

Variable Neighborhood Search for the Feature Subset Selection Problem *

Miguel García-Torres † Belén Melián-Batista* José A. Moreno-Pérez*
J. Marcos Moreno-Vega*

*Departamento de Estadística, I. O. y Computación, University of La Laguna
Facultad de Matemáticas, La Laguna, España
{mgarcia,mbmelian,jmmoreno,jamoreno}@ull.es

1 Introduction

Given a set of labelled examples, the aim of a classification problem is to assign a label to each new instance. The effectiveness of the classification is related to many factors, such as the quality of the data. The Feature Selection problem can be considered to remove redundant and irrelevant information from these data. The aim of feature selection applied to classification problems is to determine the subset of features with which classification achieves the highest effectiveness. Since the feature selection problem is an *NP*-hard optimization problem [5], for real-world data with high dimensionality, heuristics algorithms are used. Feature selection can be categorized into two approaches: the filter and wrapper models [5]. The filter approach selects the features without involving any learning algorithm. The wrapper model considers a predetermined learning algorithm (or classifier) to evaluate the features and determine the subset to be selected.

Variable Neighborhood Search (VNS) [3] is a recent metaheuristic for solving combinatorial and global optimization problems based upon a simple principle: systematic changes of neighborhood within the search. When a local minimum is reached, a shake procedure performs a random search. This determines a new starting point for running an improvement method. The process iterates until a stopping criterium is met.

This paper presents a new filter-based VNS metaheuristic, which makes use of the symmetrical uncertainty (SU) [9] measure to guide the search. The rest of the abstract is organized as follows. Next section provides a formal description of feature selection in classification and presents the filter approach. Section 3 summarises the proposed VNS. Finally, section 4 reports the computational results and conclusions.

*This research has been partially supported by the projects TIC2002-04242-C03-01 (70% of which are FEDER funds) and PI042004/088

†The research of this author has been partially supported by a CajaCanarias grant

2 Related Work

The feature space is represented by $X = \{X_j : j = 1, \dots, d\}$, where d is the dimension of X . A dataset is denoted as A and each data object is called example if label is known and instance otherwise. The set of examples is called training set T and the set of instances, test set V . Then, given the training set T , the aim of the classification task is to assign a label to each instance from the test set V .

Several classifiers have been proposed in the literature to carry out the classification task. A widely used classifier is Naive Bayes [8]. The Naive Bayes classifier consists of estimate *a posteriori* probabilities of all possible classifications and assigning the label with the highest probability. The Bayes theorem is used to estimate these probabilities. Two well known filter approaches are *Relief* [4] and *FOCUS* [1]. *Relief* assigns a relevance weight to each feature and searches for all the relevant features. *FOCUS* exhaustively examines all subsets of features, selecting the minimal subset of features sufficient to determine the label of all examples.

The proposed filter-based Variable Neighborhood Search method is based on the fact that a good feature subset has to contain features highly correlated to the class and uncorrelated to each other. Then, a feature is considered as good if it is highly correlated to the class but not highly correlated to any of the other features in the subset. In order to measure the correlation between features, it is possible to use measures based either on the classical linear correlation or on information theory. Several advantages and disadvantages of these two approaches have been discussed in [9]. The proposed VNS uses the symmetrical uncertainty (SU) correlation measure between features X and Y , which is defined in the following way. Let $P(x_i)$ be the prior probabilities for all values of X , and $P(x_i|y_j)$ be the posterior probabilities of X given the values of Y . Then the *symmetrical uncertainty* is stated as follows.

$$SU(X, Y) = 2 \left[\frac{IG(X|Y)}{H(X) + H(Y)} \right], \quad IG(X|Y) = H(X) - H(X|Y)$$

where $H(X)$ is the entropy of a variable X , $H(X|Y)$ is the entropy of X after observing values of Y and $IG(X|Y)$ is the *information gain* [7].

