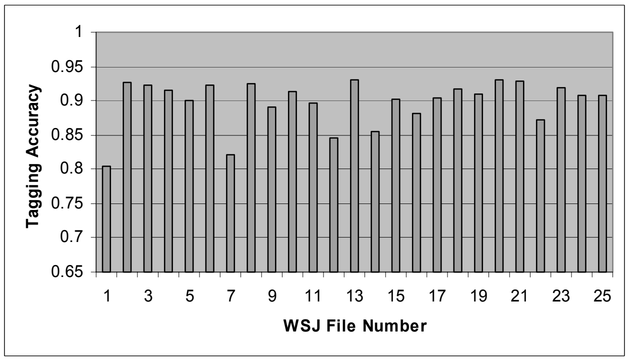

Figure 6. Tagging accuracy of the rewrite set for the individual with highest fitness in the implementation where some individual is evolved on 600,000 words given a population of 20.

5.2 All individuals evolved on 600,000 words

The final implementation allowed each individual in a population of 20 to be evolved on 600,000 words of the WSJ test corpus. As described in Section 5.3.3, this strengthened adherence to Brill's closed vocabulary assumption. Files 0 – 12 of the corpus (amounting to 628 841 words) were used on each individual when they were chosen to compete in a generation of the tournament. Due to high I/O demands and cycles per individual evolved, the tournament was specified to end after 25 generations and the population consisted of 20 individuals. (After all, no improvement was seen in accuracy after increasing population size from 20 as seen in Section 5.1.) The fitness of the best individual for each tournament generation is given in Figure 7 below.

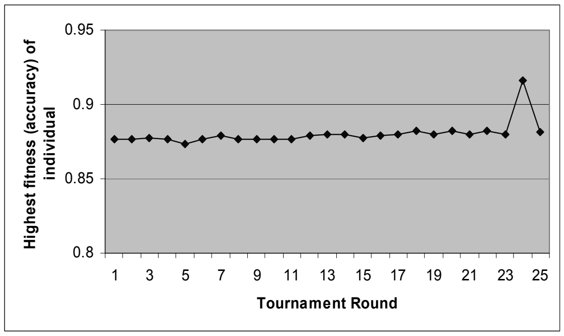

Figure 7. Highest fitness of an individual per generation in the implementation where each individual in a population of 20 is evolved on 600,000 words in a tournament of 25 generations.

Since each individual was evolved on the same 13 WSJ files, the best fitness of an individual stays extremely constant across the tournament generations except for slightly higher fitness in generation 24. The accuracy as a result of the rules for the best individual evolved on the first 13 WSJ documents is given below in Figure 8. Any file beyond 13 would contain unknown words, and thus is not graphed due to the closed vocabulary assumption.

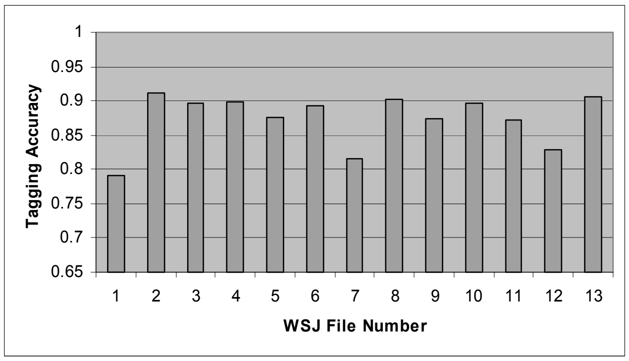

Figure 8. Tagging accuracy of the rewrite set for the individual with the highest fitness in the implementation where each individual in a population of 20 is evolved on 600,000 words.

The accuracy of the best individual over the 13 WSJ documents files on which it was evolved is 0.873, which is not even more accurate than the implementation where a population of 20 had some individual evolved on 600,000 words. This method likely results in over-fitting of the solution, so no tournaments over 25 generations were conducted.

6. CONCLUSIONS

Table 1 below summarizes the parameters of the different implementations and the most important end result of the experiments—the overall tagging accuracy of the resulting list of rewrite rules. Three implementations of the GA Brill tagger were tried. In the first, a population of 20 and 100, respectively, were involved in a tournament until some member of the population was exposed to 600,000 words. The first implementation was actually the most successful, providing an accuracy of 0.898 over all the files in the WSJ corpus test set. The final implementation had a population of 20 repeatedly evolved on approximately 600,000 words (actually files 0 – 12 in the WSJ test set) when they were selected to compete in each of 25 generations.

Interestingly, the implementation with the highest accuracy among the GA Brill taggers involved a low population number with a comparatively moderate amount of processing (the tournament ended after one individual was evolved on 600,000 words). Neither the strategy of increasing the pool of possible solutions (by increasing population size) nor allowing a closer adherence to the closed vocabulary assumption (by always evolving each individual on 600,000 words) increased the tagging performance.

These results indicate factors to take into consideration when designing future GA Brill tagging systems: The latter could mean that the algorithm can be over-fitted, and the former likely indicates that most of the work in achieving a solution is done by the operators rather than exploiting initial material in the chromosomes. All implementations were successful in that they provided definite improvement over the initial tagging that was simply based on word frequencies (0.803).

