Department of Computer Science
George Mason University Technical Reports

4400 University Drive MS#4A5
Fairfax, VA 22030-4444 USA
http://cs.gmu.edu/ 703-993-1530

Multiobjective Optimization of Co-Clustering Ensembles

Francesco Gullo gullo@yahoo-inc.com
AKM Khaled Ahsan Talukder atalukde@gmu.edu
Sean Luke sean@cs.gmu.edu
Carlotta Domeniconi carlotta@cs.gmu.edu
Andrea Tagarelli tagarelli@deis.unical.it

Technical Report GMU-CS-TR-2012-3

Abstract

Co-clustering is a machine learning task where the goal is to simultaneously develop clusters of the data and of their respective features. We address the use of *co-clustering ensembles* to establish a consensus co-clustering over the data. As is obvious from its name, co-clustering is naturally multiobjective. Previous work tackled the problem using both rudimentary multiobjective optimization and expectation maximization, then later a gradient ascent approach which outperformed both of them. In this paper we develop a new preference-based multiobjective optimization algorithm to compete with the gradient ascent approach. Unlike this gradient ascent algorithm, our approach once again tackles the co-clustering problem with multiple heuristics, but also applies the gradient ascent algorithm's joint heuristic as a preference selection procedure. As a result, we are able to significantly outperform the gradient ascent algorithm on feature clustering and on problems with smaller datasets.

1 Introduction

This paper describes a novel application of multiobjective optimization to the problem of producing optimal *co-clustering ensembles*. Co-clustering is an unsupervised machine learning technique for identifying groups of *objects* related by their similar *feature values*. Co-clustering is distinguished from ordinary clustering in that it simultaneously discovers clusters of similar objects with regard to the values, as well as clusters of similar features with regard to the objects related by them. Collectively, these cluster pairs do a better job of identifying the underlying structure in the data.

An example of co-clustering may be drawn from DNA microarray analysis. Consider a large 2-dimensional matrix whose rows are patients with various diseases and other conditions, and whose columns are various genes. Each cell in the matrix describes the degree to which a given gene is expressed in a given patient. The objective is to extract combinations of genes with regard to patients which suggest possible genetic sources of disease. Using standard clustering techniques we could attempt to find the largest clusters of patients which share gene combinations in common; and (though it is less common) we could likewise attempt to find clusters of genes with regard to patients which collectively have them. But co-clustering would seek to do both in combination: to find clusters of patients with respect to specific clusters of genes, where each cluster reinforces the other.

Ensemble learning is an approach to producing higher quality learned solutions by combining the results of several different kinds of learning algorithms. Ensemble learning is particularly helpful in overcoming learning bias among various co-clustering algorithms. In the above example, we might apply various algorithms to the DNA microarray to produce a set of co-clustering solutions, one per algorithm. These would be fed into an ensemble learning algorithm to produce a *consensus co-clustering* which ideally would be superior to any one member of the ensemble.

Many ensemble algorithms are essentially optimizers, hunting for sets of weights which produce a consensus which is as optimal as possible according to one or more heuristic criteria. The possibility of multiple heuristics makes ensemble learning easily amenable to multiobjective optimization. But in the co-clustering scenario, this possibility is made explicit, since there are *two ensemble clusterings* being performed, each according to a different set of heuristic criteria.

In prior work [16, 17, 18], a subset of the authors of this paper applied both a rudimentary multiobjective optimization, the EM (Expectation Maximization) algorithm, and custom gradient ascent approaches to search

for optimal consensus co-clusterings. The multiobjective optimization technique bred candidate consensus co-clusterings and assessed their fitness according to two heuristic objective functions: one which examined how well the co-clustering clustered the features, and another which examined how well it clustered the objects. The EM and gradient ascent algorithms instead used a single combined heuristic function. The early multiobjective algorithm suffered from two problems: first, later versions of the combined heuristic objective were more discriminating than either of the individual objectives. Second, the multiobjective optimization technique produced a Pareto Front of solutions: but at the end of the day a co-clustering algorithm must produce a *single* solution which is then tested for generalization accuracy. As a result, the gradient ascent algorithms generally proved superior.

In this paper we will revisit the multiobjective optimization approach with a more advanced technique based on a customized version of the NSGA-II algorithm. This version uses the two heuristics, originally used in the earlier multiobjective optimization approach, to build the Pareto Front (NSGA-II's "archive"), but then uses the advanced combined heuristic function to select parents from the front to breed new children. Additionally, each child is hill-climbed using the combined heuristic function. Finally, and crucially, we provide a single solution at the end of the run by returning the member of the final Pareto Front which maximized the combined heuristic function. This, plus new mutation and crossover algorithms, enables the procedure to achieve significantly better performance than the gradient ascent methods (the current state of the art) in a number of problems and criteria, though it still falls short in others.

2 Previous Work

Clustering Ensembles Clustering is the key step for many tasks in machine learning. It is well known that off-the-shelf clustering methods may discover different patterns in a given set of data. This is because each algorithm has its own bias due to the optimization of different criteria. Furthermore, there is no ground truth to validate the result.

Recently the use of *clustering ensembles* has emerged as a technique for overcoming problems with clustering algorithms. A clustering ensemble technique is characterized by two components: the mechanism to generate diverse partitions, and the consensus function to combine the input partitions into a final clustering. Diverse partitions are obtained from multiple applications of any single algorithm with different initializations [13, 25, 26], or on various bootstrap samples of the available data [12, 27], or from the application of different algorithms to the same data set [14].

