

# Large Scale Feature Selection Using Modified Random Mutation Hill Climbing

Michael E. Farmer<sup>§</sup>, Shweta Bapna<sup>§</sup>, and Anil K. Jain\*

<sup>§</sup> Eaton Corporation

\* Michigan State University

Email: farmerm3@msu.edu, bapnashw@egr.msu.edu, jain@cse.msu.edu

## Abstract

*Feature selection is a critical component of many pattern recognition applications. There are two distinct mechanisms for feature selection, namely the wrapper method and the filter methods. The filter methods are generally considered inferior to the wrapper methods since the wrapper methods directly integrate the classifier to be used, and because most filter methods do not directly address feature correlation. Wrapper methods are, however, computationally more demanding than filter methods. One of the popular methods for wrapper-based feature selection is random mutation hill climbing. It performs a random search over the feature space to derive the optimal set of features. We will describe two enhancements to this algorithm, one that will improve its convergence time and the other that will allow us to bias the results towards either higher accuracy or lower feature count. We will apply the algorithm to a real-world massive-scale feature selection problem involving the image classification problem associated with suppressing airbags for children. We will provide classification results on an image database of nearly 4,000 images that indicate the advantages of the proposed method.*

## 1. Introduction

Two common and distinct mechanisms for feature selection have been devised to make the problem computationally manageable: the wrapper method and the filter method [3][6][7]. Wrapper methods use the actual classifier, and its resultant probability of error, to select the feature subsets. The feature selection algorithm is wrapped inside the classifier. Filter methods analyze features independently of the classifier and use a ‘goodness’ metric to decide which features should be kept. Because of their use of the classification results as a metric, wrapper methods generally perform better than filter methods.

One taxonomy for feature selection is the size of feature space into the following three categories [9]:

1) small-scale feature space ( $N < 20$ )

2) medium-scale feature space ( $20 < N < 50$ )

3) large-scale feature space ( $N > 50$ ),

where  $N$  is the number of given features. With the current state-of-the-art processors, it is possible to extract many features from an image in real-time. For example, computing the moments of an image generates 703 features for up to the 36<sup>th</sup> order, and 1081 features for up to the 45<sup>th</sup> order. Likewise, appearance-based object recognition methods can generate a large number of features, where sub-sampling a 400x320 image at every 10<sup>th</sup> pixel generates 1280 features. Therefore, we propose an extension of the above taxonomy to include a fourth category, namely *massive-scale* feature selection, for datasets on the order of 1,000 features, or more.

Another useful taxonomy for feature selection divides these methods into three classes: (i) complete, (ii) heuristic, and (iii) random [3]. While the literature has shown no clear superiority of any particular feature selection method, some are more suitable for large-scale applications than others. Obviously, any feature selection method that would approximate a complete, or exhaustive, search in these massive data spaces is infeasible since, for the 36<sup>th</sup> order moments example, there would be  $4.2e+211$  possible combinations. Of course, any non-exhaustive search method cannot guarantee to find the optimal feature set [3][6][7][8].

Jain and Zongker [2] found that the heuristic methods (forward sequential search) performs, best in large data sets, while Kudo and Sklansky [9] found the random methods (genetic algorithms) are superior for large-scale problems [2][9]. Note, however, that their definition of large-scale was based on the definition above ( $> 50$  features). For massive-scale feature selection problems these popular methods can also be too computationally demanding for practical training times. Therefore a means to speed-up the wrapper method is required. Another random search method has been used as well, namely random mutation hill climbing (RMHC), however, it is not quite as powerful as a genetic algorithm [4].

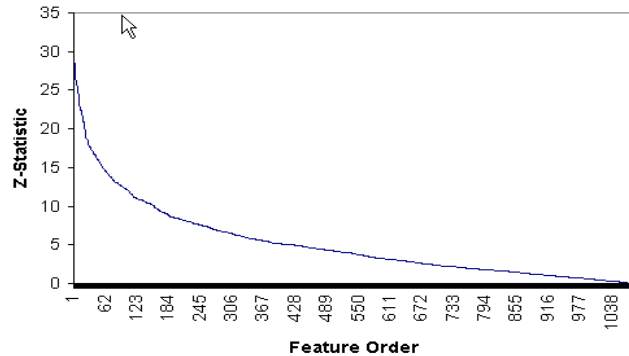
The aim of this paper is to show that the random mutation hill climbing method can be enhanced in terms of its speed by adopting methods from simulated annealing,

while its ability to dramatically reduce the feature count can be easily improved by a modification to the figure of merit.

