# Evolving Coevolutionary Classifiers under large Attribute Spaces*

John Doucette, Peter Lichodzijewski and Malcolm Heywood†

14–16 May 2009

**Abstract**

Model-building under the supervised learning domain potentially face a dual learning problem of identifying both the parameters of the model and the subset of (domain) attributes necessary to support the model: or an embedded as opposed to wrapper or filter based design. Genetic Programming (GP) has always addressed this dual problem, however, further implicit assumptions are made which potentially increase the complexity of the resulting solutions. In this work we are specifically interested in the case of classification under very large attribute spaces. As such it might be expected that multiple independent/ overlapping attribute subspaces support the mapping to class labels; whereas GP approaches to classification generally assume a single binary classifier per class, forcing the model to provide a solution in terms of a single attribute subspace and single mapping to class labels. Supporting the more general goal is considered as a requirement for identifying a 'team' of classifiers with *non-overlapping* classifier behaviors, thus each classifier responds to different subsets of exemplars. Moreover, the subsets of attributes associated with each team member might utilize a unique 'subspace' of attributes. This work investigates the utility of coevolutionary model building under the case of classification problems with attribute vectors consisting of 650 to 100,000 dimensions. The resulting team based coevolutionary evolutionary method – Symbiotic Bid-based (SBB) GP – is compared to alternative embedded classifier approaches of C4.5 and Maximum Entropy Classification (MaxEnt). SBB solutions demonstrate up to an order of magnitude lower attribute count relative to C4.5 and up to two orders of magnitude lower attribute count than MaxEnt while retaining comparable or better classification performance. Moreover, relative to the attribute count of individual models participating within a team, no more than six attributes are ever utilized; adding a further level of simplicity to the resulting solutions.

*Paper published in Genetic Programming Theory and Practice - VII: 2009, R. Riolo, U.-M. O'Reilly and T. McConaghy (eds) Genetic and Evolutionary Computation Series – 

†Faculty of Computer Science, Dalhousie University. Halifax. NS. Canada.

# 1 Introduction

Team or ensemble based frameworks for machine learning may be used to provide explicit support for the 'divide and conquer' metaphor of problem decomposition. Thus under a classification problem domain, rather than assuming a single model-based classifier[1] per class, the process of credit assignment is able to actively decompose the problem as originally posed. The resulting solution engages multiple classifiers to provide the same class label, but in the case of this work we do so while seeking an explicitly *non-overlapping* interaction between classifiers. Such a non-overlapping behavioral requirement implies that the team of classifiers associated with the same class respond to different partitions of the exemplars comprising the class in question[2]. Thus, under such an approach, the overall solution is potentially much simpler than assuming a single classifier per class. For the purposes of this work the simple solution property has at least two specific properties: (1) the complexity of individual classifiers associated with the same class is less than that when a priori forcing a *single* classifier to represent *each class*, and; (2) the attributes/ features[3] indexed by a team member need only be a subset of the total attributes utilized under the single classifier per class approach. The net result is that the transparency of a solution increases relative to non-team based classifiers and a wider acceptance of machine learning solutions might be expected in general.

Recent advances to team based evolutionary model building appear to represent a particularly appropriate approach for realizing both of the above simplification properties simultaneously. To date, however, there has been little effort to investigate the utility of such models to problem domains with hundreds to hundreds of thousands of attributes. With these goals in mind, we begin by reviewing advances in team-based evolutionary model building under the classification domain (Section 2). Section 3 summarizes the properties of the Symbiotic Bid-Based (SBB) model of coevolutionary machine learning as employed in this study. The evaluation methodology is established in Section 4, where this includes the details of data sets employed and a summary of two alternative classification methodologies that also support the embedded identification of attribute sets (C4.5 and Maximum Entropy Classification). Results of the empirical benchmarking study follow in Section 5, with conclusions and future work in Section 6.

[1] By 'model-based' representation we imply that individuals are required to discover a mapping from the original attribute space to the output space.

[2] Hereafter 'team' and 'ensemble' will be used interchangeably with the non-overlapping behavioral constraint implicit.

[3] The term 'attribute' and 'feature' have became interchangeable in the general literature; although in some works 'feature space' is distinct from the original attribute vector associated with the application domain. In the following we will associate a 'feature count' with all zero argument terms included in a solution, thus including attributes explicitly included in the classifier as a subset.

## 2 Related Work

When faced with a data set comprising of a large potential number of attributes one of two methods are generally employed: filter or embedded [15]. Filter methods divide the overall task into two *independent* steps, attribute subset identification and then classification; a process that potentially makes the overall task computationally faster at the potential expense of overall accuracy. Conversely, the embedded approach takes the view that by performing *both* tasks in one step, as part of a single *integrated* process of learning, the subset of attributes most appropriate for the model of classification can be explicitly identified. A third approach – wrapper methods – use the classification model to iteratively evaluate suggested attribute subsets, but without integrating the two steps within a single learning algorithm; thus any classification algorithm would suffice for evaluation of the suggested attribute subset. However, such methods do not appear to see much utility in practice.