One popular methodology to build a consensus function utilizes a co-association matrix [27, 36]. Alternatively, voting procedures have been considered to build consensus functions [11]. A different popular mechanism for constructing a consensus maps the problem onto a graph-based partitioning setting [34, 9, 32].

Co-Clustering Clustering is seriously hampered by the so-called *curse of dimensionality*. Various clustering algorithms can handle data with low dimensionality, but as the dimensionality of the data increases, these algorithms tend to break down. A common scenario with high-dimensional data is that several clusters may exist in different subspaces comprised of different combinations of features. To capture such local structure of the data, different *co-clustering* methods have been proposed.

Co-clustering methods fall into bottom-up or top-down approaches. Bottom-up methods find subspaces recognized as "interesting", and assign each data to the most similar subspace [33, 28]. Top-down approaches find the subspace to be associated to each cluster during the clustering stage. Top-down methods belong to histogram-based [29], density-based [31, 4, 38], hierarchical [37, 1], and EM-like [3, 2] approaches. These methods provide clustering solutions which are hard at the data level and have feature-to-cluster assignments equally weighted. Recent studies have focused on algorithms able to produce soft data clusterings (e.g., [28, 5]), and/or clusterings having feature-to-cluster assignments unequally weighted (e.g., [10, 5]).

Multiobjective Optimization and Clustering Viewed from a Bayesian perspective — as is now common — unsupervised machine learning is inherently an optimization task: one is trying to fit the best model to a sample of data. The definition of "best" is absolute: generalization performance with respect to the full universe of data points. But machine learning algorithms do not know this *a priori*, and instead must rely on heuristic assessments regarding the quality of their model and parameters, such as: goodness of fit with regard to the sample data, model parsimony, and so on.

As such there is various previous literature which has tackled clustering from an evolutionary computation perspective, and most commonly, from a multiobjective one. Kin et al [23] tackled the problem of optimizing the right model framework to fit to the data using a combination of multiobjective evolutionary optimization and local search (K-means, EM). Handl and Knowles [19] applied multiobjective optimization to examine the trade-off between number of clusters and cluster solution quality. Oliveira et al examined cluster ensemble discovery by using multiple objective optimization to optimize cluster features [30]. Jin and Sendhoff [21] applied multiobjective optimization in a wide variety of machine learning contexts, including unsupervised clustering. However no prior literature, to our knowledge, has ex-

amined co-clustering or co-clustering ensembles from an evolutionary or multiobjective optimization standpoint, other than the approach described earlier in [16].

3 Building a Co-Clustering Ensemble

We begin with a formal description of a co-clusterings and co-clustering ensembles.

Definition 1 (Co-Cluster) *Let $\mathcal{D}$ be a set of data objects, where each $\vec{o} \in \mathcal{D}$ is a vector in a feature space $\mathcal{F} = \{1, \ldots, |\mathcal{F}|\}$. A co-cluster C defined over $\mathcal{D}$ is a pair $\langle \Gamma_C, \Delta_C \rangle$:*

- *Γ_C denotes the object-based representation of C. It is a $|\mathcal{D}|$-dimensional real-valued vector whose components $\Gamma_{C,\vec{o}} \in [0,1]$, $\forall \vec{o} \in \mathcal{D}$, represent the object-to-cluster assignment of $\vec{o}$ to C, i.e., the probability $\Pr(C|\vec{o})$ that object $\vec{o}$ belongs to C;*
- *Δ_C denotes the feature-based representation of C. It is a $|\mathcal{F}|$-dimensional real-valued vector whose components $\Delta_{C,f} \in [0,1]$, $\forall f \in \mathcal{F}$, represent the feature-to-cluster assignment of the f-th feature to C, i.e., the probability $\Pr(f|C)$ that the f-th feature belongs to the subspace of features associated with C.*

Definition 2 (Co-Clustering Solution) *A co-clustering solution $\mathcal{C}$ defined over $\mathcal{D}$ is a set of co-clusters that satisfy the following conditions:*

$$\sum_{C \in \mathcal{C}} \Gamma_{C,\vec{o}} = 1, \ \forall \vec{o} \in \mathcal{D} \quad \textit{and} \quad \sum_{f \in \mathcal{F}} \Delta_{C,f} = 1, \ \forall C \in \mathcal{C}$$

Note that Definition 1 includes both *soft* and *hard* data clusterings, as well as features-to-clusters assignments which may have either equal or different weights. More precisely, $C = \langle \Gamma_C, \Delta_C \rangle$ is a hard (soft) co-clustering solution if $\Gamma_{C,\vec{o}} \in \{0,1\}$ ($\Gamma_{C,\vec{o}} \in [0,1]$), $\forall \vec{o} \in \mathcal{D}$. For the all-equal-weight case, if a feature f is relevant for C then $\Delta_{C,f} = 1/R$, where R is the total number of relevant features. Otherwise, $\Delta_{C,f} = 0$.

A *Co-clustering Ensemble* $\mathcal{E}$ is a collection of co-clustering solutions. Note that $\mathcal{E}$ does not carry any information on the ensemble generation strategy, nor on the original feature values of the objects in $\mathcal{D}$. Furthermore, each co-clustering solution in $\mathcal{E}$ may contain in general a different number of clusters.