Table 1. Overall tagging accuracy of the list of rewrite rules resulting from each implementation on the Pen Treebank Wall Street Journal Corpus

Population Size	Criterion to end tournament	Files in each generation used for evolution	Average tagging accuracy
20	600,000 words by some individual	Randomly chosen documents	0.898
100	600,000 words by some individual	Randomly chosen documents	0.869
20	25 generations	Documents 0-12	0.873

Table 2 below contrasts the results of this work with previous POS tagging solutions. The table indicates that no implementation has provided the accuracy of Brill's original tagger (97.0%), but this is unsurprising since Brill's tagger guarantees the best ordering of its rules through the greedy search. However, such a greedy search may not always be computationally viable if the corpus is large. Also, the *a priori* rules may not be well fitted to the peculiarities of different corpuses, and the GA solution can tailor rules to the corpus. The GA Brill hybrid solution thus combines the proven performance of a transformation rule-based approach with the adaptability and flexibility of an evolutionary approach. The only known evolutionary computation-based attempt at POS tagging that would share these benefits was the inductive logic program evolutionary algorithm (ILP-EA hybrid) of Reiser and Riddle [8], but it was not competitive with the GA Brill results (compare rows 2 and 3).

Table 2. Accuracy of POS tagging solutions on the Penn Treebank Wall Street Journal text

Implementation	Evolution	Closed Vocabulary	Accuracy
ILP	6000 per tag	No	73.8%
ILP-EA	6000 per tag	No	76%
GA/Brill	600,000	No	89.8%
Brill Tagger	600,000	Yes	97.0%
Markov Model	1 million	Yes	96.7%

7. FUTURE WORK

Future work may include a detailed analysis of the accuracy of the rewrite rules themselves, and whether certain rules could be removed. In other words, ineffective portions of the individual (introns) could be removed to reduce the rule set needed for tagging. This analysis could also be extended to determine, among the useful tagging rules, how many of them could be removed before a significant loss of accuracy occurred. Brill found that as the end of the ranked list is approached, the tagging rules contribute less and less to accuracy [3]. In fact, the use of the first 200 rules as opposed to the first 100 only yielded an increase in accuracy of 0.2% when lexicalized rules were included. The degree to which this rule is true for solutions where the ranking is produced as a phenomenon of evolutionary processes needs to be determined.

While the accuracy of the GA Brill algorithm is encouraging, there is also the opportunity to improve significantly upon it in future work. Different cost functions to determine fitness are possible as an alternative to raw count of incorrect tags, e.g., sum-squared error. There is also the opportunity to make the rules generated even more adaptive and better suited to the corpus: The bit string that represents each rule can have portions of it function so that they determine which word indices the transformation template seeks, meaning that the neighborhood of the triggering environment can extend either left or right to any degree desired.

In this preliminary work the parameters of the GA were tailored to compare its performance with the original Brill tagger while attempting to match Brill's original constraints. A number of experiments could be conducted to establish the potential of a GA on the POS tagging problem without such constraints. Even using these constraints, we do see significant improvement from initial accuracy, but the population of solutions is not given enough time to fully converge. A larger population size ought to be tried, with a suitable increase in number of generations given for the evolution to take place. Since the population begins at over 80% accuracy, it may also be beneficial to use a fair amount of crossover initially for global search and proceed to gradually decrease crossover rates while increasing mutation rates to perform local search once global search optima are found. Another interesting possibility is to explore the use of novel crossover and mutation operators that may allow similar performance with smaller populations and fewer generations.

Finally, we note that the use of active learning strategies for filtering the dataset would have a significant computational speedup in the evaluation of fitness. Moreover, they have been shown to improve the accuracy of the resulting model [5]. Future work will also attempt to use these methods to boost the accuracy of the GA Brill hybrid system.

8. ACKNOWLEDGEMENTS

The authors gratefully acknowledge the support of a NSERC PGS-B and Honorary Izaak Walton Killam Scholarship (Garnett Wilson), and the CFI New Opportunities and NSERC research grants (Dr. M. Heywood). We would also like to thank Dr. Valdo Keselj for his questions and providing the Wall Street Journal corpus, as well as the referees for their helpful comments.

9. REFERENCES

[1] Araujo, L. Part-of-Speech Tagging with Evolutionary Algorithms. *International Conference on Intelligent Text Processing and Computational Linguistics (CICLing 2002), Lecture Notes in Artificial Intelligence 2276* (2002) 230-239.

[2] Araujo, L. Studying the Advantages of a Messy Evolutionary Algorithm for Natural Language Tagging. *Proceedings of the International Genetic and Evolutionary Computation Conference (GECCO 2003), Lecture Notes in Computer Science 2724* (2003) 1951-1962.

[3] Brill, E. Transformation-Based Error-Driven Learning and Natural Language Processing: A Case Study in Part-of-Speech Tagging. *Computational Linguistics, 21, 4* (1995) 543-565.

[4] Curran, J.R., Wong, R.K. Formalisation of Transformation-based Learning. *Proceedings of the 2000 Australian Computer Science Conference (ACSC 2000)* (Canberra, Australia, 2000). 51-57.

[5] Gathercole, C., Ross, P., Dynamic Training Subset Selection for Supervised Learning in Genetic Programming. *Parallel Problem Solving from Nature III (PPSN'94)*, *Lecture Notes in Computer Science, 866* (1994) 312-321.

[6] Losee, R.M. Learning Syntactic Rules and Tags with Genetic Algorithms for Information Retrieval and Filtering: An Empirical Basis for Grammatical Rules. *Information Processing & Management, 32, 2* (1996) 185-197.

[7] Manning, C. D., Schütze, H., *Foundations of Statistical Natural Language Processing*. Cambridge, MA: MIT Press (2002).

[8] Reiser, P.G.K., Riddle, P.J. Evolution of Logic Programs: Part-of-Speech Tagging. *Proceedings of the Congress on Evolutionary Computation, 2* (1999) 1338-1346.

[9] Weischedel, R., Marie, M., Schwartz, R., Ramshaw, L., and Palmucci, J. Coping with ambiguity and unknown words through probabilistic models. *Computational Linguistics, 19, 2* (1993) 359-382.