

Bottom-up Evolutionary Subspace Clustering

Ali Vahdat and Malcolm Heywood and Nur Zincir-Heywood

Abstract—The ultimate goal of subspace clustering algorithms is to identify both the subset of attributes supporting a cluster and the location of the cluster in the subspace. In this work a generic evolutionary approach to *bottom-up* subspace clustering is proposed consisting of three steps. The first applies a non-evolutionary clustering algorithm attribute-wise to establish the lattice from which subspace clusters will be designed. In the second step a multi-objective Genetic Algorithm (MOGA) is used to evolve good candidate subspace clusters (CSC) through a combinatorial search w.r.t. the attribute-wise lattice from step 1. The third step then searches in the space of CSC from the population of the the first MOGA to find the best combination of subspace clusters, again under a MOGA formulation. Important properties of the approach are that a standard clustering algorithm is deployed in step one to build the initial lattice of attribute-wise clusters. This helps to decouple the computational expense of clustering using Evolutionary Computation, with the MOGA applied in steps 2 and 3 building clusters through a combinatorial search relative to the original lattice parameters. Benchmarking on data sets with tens to hundreds of attributes illustrates the feasibility of the approach.

I. INTRODUCTION

The general goal of any clustering algorithm is to extract the underlying structure of the data i.e., grouping data points such that an a priori cost function is optimized. There are many approaches to this task, but it is only relatively recently that the wider task of simultaneously identifying attributes as well as the cluster locations has been considered [1]. Factors driving this development include the increasing frequency of applications domains with large attribute counts e.g., document analysis, web pages, and multi-media. The large dimensionality gives rise to the observation that the density or support for clusters might decrease i.e., clusters lie within possibly unique subsets of the entire attribute space. There are therefore at least two parts to this problem. Firstly, entire subsets of attributes might be redundant from the data description/ clustering perspective. Secondly, as the dimensionality increases then the selectivity of distance functions tend to break down. Thus, from the perspective of a candidate cluster, redundant dimensions are sources of noise with points appearing to be equally distant irrespective of their cluster membership [1]. Finally, if subspace clusters can be located, then it is much easier to visualize individual subspaces than the union of (higher dimensional) subspaces.

There are two major categories of subspace clustering based on their search strategy. *Top-down* algorithms find an initial clustering in the full set of dimensions and evaluate the subspaces of each cluster, iteratively improving the results.

Bottom-up approaches find dense regions in low dimensional spaces and combine them to form clusters. To do so, bottom up methods frequently make use of data structures such as grids, windows, cells, or units and iteratively combine the content to identify higher dimensional regions of density [1]. The insight motivating this work is that, once the initial data structure has been identified, the search for appropriate higher dimensional subspace clusters is combinatorial, and Genetic Algorithms (GA) are very effective at combinatorial optimization problems. A short review of bottom-up approaches is made in Section II.

Conversely, top-down approaches start by considering clusters composed from all attributes. Attributes are incrementally removed and the quality of the ensuing cluster(s) measured until the performance reaches a plateau, as in wrapper style algorithms. Naturally, since clusters are initially composed from all the attributes, the selection of appropriate distance functions is very important and might represent a significant computational overhead.

The general approach pursued here for Evolutionary Subspace Clustering (ESC) assumes a bottom-up methodology in which a classical clustering algorithm is first applied to each attribute independently to establish the initial ‘lattice’ of candidate 1-dimensional clusters from which subspace clusters will then be composed. The lattice captures the regions of high density with respect to each attribute. The process of identifying the subspace clusters takes the form of a combinatorial search for clusters described in terms of the regions identified by the lattice. In order to achieve this, a two stage evolutionary process is assumed. Stage 1 applies a Multi-Objective GA (MOGA) to identify independent subspace clusters subject to the constraint that clusters in the Pareto front have minimal overlap in the exemplars assigned to each other. Stage 2 applies an independent MOGA to find the best combination of subspace clusters that will then form the ultimate solutions. The details of ESC are presented in Section III.

Experimental methodology and resulting performance evaluation are presented in Sections IV and V respectively. Particular attention is given to the construction of data sets with known distributions of subspace clusters – but without resorting to purely artificial data – while also explicitly modeling the impact of noise in the regions without identifiable subspaces. The resulting data sets consist of tens to hundreds of attributes. Performance evaluation demonstrates that ESC is able to identify the correct subspaces with detection rates in the order of 90 percent while utilizing as little as 2 to 35 unique attributes in total i.e., across all subspace clusters. Computational requirements on the larger subspace is in the order of 2 minutes per run (worst case) over all stages of the

Ali Vahdat, Malcolm Heywood, and Nur Zincir-Heywood are with the faculty of computer science, Dalhousie University, 6050 University Ave., Halifax, NS, Canada, B3H1W5. email: {vahdat, mheywood, zincir}@cs.dal.ca.

framework. A concluding section summarizes the ensuing findings (Section VII).

II. RELATED WORK

CLIQUE [2], ENCLUS [3], MAFIA [4], and CLTREE [5] are among the most popular bottom-up subspace clustering algorithms, whereas PROCLUS [6], ORCLUS [7], and COSA [8] are among the most well-known top-down approaches. However, none of these approaches use evolutionary algorithms to locate the subspace clusters. Conversely, the work by Kim *et al.* [9] represents a case in which a GA was employed. Specifically, a MOGA is used to define both an attribute subset (under a direct binary encoding) and suggest the number of clusters, k . The k -means clustering algorithm is then used to design the corresponding clusters for that particular individual. As a consequence, all the k clusters share the same attribute space, and cluster formation naturally assumes the standard greedy heuristic for cluster formation once the attribute subspace and cluster count is defined.

