Improving Statistical Measures of Feature Subsets by Conventional and Evolutionary Approaches *

Helmut A. Mayer[2], Petr Somol[1], Reinhold Huber[3], and Pavel Pudil[1]

[1] Department of Pattern Recognition, UTIA Prague, 182 08 Prague, Czech Republic
[2] Department of Computer Science, University of Salzburg, A–5020 Salzburg, Austria
[3] Department of Mathematics, University of Salzburg, A–5020 Salzburg, Austria

Abstract. In this paper we compare recently developed and highly effective sequential feature selection algorithms with approaches based on evolutionary algorithms enabling parallel feature subset selection. We introduce the oscillating search method, employ permutation encoding offering some advantages over the more traditional bitmap encoding for the evolutionary search, and compare these algorithms to the often studied and well–performing sequential forward floating search. For the empirical analysis of these algorithms we utilize three well–known benchmark problems, and assess the quality of feature subsets by means of the statistical Bhattacharyya distance measure.

1 Introduction

The problem of selecting a "good" subset of features from the total of available features arises in a great variety of problem domains in science, engineering, and economy. As the number of possible subsets increases exponentially with the number of total features, a variety of deterministic and nondeterministic *Feature Selection* (FS) algorithms have been developed in order to escape the *Curse of Dimensionality*.

In this work we would like to compare some very efficient sequential FS algorithms with parallel FS techniques based on *Evolutionary Algorithms* (EAs). Basically, a sequential FS algorithm adds a feature to or leaves out a feature from the subset to be constructed in an iterative manner. Hence, the feature subset generation depends on initial and intermediate subsets. A parallel FS algorithm constructs a complete feature subset at once. The latter can be achieved by EAs which are generally believed to be well–suited for nonlinear, high–dimensional problems of exponential complexity (Schwefel, 1995; Mitchell, 1996).

The two basic categories of approaches to FS are the *Filter* and the *Wrapper* approach (John et al., 1994). The crucial point in discriminating these methods

* This work has been supported by *AKTION Österreich – Tschechische Republik* under grant AKTION 23p20: "Comparison of statistical and evolutionary approaches to feature selection in pattern recognition"

is the absence or presence, respectively, of a classifier to assess the quality of a feature subset. Either, a statistical measure independent of a classifier is employed (filter approach), or the error rate of a classifier determines the usefulness of a feature subset (wrapper approach). In this work we adopt the filter approach, as our main goal is to compare the performance of conventional to evolutionary FS algorithms. The additional use of specific classifiers (and training algorithms) would increase the already complex FS process interactions, and make it even more difficult to isolate the effectiveness of the FS algorithms to be compared.

For experimental comparisons we employ the Bhattacharyya (B–) distance measure (Fukunaga, 1990), and investigate algorithm performance on three dichotomous classification problems from the real world with the total number of features ranging from the single best feature to the (almost) full feature data set.

1.1 Formulation of the Feature Selection Problem

Following the statistical approach to pattern recognition, we assume that a pattern or object described by a real D-dimensional vector $\mathbf{x} = (x_1, x_2, \cdots, x_D)^{\mathrm{T}} \in \mathcal{X} \subset \mathcal{R}^D$ is to be classified into one of a finite set of C different classes $\Omega = \{\omega_1, \omega_2, \cdots, \omega_C\}$. The patterns are supposed to occur randomly according to some true class conditional probability density functions (pdfs) $p^\star(\mathbf{x}|\omega)$ and the respective *a priori* probabilities $P^\star(\omega)$. Since the class conditional pdfs and the *a priori* class probabilities are rarely known in practice, it is necessary to estimate these probability functions from the training sets of samples with known classification.

If the pdfs are a priori known to be unimodal, probabilistic distance measures, e.g., Mahalanobis or Bhattacharyya distance, may be appropriate to evaluate the quality of a feature subset. As pointed out by Siedlicki and Sklansky (1988) the error rate with respect to the chosen measurement criterion $J(\cdot)$ is even better (computational feasibility provided) (Siedlecki and Sklansky, 1988).