## 2. Random Mutation Hill Climbing (RMHC)

Random Mutation Hill Climbing is a member of the family of random search optimization tools that include methods such as simulated annealing, random mutation, and genetic algorithms [4]. Random search algorithms derive their power from the ability to search the optimization space in a random manner, which makes them inherently immune to local minima. The difficulty of the random methods is that the randomness must be controlled to ensure the method converges, while allowing it to be free enough to allow ‘complete’ coverage of the overall search space.

It is easier to control the randomness if the search surface is relatively well behaved. Figure 1 shows a typical decay in the Mann-Whitney z-statistic across the complete feature set for the application defined in Section 5. The Z-statistic is a convenient non-parametric statistical method for estimating the discrimination between feature distributions. Notice there is a very gradual decay in the discrimination ability. Thus the randomness of the search space should be modest, since the addition of any one feature should have a finite effect on the outcome. Additionally, Kudo and Sklansky’s comment that reducing the dimension of the feature space *always* results in some loss in discrimination ability in real-world applications, which implies the features in most applications will have a similar discrimination ranking behavior [9].



**Figure 1. Mann-Whitney statistic versus feature ranking.**

For the random mutation hill climbing algorithm, the complete set of features is represented by a binary string of length  $N$ , where a bit in the string is set to ‘1’ if it is to be kept, and set to ‘0’ if it is to be discarded, and  $N$  is the original number of features [4]. The key free parameter to set when using an algorithm such as random mutation is the number of bits,  $M$ , that are allowed to randomly

change at each iteration. The most conservative approach is to only allow a single bit to change per pass [4]. The algorithm operates as follows:

- 1) Initialize a binary string,  $S$ , of length  $N$ , where  $M$  features are marked as used, ‘1’ and the remaining  $N-M$  are ‘0’.
- 2) Test binary string,  $S$ , for fitness  $F(S)$  using the probability of correct classification.
- 3) Randomly mutate  $M$  bits in the binary string,  $S$ .
- 4) Return to step (2) and continue until either the fitness goal is reached or the maximum number of iterations is reached.

Since this is a wrapper algorithm, the definition of the fitness function for the basic method is simply the classification error:

$$F(S) = P_{error}(S), \quad (1)$$

where  $S$  is the set of currently utilized features.

## 3. RMHC Speed Enhancements

We adopt a method that is loosely motivated by simulating annealing where a system is cooled over time [10]. We implement this concept of cooling by reducing the number of features that can be mutated at each iteration. This allows us to get the benefit of rapidly changing the mix of the features in the early iterations, and then more slowly changing the set of features as the system converges to a solution. The number of features to mutate at any iteration is:

$$M = M_{max} * \min\left(\frac{(I_{max} - i_{current})}{I_{max}}, P_{error}(S)\right), \quad (2)$$

where  $M_{max}$  is the maximum allowed value for the number of features to mutate,  $I_{max}$  is the maximum number of iterations,  $i_{current}$  is the current iteration, and  $P_{error}(S)$  is the current error rate.

## 4. Dimensionality Reduction Enhancements

The definition of the fitness function is also required for random mutation hill climbing. Since this is a wrapper algorithm, the fitness function must be a function of the classification accuracy. Note the fitness function should also be a function of the number of features remaining or else there will be no explicit incentive to reduce the number of features. A natural fitness function is then:

$$F(S) = \alpha \cdot P_{error}(S) + (1 - \alpha) \cdot \frac{|S|}{N}, \quad (3)$$

where  $N$  is the original number of features,  $S$  is the current

set of features,  $|S|$  is the cardinality of  $S$ , and  $0 \leq \alpha \leq 1$  is the relative weighting factor between dimensionality reduction and error rate. Thus, this fitness function is the weighted average of the classification error and the fraction of the features used. The goal in the random search is to drive both of these values to zero simultaneously, but at some point there is clearly a trade-off between dimensionality reduction and error rate. The parameter  $\alpha$  is set based on how aggressively the algorithm reduces the number of features. A larger  $\alpha$  encourages the final solution to be based more on the resultant classification error, and a smaller  $\alpha$  encourages the final solution to use fewer features at the expense of classification accuracy.