III. METHODOLOGY

The proposed approach to Evolutionary Subspace Clustering (ESC) assumes a bottom-up approach to subspace clustering and is based on three phases completed in a single pass (as opposed to iteratively): Attribute-wise Clustering, Evolving independent subspace clusters, and Cluster combination; and are detailed as follows.

A. Attribute-wise Clustering

Bottom-up methods require that a lattice/ grid/ cells from which the ensuing subspace clusters are composed be identified as the initial step. This represents an implicit set of tradeoffs, with the quality of the ensuing ‘discretization’ of the original attributes biasing the quality of the resulting clusters. Moreover, this process is also sensitive to computational overheads. The view is taken in this work that a traditional clustering algorithm, applied to each attribute independently (i.e., attribute-wise) is sufficient for this task, providing that the clustering algorithm is able to identify the ‘appropriate’ number of clusters as part of the attribute-wise clustering activity. With these requirements in mind the X -means generalization of the original k -means algorithm is assumed [10]. Following application of X -means, we have each attribute described in terms of a number of cluster centroids and a corresponding (nearest neighbor) allocation of exemplars to centroids, and therefore standard deviation. See for example Figure 1. It is the combination of attribute and centroid indexes which will then be used to build clusters and, by extension, sets of clusters in the following two phases of ESC.

At this point we are also in a position to test for degenerate attribute-wise clusters. Specifically, any clusters with zero standard deviation are considered cases of the ‘meaningless zeros’ problem. That is, under the subspace clustering problem, attributes with a zero attribute value are frequently used to denote the lack of data for a given attribute. Under

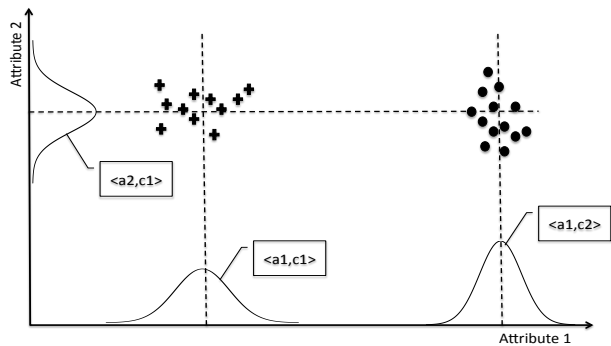


Fig. 1. Result of attribute-wise clustering. Resulting lattice described by 1-d attribute-cluster pairs $\langle a_i, c_j \rangle$.

sparse data sets a significant amount of the data might consist of such data. Moreover, given the frequency of such data, it is then highly likely that bottom up subspace clustering algorithms will devote a lot of resource to clustering such information. Unfortunately, this has little to do with describing the meaningful (non-zero) data content. This is therefore an example of where a priori meta information is able to guide the bottom-up process of subspace cluster formation.

B. Evolving independent subspace clusters – MOGA(a)

The goal of this step is to establish a population of ‘good’ candidate subspace clusters (CSC) in the space of possible subspace clusters. To do so, the search for subspace clusters is formulated as an evolutionary combinatorial optimization problem. Thus, as each cluster is required to meet generic properties of cluster quality, the use of a MOGA is justified.

CSC individuals design a subspace cluster using a pairwise gene in which each attribute and attribute-wise cluster index is specified as a pair of integers ($\langle \text{attribute}, \text{cluster} \rangle$ or $\langle a, c \rangle$). A variable length representation is assumed, thus the number of attributes supporting a subspace cluster is free to evolve. Specifically, each individual is of the form,

$$\text{Individual}(i): \quad SC_i = \{g_1, \dots, g_p\}, \\ g_j = \langle a_j, c_j \rangle \text{ and } 1 < p < P$$

and P is the number of attributes.

Subspace clusters are formed independently from each other, hence objectives are formulated to reward both diversity (very unlikely that a single cluster will capture all data set properties) and intra-cluster properties (exemplars which are explicitly associated with an individual should result in a ‘tight’ distribution), or

- 1) *Cluster Compactness (CC)*: With respect to the subset of exemplars that are common to all attributes identified by the CSC, we assume the following measure of ‘distortion’ or intra-cluster variation,

$$CC(SC) = \frac{1}{p} \sum_{i=1}^p dis(a, c; i) \quad (1)$$

where p provides a linear normalization relative to the number of attributes included; $dis(a, c; i)$ is the Euclidean distance between centroid c , attribute a and exemplar i under the constraint that the subset of exemplars considered are common to each of the attributes supporting the CSC. This information was collected during the nearest neighbor allocation step of attribute-wise cluster creation (Section III-A). More formally this process is defined by,

$$dis(a, c; i) = \sum_{x_i \in T} \begin{cases} d(x_i, \mu_{SC}); & \forall a_j \in SC | x_i \in a_j \\ 0; & \text{otherwise} \end{cases} \quad (2)$$

where μ_{SC} is the centroid of the CSC cluster as defined by the p centroid–attribute pairs; T is the total set of data points that are common to all attributes of CSC SC_i as identified during the attribute-wise nearest neighbor allocation process of Section III-A; and $d(\cdot, \cdot)$ is the Euclidean distance function. Thus, by minimizing this objective we give higher fitness to compact (spherical) subspace clusters.

