A Corpus-Based Natural Language Grammar Optimization Approach using Agent-based Evolutionary Computing

Guy De Pauw
CNTS - Language Technology Group
University of Antwerp - Campus Drie Eiken
Antwerp - Belgium
guy.depauw@ua.ac.be

Abstract
This paper describes one of the first research efforts that considers agent-based evolutionary computing as a machine learning tool for data-driven grammar optimization. This method employs a society of autonomous agents in an evolutionary environment (GRAEL) to yield better overall parsing scores. Experiments show that the GRAEL approach is able to improve parsing accuracy on both the ATIS and the WSJ datasets, comparing favorably to the established ensemble learning techniques of bagging and boosting.

1 Introduction
An expanding body of work in the field of machine learning of natural language is investigating the effect of algorithm bias, as well as that of the bias imposed by the information source itself on the classification accuracy of the algorithms (Breiman, 1996). As a result, many researchers try to develop and apply techniques to overcome these limitations, most often by using ensemble classifier methods. These techniques have indeed proved to be particularly useful when dealing with natural language problems, as a combination of classifiers is often able to provide a better coverage for the large datasets and expansive arrays of features that are paramount in describing this difficult and disparate domain that typically features a considerable amount of sub-regularities and exceptions. Not only system combination and cascaded classifiers are well-established methods in the field of Machine Learning for natural language (van Halteren and Daelemans, 2001; Tjong Kim Sang et al., 2000), also the techniques of bagging and boosting (Breiman, 1996) have been used successfully on a number of natural language classification tasks (Abney et al., 1999; Henderson and Brill, 2000). These techniques hold in common that in no way do they alter the actual content of the information source of the predictor. Simply by re-distributing the data, different resamplings of the same classifier are generated to create a combination of classifiers.

The GRAEL\(^1\) system described in this paper, is a distributed evolutionary computing approach to grammar induction and optimization (De Pauw, 2002). In its most basic form, GRAEL is similar to the aforementioned ensemble learning techniques in that it does not alter the content of the information source itself, but merely achieves a re-distribution throughout a society of autonomous, communicating and co-evolving agents. But whereas it can be shown that combinatory methods can be a hit-or-miss affair in the context of a treebank grammar optimization task, GRAEL seems able to achieve an unequivocal increase in parsing accuracy over the baseline method. This paper evaluates the agent-based evolutionary computing approach of GRAEL as a machine learning tool for probabilistic grammar optimization and compares it to the combination techniques of bagging and boosting for a treebank grammar optimization task.

2 Agent-Based Evolutionary Computing for Grammar Optimization
When considered as a machine learning technique, evolutionary computing is an interesting method that is in theory able to overcome the problem of local maxima in finding a solution to a particular problem, by recombining and mutating individuals in a society of possible solutions. This has made it an attractive

\(^1\)GRAmmar EvoLution
technique for problems involving large, complicated and non-linearly divisible search spaces. The field of evolutionary computing has however always seemed reluctant to deal with natural language and natural language syntax in particular. The fact that syntax is in essence a recursive, non-propositional system, dealing with complex issues such as long-distance dependencies and constraints, has made it difficult to incorporate it in typical evolutionary systems such as genetic algorithms. A limited amount of GA-related syntactic research has focused on linguistic data (Smith and Witten, 1996; Araujo, 2002), but none of these systems are suited to a generic (treebank) grammar optimization task, mainly because the grammatical formalism and evolutionary processes underlying these systems are designed to fit a particular task.

The GRAEL environment provides a suitable framework for the induction and optimization of any type of grammar for natural language in an evolutionary setting. A GRAEL society contains a population of parser-agents in a virtual environment, each containing their own grammar. Through an extended series of inter-agent interactions, these grammars are updated using a form of error-driven learning. While the evolutionary computing approach of GRAEL is able to define the quality of the grammars that are developed over time, the agent-based aspect of GRAEL ensures that the grammar optimization is grounded in the practical task of parsing itself, providing guidance in the search of a global maximum.