The enhanced random mutation hill-climbing algorithm now operates as follows:

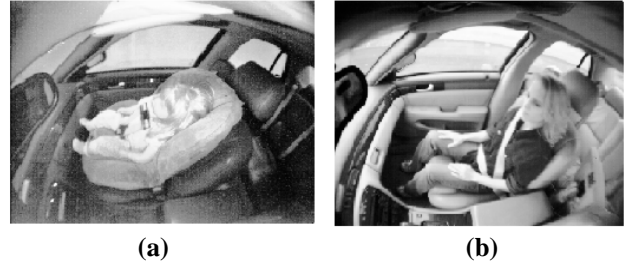
- 1) An initial vector of utilized features is a randomly-generated binary vector, where  $M$  features are marked as used.
- 2) Test currently utilized features for weighted average fitness  $F(S)$  in Eq. 3.
- 3) Compute the current value of  $M$  based on current  $F(S)$  and the current iteration number.
- 4) Randomly mutate  $M$  bits in currently utilized feature vector.
- 5) Return to step (2) and continue until either the fitness goal is reached or the maximum number of iterations is reached.

## 5. Experimental Results

Recently, considerable attention has been paid to developing ‘smart’ airbags for automobiles where a using a computer vision system is used to determine not if it is safe to deploy the airbag in a crash event, based on the type of occupant in the seat [1]. In this application our goal is to differentiate between an infant in a forward or rear-facing car seat from an adult as shown in Figure 1. If the occupant is an infant then we disable the airbag, while if it is an adult we enable the airbag, unless they are too close to the airbag. Children are either classified as an infant if very small, or classified as an adult and then the airbag is disabled if they are too close at the time of deployment. Thus we have a 2-class problem with which to test our proposed feature selection method.

The training and test database consisted of 2597 infant images and 983 adult images. The features used for the classification are the Legendre moments of the edge-detected images as defined in [1]. The moments to the 45<sup>th</sup> order were used to allow the greatest image reconstruction accuracy. The 45<sup>th</sup> order moments generate a feature space with 1081 dimensions. Clearly, for any real-time implementation, and also to avoid the curse of dimensionality, a significant reduction in the number of features is necessary.

The performance of the feature selection procedure has been validated using a 50/50 cross validation testing where the dataset was randomly divided into training and test sets of equal size. We ran the 50/50 testing for 10 iterations of random re-sampling. The classifier used for this testing was a k-nearest neighbor classifier with  $k=9$ .



**Figure 2. Two-class occupant classification problem, (a) infant, (b) adult.**

Table 1 through Table 4 show the results of varying the dimensionality reduction weighting factor,  $\alpha$ , from 0.2 to 1.0. As can be seen from Tables 1 through 4, the trade-off between the number of features and the classification accuracy need not dramatically affect the system performance. Notice that reducing  $\alpha$  from  $\alpha = 1.0$  to  $\alpha = 0.8$  reduces the final number of features by a factor of two with no net reduction in classification accuracy.

**Table 1. Results for  $\alpha = 0.2$  and  $M=27$ .**

	Predicted Infant	Predicted Adult
True Infant	1313	6
True Adult	20	481

**Table 2. Results for  $\alpha = 0.6$  and  $M=5$ .**

	Predicted Infant	Predicted Adult
True Infant	1319	0
True Adult	14	475

Figure 2 summarizes the resultant performance of the classification system versus the weighting parameter,  $\alpha$ . Notice that even for  $\alpha = 0.2$  the performance of the system is still better than 98% correct while the number of features used has fallen by a factor of five. Clearly, by adding in the new weighting factor, the random mutation algorithm is able to more optimally select the features with very little loss in system performance.

Figure 3 shows the resultant improvement in convergence time of the system by allowing the number of features mutated during each iteration to be cooled over time. The improvement in convergence time is nearly a factor

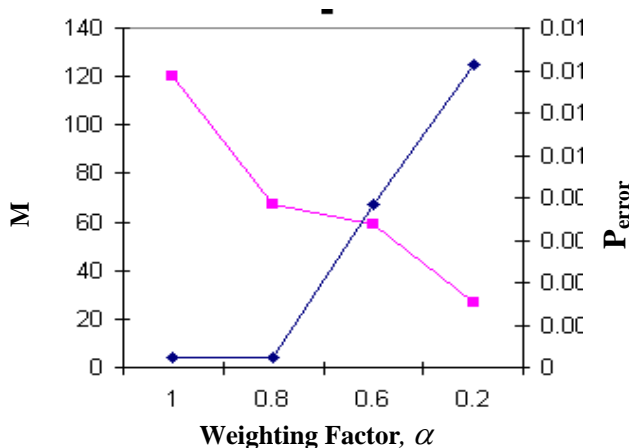
of 10 for a system that initially randomly selects 8 features per iteration, and then eventually cools to the level of one feature per iteration. This is in stark contrast to an algorithm such as floating forward sequential search, where the execution time dramatically increases with the number of features selected during each iteration, and all combinations of these must be processed, which results in a combinatorial growth in the processing time.