- 2) *Exemplar Count (EC)*: The number of exemplars mapped to a CSC should be maximized. In order to estimate this through a bottom up process, the number of common exemplars shared across the union of each attribute contributing to a CSC is utilized.¹ Naturally, this process is analogous to that of estimating the Cluster Compactness, but this time the summation is over a binary distance function, or

$$EC(SC) = \sum_{x_i \in T} \begin{cases} 1; & \forall a_j \in SC | x_i \in a_j \\ 0; & \text{otherwise} \end{cases} \quad (3)$$

- 3) *Unique Exemplars (UE)*: One byproduct of Pareto multi-objective algorithms commonly utilized for MOGA is that at the end of each generation the ‘solution’ (typically) takes the form of a set of individuals – in this case subspace clusters – considered to be Pareto equivalent, instead of a single ‘best’ individual. We utilize the individuals comprising the *previous* Pareto front to prioritize CSCs from the *current* generation that index exemplars that are *not* present in the previous Pareto front.

$$UE_k = \frac{|\{x_i \in SC_k \cap x_i \notin SC_{j \neq k} \cap SC_j \in PF_{t-1}\}|}{|\{x_i \in SC_j \cap SC_j \in PF_{t-1}\}|} \quad (4)$$

where $|\cdot|$ denotes the cardinality of the set, SC_j denotes subspace cluster j , and $PF(t-1)$ denotes membership of the previous Pareto front. A subspace cluster that indexes many exemplars that are not indexed by other CSCs on the Pareto front receives a larger value; hence, this objective should be maximized.

¹Again making use of the information collected during the nearest neighbor allocation step of attribute-wise cluster creation (Section III-A).

The MOGA implementation could naturally make use of any recent development of research in this area (see for example [11] for a survey of the field). The only property we considered of particular relevance was a need for elitism. This work assumes the popular NSGA-II [12] model, however, there is nothing in NSGA-II which ties ESC to this particular MOGA.

The crossover operator takes the form of 1-point crossover with pruning/ repair. Specifically, once crossover is applied, there might be more than one attribute-wise cluster from the same attribute indexed by the individual; whereas we consider subspace clusters to be based on a unimodal support from each attribute, thus use pruning to delete any duplicates. The mutation operator replaces a gene with feasible attribute-wise cluster values selected with uniform p.d.f. In other words the mutation operator creates an attribute-wise cluster index which does not exist in the individual and introduces one of its cluster centroids.

C. Cluster combination – MOGA(b)

The goal of MOGA(a) was to build a population of ‘good’ candidate subspace clusters, thus the last step is to define a combination of CSCs that provide the best final clustering solution. This is a combinatorial search in the space of all CSC individuals from the final generation of MOGA(a) and is again conducted under a Pareto multi-objective context; hereafter denoted MOGA(b).

Each individual in MOGA(b) consists of q integers each indexing one individual from the final population of MOGA(a). Thus, each MOGA(b) individual denotes a set of CSCs, or

$$\text{Individual}(i): \quad CS_i = \{SC_1, \dots, SC_q\}, \\ q \in \{1, \dots, Q\} \text{ and } Q = |MOGA(a)|$$

where $|MOGA(a)|$ is the size limit of the MOGA(a) population. MOGA(b) also assumes a variable length representation where this is equivalent to letting the algorithm evolve the optimal number of subspace clusters in a clustering solution.

In the case of general clustering optimization criteria, we consider two categories [13], [1]: intra-cluster distance and connectedness. Clustering algorithms based on intra-cluster distance are designed to form compact spherical clusters by minimizing the ‘distance’ between data items of the same cluster or data items and cluster representatives, as in k -means. Conversely, a cluster connectedness objective forms clusters in which data items “close to each other” fall into the same cluster, as in single link agglomerative clustering. This category of clustering algorithms can find clusters of arbitrary shape, however, they might fail when clusters are close to each other.

Since clustering algorithms relying on a single optimization criterion may find one category of clusters but fail on the other, we assume a methodology based on objectives from both categories for MOGA(b) as in the work of Handl and Knowles [13]. In fact these two objectives are complementary, implying that the first objective, *compactness*,

tries to include spherical clusters whereas *connectivity* tries to include clusters in which neighbouring data items share the same cluster, regardless of cluster shape. Specifically:

- 1) *Solution intra-cluster distortion*: A new nearest neighbor allocation of exemplars to cluster centroids is performed against the set of candidate cluster centroids indexed by the MOGA(b) individual.² Each exemplar of the data set is allocated to one of the q subspace clusters of the clustering solution. The distance between an exemplar and its subspace cluster centroid is defined over the attributes specific to each subspace cluster. The sum of all distances is the overall distortion value of a MOGA(b) individual, or

$$Dis(CS) = \sum_{i=1}^q \sum_{j=1}^{|SC_i|} dis(x_j, \mu_{SC_i}) \quad (5)$$

where q is the number of CSCs in a MOGA(b) solution, $|SC_i|$ is the number of exemplars allocated to SC_i and μ_{SC_i} is the centroid of SC_i .

- 2) *Solution Connectivity*: While the first objective tries to build spherical subspace clusters to minimize the distortion value, the second objective tries to give more weight to subspace clusters that allocate neighboring exemplars to the same subspace clusters [13].

$$Con(CS) = \sum_{i=1}^N \sum_{j=1}^M \begin{cases} \frac{1}{j}; & \text{IF } \theta(x_i, NN_{ij}) \\ 0; & \text{otherwise} \end{cases} \quad (6)$$

where N is the exemplar count of the data set, M is the number of nearest neighbor exemplars to x_i considered for connectivity which we define as $\max(10, 0.01 \times N)$. NN_{ij} is the j th nearest neighbor exemplar to x_i , and the θ function is defined as:

$$\theta(x_i, NN_{ij}) = \#SC_k : x_i \in SC_k \cap NN_{ij} \in SC_k \quad (7)$$

A large value of connectivity indicates that there are a lot of neighboring exemplars placed in different clusters, whereas a small value of connectivity indicates that only a few neighboring exemplars are allocated to different subspace clusters [13]. Thus, this objective is to be minimized.