1.2 Bhattacharyya Distance for Feature Selection

In the following formulation B–distance measures the separability of normal distributions for two classes indexed by i and k (Fukunaga, 1990):

$$B_{ik} = \frac{1}{8}(\mu_i - \mu_k)^t \Sigma^{-1}(\mu_i - \mu_k) + \frac{1}{2}\ln\left(\frac{|\Sigma|}{|\Sigma_i|^{\frac{1}{2}}\,|\Sigma_k|^{\frac{1}{2}}}\right), \quad \Sigma = \frac{\Sigma_i + \Sigma_k}{2}, \quad (1)$$

where μ_i, μ_k are the feature mean vectors and Σ_i and Σ_k denote class covariance matrices for classes i and k, respectively. We point out, that a more general distance measure, such as the Chernoff distance, is in general closer to the error rate than B–distance, on the other hand such a measure is not easy to obtain (Kailath, 1967).

2 Feature Selection Algorithms

Assuming that a suitable criterion function $J(\cdot)$ has been chosen to evaluate the effectiveness of feature subsets, FS is reduced to a search for a (sub)optimal feature subset based on the selected measure.

Although an exhaustive search is a sufficient procedure to guarantee the optimality of a solution, in many realistic problems it is computationally prohibitive. Therefore, in practice one has to rely on computationally feasible procedures to avoid exhaustive search for the price of suboptimal results. A comprehensive list of suboptimal procedures and the corresponding formulas can be found in (Devijver and Kittler, 1982). An excellent taxonomy of currently available FS methods in pattern recognition is presented in (Jain and Zongker, 1997).

The strategies used to find a useful subset of features range from the simple but popular sequential forward (SFS) and sequential backward selection (SBS), to more sophisticated but computationally more expensive algorithms. In the following section we will describe three of the latter which will then be compared on three benchmark problems.

2.1 Sequential Forward Floating Search

The sequential forward floating search (SFFS) Pudil et al., 1994 can be viewed as SFS with backtracking. SFS is a simple greedy algorithm starting with an empty feature subset, then iteratively adding the feature which maximizes the evaluation criterion of the feature subset up to a specified target size t. Obviously, this procedure does not account for nonlinear interactions of features, hence SFFS offers the following improvements:

After adding a feature x_i to a subset of size k with $J(k)$ in SFS manner (inclusion), SFFS tries to find a feature x_j with $j \neq i$ which can be excluded so that the new $J'(k) > J(k)$ (conditional exclusion). This step is continued as long as a feature can be excluded under the above condition which demands that the best values of $J(\cdot)$ are recorded for each subset size (continuation of conditional exclusion). If an exclusion step is not successful, inclusion continues until the algorithm has “floated” through all possible subset sizes or has reached a certain target size t.

While SFFS starts with an empty feature subset dominantly searching in forward direction, the sequential backward floating search (SBFS) “floats” in backward direction and analogously can be viewed as SBS with backtracking (Pudil et al., 1994).

2.2 Oscillating Search

A very recent development is the oscillating search (OS) algorithm (Somol and Pudil, 2000). Most of the known suboptimal FS methods are based on step–wise adding of features to an initially empty feature set, or on step–wise removing features from an initial set of all features. A single search direction – forward or backward – is usually preferred. It is apparent that all these algorithms spend a lot of time testing feature subsets having cardinalities far distant from the required cardinality d.

Unlike other methods, OS is based on repeated modification of the current subset X_d of d features. This is achieved by alternating the *down–* and *up–swings*. The down–swing removes o “worst” features from the current set X_d to obtain a new set X_{d-o} at first, then adds o “best” ones to X_{d-o} to obtain a new current set X_d. The *up-swing* adds o "good" features to the current set X_d to obtain a new set X_{d+o} at first, then removes o "bad" ones from X_{d+o} to obtain a new current set X_d again. The intitial subset X_d is generated randomly, and an up– and down–swing is achieved by SFS and SBS, respectively.

Let us denote two successive opposite swings as an *oscillation cycle*. Then, the oscillating search consists of repeating oscillation cycles. The parameter o is termed *oscillation cycle depth* and should initially be set to 1. If the last oscillation cycle did not find a better subset X_d of d features, the algorithm increases the oscillation cycle depth by setting $o = o{+}1$. Whenever any swing finds a better subset X_d of d features, the depth value o is restored to 1. The algorithm terminates, when the value of o exceeds the user–specified limit Δ. Besides the basic OS algorithm described here, variants can be found in (Somol and Pudil, 2000).