Whether one of the two methods is pursued over the other is often based on additional factors such as the ultimate cost of model building or the availability of expert knowledge appropriate for reducing the size of the attribute space. Moreover, some models of classification have a bias towards including all attributes and then simplifying (e.g., neural networks and SVM models); whereas other models of machine learning begin with a bias towards including a low number of attributes and incrementally include more until an 'optimal' classification performance is achieved (e.g. decision tree induction and evolutionary methods of model building).

This work naturally assumes an embedded approach under the hypothesis that evolutionary methods for constructing models of classification provide a suitable basis for incremental attribute identification. Indeed, previous works have demonstrated that both Genetic Algorithms (GA) and Genetic Programming (GP) are appropriate for attribute subset identification/ attribute creation [13], [24], [26]. In each case evaluation was limited to problem domains with tens of attributes. Moreover, such approaches to classification still fall short of the overall objective pursued in this work as the solution takes the form of a single classifier per class. That is to say, solutions fail to support transparency under the aforementioned two properties of: (1) team-based classifier decomposition through non-overlapping behaviors, and; (2) the identification of (potentially) independent attribute subsets by each team member. More recently, GP was used as a pairwise attribute selector in combination with statistical feature selection and a linearly weighted bi-objective fitness function for wrapper based attribute selection under a Bayes model of classification and dimensionality in the order of thousands of attributes [19]. The work reported here concentrates on the single step embedded approach to classification–attribute selection.

In order to support problem decomposition under evolutionary methods various metaphors have been investigated, including learning classifier systems [2], cooperative coevolution [21], GP teaming [3], [25], and various evolutionary approaches for building ensembles [12]. Some of the generic difficulties faced in attempting to compose such models under the supervised learning domain

of classification include: establishing how many classifiers to include per class; defining an appropriate credit assignment policy; deciding how to combine multiple individuals once identified; and simultaneously scaling the model for efficient evolution over large data sets. Specifically, the generic model of cooperative coevolution established by Potter and de Jong assigns an independent population per 'team member' [21]. Thus, *a priori* knowledge is necessary in order to specify the number of individuals required to participate in the classwise decomposition. The same constraint has limited teaming metaphors under GP [3], [25]. In the case of evolutionary ensemble methods a common requirement is to hold multiple independent runs to produce each member of the ensemble, where this often implies suitable computational support, especially when scaling to large data sets [9]. Moreover, the generic ensemble learning approach does not guarantee that the resulting learners will have non-overlapping behaviors [11], [25]. Indeed, in order to guarantee diversity in the ensemble, techniques such as strongly typed GP [14], local membership functions [18], or negative correlation [4] have been proposed; all under the context of Multi-objective fitness formulations.

With the above discussion in mind, the approach to evolving a team of learners under the classification domain will assume the Symbiotic Bid-based (SBB) framework for model building under discrete domains [17]. Such an approach provides problem decomposition without pre-specifying the nature of the decomposition (c.f. the number of cooperating learners per class) and scales to large data sets care of a competitive coevolutionary mechanism. Section 3 will summarize the characteristics of the SBB learning algorithm.

# 3 Symbiotic Bid-Based framework

## 3.1 Motivation and Methodology

The framework typically assumed for applying model based cases of evolution – such as Genetic Programming (GP) – to the supervised learning domain of classification requires an individual to map exemplars from an attribute space to a class label space. An individual's program expresses the mapping. However, this is not the case under the bid-based GP framework [16]. Instead the task is divided into two components: (1) deciding *which* exemplars to label, or the *bid*, and (2) suggesting class label, or the *action*. In the case of the individual's action, the assumption is made that an individual will always be associated with the same action (class label). Thus at initialization, a problem with $C$ classes results in $\frac{PopSize}{C}$ individuals in the population being pre-assigned to each class. The assignment is defined by assigning a scalar $a$ to each individual at initialization. Scalars are selected with uniform probability over the interval $\{1, ..., C\}$. The actions are *not adapted* during evolution. Conversely, the task of deciding which subset of exemplars to label is expressed in terms of a bid. The individual with maximum (winning) bid suggests their pre-assigned action as the class label. Individuals suggesting an action $a$ that matches the exemplar

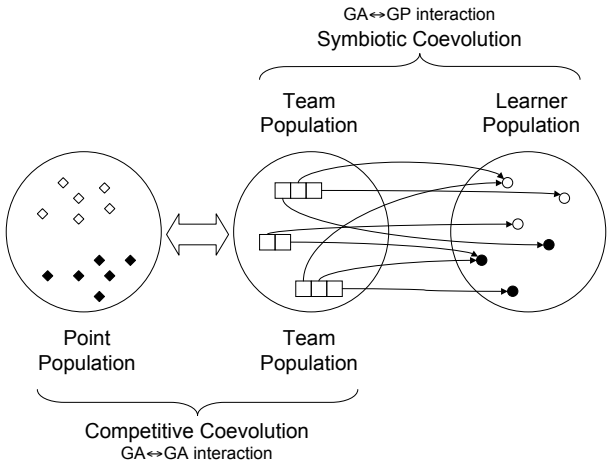



Figure 1: Architecture of Symbiotic Bid-based GP. Black/ white diamonds denote exemplars from different classes; Black/ white circles denote programs with different actions.

class label are rewarded, whereas individuals winning the bid, but not providing a class matching action are penalized.