The crossover operator of MOGA(b) takes the form of 1-point crossover in which two new offspring are created by merging the first (second) half of the first (second) parent with the second (first) half of the second (first) parent. Duplicate CSC indexes are deleted from the resulting offspring. The mutation operator replaces one of the CSC indexes of an individual with a CSC index either from the Pareto front of MOGA(a) or the last population of the MOGA(a) depending on the parameter settings of the ESC algorithm.

²This nearest neighbor allocation is independent of the attribute-wise nearest neighbor allocation performed in Section III-A and reflects the true allocation of exemplars to clusters as the set of q CSC is now explicitly defined.

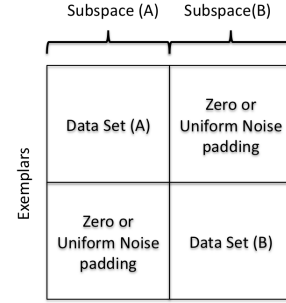


Fig. 2. Construction of subspace benchmarking data set.

IV. EXPERIMENTAL METHODOLOGY

A. Data Sets and ESC Parameterization

In order to provide a known ‘ground truth’ with respect to the structure of the underlying data, we take the following approach to explicitly building subspace clusters within the data sets used for benchmarking purposes. Real data sets from the UCI repository are selected, and attributes pre-processed to standardize the dynamic range for each attribute to the unit interval $[0, 1]$. Specifically, a measure of distortion appears as one of the objectives during MOGA(a) and MOGA(b), thus attribute standardization mitigates against one attribute artificially appearing more significant than another. Pairs of data sets are then combined with dissimilar attribute counts, resulting in a matrix style of combination in which each exemplar only consists of attributes from one subspace cluster (data set) and not the other, Figure 2. Regions corresponding to the subspaces naturally need padding with suitable values. Two scenarios are considered. Either zero values are employed or random noise³ under a uniform p.d.f.; hereafter referred to as the zero-padded and noise-padded data sets respectively. Moreover, in the case of the noise source model, the attributes from the subspace clusters with zero values are also replaced with data from the noise source. This naturally presents a more difficult problem than the case of zero padding as a distortion is introduced into the true subspaces as well as the ‘framing effect’ provided by the zero attribute information now being removed.

In this initial study UCI data sets are selected to represent subspace clusters with tens or hundreds of attributes and differing amounts of implicit ‘sparsity’ (Table I). Specifically, Iris has only 4 attributes whereas Ionosphere has 34, resulting in a benchmarking data set with 38 attributes; hereafter referred to as Iris-Ion. Moreover, approximately ten percent of the attribute values in Ionosphere are zero values, which under the above noise padding model for subspace construction, are also replaced with noise. The second subspace data set consists of the union of Musk and Ionosphere (Musk-Ion), resulting in a much larger attribute space of 200 attributes (Table I). The first data set is balanced, with only a subset of exemplars from the larger Ionosphere data set appearing;

³Distributed over the unit interval to match the range employed for the subspace clusters.

TABLE I

CONSTRUCTION OF BENCHMARKING DATA SETS. ‘ATTR’ AND ‘INST’ DENOTE ATTRIBUTE AND INSTANCE COUNTS RESPECTIVELY; ‘IRIS,’ ‘IONOSPHERE’ AND ‘MUSK’ ARE THE ORIGINAL SOURCE DATA SETS; ‘RESULT’ IS THE CORRESPONDING SUBSPACE DATA SET.

Data Set	Iris		Ionosphere		Musk		Result	
Parameter	Attr	Inst	Attr	Inst	Attr	Inst	Attr	Inst
Iris-Ion	4	150	34	150	–	–	38	300
Musk-Ion	–	–	34	351	166	476	200	827

whereas Musk-Ion has a mild exemplar imbalance in the representation from each subspace (57.6% Musk).

In total there are therefore 4 benchmarking data sets: Iris-Ion and Musk-Ion each combined into pairwise subspaces under either zero-padding or uniform noise. This then results in a total of 8 experiments as we consider two ESC scenarios. In the first case, MOGA(b) is limited to building solutions from the CSCs lying in the Pareto front of MOGA(a). In the second case MOGA(b) is allowed to index CSCs from the entire final population of MOGA(a). Each experiment comprises of 100 ESC runs using the common parameterization of 100 individuals per population, 100 generations, and 100 percent likelihood of applying crossover and mutation.

B. Performance Measures

Performance will be assessed from two generic perspectives: the ability to build clusters that separate the two subspaces, and solution complexity. Given that the ground truth for membership of the two subspace clusters comprising Iris-Ion and Musk-Ion is explicitly known we can measure the quality of subspace clusters in terms of the ability to associate subspace clusters with each of the original source data sets. The performance metric of balanced detection rate will be used for this purpose, where this resists the impact of class imbalance under Musk-Ion (degenerate solutions would have a best case performance of 50 percent). Thus, a solution has a balanced detection rate of $\frac{1}{|S|} \sum_{i \in S} \text{detection}(i, S)$ where $|S|$ is the number of subspaces (always 2) and $\text{detection}(i, S)$ is the detection rate of the individual with respect to subspace S exemplar i . The properties of the MOGA(b) objectives in the final population can also be measured, where this gives some qualitative information regarding to what degree satisfaction of the subspace clustering objectives correlates with good post training performance.