2.3 Evolutionary Algorithms for Feature Selection

When employing EAs for FS, complete subsets are generated in parallel potentially eliminating the problems inherent to sequential FS methods. Basically, a start population of EA chromosomes (feature subsets) is generated randomly, and each individual receives a fitness according to the evaluation criterion $J(\cdot)$. The fitness determines the probability of an individual to be selected for mating, where (usually) two individuals exchange their genetic information. The offspring then undergo a mutation operation and form the next generation. This evolutionary cycle is repeated for a user–defined number of generations, and the best individual (feature subset) of all generations represents the final solution.

In the basic approach to feature selection using EAs a feature subset is encoded as a binary vector (*Bitmap Encoding*) $\boldsymbol{a} = (a_1, \ldots, a_d)$, where $a_i = 1$ indicates the presence of the $i-$th feature in the subset, while the absence of the $i-$th feature is expressed by $a_i = 0$. The bitmap encoding is well suited for an evolutionary feature selection technique based on a wrapper approach (Siedlecki and Sklansky, 1989).

In (Punch et al., 1993) the bitmap encoding was generalized to a weighted encoding, where features were assigned different weights resulting in a modification (warping) of the feature space for a k–NN classifier. In (Yang and Honavar, 1997) bitmap encoding has been employed for the evolutionary search of feature subsets for an ANN classifier. A mixture of filter and wrapper approach is presented in (Chaikla and Qi, 1999), where an EA with a fitness function comprising k–NN classifier accuracy and multiple correlation coefficients is employed for FS. Although all these works could report on improvements in accuracy or subset size, none of them compared evolutionary to conventional FS methods.

With the filter approach and the inherent monotony of the B–distance measure we are using in this work, bitmap encoding together with a fitness function only assessing B–distance would simply result in convergence to the full feature set. A penalty or cost term could be introduced in the fitness function $f(\boldsymbol{a}) = J(\boldsymbol{a}) + p(\boldsymbol{a})$, where the function $p(\boldsymbol{a})$ depends on the number of features present in $\boldsymbol{a}$. The penalty term $p(\boldsymbol{a})$ can then be used to favor feature subsets of a given cardinality.

Intuitively, for the experimental framework in this work the use of a penalty term unnecessarily complicates the evolutionary search, as the EA has to find a subset size which is known from the very beginning. Moreover, as soon as the EA arrives at the correct size of the feature subset it will likely stay with this solution, because only very specific interactions of genetic operators will allow the transition to a different solution of the given target size. Experimental results confirmed these assumptions, hence we introduce a permutation encoding method.

3 Permutation Encoding

As outlined in Section 2.3, instead of the more traditional bitmap encoding for the generation of subsets, we experimented with variants of permutation encoding which is primarily used for order problems such as the *Traveling Salesperson Problem* (TSP). The bases [1] of the EA chromosome are integers building a permutation as shown in Figure 1.

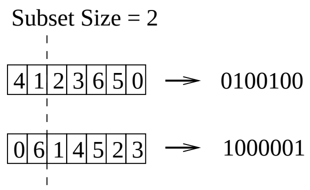

Fig. 1. Permutation encoding for the generation of subsets.

For the generation of a subset of a specific target size t the chromsome is scanned from left to right, and the first t bases representing a feature index are used to construct the subset. The permutation encoding ensures that each feature taken into account is different, and always yields a subset of the given target size t.

In order to preserve the permutation property of the chromosomes, specific genetic operators have been devised. The mutation operator simply exchanges

[1] Many researchers use the term *gene*, but a wild type gene is a much more complex structure, and the term *base* is equally a synonym for an atomic information unit in biology.

two random bases on the chromosome with a given mutation rate p_m (usually in the range of $0.001 - 0.01$). One of the most prominent crossover operators for permutation encoding is the *Partially Matched Crossover* (PMX) proposed in (Goldberg and Lingle, 1985). Its basic mechanisms are presented in Figure 2.