The most recent form of the bid-based framework – hereafter Symbiotic Bid-based (SBB) – makes extensive use of coevolution [17], with a total of three populations involved: a population of points, a population of learners, and a population of teams (Figure 1). Specifically, individuals comprising a team are specified by the team population, thus establishing a symbiotic relationship with the learner population. Only the subset of individuals indexed by an individual in the team population compete to bid against each other on training exemplars. The use of a symbiotic relation between teams and learners makes the credit assignment process more transparent than in the case of a population wide competition between bids (as used in the earlier variant of the model [16]). Thus, variation operators may now be defined for independently investigating team composition (team population) and bidding strategy (learner population). The third population provides the mechanism for scaling evolution to large data sets. In particular the interaction between team and point population is formulated in terms of a competitive coevolutionary relation [6]. As such, the point population indexes a subset of the training data set under an active learning model (i.e. the subset indexed varies as classifier performance improves). Biases are enforced to ensure equal sampling of each class, irrespective of their original exemplar class distribution [7]; whereas the concept of Pareto competitive coevolution is used to retain points of most relevance to the competitive coevolution of teams.

### 3.2 SBB Algorithm

The SBB model of evolution generates $P_{gap}\%$ new exemplar indexes in the point population and $M_{gap}\%$ new teams in the team population at each generation. Specifically, individuals in the point population take the form of indexes to the training data and are generated stochastically (subject to the aforementioned class balancing heuristic). New teams are created through variation operators applied to the current team population. Fitness evaluation evaluates all teams against all points with $(1 - P_{gap})\%$ points and $(1 - M_{gap})\%$ teams appearing in the next generation. Pareto competitive coevolution ranks the performance of teams in terms of a vector of outcomes, thus the Pareto non-dominated teams are ranked the highest [6]. Likewise, the points supporting the identification of non-dominated individuals (distinctions) are also retained. In addition, use is made of competitive fitness sharing in order to bias survival in favor of teams that exhibit uniqueness in the non-dominated set (Pareto front).

Evaluation of team $m_i$ on a training exemplar defined by point population member $p_k$ results in the construction of an outcome matrix $G(m_i, p_k)$ in which unity implies a correctly classified exemplar, and zero an incorrectly classified exemplar. The ensuing distinction matrix details the pairwise outcome of each team over all exemplars sampled by the point population, or,

$$\begin{cases} 1 & \text{if } G(m_i, p_k) > G(m_j, p_k) \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

where unity implies that point $p_k$ 'distinguishes' between team $m_i$ and $m_j$. The ensuing Pareto competitive coevolutionary process identifies the non-dominated teams and points supporting their identification.

Denoting the non-dominated and dominated points as $F(P)$ and $D(P)$ respectively, the SBB framework notes that as long as $F(P)$ contains less than $(1-P_{gap})\%$ points, all the points from $F(P)$ are copied into the next generation. On the other hand, if $F(P)$ contains more points than are allowed to survive, then the following fitness sharing heuristic is imposed to rank the collection of non-dominated points [23],

$$\sum_i \frac{d_k[i]}{1 + N_i} \tag{2}$$

where $d_k[i]$ is the $i$th entry of the distinction vector for $p_k$; and $N_i$ is the sum of the $i$ *th* entries over the distinction vectors across all points in $F(P)$ i.e., the number of points making the same distinction. Thus, points making the same distinction are weighted less than points making unique distinctions.

An analogous process is repeated for the case of team selection, with $(1 - M_{gap})\%$ individuals copied into the next generation. Naturally, under the condition where the (team) non-dominated set exceeds this fraction, the fitness sharing ranking employs $F(M)$ and $D(M)$ in place of $F(P)$ and $D(P)$ respectively. The resulting process of fitness sharing under a Pareto model of has been

shown to be effective at promoting solutions in which multiple models cooperate to decompose the original $|C|$ class problem into a set of non-overlapping behaviors [16], [17].

Finally, the learner population of individuals expressing specific bidding strategies employs a linear representation. Bid values are standardized to the unit interval through the use of a sigmoid function, or $bid(y) = (1 + \exp -y)^{-1}$, where $y$ is the real valued result of program execution on the current exemplar. Variation operators take the form of instruction add, delete, swap and mutate; applied with independent likelihoods, under a uniform probability of selection. When an individual is no longer indexed by the team population it becomes extinct and deleted from the learner population. Conversely, during evaluation of the team population, exactly $M_{gap}\%$ children are created pairwise care of team based crossover. Learners that are common to both child teams are considered to be the candidates for retention. Learners not common to the child teams are subject to stochastic deletion or modification; with corresponding tests for deletion/ insertion at the learner population. The instruction set follows from that assumed in [17] and consists of eight opcodes ($\{cos, exp, log, +, \times, -, \div, \%\}$) operating on up to 8 registers, as per a linear GP representation.

# 4 Evaluation Methodology

The Evaluation Methodology is first considered from two perspectives, the selection of data sets appropriate for performing the comparison, and identification of alternative models for establishing a realistic baseline of performance. Parameterization of the SBB model is briefly discussed and the metrics deployed for evaluating performance post training are presented.