Complexity will be measured from the perspective of the number of subspace clusters and attribute count of the clusters. Again, each data set is known to be based on two explicitly embedded attribute subspaces, however, this does not preclude the process of subspace clustering discovering additional subspaces which may impact the total number of subspaces per solution, potentially gaining additional insight to the distribution of data. With regards to attribute counts frequency histograms will be used to summarize the number of attributes typically employed over ESC solutions.

TABLE II

SUMMARY STATISTICS FROM COMPLETE STUDY. ‘SC COUNT’ IS THE NUMBER OF SUBSPACE CLUSTERS PER SOLUTION; ‘DETECTION’ IS THE BALANCED DETECTION RATE; ‘ATTRIBUTE COUNT’ IS THE NUMBER OF ATTRIBUTES PER SOLUTION

Iris-Ion data set							
–	–	SC count		Detection		Attribute count	
CSC	padding	Med	Max	Med	Max	Min	Avg
PF	zeros	3	22	0.937	0.99	2	4.85
PF	noise	5	25	0.92	1.0	2	6.37
Full	zeros	2	25	0.903	0.997	2	4.76
Full	noise	3	25	0.873	1.0	2	6.47
Musk-Ion data set							
PF	zeros	3	11	0.946	0.99	3	3.88
PF	noise	6	25	0.903	1.0	2	8.15
Full	zeros	3	25	0.935	1.0	2	6.32
Full	noise	4	25	0.919	1.0	2	8.05

V. EMPIRICAL EVALUATION

ESC solutions are taken across the entire Pareto front of MOGA(b) on the last generation, naturally the designer would normally filter this information, however, to minimize any bias in the reporting, all Pareto non-dominated individuals are considered in compiling the following results. Table II summarizes the overall performance of ESC solutions across all 8 experiments in terms of the median subspace clusters per solution, median and maximum detection rates, and the average number of (unique) attributes per solution i.e., attribute support. In all cases ESC results in concise solutions, both from the perspective of subspace cluster count, and in terms of attribute support. Likewise cluster quality, as measured in terms of the balanced detection rate, is also very consistent. On cross referencing the MOGA(b) training objectives of distortion and connectivity with the post training metric, it was also apparent that solutions which focused on minimizing distortion or connectivity alone were not correlated with the better detection rates (see the following subsections). A result that supports including both measures of cluster quality during MOGA(b) evolution. The other general trend apparent is that increasing the set of candidate subspace clusters that MOGA(b) may index from MOGA(a) – that is candidate subspace clusters from the Pareto front versus the entire population of MOGA(a) – does not lead to significant improvements. In the following subsections we provide the associated context for these general trends.

A. Iris-Ion data set

The Iris-Ion benchmark uses equal exemplar counts per subspace cluster but only a very small attribute support for the Iris subspace.

1) *MOGA(b) limited to CSC individuals from MOGA(a) Pareto front:* The violin/ quartile box plots of Figures 3 and 4 summarize complexity and cluster quality of the resulting solutions under zero padding and noise padding scenarios respectively. Both methods demonstrate a high class-wise detection rate, whereas the principle difference between the two scenarios lies in the number of subspace

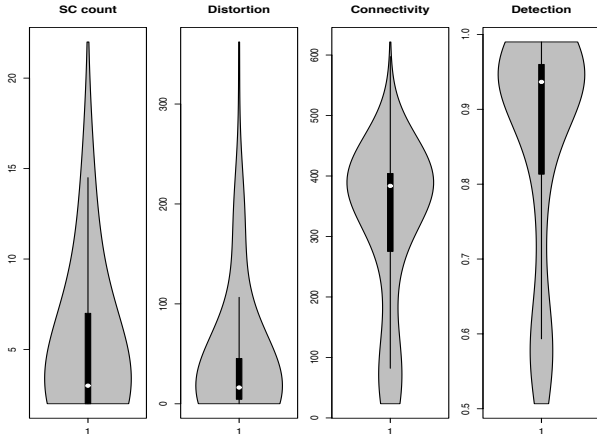


Fig. 3. Cluster quality on Iris-Ion data set. Limited MOGA(b) indexing under zero padding scenario.

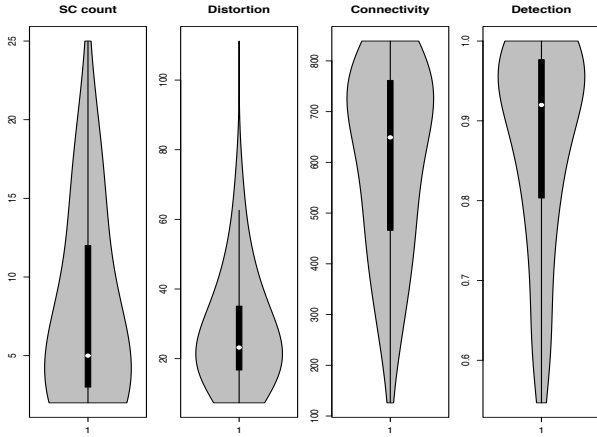


Fig. 4. Cluster quality on Iris-Ion data set. Limited MOGA(b) indexing and noise padding scenario.

clusters typically returned. When noise appears the number of subspace clusters increases from 3 to 5. Naturally, the cluster quality objectives used during training (distortion and connectivity) also record a relative increase in their values as noise based padding is introduced.