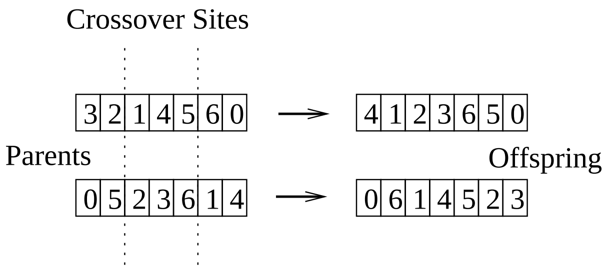

Fig. 2. Partially Matched Crossover (PMX).

Generally, for crossover two parent chromosomes are selected, and crossover is performed according to a user–defined crossover rate p_c (usually in the range of $0.6 - 1.0$). If no crossover occurs, the two parents are simply copied to two offspring chromosomes. In the crossover phase two crossover sites are selected randomly (sites are the same for both parents). Then, the bases in between the crossover sites are exchanged. Up to this point we have exactly described the very common *2–point Crossover*, but if the bases were only exchanged, the permutation property would be lost. Thus, each base to be copied to the other parent is searched in that parent and swapped with the base currently at the locus, where the exchange takes place (just like a single mutation). In doing so the partial order between the crossover sites can be exchanged between the parents without corrupting the permutation property.

As can be seen in Figure 1 parts of the chromosome are not expressed. If both crossover sites fall into this region, it might appear that crossover does not change the expressed feature subset. But assuming an exemplary target subset size $t = 2$ in Figure 2, it can be observed that with PMX the features in the subset can be altered even under this condition. Though, a mutation in an unexpressed region of the chromsome does not effect the encoded subset, it can indirectly take an influence by means of a subsequent PMX crossover.

4 Benchmark Problems

A brief description of the data sets used for FS experiments is given below. The main criterion for selecting these specific benchmarks is the rather high number of features challenging the search capabilities of the FS algorithms to be compared.

Breast This is the diagnostic Wisconsin Diagnostic Breast Center database containing 284 examples. From 30 features a prediction into the classes malignant and benign is aspired (from the UCI Machine Learning Repository) (Blake and Merz, 1998).

Sonar This data set is used to classify sonar signals into signals bounced off a metal cylinder and those bounced off a roughly cylindrical rock (Gorman and Sejnowski, 1988). It has 105 examples with 60 continuous inputs (from the CMU neural networks benchmark collection).

Mammo This is a mammogram data set from the Pattern Recognition and Image Modeling Laboratory at the University of California, Irvine, containing 86 examples. From 65 features a prediction into the classes malignant and benign is aspired (from the UCI Machine Learning Repository) (Blake and Merz, 1998).

4.1 Experimental Setup

Experiments have been run for the three benchmark data sets using the three FS algorithms to be compared. For each problem all subset sizes from $t = 1$ to $t = n - 1$ (with n being the total number of features) have been investigated. The following parameters have been used with all the experiments in this paper:
SFFS Parameters: Initial subset size = 2 (generated by SFS), Runs = 1 (deterministic behavior).
OS Parameters: $\Delta = 50\%$ (of the number of features), Runs = 20.
EA Parameters: Population Size = 50, Generations = 100, Crossover Probability $p_c = 0.6$, Mutation Probability $p_m = 0.01$, Crossover = PMX, Selection Method = Binary Tournament, Runs = 20 (the EA parameters are fairly standard and are not based on extensive experiments).

5 Experimental Results

For the *Breast* data set containing 30 features, we were able to compute the optimal subset of each cardinality by means of a yet unpublished *Branch–and–Bound* method. Thus, Figure 3 shows the differences (error) of the compared algorithms to the optimal subsets.

It can be observed that OS yields the smallest mean errors, followed by SFFS, and EA. When looking at the best results of OS and EA, the optimal result was always found within the 20 runs spent. As SFFS is deterministic, the best results are identical to the mean results shown in Figure 3 (left). The relatively large mean error for EA and a $t = 2$ might be a side effect of permutation encoding, as most mutations fall in an unexpressed region of the chromosome, and the population quickly converges to a local optimum. However, with bitmap encoding it is even difficult to find any solution with subset size $t = 2$!