## 4.1 Data Sets

Data sets with large attribute spaces are frequently encountered under the context of document analysis (information retrieval), speech recognition, bioinformatics, and image processing. In this work, we make use of data sets from the domains of document analysis and image processing. In the case of the image processing domain, the 'Multifeature' and 'Gisette' data sets were employed [1], where both pertain to the recognition of handwritten digits (Table 1) and used 'as is' with no pre-processing applied. The Multifeature data set is a 10 class problem with each class equally represented; whereas Gisette is a binary classification problem in which 55% (45%) of the exemplars are in-class (out-class). Moreover, Gisette has the additional property that half of the attributes (2,500) are 'probes,' thus redundant from the perspective of building an appropriate classification model.

In the case of the document analysis domain, three binary classification problems were composed from the UCI Bag-of-words data set [1]. The data set is comprised from a series of distinct document repositories. The repository content are unlabeled, however, it is known from which repository a document is

Table 1: Data set properties.

| Data set | Exemplar Count train (test) | Feature Count |
|---|---|---|
| Handwritten character recognition | | |
| Multifeature | 1,510 (490) | 649 |
| Gisette | 6,000 (1,000) | 5,000 |
| Document Classification: Bag-of-words | | |
| NIPS | 7,000 (3,500) | 12,419 |
| Enron | 7,000 (3,500) | 28,102 |
| NY Times | 7,000 (3,500) | 102,660 |

sourced. Thus we first combine the common words from the NIPS, Enron and New York Times (NYT) repositories; whereas the three binary classification problems entail distinguishing documents in the NIPS/ Enron/ NYT repository from the combination of all three. In each case document files were normalized with respect to the target class. Thus under the goal of distinguishing the Enron repository from NIPS and NYT, only words appearing in the Enron vocabulary were used to build the corresponding bag-of-words across all three document repositories. The resulting documents were labeled as in-class if they came from the set of documents that originated the vocabulary or out-class otherwise. This resulted in the largest attribute spaces deployed during the ensuing performance evaluation, Table 1. Class representation was also generally unbalanced with in-class representation at 14, 29 and 43 percent respectively for NIPS, Enron and NYT.

## 4.2 Comparator Models of Classification

In establishing a set of baseline classifiers we considered two alternative examples of models that operate under an explicitly embedded paradigm and are widely utilized under large attribute space domains: decision tree induction and Maximum Entropy Classifiers (MaxEnt). Both models make use of entropy frameworks for model building. However, decision tree induction – C4.5 – naturally assumes a greedy incremental non-linear model building methodology. As such this gives the model the explicit ability to trade off model complexity/ feature count with classification performance. Conversely, MaxEnt classifiers are based on a linear model and might therefore be expected to utilize many more attributes relative to non-linear models such as C4.5 or SBB. However, they have repeatedly been shown to be very accurate under domains with high feature counts, even relative to methods incorporating SVM models of classification [10]. Indeed, both SVM and MaxEnt are large margin classifiers, with the SVM approach formulated for exemplar optimization and MaxEnt formulated for attribute selection [10].

Finally, in both cases we also consider the impact of model pruning on the

classification performance of the resulting models, with the goal of establishing to what degree the baseline models can approach the feature counts returned under SBB solutions. In the following subsections we provide background on the parameterization/ modifications necessary prior to benchmarking and a summary of the C4.5 and MaxEnt approaches.

#### 4.2.1 C4.5 Decision Tree Induction

C4.5 is a widely used model for the construction of decision trees under a recursive algorithm in which attributes are incrementally added to the model care of their respective maximum normalized information gain relative to class label [22]. The deployment used here is essentially the original code from Quinlan with modifications to support wider ranges of (confidence value) pruning than would normally be the case. In order to support efficient operation under the larger data sets, modification was necessary of the code in order to accept implicit data formats. Naturally, extensive evaluation was performed to verify that results remained consistent with the original version under smaller attribute dimensions. Such formats are widely used in text classification domains where they provide a significant reduction on memory requirements under sparse data sets.

#### 4.2.2 Maximum Entropy Classifier

MaxEnt methods are either based on a conditional distribution, $P(y|x)$, [20] or a joint distribution, $P(y, x)$, [10]. Moreover, MaxEnt models' each class independently, thus the conditional probability of a binary problem becomes,

$$P(y = +1|x) = \frac{\exp(y(w^+)^T x)}{Z(x)} \tag{3}$$

for the in-class exemplars, and,

$$P(y = -1|x) = \frac{\exp(y(w^-)^T x)}{Z(x)} \tag{4}$$

for the out-class exemplars. However, $Z(x) = \exp(y(w^+)^T x) + \exp(y(w^-)^T x)$, thus a conditional MaxEnt classifier reduces to a logistic classifier i.e., a sigmoid function applied to the linear combination of weights $w = (w^+) + (w^-)$, with an exponentially weighted error term,

$$E_{\log}(y_i w^T x_i) = \log(1 + \exp(-2y_i w^T x_i)) \tag{5}$$

This is the most common formulation and will be employed here. In addition, the frequently employed $l_2$ Gaussian regularization factor for reducing the likelihood of overfitting will be assumed [20], [10], [15].