In the data-driven GRAEL experiments, the grammatical knowledge of the agents in the society is bootstrapped by using an annotated natural language corpus. At the onset of such a data-driven GRAEL society, the syntactic structures of a treebank are randomly distributed over the agents, so that each agent holds a number of tree-structures in memory. These structures enable the agents to form sentences, as well as provide grammars that allow them to analyze other agents’ sentences.

The actual interaction between agents is implemented in language games: an agent (ag1) presents a sentence to another agent (ag2). If ag2 is able to correctly parse ag1’s sentence, the communication is successful. If on the other hand, ag2 is lacking the proper grammatical information to parse the sentence correctly, ag1 shares the necessary information to arrive at the proper solution.

Probabilistic Redistribution through Error-Driven Learning

Figure 1 displays a toy example of such a language game. In this example, a “treebank” of two structures has been distributed over a society of two agents. The two agents engage in a language game, in which ag1 presents the sentence “I offered some bear hugs” to ag2 for parsing. At this point in time, ag2’s grammar does not contain the proper grammatical information to parse this sentence the way ag1 had intended and so ag2 will return an incorrect analysis, even though it is consistent with its own grammar. Consequently, ag1 will try and help ag2 out by revealing the minimal correct substructure of the correct, i.e. gold standard, parse that should enable ag2 to arrive at a better solution. If more than one such substructure
is found, a random choice is made. If no such substructure can be found, \texttt{ag1} provides the entire analysis.

The grammatical structure provided by \texttt{ag1} is incorporated in \texttt{ag2}'s grammar, who will try to parse the sentence again with the updated knowledge. If \texttt{ag2} were to provide the same erroneous analysis again, \texttt{ag1} will reveal the same substructure, even though \texttt{ag2} already possesses this information. \texttt{ag2} therefore does not gain any new grammatical knowledge this time, but nevertheless incorporates the substructure in his grammar. As the multiple occurrences of this substructure in his grammar will increase its probability, it will therefore be more likely be used in \texttt{ag2}'s subsequent parsing attempts. When \texttt{ag2} is finally able to provide the correct analysis (or is not able to after a certain number of attempts), either \texttt{ag1}'s next sentence will be parsed, or two other agents in the GRAEL society will be randomly selected to play a language game.

These interactions introduce a concept of error-driven knowledge sharing and cause the agents' grammars to grow rapidly, so that the datasets can grow very large in a short period of time. Even though a GRAEL society can in principle have a single epoch type run, the more beneficial setting involves the introduction of new generations. This allows the society to purge itself of bad agents and build new generations of good parser agents, who hold a fortuitous distribution of grammatical knowledge. This involves the use of fitness functions that can distinguish good parsing agents from bad agents. A number of fitness functions were defined and evaluated (De Pauw, 2002), the most important of which relate to average F-scores recorded during inter-agent communicative attempts, as well as the average F-scores achieved by the agent on a held-out test set after each experimental run. The use of generations ideally makes sure that good grammatical knowledge is retained throughout different generations, while useless grammatical knowledge will be marginalized over time. A full overview of all the evolutionary parameters in the GRAEL environment would fall beyond the scope of this paper. We will instead outline the features that are most relevant within the context of the experiments outlined in this paper in Section 4.

3 Bagging and Boosting Treebank Parsers

As a grammar optimization method, GRAEL is related to the established ensemble learning techniques of bagging (Breiman, 1996) and boosting (Freund and Shapire, 1996; Abney et al., 1999). The bagging method tries to yield a performance increase over a dataset, by creating an ensemble of classifiers, each trained on a re-sampling of the initial dataset. These re-samplings are compiled by randomly selecting instances (with replacement) from the training set and placing them in a number of new datasets, equal in size to the original dataset. Each of these classifiers will now introduce a particular bias towards the final decision of the ensemble classifier, ideally resolving the potentially harmful bias of the initial dataset.