In terms of computational cost (Figure 3 (right)) OS is the most expensive, but also delivers the best results. However, increase of the number of generations of the EA would further improve its solutions.

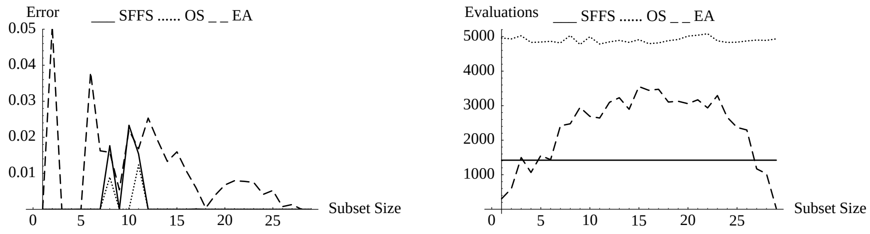

Fig. 3. *Breast* – Mean errors (left) and mean number of evaluations to find the best B–Distance (right) of Sequential Floating Forward Search (SFFS), Oscillating Search (OS), and Evolutionary Algorithm (EA) (averaged on 20 runs).

The best results of the compared FS algorithms and the corresponding computational cost for the *Sonar* data set is depicted in Figure 4.

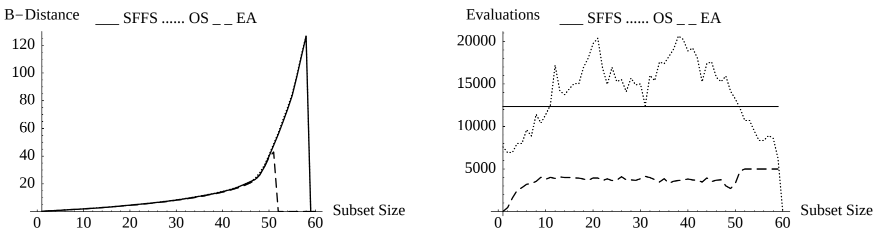

Fig. 4. *Sonar* – Best B–distances (left) and mean number of evaluations to find the best B–Distance (right) of Sequential Floating Forward Search (SFFS), Oscillating Search (OS), and Evolutionary Algorithm (EA) (averaged on 20 runs).

Although, the best results found seem to be very similar in Figure 4 (left), a closer look at the numbers reveals the same order of performance as for the other data sets. The sudden decrease of the B–distance [2] is a strong indicator that some of the features in the data set are linearly dependent. Obviously, EA exhibits these problems earlier which remains to be studied, but a possible explanation is the more "intelligent" inclusion and exclusion of features with OS, whereas EA performs a blind search.

Very similar things can be said about the results for the *Mammo* data set shown in Figure 5.

Again, OS and EA deliver the best results, but starting with a subset size around $t = 30$ EA marginally drops behind SFFS. A main reason for that behavior might be the increased complexity of that problem (65 features), while keeping the number of generations of the EA fixed at 100, which even compared to SFFS results in a much lower number of subset evaluations. Accordingly, the

[2] The numerical value of 0.0 is arbitrary and is used by our software to indicate impossible matrix operations.

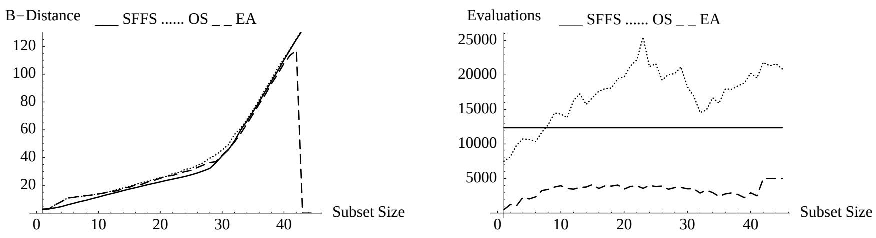

Fig. 5. *Mammo* – Best B–distances (left) and mean number of evaluations to find the best B–Distance (right) of Sequential Floating Forward Search (SFFS), Oscillating Search (OS), and Evolutionary Algorithm (EA) (averaged on 20 runs).

computational cost (Figure 5 (right)) clearly confirms the well established fact that EAs find good solutions in a short time (but are less effective in honing these solutions).