The foregoing description establishes the basis for the definition of the error term, but says nothing about the scheme employed for adapting the free parameters, $w$. One of the very nice properties of MaxEnt methods is that the

Table 2: Parameterization of the Symbiotic Bid-Based model.

| Parameter | Value |
|---|---|
| Team/ Point population | 90 |
| Point replacement ($P_{gap}$) | 1 / 3 |
| Team replacement ($M_{gap}$) | 2 / 3 |
| Max. Team size | 100 |
| Prob. Team add/ remove/ swap | 0.1 |
| Prob. Learner add remove/ swap/ mutate | 0.1 |
| Max. Generations | 30,000 |
| Number of Trials | 40 |

constrained multi-objective formulation results in a single unimodal objective search space [20]. As such, gradient based optimization routines are sufficient. However, it is still important to address stability issues (c.f. sparse training data) and direct inversion of the Hessian matrix is generally not possible. In this work, we make use of a recent Conjugate Gradient (binary) and BFGS (multi-class) implementation, or MegaM, [5] – rather than the originally widely employed Improved Iterative Scaling (IIS) routine [20] – where MegaM provides a considerable speedup over the IIS methodology.

Finally, we note that pruning was applied post training through the application of a simple thresholding scheme in which attributes with free parameters below the threshold were ignored. Such a simplistic scheme was deemed sufficient for qualifying to what degree the resulting linear model was reliant on the overall composition of the attribute space (as opposed to the potential ability of a non-linear model to compose features from a smaller subset of the total attribute space).

## 4.3 SBB Configuration

Relative to the original SBB configuration the most significant modification necessary to undertake this work was to: (1) provide support for implicit data formats, and; (2) extend the range of attributes learners may index from 64 to over 100,000. Parameterization of the model essentially follows that of the original work [17], but with larger team sizes appearing here, and is summarized in Table 2. A distribution of the source code and data is available (http://www.cs.dal.ca/~mheywood/Code/SBB/).

## 4.4 Post Training Performance Metrics

Post training performance will be assessed from the perspective of classification and feature count. In the case of classification performance we make use of detection (sensitivity) as measured class-wise, resulting in a multi-class measure of detection. Thus, defining the class specific detection rate as $DR(i) = \frac{tp(i)}{tp(i)+fn(i)}$

where *tp(i)* and *fn(i)* are the true positive and false negative counts under class $i \in \{1, ..., C\}$, leads to the following definition for class-wise detection,

$$CW\text{-}detection = \frac{DR(1) + \ldots + DR(C)}{C} \tag{6}$$

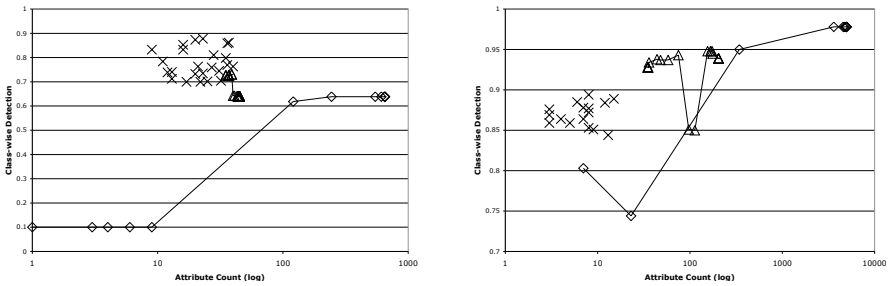
Such a measure is independent of the distribution of exemplars per class. Thus under an imbalanced binary data set in which 95% (5%) of the exemplars were out-class (in-class) a degenerate classifier might label all exemplars as the out-class and achieve an accuracy of 95%; whereas the CW-detection metric would return a value of 50% or more generally $\frac{1}{C}$. Feature count will be measured in terms of the number of zero argument terms included in the model i.e., the number of constants and unique domain attributes actually utilized.

# 5 Benchmarking Results

Benchmarking results will be summarized in terms of 2-D scatter plots of CW-detection versus Feature count. SBB solutions are plotted per run; C4.5 and MaxEnt solutions are plotted for increasing levels of pruning c.f. the pruning threshold of C4.5 and the post training thresholding of the model free parameters in the case of MaxEnt. As such the C4.5 and MaxEnt results are likely to span from complex but most accurate, to the simplest achievable but (relatively speaking) least accurate. This is further emphasized by linking the points formed by pruning C4.5 and MaxEnt solutions to provide a corresponding performance curve. Points which tend to the top left of a curve will naturally dominate the performance of other points in a manner similar to that used to interpret ROC curves. However, the interaction between CW-detection and attribute count will not necessarily result in a monotonic curve. Finally, SBB solutions will naturally result in a distribution of points, care of the multiple stochastic sources of variation implicit in GP, thus the *training* partition is used to identify the top 50 percent of solutions for which test evaluation is performed.