The method of boosting is related to bagging as an ensemble learning algorithm in that it also uses a collection of classifiers, but adds an element of internal feedback to the process by keeping track of misclassified instances to consequently assign them more weight in the subsequent resampling.

The bagging and boosting approaches have previously been applied to the treebank parsing problem by (Henderson and Brill, 2000). Their experimental results showed a clear increase in F-score when using a system consisting of 15 bags, while an adapted version of the classic AdaBoost algorithm was unable to yield a significant performance increase over either the bagging approach, or the baseline model. Rather surprisingly the baseline method still outperforms the bagged and boosted systems by a significant margin on exact match accuracy. To enable a direct comparison with the GRAEL method, we will re-evaluate the concepts coined by (Henderson and Brill, 2000) and provide new experimental results using a different parser and data.

Relation to GRAEL

Bagging a treebank parser involves creating a number of treebanks that differ from each other in terms of the distribution of the grammatical units in the individual "bags". Some structures will not be featured in some bags, while they may occur multiple times in another bag. The GRAEL method appears to be similar in concept
to bagging: in the GRAEL system a collection of tree-structures is randomly distributed over a group of agents. But in a bagging approach, the newly created data sets are roughly the same size as the original data set, thanks to the creation of replicates with replacement, whereas the number of tree-structures that agents at the onset of a GRAEL society hold, equals the number of trees in the original data set, divided by the number of agents in the society.

The agents replenish their grammars through an extended series of language games, which for a limited period of time\(^2\), produces a number of agents that have grammars that are fairly different from each other in terms of the distribution of grammatical structures contained. Disregarding the development of the agents over time and only considering the grammars in the society at that moment, one might think of the GRAEL-society as a collection of bags. The difference with the bagging approach however is twofold: whereas bagging tries to find a collection of data sets that, considered as a whole, approach the distribution of the original training set, the grammars in the GRAEL-society at that point were not optimized to mirror the original training set, but rather to perform a particular task, i.e. parsing. The second difference with bagging is that there is no majority voting in a GRAEL-society: typically, only one agent is picked to parse the test set. We will however also describe an experiment in Section 6 in which agents in a society are “bagged”.

Boosting as an ensemble learning method also mirrors some aspects of the GRAEL approach. There is obviously the common concept of distributing a group of tree-structures to provide a number of different resamplings of the original training corpus. But boosting further ties in with GRAEL by effectively implementing a form of error-driven learning, providing a re-distribution of grammatical structures based on errors made by the parser. Whereas bagging tries to create an aggregate of bags that approaches the original training set distribution, boosting can optimize the distribution to achieve higher accuracy scores in parsing.

This is very similar to what is going on in GRAEL, albeit in a different way: if \(ag_2\) parses a sentence incorrectly, \(ag_1\) will cause \(ag_2\) to increase the weight of the relevant (sub)structure in his grammar. GRAEL therefore also clearly incorporates a form of error-driven learning. The difference with boosting lies in the fact that the agents start out with grammars that are totally distinct from one another. The language games consequently make sure that through a form of inter-agent error-driven learning, grammatical information is shared in a way that optimizes the distribution of probability mass for actual parsing, rather than to mirror the distribution of the original training set.

4 Experimental Setup

We used two data sets from the Penn Treebank (Marcus et al., 1993). The main batch of experiments was conducted on an edited version of the small, homogeneous ATIS-corpus, which consists of a collection of annotated sentences recorded by a spoken-dialogue system. The larger Wall Street Journal Corpus, a collection of annotated newspaper articles, was used as a further proof-of-the-principle experiment. The common division between training set (Section 02-21) and test set (Section 23) was used.

We used the parsing system PCFG+PMFG (De Pauw, 2000), which combines a cky parser (Chappelier and Rajman, 1998) and a post-parsing parse forest ranking scheme that employs probabilistic information as well as a memory-based operator that maximizes for each parse the number of nodes that can be retrieved from memory. This memory-based approach to Data-Oriented Parsing (Bod, 1998) ensures that larger syntactic structures are used as the basis for parsing, with a minimal loss of computational efficiency over regular PCFGs. Parsing was performed on part-of-speech tag sequences.