6 Outlook

The results presented in this paper are quite encouraging considering that SFFS has been evaluated as one of the best available FS algorithms in (Jain and Zongker, 1997). Not only OS and EA generate better results than SFFS, but EA also takes comparable, if not smaller computation time. Clearly, these results have to be confirmed with a number of additional data sets and the algorithms should also be investigated in the environment of a wrapper approach employing a number of different classifiers. A very promising future research direction could be the hybridization of EA and OS combining the speed of an EA to find a good solution with the ability of the OS to improve an existing feature subset.

7 Acknowledgements

We would like to thank the executives of the Aktion board for their very friendly and motivating support of this inter–European research initiative. Many thanks to our project colleagues Jiri Grim and Roland Schwaiger for their ideas having already initiated new lines of research within our cooperation. We also thank Markus Amersdorfer, Martin Angerer, Sandor Herramhof, and Harald Schweiger for development of a Java prototype for evolutionary feature selection in their software engineering course.

References

Blake, C. and Merz, C. (1998). http://www.ics.uci.edu/~mlearn/mlrepository.html. WWW Repository, University of California, Irvine, Dept. of Information and Computer Sciences.

Chaikla, N. and Qi, Y. (1999). Genetic Algorithms in Feature Selection. In *IEEE International Conference on Systems, Man, and Cybernetics*, pages V – 538–540. IEEE.

Devijver, P. and Kittler, J. (1982). *Pattern Recognition: A Statistical Approach.* Prentice.

Fukunaga, K. (1990). *Introduction to Statistical Pattern Recognition.* Academic Press.

Goldberg, D. E. and Lingle, R. (1985). Alleles, Loci, and the Traveling Salesman Problem. In Grefenstette, J. J., editor, *Proceedings of the First International Conference on Genetic Algorithms and their Applications*, pages 154–159. Texas Instruments, Inc. and Naval Research Laboratory, Lawrence Erlbaum Associates.

Jain, A. and Zongker, D. (1997). Feature Selection: Evaluation, Application and Small Sample Performance. *IEEE Transactions on PAMI*, 19(2):153–158.

John, G., Kohavi, R., and Pfleger, K. (1994). Irrelevant Features and the Subset Selection Problem. In *Proceedings of the Eleventh International Conference on Machine Learning*, pages 121–129, San Mateo, CA. Morgan Kaufmann.

Kailath, T. (1967). The divergence and bhattacharyya distance measures in signal selection. *IEEE Transactions on Communications Technology*, 15(1):52–60.

Mitchell, M. (1996). *An Introduction to Genetic Algorithms.* Complex Adaptive Systems. MIT Press, Cambridge, MA.

Pudil, P., Novovičová, J., and Kittler, J. (1994). Floating search methods in feature selection. *Pattern Recognition Letters*, 15:1119–1125.

Punch, W. F., Goodman, E. D., Pei, M., Chia-Shun, L., Hovland, P., and Enbody, R. (1993). Further Research on Feature Selection and Classification Using Genetic Algorithms. In Forrest, S., editor, *Fifth International Conference on Genetic Algorithms*, pages 557–564, San Mateo, CA. Morgan Kaufmann.

Schwefel, H.-P. (1995). *Evolution and Optimum Seeking.* Sixth–Generation Computer Technology Series. Wiley, New York.

Siedlecki, W. and Sklansky, J. (1988). On automatic feature selection. *International Journal of Pattern Recognition and Artificial Intelligence*, 2(2):197–220.

Siedlecki, W. and Sklansky, J. (1989). A Note on Genetic Algorithms for Large–Scale Feature Selection. *Pattern Recognition Letters*, 10:335–347.

Somol, P. and Pudil, P. (2000). Oscillating Search Algorithms for Feature Selection. In *Submission to the 15th International Conference on Pattern Recognition, Barcelona.*

Yang, J. and Honavar, V. (1997). Feature Subset Selection Using a Genetic Algorithm. In *Genetic Programming*, pages 380–385.