Heuristic Multi-Objective Functions Though ultimately the consensus clustering will be evaluated based on its generalization performance (Θ_f, Θ_o, and Θ_{of}, as discussed later), this information is of course not available to it during the learning task. Thus our multi-objective optimizer will rely on multiple heuristics which assess its performance.

The heuristics are derived this way. A consensus co-clustering $\mathcal{C}^* = \langle \Gamma^*, \Delta^* \rangle$ derived from an ensemble $\mathcal{E}$ should meet two different requirements: $\mathcal{C}^*$ should capture the underlying clustering structure of the data, on the one hand through the data clusterings of the solutions in $\mathcal{E}$, and on the other through the assignments of features to clusters in the clusterings of $\mathcal{E}$. To capture both sides of the components in $\mathcal{E}$, the co-clustering ensemble problem can be naturally formulated as a two-objective optimization problem:

$$\mathcal{C}^* = \arg\min_{\mathcal{C}} \left\{ \Psi_o(\mathcal{C}, \mathcal{E}), \ \Psi_f(\mathcal{C}, \mathcal{E}) \right\} \tag{1}$$

where Ψ_o and Ψ_f are two optimization functions that account for the data clusterings and the feature-to-cluster assignments of the co-clusterings in $\mathcal{E}$, respectively. Note that the only constraint in the above formulation is that $\mathcal{C}$ be a well-defined co-clustering solution, as given in Definition 2.

The issue now is how to define Ψ_o and Ψ_f. We use a *clustering-based* approach, which involves a comparison with the co-clustering solutions of the ensemble. Formally, we have:

$$\Psi_o(\mathcal{C}, \mathcal{E}) = \sum_{\hat{\mathcal{C}} \in \mathcal{E}} \overline{\psi}_o(\mathcal{C}, \hat{\mathcal{C}}) \qquad \Psi_f(\mathcal{C}, \mathcal{E}) = \sum_{\hat{\mathcal{C}} \in \mathcal{E}} \overline{\psi}_f(\mathcal{C}, \hat{\mathcal{C}}) \tag{2}$$

where $\overline{\psi}_o$ ($\overline{\psi}_f$) is a function that measures the distance between two co-clustering solutions $\mathcal{C}'$ and $\mathcal{C}''$ in terms of their corresponding object-based clusterings (feature-to-cluster assignments):

$$\begin{aligned} \overline{\psi}_o(\mathcal{C}', \mathcal{C}'') &= \frac{1}{2}\Big(\psi_o(\mathcal{C}', \mathcal{C}'') + \psi_o(\mathcal{C}'', \mathcal{C}')\Big) \\ \overline{\psi}_f(\mathcal{C}', \mathcal{C}'') &= \frac{1}{2}\Big(\psi_f(\mathcal{C}', \mathcal{C}'') + \psi_f(\mathcal{C}'', \mathcal{C}')\Big) \end{aligned} \tag{3}$$

where

$$\begin{aligned} \psi_o(\mathcal{C}', \mathcal{C}'') &= \frac{1}{|\mathcal{C}'|} \sum_{C' \in \mathcal{C}'} \Big(1 - \max_{C'' \in \mathcal{C}''} J\big(\Gamma_{C'}, \Gamma_{C''}\big)\Big) \\ \psi_f(\mathcal{C}', \mathcal{C}'') &= \frac{1}{|\mathcal{C}'|} \sum_{C' \in \mathcal{C}'} \Big(1 - \max_{C'' \in \mathcal{C}''} J\big(\Delta_{C'}, \Delta_{C''}\big)\Big) \end{aligned} \tag{4}$$

$J(\vec{u}, \vec{v}) = (\vec{u}\ \vec{v}) / (\|\vec{u}\|^2 + \|\vec{v}\|^2 - \vec{u}\ \vec{v}) \in [0,1]$ denotes the extended Jaccard similarity coefficient (also known as Tanimoto coefficient) between any two real-valued vectors $\vec{u}$ and $\vec{v}$ [20]. The solution $\mathcal{C}^*$ of Equation 1 maximizes the information shared with the components of the ensemble, both in terms of object clusterings and feature-to-cluster assignments.

Heuristic Single-Objective Functions While promising, the two-objective evolutionary co-clustering ensemble formulation has some limitations. First, it is inefficient, mostly due to the fact that the number of iterations needed to achieve good solutions is typically large. Second, it is not obvious how to set the parameters (number

of iterations, population size, etc.). Third, the results are difficult to interpret. Rather than output a single solution, the algorithm produces a set of consensus co-clusterings along the Pareto front. Furthermore, and more importantly, the two-objective approach suffers from an important conceptual issue: it evaluates the object-based and feature-based representations of a candidate solution independently of one another. However they are instead strictly coupled. This could result in the selection of solutions which are not optimal.