**Table 3. Results for  $\alpha = 0.8$  and  $M=67$**

	Predicted Infant	Predicted Adult
True Infant	1319	0
True Adult	1	500

**Table 4. Results for  $\alpha = 1.0$  and  $M=120$**

	Predicted Infant	Predicted Adult
True Infant	1319	0
True Adult	1	500

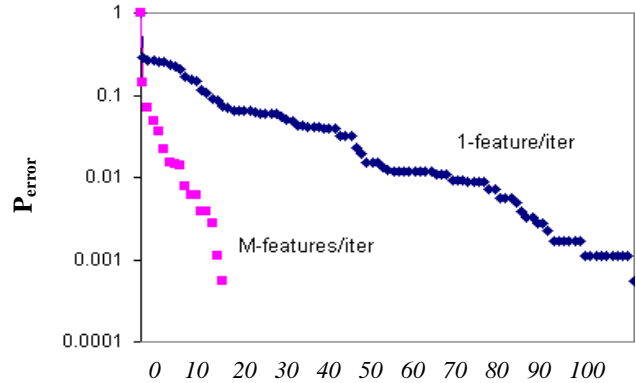


**Figure 3. Number of features and probability of error versus  $\alpha$ .**

## 6. Summary and Conclusions

We have shown that two modifications to the random mutation feature selection algorithm can both improve its feature selection ability and also greatly reduce the required processing time for the algorithm. The addition of an ad hoc cooling function, that reduces the number of features the algorithm attempts to add over time, can greatly improve the convergence time of the algorithm. Additionally, modifying the fitness function, so that it is a function of the number of features, can greatly reduce the resultant number of features that the algorithm selects, with very little impact to the classification accuracy. The

performance of our algorithm was demonstrated on a real-world problem of using a vision system to recognize the type of occupant that is in the front passenger seat of an automobile. Through our ongoing research, we plan to continue to investigate the application of feature selection algorithms to this problem, since it provides an interesting real-world application and a significant feature-selection challenge.



**Figure 4.  $P_{error}$  versus iteration number.**

## References

- [1] M. Farmer and A. K. Jain, "Occupant classification system for automotive airbag suppression", *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 756-761, 2003.
- [2] A.K. Jain and D. Zongker, "Feature selection: Evaluation, application, and small sample performance", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 19, no. 2, pp. 153-158, Feb. 1997.
- [3] M. Dash and H. Liu, "Feature selection for classification", *Intelligent Data Analysis*, vol. 1, pp. 131-156, 1997.
- [4] D. B. Skalak, "Prototype and feature selection by sampling and random mutation hill climbing algorithms", *Proc. of the Eleventh International Conference on Machine Learning*, pp. 293-301, 1994.
- [5] P. Somol, P. Pudil, J. Novovicova, and P. Paclik, "Adaptive floating search methods in feature selection", *Pattern Recognition Letters*, vol. 20, no. 11-13, pp.1157-1163, 1999.
- [6] D. Koller and M. Sahami, "Toward optimal feature selection", *Proc. Of the 13<sup>th</sup> International Conference On Machine Learning*, Morgan Kaufman, pp. 197-243, 1996.
- [7] D.W. Aha and R.L. Bankert, "A comparative evaluation of sequential feature selection algorithms", in *Learning from Data: AI and Statistics*, Springer-Verlag, 1996.
- [8] R. Kohavi and G.H. John, "The wrapper approach", In *Feature Extraction, Construction and Selection: A Data Mining Perspective*, Kluwer Academic, pp.33-50, 1998.
- [9] M. Kudo and J. Sklansky, "Comparison of algorithms that select features for pattern classifiers", *Pattern Recognition*, vol. 33, no. 1, pp. 25-41, 2000.
- [10] K. Kirkpatrick, C.D. Gelatt, and M.P. Vecchi, "Optimization by Simulated Annealing", *Science*, vol. 220, no. 4598, May 1983.