3 Filter-based Variable Neighborhood Search

Variable Neighborhood Search (VNS) [3] is a recent metaheuristic for solving combinatorial and global optimization problems based upon the simple principle of systematically changing the neighborhoods within the search. Let \mathcal{N}_k , ($k = 1, \dots, k_{max}$) be a finite set of neighborhood structures, and $\mathcal{N}_k(S)$ be the set of solutions in the k^{th} neighborhood of a solution S . Neighborhoods \mathcal{N}_k may be induced from metric functions introduced into a solution space Z . If $d(.,.)$ is this distance then take increasing values d_k , $k = 1, \dots, k_{max}$ and set $\mathcal{N}_k(S) = \{S' \in Z : d(S, S') \leq d_k\}$. Most local search heuristics use only one neighborhood structure \mathcal{N} . Therefore a series of nested neighborhoods are obtained from a single neighborhood by taking $\mathcal{N}_1(S) = \mathcal{N}(S)$ and $\mathcal{N}_{k+1}(S) = \mathcal{N}(\mathcal{N}_k(S))$, for every solution S . This means that a move to the k -th neighborhood is performed by repeating k times a move into the original neighborhood. A solution $S' \in Z$ is a *local minimum* with respect to \mathcal{N}_k if there is no

Initialization.

Find an initial solution S .

Select the set of neighborhood structures \mathcal{N}_k , for $k = 1, \dots, k_{max}$.

Choose a stopping condition.

Iterations.

Repeat the following sequence until the stopping condition is met:

(1) Set $k \leftarrow 1$.

(2) Repeat the following steps until $k = k_{max}$:

(a) Shaking.

Generate a point S' at random from the k^{th} neighborhood of S ($S' \in \mathcal{N}_k(S)$).

(b) Improvement method.

Apply some improvement method with S' as initial solution; denote as S'' the obtained solution.

(c) Move decision.

If this solution is better than the incumbent, then $S \leftarrow S''$, and $k \leftarrow 1$; otherwise, set $k \leftarrow k + 1$. Goto (a).

Figure 1: Basic Variable Neighborhood Search.

solution $S \in \mathcal{N}_k(S') \subseteq Z$ better than S' (i.e., such that $f(S) < f(S')$ where f is the objective function of the problem).

The Basic Variable Neighborhood Search (BVNS) method uses deterministic changes in the neighborhood structure for perturbation or shaking. Its steps are given in Figure 1. The stopping condition may be the maximum CPU time allowed, the maximum number of iterations, or the maximum number of iterations between two improvements.

Given a solution S , we denote by \bar{S} to the set of features not belonging to the solution S .

- *Initialization.* In order to perform the initialization step of the metaheuristic, the procedure that generates the initial solution and the neighborhood structure must be defined.

Initial solution

1. Select at random an integer value, r , between $\lfloor \alpha d / \log_2 d \rfloor$ and $\lfloor d / \log_2 d \rfloor$ with d the number of features and α a parameter fixed by the user.
2. Initialize the solution $S = \emptyset$;
3. For each feature X_j , calculate its symmetrical uncertainty with respect to the class feature, C , denoted by $SU(X_j, C)$.
4. Add to S the r features with the highest values of symmetrical uncertainty.

The k^{th} neighborhood of the solution S , $\mathcal{N}_k(S)$, consists of all the solutions that can be reached from S by exchanging k features in the solution with k features out of the solution, adding k features to the solution or removing k features from the solution.

- *Shaking.* This procedure generates a solution S' at random from the k^{th} neighborhood of S ($S' \in \mathcal{N}_k(S)$). If the number of features in the solution is smaller than k then S' is constructed by adding k features in \bar{S} to S ; if $|S|$ and $|\bar{S}|$ are both greater than k , then the new trial solution is obtained by exchanging k features of S by k features of

\bar{S} ; otherwise, S' results from removing k features of S . In these three cases the features added, removed or exchanged are selected at random.

- *Improvement method.* The improvement step is performed on the current solution S following both a removing phase and an adding phase described below, after obtaining several threshold values.

1. *Thresholds.* This phase consists of calculating thresholds and measures to decide the features that have to be removed and added. These values are obtained considering the initial solution S and are not updated along the improvement process.