In [18] an alternative single-objective function was proposed that captured this interdependence to more effectively evaluate the quality of a candidate consensus co-clustering. The object-based (Γ_C) and the feature-based (Δ_C) representations of a co-cluster C were combined to define the *subspace cluster representation matrix* X_C of C. X_C is a $|\mathcal{D}| \times |\mathcal{F}|$ matrix that stores, for each $\vec{o} \in \mathcal{D}$ and $f \in \mathcal{F}$, the probability of the intersection of the events *"object $\vec{o}$ belongs to C"* and *"feature f belongs to the subspace associated with C"*. Under the assumption of independence between the two events, this probability is equal to the product of $\Pr(C|\vec{o}) = \Gamma_{C,\vec{o}}$ and $\Pr(f|C) = \Delta_{C,f}$. Hence, given $\mathcal{D} = \{\vec{o}_1, \ldots, \vec{o}_{|\mathcal{D}|}\}$ and $\mathcal{F} = \{1, \ldots, |\mathcal{F}|\}$, matrix X_C can be formally defined as:

$$\mathrm{X}_C = \begin{pmatrix} \Gamma_{C,\vec{o}_1} \times \Delta_{C,1} & \ldots & \Gamma_{C,\vec{o}_1} \times \Delta_{C,|\mathcal{F}|} \\ \vdots & & \vdots \\ \Gamma_{C,\vec{o}_{|\mathcal{D}|}} \times \Delta_{C,1} & \ldots & \Gamma_{C,\vec{o}_{|\mathcal{D}|}} \times \Delta_{C,|\mathcal{F}|} \end{pmatrix} \tag{5}$$

The single-objective function that captures the dependence between the object-based and feature-based representation of a candidate solution is:

$$\mathcal{C}^* = \arg\min_{\mathcal{C}} \ \Psi_{of}(\mathcal{C}, \mathcal{E}) \tag{6}$$

where Ψ_{of} is a function designed to measure the "distance" of any well-defined co-clustering solution $\mathcal{C}$ from $\mathcal{E}$ in terms of both data clustering and feature-to-cluster assignments. Ψ_{of} is defined as follows:

$$\Psi_{of}(\mathcal{C}, \mathcal{E}) = \sum_{\hat{\mathcal{C}} \in \mathcal{E}} \overline{\psi}_{of}(\mathcal{C}, \hat{\mathcal{C}}) \tag{7}$$

where

$$\overline{\psi}_{of}(\mathcal{C}', \mathcal{C}'') = \frac{1}{2}\left(\psi_{of}(\mathcal{C}', \mathcal{C}'') + \psi_{of}(\mathcal{C}'', \mathcal{C}')\right) \tag{8}$$

and

$$\psi_{of}(\mathcal{C}', \mathcal{C}'') = \frac{1}{|\mathcal{C}'|} \sum_{C' \in \mathcal{C}'} \left(1 - \max_{C'' \in \mathcal{C}''} \hat{J}\left(\mathrm{X}_{C'}, \mathrm{X}_{C''}\right)\right) \tag{9}$$

In (9), the similarity between any pair C', C'' of co-clusters is computed in terms of their corresponding subspace cluster representation matrices $\mathrm{X}_{C'}$ and $\mathrm{X}_{C''}$. For this purpose, the Tanimoto similarity coefficient was generalized to operate on real-valued matrices:

$$\hat{J}(\mathrm{X}, \hat{\mathrm{X}}) = \frac{\sum_{i=1}^{|\mathrm{rows}(\mathrm{X})|} \langle \mathrm{X}_i \cdot \hat{\mathrm{X}}_i \rangle}{(\|\mathrm{X}\|_2)^2 \ + \ (\|\hat{\mathrm{X}}\|_2)^2 \ - \ \sum_{i=1}^{|\mathrm{rows}(\mathrm{X})|} \langle \mathrm{X}_i \cdot \hat{\mathrm{X}}_i \rangle} \tag{10}$$

where $\langle \mathrm{X}_i \cdot \hat{\mathrm{X}}_i \rangle$ denotes the dot product between the i-th rows of matrices X and $\hat{\mathrm{X}}$, and $\|X\|_2$ represents the so-called 2- or Euclidian-norm of X (similarly for $\hat{X}$).

Gradient Ascent Methods A heuristic approach (CB-PCE) was introduced in [18] to solve the optimization problem given in Equation (6). The heuristic computes meta-clusters (clusters of clusters) from the solutions of the ensemble, and applies a majority voting rule to assign objects and features to the meta-clusters. The resulting assignments provide the consensus co-clustering.

Unfortunately, this approach is extremely slow. An approximation of CB-PCE (FCB-PCE) that speeds up the computation of Equation (10) was introduced in [18].

4 A Preference-Based MOEA Approach to the Co-Clustering Ensemble Problem

Until recently, the concept of *Prefrence Based* Multiobjective Evolutionary Algorithm (MOEA) has gained a significant interest in the research community [22, 7, 35, 24, 15]. Here, the evolutionary algorithm is required to converge the solutions to a *particular region* (or point) of the Pareto front. In most cases, progress toward this region is made by accepting preference based information from the experimenter after every few generations of the evolutionary algorithm [7].

As discussed in [22, 35], this preference information is used to model a monotonically ascending or descending function, which then is used for the subsequent iterations of the MOEA to converge to a particular region of the objective space. To facilitate this procedure, the experimenter may provide *a priori* information, such as:

1. The exact region or point on the Pareto front where the preferred solution or solutions may exist.
2. A partial view of the Pareto front, if necessary.
3. The specific time when the experimenter will intervene and provide preference information to the algorithm.

It is usually necessary to know the above information before the start of the optimization procedure. In some cases, where such information is not readily available, the experimenter must supply at least a reference point in the objective space where the optimizer will try to converge to [8].