Experiments

The experiments described here compare the performance of two types of parsers: the parser of the fittest agent in a GRAEL society after a certain amount of evolutionary processing on the one hand and the bagging and boosting approaches on the other. For the GRAEL experiments, the training set was randomly distributed (without replacement) over a society of agents (population sizes of 5, 10, 20, 50 and 100). Next, the agents in the society engaged in

\(^2\)Parameters are defined in the GRAEL system which attempt to halt the society during this period in time (De Pauw, 2002).
a series of language games (cf. Section 2). If an agent is observed not to acquire any more rules over the course of \( n \) communicative attempts\(^3\), it is considered to be an end-of-life agent. As soon as two end-of-life agents are available that belong to the 50% fittest agents in the society, they are allowed to procreate by crossing over the grammars they have acquired during their lifespan. The top 75% most probably rules of each ancestor’s grammar is randomly recombined, creating a number of new agents. Two of these agents will take their ancestors’ slots in the society, while the others take the slot of the oldest agents among the 50% unfit agents at that point in time.

The fitness of an agent is defined by recording a weighted average of the F-score during inter-agent communication and the F-score of the agent’s parser on a held-out validation set. This information was also used to try and halt the society at a global maximum and select the fittest agent from the society. For computational reasons, the experiments on the WSJ-corpus were limited to two different population sizes and used an approximation of GRAEL that can deal with large datasets in a reasonable amount of time\(^4\).

Three different experiments were conducted for each ensemble learning technique: a system with only one bag was used as the base-line bagging experiment, which was consequently extended to a 10 bag experiment. Our main focus is the system consisting of 15 bags, as this was reported by (Henderson and Brill, 2000) to yield the best results on a training set and close to the best result on the test set in their experiments.

The method for creating the bags is the same as in (Henderson and Brill, 2000), but we employ a different method for combining the decisions made by the different bags. Instead of using the unweighted constituent voting method (Henderson and Brill, 1999) in which the individual classifiers can contribute constituents to the final decision, we used a weighted voting mechanism for full parsers as a combination technique in the bagging experiments. This implements a simple weighted majority voting method as follows: the 10 most probable parses are gathered from the parse forests of the individual parsers. Next, their respective probabilities are added. This returns an ordered miniature parse forest of at most 10*\( n \) parses. The parse with the highest probability is the parse proposed by the majority voting method.

Although (Henderson and Brill, 2000) provides an interesting application of the AdaBoost algorithm on treebank parsing, the results are such that it is not always clear what the actual effect of boosting proper is. Whereas the bagging experiment employed unweighted constituent voting to decide on the ensemble’s decision, the altered boosting algorithm involves a weighted voting scheme. Even though the boosting algorithm provides such a weight on the fly, following (Henderson and Brill, 1999), these weights must have also been implicitly present in the data of the bagging experiment. We will employ a different boosting algorithm that should provide a clearer insight into the effects of boosting proper. The adjustments of weights in our boosting experiments is defined as follows: over the course of 10 iterations, adjust the weight \( w_i \) of a sentence \( i \) by looking at its proposed parse \( P_i \) and the correct parse \( T \) as follows:

\[
\delta(t_i, t_j): \text{a function that counts the number of constituents in tree-structure } i \text{ that can also be found in tree-structure } j
\]

\[
w_i = 1 - \frac{2 \delta(P_i, T) \delta(P_j, T)}{\delta(P_i, T) + \delta(P_j, T)}
\]

This gives each sentence a weight that is equal to 1 - its F-score. When inducing a grammar for these sentences in the next iterations, the probability of each constituent will be multiplied by this weight + 1, thereby redistributing the probability mass over the grammatical structures to try and resolve previous erroneous analyses. After 10 iterations, we are left with a number of classifiers that each have a certain kind of specialty. Rather than giving the classifiers a weight in the final decision, as in (Henderson and Brill, 2000), this weight is intrinsically provided by the majority voting mechanism previously described: a miniature ordered parse forest is culled from the \( n \) best parses in the parse

\(^3\) \( n \) is by default set to the number of agents in a society

\(^4\) The degree to which this negatively affects performance is the object of future research.
forest of each classifier. A classifier that is sure of its decision will express this in the probability of the tree-structures it proposes, thereby supplying a bigger weight towards final classification in the combination method.