For each feature $X_i \in S'$ and $X_i \in \bar{S}'$, we calculate the values m_i and \bar{m}_i , respectively

$$m_i = \max_{X_j \in S', X_j \neq X_i} SU(X_i, X_j) \quad \bar{m}_i = \max_{X_j \in \bar{S}'} SU(X_i, X_j)$$

Let $\lambda_{S'}$ be the threshold that allows us to determine the features to be removed from the solution. Let $\lambda_{\bar{S}'}$ be the threshold that allows us to determine the candidate features to be added to the solution. Given the fixed values λ_1 and λ_2 , $\lambda_{S'}$ and $\lambda_{\bar{S}'}$ are defined as

$$\lambda_{S'} = \lambda_1 \max_{X_i \in S'} m_i \quad \lambda_{\bar{S}'} = \lambda_2 \max_{X_i \in \bar{S}'} \bar{m}_i$$

Let S'_R be the set of features, $X_i \in S'$, that satisfy the following rule

$$m_i > \max\{\lambda_{S'}, SU(X_i, C)\}$$

and let \bar{S}'_A be the set of features, $X_i \in \bar{S}'$, such that

$$\bar{m}_i < \lambda_{\bar{S}'}$$

2. *Removing phase.* Let $S'' = S' \setminus S'_R$ be the solution obtained from S' by removing the features in S'_R .
3. *Adding phase.* With the purpose of choosing the features in \bar{S}'_A to be added to the solution in this phase, we calculate the maximum symmetrical uncertainty value between each feature in S'' and the class feature that is defined as

$$M_{S''} = \max_{X_i \in S''} SU(X_i, C)$$

Then a candidate feature $X_i \in \bar{S}'_A$ is included in S'' if it satisfies the following rule.

$$SU(X_i, C) > M_{S''}$$

If this rule is satisfied, then $S'' = S'' \cup \{X_i\}$, the value $M_{S''}$ is updated and repeat this process for all the features in \bar{S}'_A .

- *Move decision* In order to evaluate the goodness of a solution S , we use the correlation measure given by the equation 1 proposed in [2].

$$G_S = \frac{|S| < SU(X, C) >}{\sqrt{|S| + |S|(|S| - 1) < SU(X, X) >}} \quad (1)$$

Then the process moves to solution S'' if $G_{S''} > G_{S'}$.

4 Computational experience

The aim objective of our computational experience was to corroborate that the VNS is able to reach a good subset of features to perform the classification task either by itself or as a starting point to get a higher set of features. With this purpose, the computational experience was carried out in two phases. In the first phase, we compare the accuracy percentages and the number of features obtained by both VNS and VNS with a post-processing step (VNS-CFS). This last procedure consists of applying CFS to the solution given by the VNS. CFS (Correlation-based Feature Selection) evaluates the merit of the subset of features applying the equation 1 and uses a heuristic to include locally predictive features and avoid the re-introduction of redundancy (see [2]). This experiment is performed on several artificial data sets, for which the set of relevant features is known. The results obtained in this phase are summarized in table 1. In the second phase, we compare the VNS-CF with the filter algorithm CFS-SF. CFS-SF starts with an empty subset of features and at each step, it adds to the subset the feature that maximizes the value G_S given by the equation 1. This process is iterated until no improvement is possible. The results obtained in this phase are summarized in table 1, that reports real-world data sets. We implemented our filter-based Variable Neighborhood Search and used the algorithms CFS and CFS-SF provided by the Weka learning project [8].

Tables 1 and 2 report the values F , C and I for each data set that give the number of features, instances and classes respectively. The four rows reported for each algorithm in both tables summarizes the 5×2 cross-validation accuracy percentage, its standard deviation, the average number of features and its standard deviation. The data sets considered in our computational experiments were obtained from the UCI repository [6], from which full documentation about all data sets can be obtained. VNS parameters considered were $k_{max} = 4$, $\lambda_1 = \lambda$ and $\lambda_2 = 1 - \lambda$.

For the first experiment with artificial data sets, we found that for four data sets (monks-1, monks-2, monks-3 and tic-tac-toe) the VNS is able to obtain a set of features with which the accuracy percentages are similar to those obtained by VNS-CFS. Then the post-processing step based on CFS is not able to significantly increase the accuracy percentages by adding new features to the subset provided by the VNS. On the other hand, for data sets led24 and led7, the VNS does not present a good behavior, since several highly correlated features are required to obtain good accuracy percentages.

The results summarized in table 2 corroborate the effectiveness of the proposed VNS-CFS since it reaches a significantly smaller subset of features than the CFS-SF with similar accuracy percentages for most data sets.