The problem of finding an optimal co-clustering ensemble is bounded by similar set of constraints: we are only equipped with a function (or a set of functions) that may help us to reach a preferred region on the Pareto front. In our case there are three such heuristic objective functions: $\Psi_o(\mathcal{C},\mathcal{E})$, the heuristic assessment of the data clustering; $\Psi_f(\mathcal{C},\mathcal{E})$, the heuristic assessment of the feature clustering; and $\Psi_{of}(\mathcal{C},\mathcal{E})$, the blended function of $\Psi_o(\mathcal{C},\mathcal{E})$ and $\Psi_f(\mathcal{C},\mathcal{E})$ that assesses both criteria. Like preference-based methods, in the end the population must be reduced to a single *final* solution. However unlike those methods, we do not know beforehand the exact point on the Pareto front to converge to, and no experimenter is available to provide preference information. As a result, if we optimized using all three objective functions, we would have a Front and no way to collapse it to a single result.

To decompose a typical co-clustering ensemble problem into a preference based multiobjective optimization problem, we have selected the third objective $\Psi_{of}(\mathcal{C},\mathcal{E})$ as the preference function (among other things, as discussed later). We use $\Psi_o(\mathcal{C},\mathcal{E})$ and $\Psi_f(\mathcal{C},\mathcal{E})$ to build the front. This approach seemed to be reasonable in the PCE context for the following reasons:

1. As $\Psi_{of}(\mathcal{C},\mathcal{E})$ is a combined criterion for assessing a consensus clustering solution, an individual with better $\Psi_o(\mathcal{C},\mathcal{E})$ and $\Psi_f(\mathcal{C},\mathcal{E})$ may help to find a better overall solution.
2. This approach will make the "reference points" or "preference points" in Preference-based MOEA unnecessary.
3. No intervention is required from the experimenter during the evolutionary run.

5 Optimization and Multiobjective Optimization

We tried two evolutionary algorithms for optimization: a traditional genetic algorithm (GA) with tournament selection, and the Nondominated Sorting Genetic Algorithm II (NSGA-II) [6]. As the NSGA-II algorithm universally dominated the GA in performance, we do not discuss the GA here.

In our situation we have three objective functions, defined more precisely earlier in the paper:

- $\Psi_o(\mathcal{C},\mathcal{E})$, a heuristic optimization function over object clusters, where lower values are preferred.
- $\Psi_f(\mathcal{C},\mathcal{E})$, a heuristic optimization function over feature clusters, where lower values are preferred.
- $\Psi_{of}(\mathcal{C},\mathcal{E})$, a heuristic optimization function jointly over both object and feature clusters, where lower values are preferred.

Algorithm 1 MOEA Based Co-Clustering Ensemble

Require: Randomly generated parent population P_t at generation t with population size M.
Ensure: After t_{max} number of iteration, individual I will represent solution of the problem.

```
Initialize child population, R_t := ∅
while t ≤ t_max do
    Create mixed population, S_t := P_t ∪ R_t
    F := Non-dominatedSort(S_t, Ψ_f, Ψ_o), create φ number of fronts. i.e. F := {F_φ, F_{φ−1} ... F_1}
    P_t := ∅ and i := 1
    repeat
        Assign crowding distance on F_i
        P_t := P_t ∪ F_i
        i := i + 1
    until |P_t| + |F_i| ≤ M
    Apply crowding distance sorting on F_i
    P_t := Choose the first (M − |P_t|) individuals from F_i
    R_t := ApplyCrossover-with-HillClimb(P_t, Ψ_of(C,E)))
    R_t := ApplyMutation-with-HillClimb(R_t, Ψ_of(C,E)))
    t := t + 1
end while
I := pick the best individual from R_t w.r.t. Ψ_of(C,E)
```

The NSGA-II algorithm used $\Psi_o(\mathcal{C},\mathcal{E})$ and $\Psi_f(\mathcal{C},\mathcal{E})$ as its two objective functions for purposes of building the archive: but used $\Psi_{of}(\mathcal{C},\mathcal{E})$ as the objective function for assembling the population from the archive via tournament selection.

Algorithm 1 gives a formal description of our approach, a variant of the NSGA-II algorithm. We used the single objective preference based $\Psi_{of}(\mathcal{C},\mathcal{E})$ function during NSGA-II's tournament selection (of size 2). Our algorithm deviated somewhat from NSGA-II as follows. When breeding individuals, we selected two parents from the archive, then crossed them over, producing a single child, which was added to the population. After the population was full, we then mutated every individual with a 0.25 probability.

Additionally, every 25 generations we added a bit of hill-climbing as follows. When breeding individuals, we again selected two parents from the archive, then crossed them over, producing a single child, but only the best of the child and its two parents were added to the population. After the population was full, we then mutated every individual with a 0.1 probability, but the child only replaced the parent if it was superior.

Representation and Breeding Our candidate consensus co-clusterings are soft feature clusterings: thus they contain weights suggesting the degree to which features belong to each cluster. For purposes of evolutionary optimization, the genotype takes the form of two 2-D arrays of real numbers. The first 2-D array holds the *object clustering*: with one row for each cluster and one column for each object. Each row holds a distribution of weights for the objects in that cluster. Likewise the

second 2D array holds the *feature clustering*: one row for each cluster and one column for each feature.