Figures 5 and 6 summarize the normalized frequency⁴ of attribute indexing across all 100 runs for the zero and noise padding scenarios respectively. There appears to be consistency with which subspace clustering weight the attributes, however, the use of noise decreases the relative significance of attributes 12 and 28 with respect to the no noise case. Thresholding the attribute frequencies at 1% leaves 8 (10) significant attributes under the zero (noise) padding cases respectively; where these are consistently identified in all runs. One other aspect of interest is that the addition of noise actually resulted in support for attributes corresponding to the Iris subspace disappearing (attributes 1 to 4 are not indexed). Thus, support for the Iris partition has been ‘washed out’ by the noise source – probably due to the dynamic range of the

⁴Normalized relative to the largest attribute count encountered.

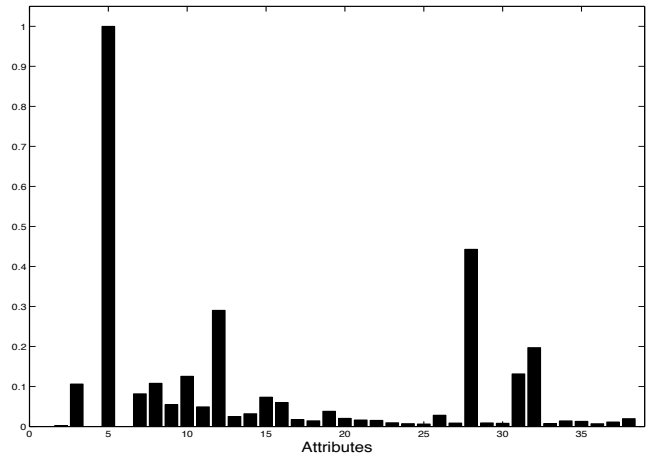


Fig. 5. Attribute support under Iris-Ion data set. Limited MOGA(b) indexing and zero padding scenario.

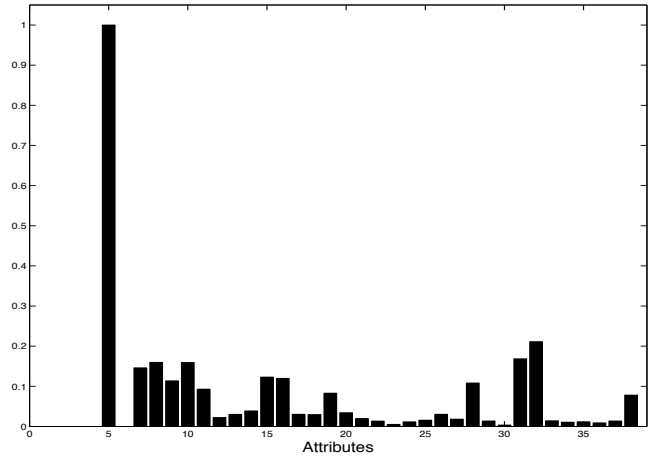


Fig. 6. Attribute support under Iris-Ion data set. Limited MOGA(b) indexing and noise padding scenario.

noise matching that of the Iris attributes, but representing nearly 90% of the available attributes.

2) *MOGA(b) indexing any CSC from MOGA(a)*: Increasing the number of candidate subspace clusters from MOGA(a) so that MOGA(b) might compose solutions beyond that of the MOGA(a) Pareto front may potentially result in better overall solutions at MOGA(b). Specifically, the nearest neighbor allocation of exemplars to clusters now takes into account other clusters contributing to the overall solution; where this is naturally not possible during MOGA(a). In the case of the violin/ boxplot for the zero padded scenario (not shown), the general picture established under Section V-A.1 was unchanged (also reflected in the general summary of Table II). Conversely, under the noise padding scenario a 3 percent reduction in balanced detection rate was evident. Moreover, the count of subspace clusters also increased. Distortion value remained very similar (w.r.t. that established in Section V-A.1) whereas connectivity actually underwent a net elongation towards lower values.

Naturally, increasing the set of possible CSCs from

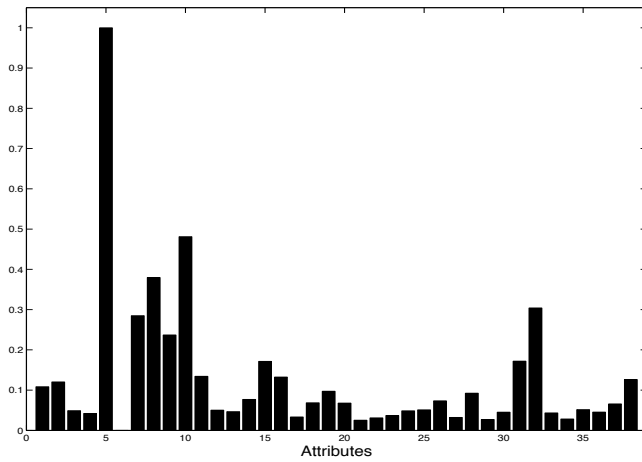


Fig. 7. Attribute support under Iris-Ion data set. Population wide MOGA(b) indexing and noise padding scenario.

MOGA(a) did result in more attribute diversity across all possible solutions. Figure 7 illustrates this under the case of the noise based padding scenario (compare to Figure 6). However, this does not result in an increase in the number of attributes typically appearing per solution (Table II), with no change to the overall distribution of attribute preferences.

B. Musk-Ion data set

The Musk-Ion benchmark is unbalanced and represents a much wider total attribute space than Iris-Ion.

1) *MOGA(b) limited to CSC individuals from MOGA(a) Pareto front*: As per the Iris-Ion data set, violin/ quartile box plots summarize complexity and cluster quality of the resulting solutions under zero padding and noise padding scenarios (Figures 8 and 9 respectively). Similar trends are apparent, with the noise padding scenario resulting in more subspace clusters appearing and a lower overall cluster purity (balanced detection rate). There is now also a pronounced tail/ second peak to the balanced detection rate; again more so in the case of the noisy data scenario. This is to be expected as we are considering all members of the MOGA(b) Pareto front as solutions, whereas in practice solutions that fail to balance distortion, and connectivity would be filtered i.e., preference for the knee of the Pareto front.