## 5.1 Character Recognition data sets

Figure 2 characterizes performance of the three classifiers under the two Character Recognition data sets considered in this study (Multifeature and Gisette). In the case of the smaller attribute space of the Multifeature data set SBB solutions appear to be stronger in terms of both simplicity and classification performance with the equivalent of up to nine of ten classes correctly classified while using 10 to 70 of the 649 attributes. C4.5 and MaxEnt managed to classify the equivalent of 6 to 7 of the classes, with MaxEnt degenerate under the higher levels of pruning. Moreover, outside of the degenerate solutions (i.e., CW-detection of 0.1) there is a clear ordering of model feature count from SBB (simplest) to C4.5 to MaxEnt (largest feature utilization). This pattern is also reflected under Gisette, with the inclusion of additional attributes improving CW-detection from 90% (SBB) to 95% (C4.5) to 98% (MaxEnt). The penalty




(a) Multifeature (b) Gisette

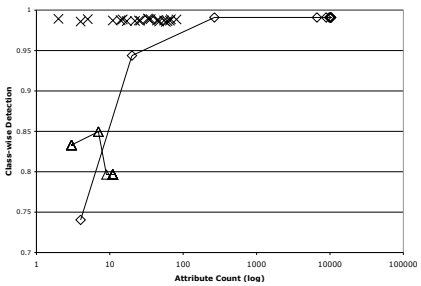
Figure 2: Test CW-detection versus complexity on the Character Recognition data sets. Feature counts are 649 (Multifeature) and 5,000 (Gisette). $\times$ denote solutions from SBB; $\triangle$ denote solutions from C4.5; and $\Diamond$ denote solutions from MaxEnt.

paid for the increased classification performance appears in terms of attribute count, where MaxEnt generally utilizes thousands of attributes whereas SBB generally uses no more then 15. We also note that all but the larger MaxEnt models managed to index less than half of the attributes under Gisette; where half of the attributes are known to be redundant/ duplicate probes although the exact identity of the probes remains concealed [1].

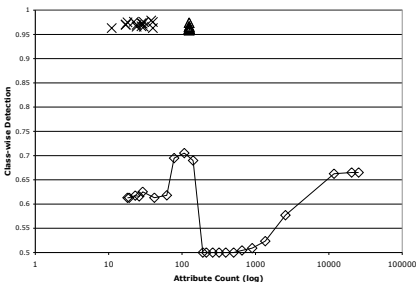
## 5.2 Bag-of-words data set

Figure 3 characterizes performance under the NIPS, Enron and NY Times (NYT) domains from the UCI 'Bag-of-words' data set; as formulated in terms of three independent binary classification problems. In the case of NYT, results are expressed in terms of MaxEnt and SBB alone; C4.5 requiring more memory capacity than was available under the 2GB RAM limit imposed by the computing platform. In this case the simple SBB models are generally able to reach the classification performance identified under MaxEnt. Indeed, they are within 1% of MaxEnt under NIPS, about 25% more accurate under Enron and within 5 to 2% of MaxEnt under NYT while utilizing 15 to 90 out of a possible 12,419 (NIPS), 28,102 (Enron) or 102,660 (NYT) attributes. Moreover, at each level of model feature count identified by SBB, the SBB models dominated the corresponding MaxEnt solution. C4.5 found solutions with the same attribute counts as SBB under NIPS, but were about 15% less accurate; whereas under Enron C4.5 were as accurate, but only by including 50 more attributes than SBB.

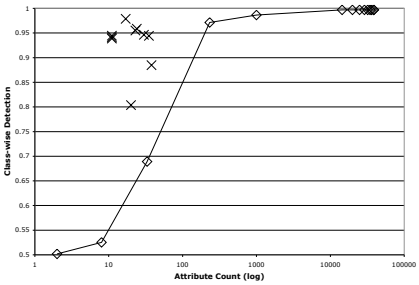
In the case of MaxEnt one characteristic of interest was the pair of peaks appearing under the Enron category. Rather than pruning resulting in a monotonic decline in performance (as in all previous cases) a series of two monotonic performance profiles are identified. Moreover, it is the curve from the more complex MaxEnt model which begins at a lower level of classification performance




(a) NIPS

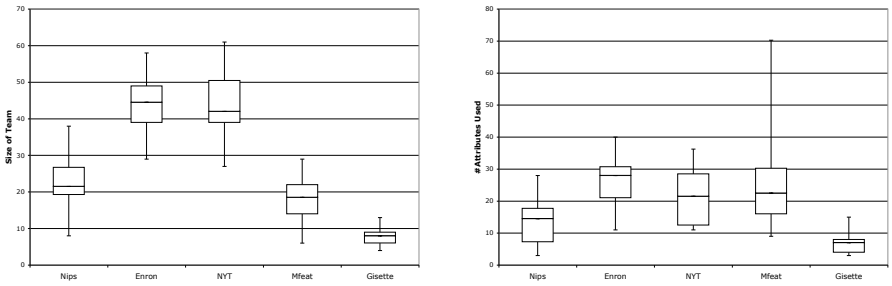



(b) Enron




(c) NY Times (SBB and MaxEnt only)

Figure 3: Test CW-detection versus complexity on the Bag-of-words data sets. Feature counts are 12,419 (NIPS), 28,102 (Enron) and 102,660 (NY Times). $\times$ denote solutions from SBB; $\triangle$ denote solutions from C4.5; and $\Diamond$ denote solutions from MaxEnt.