Crossover Crossover is performed on two parent but produces only one child (the other child is discarded). For each row, we decide, with an independent crossover probability P_{C1},[1] whether to cross over that row among the parents. If so, we perform parameterized uniform crossover among the elements of the row with a crossover probability P_{C2}.[2] This is followed by renormalizing both rows so that each sums to one.

Mutation For each row, we decide, with an independent mutation probability $P_M = 0.25$, whether to mutate that row. If we decide to do so, then we add random Gaussian noise, of standard deviation $\sigma_M = 0.1$, to each element in the row. If the chosen noise would move the element to less than 0.0 or greater than 1.0, it is rejected and a new noise value is selected. After convolving the entire row with noise, we renormalize the row so that its elements sum to one.

Collapsing the Pareto Front NSGA-II does not produce a single solution, but rather produces a Pareto front of solutions. However while multiple objectives are useful in solution discovery, at the end of the day the algorithm must produce a *single* co-clustering solution which is then tested for generalization. In earlier work [16] we ignored this issue and simply assessed the quality of the final solution as the average generalization score over each member of the front or of the population. However here we take another tack. Once NSGA-II has produced a Pareto front, we then return the solution whose $\Psi_{of}(\mathcal{C}, \mathcal{E})$ score is highest.

6 Experiments

We performed experiments comparing our revised MOEA methods with the Gradient Ascent methods described in [17, 18]. The experiments were done on six benchmark datasets from the UCI Machine Learning Repository[3] (Iris, Wine, Glass, and E. Coli) and on one time-series dataset from UCR Time Series Classification Page[4] (Trace Data). Because of very high time cost involved, we performed only ten independent runs for each combination of technique and dataset, and collected the results. As both the gradient-based and the multiobjective evolutionary approaches are stochastic in nature, we performed non-parametric two-tailed t-tests to determine statistical significance. *P*-values are shown in Figure 1.

[1] P_{C1} was set to the reciprocal of the total number of objects. That is, $P_{C1} = 1/|\mathcal{D}|$.

[2] P_{C2} was set to 0.5

[3] http://archive.ics.uci.edu/ml/

[4] http://www.cs.ucr.edu/∼eamonn/time_series_data/

Measurement Unusually for a machine learning method, a co-clustering has *two* equally important measures of accuracy: how well it properly clusters the object data, and how well it properly clusters the features.

The Θ_o, Θ_f, and Θ_{of} measures are formulated to assess the similarity of the consensus clustering C with a reference classification $\hat{C}$ drawn from existing labelled validation data. The measures are computed in terms of *Normalized Mutual Information* (NMI) [34] by taking into account their object-based representations (Θ_o) and feature-based representations (Θ_f), or both (Θ_{of}). NMI is a commonly used measure to assess the quality in clustering in general. The formal description of these measures may be found in [18]. Θ_{of} is the final, comprehensive assessment measure that takes into account both co-clustering aspects.

Results Table 1 summarizes the performance results. Our preference based MOEA model tends to attain better consensus co-clustering in terms of Θ_f and Θ_{of} on the simpler problems (Iris, Wine, Glass). However on E. Coli and the much more difficult TraceData, it could not outperform the gradient ascent algorithm.

This was the opposite of what we had expected. We had hypothesized that local optima would trap the gradient ascent algorithm on the more complex problems, but on the simpler problems a straightforward gradient ascent should outperform a global algorithm like the MOEA. This suggests that the underlying spaces may not be what we had expected.

We note that the overall Θ_{of} performance of the MOEA was largely based on its very strong Θ_f scores: indeed it was *never* able to outperform its gradient ascent counterpart in Θ_o. Nonetheless, the Θ_f scores were strong enough to easily outweigh the Θ_o differences.

One issue we faced was the fact that while the Ψ_o and Ψ_f functions were very basic, the Ψ_{of} function is more sophisticated and likely more powerful for optimization. In order to improve our results for the more complex data sets, and also to successfully win on all three fronts instead of just Ψ_f and Ψ_{of}, we will need to improve these functions to make them more accurate.

7 Conclusion

While in early experiments [16], a trivial MOEA outperformed an Expectation Maximization approach, in later implementations a gradient ascent formulation produced much better-performing models than either of them [17, 18]. However we believed, and still believe, that a more sophisticated MOEA can ultimately outperform a gradient ascent approach as it works around local optima.

To this end we have developed a Preference-based MOEA which develops the Pareto front using basic heuristics on the features and objects, but also applies,