5 Results

Table 1 displays the exact match accuracy and F-scores for the baseline model, a standard PCFG+PMPI parser using a grammar induced from the training set. It also displays scores of the GRAEL system for different population sizes. We notice a significant gain for all GRAEL models over the baseline model, but increasing population size over 20 agents seems to decrease Exact Match accuracy.

Looking at the results of the bagging experiments, we also notice a considerable increase when using a bagging approach on the ATIS-corpus. Using only one bag, exact match accuracy is much lower for the ATIS-corpus. This can probably be attributed to the fact that ATIS, although fairly homogeneous, is a small treebank, so that the absence of certain key structures may well deteriorate parsing accuracy on a test set. Using 10 bags on the ATIS-corpus improves on the baseline model, while using 15 bags further improves on the accuracy. The bagging approach however is not able to outperform the GRAEL-approach on the ATIS-corpus.

The situation is more or less reversed for the WSJ-corpus: using only one bag does not degrade performance significantly over the baseline model. But the system consisting of 15 bags outperforms GRAEL by a significant margin. Nevertheless, GRAEL (100 agents) parses 69 more sentences completely correct than a 15-bag system does.

Table 1 describes the results of the boosting experiment compared to the GRAEL experiments and the bagging experiments. Again, we notice a significant performance increase when using 10 or 15 bags. Boosting using only one bag significantly reduces accuracies for the ATIS, as well as the WSJ-corpus. The 10 to 15 bags systems achieve a similar result on the ATIS-corpus as bagging, i.e. lower than the GRAEL approach. For the WSJ-corpus, it is a close tie between boosting and GRAEL. GRAEL does outperform boosting on exact match accuracy, but the F-scores are very close. Neither of them can outperform bagging on this account.

Since there is a large random factor at work in bagging and boosting, as well as GRAEL, we took the best configuration of the bagging method and GRAEL and repeated the experiment five times on the ATIS data set to test the stability of the methods as a grammar optimization technique. The GRAEL system (50 agents) achieved a similar result in every experimental run, with F-scores ranging from 92.1% to 92.2%. The bagging approach (15 bags) however seemed less stable, with reported F-scores as low as 90.9% and as high as 91.9%.

6 “Bagging” Agents

One final experiment tried to combine the sensibilities of bagging and GRAEL by considering agents as bags and applying a majority voting method on their decision. The basis for this experiment was the 50-agent society: we vary the number of bags for this experiment as well, by considering the n fittest agents of a GRAEL-society as bags. The previously described weighted majority voting mechanism consequently compiles an ordered miniature parse forest and produces the most probable tree-structure.

Table 2 displays the results for this experiments. The first line (1 bag) expresses the standard GRAEL accuracy, achieved by the single fittest agent in the society. The 5, 10 and 15 bags-system yields a slight performance increase on the accuracies of the ATIS-corpus. But for both datasets, a 50 bags-system, i.e. using all agents in the society, decreases performance.

The 5, 10 and 15-bags systems however yield some significant performance increases on the WSJ-data set. Exact Match accuracy advances significantly and so does the F-score. Using majority voting, the 50-agent society is now able to climb up to a 100-agent society in terms of F-score. Interestingly, a side-experiment that used 10 “bags” from a 100-agent society yielded no performance increase over the 50-agent counterpart whatsoever.