Figures 10 and 11 summarize the normalized frequency of attribute indexing under the Musk-Ion benchmark across all 100 runs for the zero and noise padding scenarios respectively. Both the no-noise and noise padding scenarios result in more attributes being ignored entirely than under the smaller Iris-Ion benchmark. However, there are also some significant changes to the attributes indexed, indicating that the introduction of noise results in previously useful attributes being replaced with selections that are more robust to the noise source. This is to be expected as the zero attributes resulted in the zero instances being replaced with noise source values, effectively rendering the attribute useless as a robust attribute candidate. That ESC is able to filter such cases is therefore considered in a positive light.

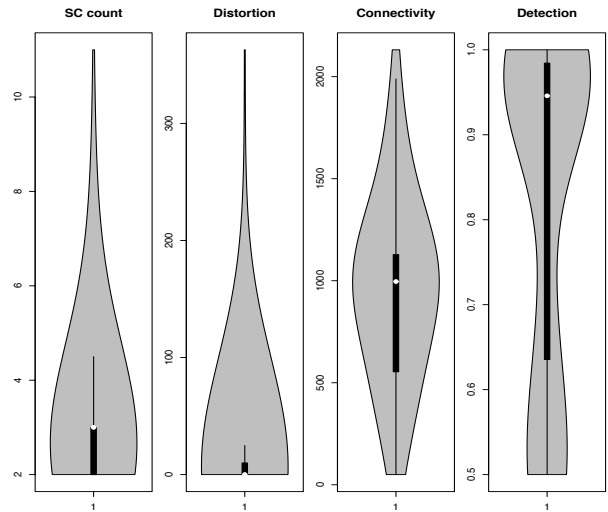


Fig. 8. Cluster quality on Musk-Ion data set. Limited MOGA(b) indexing under zero padding scenario.

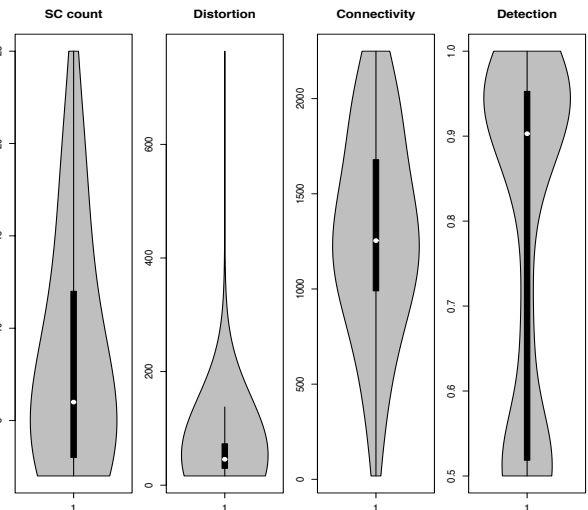


Fig. 9. Cluster quality on Musk-Ion data set. Limited MOGA(b) indexing and noise padding scenario.

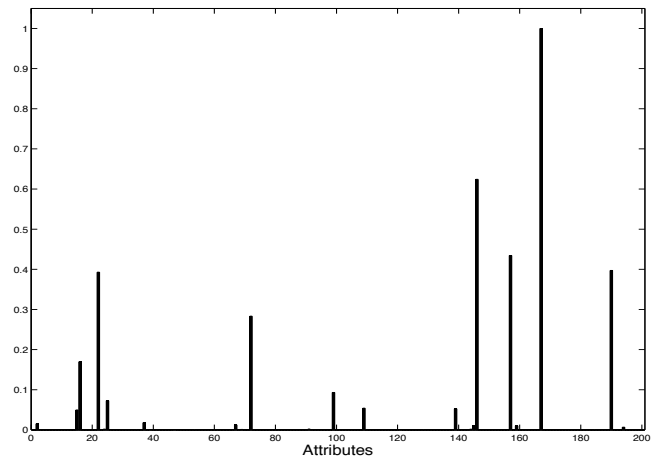


Fig. 10. Attribute support under Musk-Ion data set. Limited MOGA(b) indexing and zero padding scenario.

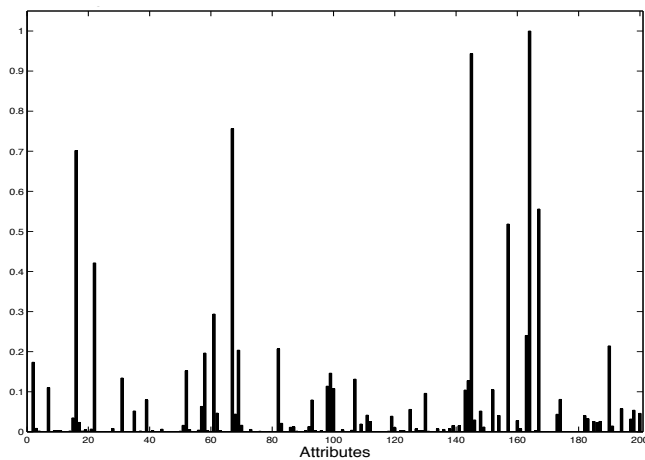


Fig. 11. Attribute support under Musk-Ion data set. Limited MOGA(b) indexing and noise padding scenario.