(a) SBB team size (num. team members)

(b) SBB total num. unique attributes indexed per team

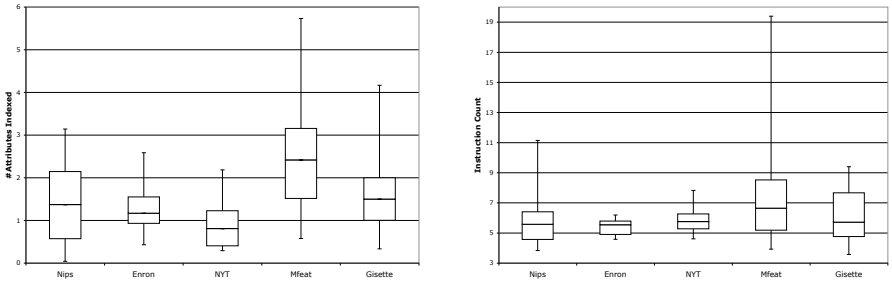
Figure 4: Summary of SBB Team complexity on each data set.

and decays to degenerate solutions; whereas the second MaxEnt peak identified solutions that were simpler and avoided degenerate solutions.

## 5.3 SBB team properties

Figure 4 summarizes complexity across the SBB teams as a whole. Gisette appears to require the lowest team complexity, although some of this characteristic is undoubtedly due to the problem with the smallest attribute count (Multi-feature) also being a ten class problem; thus requiring a greater diversity in model behavior. The three larger 'bag-of-words' data sets did establish some correlation between total attributes indexed over the entire team and attribute count of the original problem domain, Figure 4. However, individual models participating in the team generally indexed no more than in the case of Multi-feature or Gisette, Figure 5.

Figure 5 summaries complexity as measured per individual. It is apparent that the total number of unique attributes indexed by individual team members is very low, or typically less than 4 attributes. This makes for very simple rules that are able to act independently from each other; as opposed to C4.5 which builds a single monolithic solution from a hierarchy of decisions, thus building up to quite complex rules as the tree depth increases. Indeed, the simples SBB rules tend to be of the form "if attribute X appears in document, it is about topic Y". In the case of several data sets we note that some individuals are returned that do not index any attributes. In these cases, the team member is bidding a constant value, leaving the bids from the alternate action (class) and/ or same action (class) to provide the counter balancing bid strategy. Finally, we note that relative simplicity in terms of attribute count is not being traded for greater model complexity. Specifically, after removal of structural introns, team members generally consisted of between 4 to 9 instructions, thus, not detracting from the overall simplicity of SBB solutions, Figure 5 (b).




(a) SBB total num. unique attributes indexed per team member

(b) SBB instruction count per team member

Figure 5: Summary of SBB team member (learner) complexity on each data set. Box boundaries denote 1st, 2nd (median) and 3rd quartiles. Whiskers denote max and min.

# 6 Conclusions

A case is made for the utility of evolutionary model based teaming (or ensembles) under classification problems described over large attribute spaces. The initial hypothesis was that when teams are explicitly designed to seek non-overlapping behaviors, assuming an evolutionary bias to model building would enable the resulting teams to provide very simple solutions without compromising classification performance. Benchmarking conducted under data sets selected from the domains of character recognition and document retrieval (as represented under a bag-of-words vector space model) appears to support this hypothesis. In particular the SBB paradigm of evolutionary teaming/ ensemble generation tends to be more effective at balancing classification performance versus feature count. Conversely, the domain standard of MaxEnt Classification can be counted on to maximize classification performance at the expense of attributes indexed; whereas the C4.5 model of classification appears to build models with an intermediate level of complexity and classification performance.

Key properties from SBB supporting this result take at least three forms: (1) Active learning – evolving solutions directly over the entire training partition would represent a prohibitively expensive computational overhead. In this work a Pareto competitive coevolutionary approach was assumed although alternatives such as host-parasite models or stochastic sampling would also be appropriate. (2) Cooperative problem decomposition – a wide range of ensemble methods exist, however, most do not support non-overlapping models of problem decomposition. Instead widespread use is made of post training voting schemes. This results in multiple models responding to each exemplar and clarity of the solution is lost (c.f. a weak learner metaphor). The SBB algorithm specifically addresses this problem by using fitness sharing to discount the Pareto evaluation before ranking solutions. A learning bias supporting the reward of teams consisting of non-overlapping bidding strategies is therefore es-

tablished. (3) Clear paths of credit assignment – unlike the traditional process of classification through mapping exemplars to a class membership value, the SBB approach explicitly separates the generally combined tasks of "what to do" (action) and "when to do it" (bid). Moreover, this is reinforced by assuming a symbiotic model which explicitly separates the tasks of optimizing team membership and evolving bidding policy. Without such a separation the bidding competition responsible for establishing cooperative team behavior would have to take place across an entire population, thus each time children are created the current team interaction would face disruption.

Natural extensions of the current study might consider the case of biomedical data sets [19] or investigate the impact of attribute support on the relative cost of model complexity, where this appears to be of particular importance to non-linear classifiers such as the SVM [8]. More generally, we are also interested in the utility of the SBB framework to problem domains with temporal discounting (reinforcement learning). Such a context might also benefit from the ability to pose solutions in terms of teams, as indicated by ongoing research in multi-agent systems in general.