7 Concluding Remarks

The results clearly show that all methods described here improve accuracy on both the ATIS and the WSJ dataset. GRAEL outperforms bagging and boosting on all accounts in the ATIS-
experiment. On the WSJ corpus, however, bagging has the advantage over GRAEL on a constituent level. (Breiman, 1996) indeed claims that a method like bagging is less suited for stable predictors. ATIS as a small and homogeneous corpus may therefore not be a suitable dataset for bagging/boosting methods, as it does not provide a sufficiently large stochastic spread. The more elaborate WSJ corpus on the other hand would seem to provide less stable predictors (grammars), hence bagging's edge over GRAEL.

Even though, bagging outperforms GRAEL on the WSJ-corpus when it comes to the F-score, GRAEL still seems to do a better job at improving exact match accuracies. This means that GRAEL on average generates more incorrect constituents than the bagging approach does, but is better at finding a global solution for a sentence. We believe that this is due to the way in which grammatical knowledge in a GRAEL-society is shared, i.e. using (the minimally correct) substructures, which may provide a predilection for larger structures in parsing. The boosting method is not able to outperform bagging, nor GRAEL as a grammar optimization method.

The scores for GRAEL seem closely tied to those of the boosting experiments. Apart from the F-score on the WSJ-corpus, GRAEL outperforms boosting, but only by a small margin. The similarity in results may be explained by both GRAEL and the focal point of boosting, i.e. error-driven learning. Boosting re-adjusted the probability mass of grammatical structures if an error was made on them. This is more or less what is going on in GRAEL as well, but rather than enforcing the information locally, the re-adjustment on the probability mass is being made in another grammar. The slight edge GRAEL has over boosting may perhaps be explained by the fact that the non-local re-adjustment of grammatical information in GRAEL applies more favorably on the task of parsing an unseen test set. Another possible explanation might be that boosting starts out with all the grammatical information present in

<table>
<thead>
<tr>
<th>GRAEL (50)</th>
<th>ATIS</th>
<th>WSJ</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 &quot;bag&quot;</td>
<td>75.9</td>
<td>22.2</td>
</tr>
<tr>
<td>5 &quot;bags&quot;</td>
<td>77.6</td>
<td>22.9</td>
</tr>
<tr>
<td>10 &quot;bags&quot;</td>
<td>77.6</td>
<td>23.1</td>
</tr>
<tr>
<td>15 &quot;bags&quot;</td>
<td>77.6</td>
<td>23.2</td>
</tr>
<tr>
<td>50 &quot;bags&quot;</td>
<td>75.9</td>
<td>22.7</td>
</tr>
</tbody>
</table>

Table 2: "Bagging Agents" Results
the grammar, while the error-driven learning aspect of GRAEL occurs simultaneously with the grammar acquisition of the agents, making the probabilistic re-adjustment a more flexible process.

One of the main selling points for the GRAEL system seems to be its stability with respect to the grammar optimization task. In both bagging and GRAEL, there is a large aspect of randomness to the re-distribution of data. But whereas bagging seems to be more vulnerable to the effects of an unfortunate re-distribution of data, GRAEL is able to rely on the agent-based grounding to make sure that the re-distribution is guided by the task of parsing itself.

As a grammar optimization method GRAEL seems to hold up very well to the established ensemble techniques. Also note that scores reported on the WSJ dataset were achieved on an approximation of GRAEL, whereas the ATIS experiments were conducted on a full instantiation of GRAEL. This may explain the reversed situation when moving from ATIS to WSJ. In any case, it is clear that all of the methods described in this paper are able to obtain a significant performance increase over the baseline model. Preliminary experiments show that GRAEL’s optimization properties are also valid using worse parsers (PCFG) as well as the state-of-the-art system of (Collins, 1997). Future work will extend this line of experimentation to investigate the portability of the GRAEL method to other parsers and data sets.

The comparison of GRAEL and the ensemble learning techniques of bagging and boosting presented in this paper, expose the similarities between the methods. Some important differences are however apparent, indicating that GRAEL holds its own as an optimization method for corpus-induced parsing.

Acknowledgments

The research described in this paper was financed by the FWO (Fund for Scientific Research). The author would like to acknowledge Frederic Chappelier, for kindly making his parser available.

References