Data Set	Algorithm	Θ_f	*P*-value	Θ_o	*P*-value	Θ_{of}	*P*-value
Iris	MOEA-CB-PCE	**0.5564**	($> 99.8\%$)	0.0813		**0.7348**	($> 95.0\%$)
	CB-PCE	0.2332		**0.4702**	($> 99.8\%$)	0.6902	
	MOEA-FCB-PCE	**0.5444**	($> 99.8\%$)	0.1826	$\approx$	**0.2027**	($> 95.0\%$)
	FCB-PCE	0.2002		0.2655	$\approx$	0.1805	
Wine	MOEA-CB-PCE	**0.5945**	($> 99.8\%$)	0.1943		**0.7748**	($> 99.8\%$)
	CB-PCE	0.1142		**0.4119**	($> 99.0\%$)	0.3402	
	MOEA-FCB-PCE	**0.6244**	($> 99.8\%$)	0.1001		0.2381	$\approx$
	FCB-PCE	0.1463		**0.2518**	($> 99.8\%$)	0.2355	$\approx$
Glass	MOEA-CB-PCE	**0.7464**	($> 99.8\%$)	0.0834		**0.9048**	($> 99.8\%$)
	CB-PCE	0.1302		**0.4702**	($> 99.8\%$)	0.1203	
	MOEA-FCB-PCE	**0.6244**	($> 95.0\%$)	0.1026		0.2397	$\approx$
	FCB-PCE	0.4639		**0.4525**	($> 99.8\%$)	0.1193	$\approx$
E. Coli	MOEA-CB-PCE	-0.003		-0.0013		0.011	
	CB-PCE	**0.0046**	($> 99.8\%$)	**0.0894**	($> 99.8\%$)	**0.0881**	($> 99.8\%$)
	MOEA-FCB-PCE	-0.003		-0.0009		0.019	
	FCB-PCE	**0.0043**	($> 99.8\%$)	**0.105**	($> 99.8\%$)	**0.112**	($> 99.8\%$)
TraceData	MOEA-CB-PCE	-0.00241		-0.0492		-0.0587	
	CB-PCE	**0.018**	($> 99.8\%$)	**0.2347**	($> 99.8\%$)	**0.2493**	($> 99.8\%$)
	MOEA-FCB-PCE	-0.00246		-0.04956		-0.0591	
	FCB-PCE	**0.0177**	($> 99.8\%$)	**0.1854**	($> 99.8\%$)	**0.247**	($> 99.8\%$)

Table 1: Results on the different benchmark datasets. Each gradient algorithm (CB-PCE, FCB-PCE) is compared against a MOEA employing the same objective function in its tournament selector, hill-climbing, and final Pareto front reduction. $\approx$ means no statistically significant difference (of at least 95%) between the two algorithms. Bold faced results are statistically significantly superior. We measured four levels of *P*-values: 95%, 98%, 99%, and 99.8%.

as the preference function, the same joint heuristic used in the gradient ascent formulations. The results from our experiments demonstrate the efficacy of the MOEA approach in terms of feature based clustering method (i.e. in terms of Θ_f) and for simpler problems. However more work and tuning remains in order to get the MOEA to the stage where it outperforms the gradient ascent algorithms universally.

8 Acknowledgements

This work was supported in part by NSF grant 0916870.

References

[1] E. Achtert, C. Böhm, H. P. Kriegel, P. Kröger, I. Müller-Gorman, and A. Zimek. Finding Hierarchies of Subspace Clusters. In *Proceedings of the European Conference on Principles and Practice of Knowledge Discovery in Databases*, pages 446–453, 2006.

[2] C. C. Aggarwal, C. M. Procopiuc, J. L. Wolf, P. S. Yu, and J. S. Park. Fast Algorithms for Projected Clustering. In *Proceedings of the ACM SIGMOD Conference on Management of Data*, pages 61–72, 1999.

[3] C. C. Aggarwal and P. S. Yu. Finding Generalized Projected Clusters in High Dimensional Spaces. In *Proceedings of the ACM SIGMOD Conference on Management of Data*, pages 70–81, 2000.

[4] C. Böhm, K. Kailing, H. P. Kriegel, and P. Kröger. Density Connected Clustering with Local Subspace Preferences. In *Proceedings of the IEEE International Conference on Data Mining*, pages 27–34, 2004.

[5] L. Chen, Q. Jiang, and S. Wang. A Probability Model for Projective Clustering on High Dimensional Data. In *Proceedings of the IEEE International Conference on Data Mining*, pages 755–760, 2008.

[6] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In *Parallel Problem Solving from Nature (PPSN VI)*, pages 849–858. Springer, 2000.

[7] K. Deb, A. Sinha, P. Korhonen, and J. Wallenius. An interactive evolutionary multiobjective optimiza-

tion method based on progressively approximated value functions. *Evolutionary Computation, IEEE Transactions on*, 14(5):723 –739, oct. 2010.

[8] K. Deb and J. Sundar. Reference point based multi-objective optimization using evolutionary algorithms. In *Proceedings of the 8th annual conference on Genetic and evolutionary computation*, GECCO '06, pages 635–642, New York, NY, USA, 2006. ACM.

[9] C. Domeniconi and M. Al-Razgan. Weighted cluster ensembles: Methods and analysis. *ACM Transactions on Knowledge Discovery from Data*, 2(4):17:1–17:40, 2009.

[10] C. Domeniconi, D. Gunopulos, S. Ma, B. Yan, M. Al-Razgan, and D. Papadopoulos. Locally Adaptive Metrics for Clustering High Dimensional Data. *Data Mining and Knowledge Discovery*, 14(1):63–97, 2007.

[11] S. Dudoit and J. Fridlyand. Bagging to improve the accuracy of a clustering procedure. *Bioinformatics*, 19(9):1090–1099, 2003.

[12] X. Fern and C. Brodley. Random Projection for High Dimensional Data Clustering: A Cluster Ensemble Approach. In *Proc. Int. Conf. on Machine Learning (ICML)*, pages 186–193, 2003.

[13] A. Fred and A. Jain. Data Clustering using Evidence Accumulation. In *Proc. Int. Conf. on Pattern Recognition (ICPR)*, pages 276–280, 2002.