References

- [1] H. Almuallim and T.G. Dietterich, Learning with Many Irrelevant Features, Proceedings of the Ninth National Conference on Artificial Intelligence (1991) 547-552
- [2] M. Hall, Correlation-based feature selection for discrete and numeric class machine learning, Proceedings of the Seventeenth International Conference on Machine Learning, (2000) 359–366

Table 1: Accuracy percentages and number of features.

DATA SET (F, I, C)	NAIVE-BAYES						
	LED24	LED7	MONKS-1	MONKS-2	MONKS-3	TIC-TAC-TOE	CORRAL
	(24, 6400, 10)	(7, 6400, 10)	(7, 864, 2)	(7, 864, 2)	(7, 864, 2)	(9, 1916, 2)	(6, 128, 2)
	$\lambda = 1$	$\lambda = 1$	$\lambda = 1$	$\lambda = 1$	$\lambda = 0.5$	$\lambda = 0.5$	$\lambda = 0.5$
VNS	50.59 (9.61)	64.11 (6.70)	75 (1.84)	66.80 (1.43)	95.39 (5.62)	69.93 (1.47)	75.06 (6.92)
	5.28 (1.36)	5.40 (0.89)	2.62 (1.07)	2.74 (0.69)	2.60 (0.85)	1.36 (0.59)	3.06 (0.90)
VNS-CFS	70.74 (4.01)	72.16 (3.39)	75 (1.84)	66.80 (1.43)	97.22 (0.65)	70.24 (2.37)	79.06 (4.89)
	12.86 (1.56)	6.86 (0.35)	2.72 (0.96)	2.74 (0.69)	3.36 (0.79)	3.54 (0.50)	4.02 (0.47)

Table 2: Accuracy percentages and number of features.

DATA SET (F, I, C)	NAIVE-BAYES						
	ARRYTMIA	COIL2000	DNA	PROMOTERS	ISOLET	LUNG-CANCER	SPLICE
	(280, 452, 16)	(86, 5822, 2)	(181, 3186, 3)	(59, 106, 2)	(618, 1560, 26)	(57, 32, 3)	(62, 3190, 3)
	$\lambda = 0.5$	$\lambda = 1$	$\lambda = 1$	$\lambda = 0.5$	$\lambda = 1$	$\lambda = 0.5$	$\lambda = 0.5$
VNS-CFS	66.26 (2.34)	93.68 (0.28)	92.26 (1.12)	89.28 (5.94)	70.69 (4.85)	47.63 (11.83)	95.52 (0.43)
	11.98 (1.17)	5.66 (1.39)	29.80 (1.24)	5.34 (1.47)	34.84 (4.64)	7.3 (1.79)	23.32 (2.13)
CFS-SF	67.45 (2.68)	92.32 (1.40)	92.21 (1.33)	91.32 (4.55)	80.70 (3.25)	50.63 (10.39)	94.85 (1.09)
	19.6 (3.62)	11.2 (3.64)	29.8 (1.03)	5.4 (1.58)	95.3 (5.37)	8.6 (2.54)	17.1 (7.09)
ALL	64.21 (1.57)	87.21 (1.18)	93.83 (0.57)	87.36 (5.55)	81.35 (2.01)	52.5 (8.43)	95.28 (0.55)

- [3] P. Hansen and N. Mladenovic, Variable neighborhood search: Principles and applications, *European Journal of Operational Research* 130, (2001) 449-467
- [4] K. Kira, L. Rendell, The feature selection problem: Traditional methods and a new algorithm, in: *Tenth National Conference Conference on Artificial Intelligence (AAAI-92)*, MIT, 1992, pp. 129-134.
- [5] R. Kohavi, G. H. John, Wrappers for feature subset selection, *Artificial Intelligence* 97 (1-2) (1997) 273-324.
- [6] P. M. Murphy, D. W. Aha, Uci repository of machine learning. <http://www.ics.uci.edu/mlearn/MLRepository.html>
- [7] J. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann, (1993)
- [8] I. Witten, E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*, Morgan Kaufmann, 2000.
- [9] L. Yu and H. Liu, Efficient Feature Selection via Analysis of Relevant and Redundancy, *Journal of Machine Learning Research* 5 (2004) 1205-1224