## Acknowledgments

This work was conducted while John Doucette held an NSERC USRA summer scholarship and Peter Lichodzijewski held a Precarn Graduate Scholarship and a Killam Postgraduate Scholarship. Malcolm Heywood would like to thank research grants from NSERC, MITACS, and CFI and industrial sponsorship from SwissCom Innovations SA. and TARA Inc.

## References

[1] A. Asuncion and D. J. Newman. UCI Repository of Machine Learning Databases [http://www.ics.uci.edu/˜mlearn/mlrepository.html]. Irvine, CA: University of California, Dept. of Information and Comp. Science, 2008.

[2] E. Bernado-Mansilla and J.M. Garrell-Guiu. Accuracy-based learning classifier systems: Models, analysis and applications to classification tasks. *Evolutionary Computation*, 11:209–238, 2003.

[3] M. Brameier and W. Banzhaf. Evolving teams of predictors with linear Genetic Programming. *Genetic Programming and Evolvable Machines*, 2(4):381–407, 2001.

[4] A. Chandra, H. Chen, and X. Yao. *Trade-off between diversity and accuracy in ensemble generation*, chapter 19, pages 429–464. 2006. In ([12]).

[5] Hal Daumè III. Notes on CG and LM-BFGS optimization of logistic regression, August 2004. Paper and code available at http://www.cs.utah.edu/~hal/megam.

[6] E.D. de Jong. A monotonic archive for pareto-coevolution. *Evolutionary Computation*, 15(1):61–93, 2007.

[7] J. Doucette and M.I. Heywood. Gp Classification under Imbalanced Data Sets: Active Sub-sampling and AUC Approximation. In *European Conference on Genetic Programming*, volume 4971 of *Lecture Notes in Computer Science*, pages 266–277, 2008.

[8] J. Doucette, A.R. McIntyre, P. Lichodzijewski, and M. I. Heywood. Problem decomposition under large feature spaces using a coevolutionary memetic algorithm. *Manuscript under review*, 2009.

[9] G. Folino, C. Pizzuti, and G. Spezzano. GP ensembles for large-scale data classification. *IEEE Transactions on Evolutionary Computation*, 10(5):604–616, 2006.

[10] P. Haffner. Scaling large margin classifiers for spoken language understanding. *Speech Communication*, 48:239–261, 2006.

[11] K. Imamura, T. Soule, R. B. Heckendorn, and J. A. Foster. Behavioral diversity and a probabilistically optimal GP ensemble. *Genetic Programming and Evolvable Machines*, 4(3):235–253, 2003.

[12] Y. Jin, editor. *Multi-Objective Machine Learning*, volume 16 of *Studies in Computational Intelligence*. Spinger-Verlag, 2006.

[13] K. Krawiec. Genetic Programming-based Construction of Features for Machine Learning and Knowledge Discovery tasks. *Genetic Programming and Evolvable Machines*, 3(4):329–343, 2002.

[14] R. Kumar, A.H. Joshi, K.K. Banka, and P.I. Rockett. Evolution of hyperheuristics for the biobjective 0/1 knapsack problem by multiobjective Genetic Programming. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 1227–1234, 2008.

[15] T. N. Lal, O. Chapelle, J. Weston, and A. Elisseeff. Embedded methods. In I. Guyon, S. Gunn, M. Nikravesh, and L.A. Zadeh, editors, *Feature Extraction: Foundations and Applications*, pages 137–165. Springer Verlag, 2006.

[16] P. Lichodzijewski and M. I. Heywood. Coevolutionary bid-based Genetic Programming for problem decomposition in classification. *Genetic Programming and Evolvable Machines*, 9(4):331–365, 2008.

[17] P. Lichodzijewski and M.I. Heywood. Managing team-based problem solving with Symbiotic Bid-based Genetic Programming. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 363–370, 2008.

[18] A.R. McIntyre and M.I. Heywood. Cooperative problem decomposition in Pareto competitive classifier models of coevolution. In *European Conference on Genetic Programming*, volume 4971 of *Lecture Notes in Computer Science*, pages 289–300, 2008.

[19] J. H. More and B. C. White. Genome-wide genetic analysis using genetic programming. In R. Riolo, T. Soule, and B. Worzel, editors, *Genetic Programming Theory and Practice IV*, pages 11–28. Springer Verlag, 2007.

[20] K. Nigam, J. Lafferty, and A. McCallum. Using Maximum Entropy for Text Classification. In *Workshop on Machine Learning for Information Filtering (IJCAI)*, pages 61–67, 1999.

[21] M. Potter and K. de Jong. Cooperative coevolution: An architecture for evolving coadapted subcomponents. *Evolutionary Computation*, 8(1):1–29, 2000.

[22] Ross J. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

[23] C. D. Rosin and R. K. Belew. New methods for competitive coevolution. *Evolutionary Compuatation*, 5:1–29, 1997.

[24] M.G. Smith and L. Bull. Genetic Programming with a Genetic Algorithm for Feature Construction and Selection. *Genetic Programming and Evolvable Machines*, 6(3):265–281, 2005.

[25] R. Thomason and T. Soule. Novel ways of improving cooperation and performance in Ensemble Classifiers. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 1708–1715, 2007.

[26] Y. Zhang and P.I. Rockett. *Feature extraction using multi-objective genetic programming*, chapter 4, pages 75–99. 2006. In ([12]).