[14] D. Greene, A. Tsymbal, N. Bolshakova, and P. Cunningham. Ensemble Clustering in Medical Diagnostics. In *Proc. IEEE Int. Symposium on Computer-Based Medical Systems (CBMS)*, pages 576–581, 2004.

[15] G. W. Greenwood, X. Hu, and J. G. D'Ambrosio. Fitness functions for multiple objective optimization problems: Combining preferences with pareto rankings. In R. K. Belew and M. D. Vose, editors, *FOGA*, pages 437–455. Morgan Kaufmann, 1996.

[16] F. Gullo, C. Domeniconi, and A. Tagarelli. Projective clustering ensembles. In *Proceedings of the IEEE International Conference on Data Mining*, 2009.

[17] F. Gullo, C. Domeniconi, and A. Tagarelli. Enhancing single-objective projective clustering ensembles. In *IEEE International Conference on Data Mining*, 2010.

[18] F. Gullo, C. Domeniconi, and A. Tagarelli. Advancing data custering via projective clustering ensembles. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, 2011.

[19] J. Handl and J. Knowles. Exploiting the tradeoff: The benefits of multiple objectives in data clustering. In *Proceedings of the Third International Conference on Evolutionary Multi-Criterion Optimization*, pages 547–560, 2005.

[20] A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data*. Prentice-Hall, 1988.

[21] Y. Jin and B. Sendhoff. Pareto-based multiobjective machine learning: An overview and case studies. *IEEE Transactions on Systems, Man, and Cybernetics Part C*, 38(3):397–415, 2008.

[22] J.-H. Kim, J.-H. Han, Y.-H. Kim, S.-H. Choi, and E.-S. Kim. Preference-based solution selection algorithm for evolutionary multiobjective optimization. *Evolutionary Computation, IEEE Transactions on*, PP(99):1 –15, 2011.

[23] Y. Kim, W. N. Street, and F. Menczer. Evolutionary model selection in unsupervised learning. *Intelligent Data Analaysis*, 6:531–556, 2002.

[24] P. Korhonen and J. Karaivanova. An algorithm for projecting a reference direction onto the nondominated set of given points. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 29(5):429 –435, sep 1999.

[25] L. Kuncheva and S. Hadjitodorov. Using Diversity in Cluster Ensembles. In *Proc. IEEE Int. Conf. on Systems, Man and Cybernetics (SMC)*, volume 2, pages 1214–1219, 2004.

[26] L. Kuncheva, S. Hadjitodorov, and L. P. Todorova. Experimental Comparison of Cluster Ensemble Methods. In *Proceedings of the International Conference on Information Fusion*, pages 1–7, 2006.

[27] B. Minaei-Bidgoli, A. Topchy, and W. Punch. A comparison of resampling methods for clustering ensembles. In *Proceedings of the International Conference on Machine Learning: Models, Technologies & Applications*, pages 939–945, 2004.

[28] G. Moise, J. Sander, and M. Ester. Robust projected clustering. *Knowledge and Information System*, 14(3):273–298, 2008.

[29] E. K. K. Ng, A. W.-C. Fu, and R. C.-W. Wong. Projective Clustering by Histograms. *IEEE Transactions on Knowledge and Data Engineering*, 17(3):369–383, 2005.

[30] L. S. Oliveira, M. Morita, and R. Sabourin. Feature selection for ensembles using the multi-objective optimization approach. In Y. Jin, editor, *Multi-Objective Machine Learning*, pages 49–74. Springer, 2006.

[31] C. M. Procopiuc, M. Jones, P. K. Agarwal, and T. M. Murali. A Monte Carlo algorithm for fast projective clustering. In *Proceedings of the ACM SIGMOD Conference on Management of Data*, pages 418–427, 2002.

[32] K. Punera and J. Ghosh. Soft cluster ensembles. In J. V. de Oliveira and W. Pedrycz, editors, *Advances in Fuzzy Clustering and its Applications*, pages 69–90. John Wiley & Sons, Ltd., 2007.

[33] K. Sequeira and M. Zaki. SCHISM: A New Approach for Interesting Subspace Mining. In *Proceedings of the IEEE International Conference on Data Mining*, pages 186–193, 2004.

[34] A. Strehl and J. Ghosh. Cluster Ensembles: A Knowledge Reuse Framework for Combining Multiple Partitions. *Journal of Machine Learning Research (JMLR)*, 3:583–617, 2002.

[35] L. Thiele, K. Miettinen, P. J. Korhonen, and J. Molina. A preference-based evolutionary algorithm for multi-objective optimization. *Evol. Comput.*, 17:411–436, September 2009.

[36] A. Topchy, A. Jain, and W. Punch. Combining Multiple Weak Clusterings. In *Proc. IEEE Int. Conf. on Data Mining (ICDM)*, pages 331–338, 2003.

[37] K. Y. Yip, D. W. Cheung, and M. K. Ng. HARP: A Practical Projected Clustering Algorithm. *IEEE Transactions on Knowledge and Data Engineering*, 16(11):1387–1397, 2004.

[38] M. L. Yiu and N. Mamoulis. Iterative Projected Clustering by Subspace Mining. *IEEE Transactions on Knowledge and Data Engineering*, 17(2):176–189, 2005.