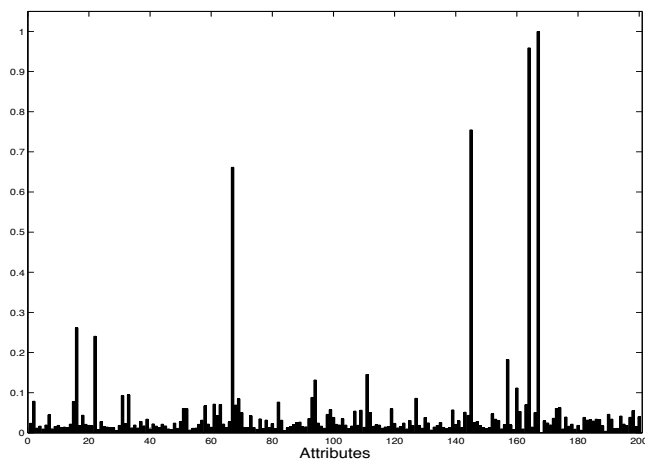


Fig. 12. Attribute support under Musk-Ion data set. Population wide MOGA(b) indexing and noise padding scenario.

2) *MOGA(b) indexing any CSC from MOGA(a)*: On introducing all MOGA(a) individuals as potential CSCs for MOGA(b) there is some evidence for a *reduction* to the total number of subspace clusters per solution (Table II). This is reflected in a corresponding shifting down in the violin/ box plot distribution (not shown), but otherwise there is no significant variation relative to the distributions returned under the results reported in Section V-B.1. Likewise, in the case of the frequency with which attributes are indexed (over all runs and entire content of the MOGA(b) Pareto front) a floor of infrequently indexed attributes occurs, but otherwise the distribution remains unchanged. Figure 12 illustrates this under the case of the noise based padding scenario (compare to Figure 11). As before, this does not result in an increase in the number of attributes typically appearing per solution (Table II) and the overall distribution of attribute preferences remains largely unchanged.

VI. CONCLUSION

Subspace clustering under arbitrary degrees of freedom represents a new avenue of research for Evolutionary Com-

putation. Previous work assumes a common subspace for all clusters and apply a classical (greedy) clustering algorithm in the fitness evaluation stage of evolution e.g., [9]. Instead this work follows the bottom up approach to subspace clustering and therefore avoids applying the classical clustering algorithm in the inner loop of fitness evaluation. The resulting ‘lattice’ of 1-d attribute clusters forms the basis for representing the subspace clustering problem as a combinatorial search. Two separate applications of a MOGA are then used to identify candidate subspace clusters and combinations of these subspace clusters respectively. Most of the computational cost is now associated with evaluating the connectivity objective of the second MOGA, although worst case computational requirements for both applying MOGA are just over 2 minutes on a Dell laptop computer.

A framework is developed for testing the resulting ESC algorithm, with particular attention paid to the representation of noise, a property that is rarely considered in previous studies. Solutions returned appear to be effective at separating the embedded clusters and consist of subspace clusters with very low attribute support.

VII. ACKNOWLEDGEMENT

This work is supported by ISSNet - NSERC Strategic Network and the CFI new opportunities program. All research was conducted at the Dalhousie Faculty of Computer Science NIMS Laboratory, <http://www.cs.dal.ca/projectx>.

REFERENCES

- [1] L. Parsons, E. Haque, and H. Liu, “Subspace clustering for high dimensional data: A review,” *ACM SIGKDD Explorations*, vol. 6, pp. 90–105, 2004.
- [2] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan, “Automatic subspace clustering of high dimensional data for data mining applications,” in *Proceedings of ACM Management of Data*, 1998, pp. 94–105.
- [3] C. H. Cheng, A. W. Fu, and Y. Zhang, “Entropy-based subspace clustering for mining numerical data,” in *Proceedings of ACM Knowledge Discovery and Data Mining*, 1999, pp. 84–93.
- [4] S. Goil, N. Nagesh, and A. Choudhary, “Mafia: Efficient and scalable subspace clustering for very large data sets,” North Eastern University, Tech. Rep. CPDC-TR-9906-010, 1999.
- [5] B. Liu, Y. Xia, and P. S. Yu, “Clustering through decision tree construction,” in *Proceedings of the International Conference on Information and Knowledge Management*, 2000, pp. 20–29.
- [6] C. C. Aggarwal, J. L. Wolf, P. S. Yu, C. Procopiuc, and J. S. Park, “Fast algorithms for projected clustering,” in *Proceedings of ACM Management of Data*, 1999, pp. 61–72.
- [7] C. C. Aggarwal and P. S. Yu, “Finding generalized projected clusters in high dimensional spaces,” in *Proceedings of ACM Management of Data*, 2000, pp. 70–81.
- [8] J. H. Friedman and J. J. Meulman, “Clustering objects on subsets of attributes,” <http://citeseer.nj.nec.com/friedman02clustering.html>, 2002.
- [9] Y. Kim, W. Street, and F. Menczer, “Feature selection for unsupervised learning via evolutionary search,” in *Proceedings of ACM Knowledge Discovery and Data Mining*, 2000, pp. 365–369.
- [10] D. Pelleg and A. Moore, “X-means: Extending k-means with efficient estimation of the number of clusters,” in *Proceedings of the International Conference on Machine Learning*, 2000, pp. 727–734.
- [11] K. Deb, *Multi-Objective Optimization using Evolutionary Algorithms*. John-Wiley and Sons, 2002.
- [12] K. Deb, A. Pratab, S. Agarwal, and T. Meyarivan, “A fast and elitist multiobjective genetic algorithm: NSGA-II,” *IEEE Transactions and Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.
- [13] J. Handl and J. Knowles, “An evolutionary approach to multiobjective clustering,” *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 1, pp. 56–76, 2007.