A Dependency-Based Search Strategy for Feature Selection

Ahmed Al-Ani

Faculty of Engineering and Information Technology
University of Technology, Sydney, Australia
ahmed@eng.uts.edu.au

Abstract

Feature selection has become an increasingly important field of research. It aims at finding optimal feature subsets that can achieve better generalization on unseen data. However, this can be a very challenging task, especially when dealing with large feature sets. Hence, a search strategy is needed to explore a relatively small portion of the search space in order to find "semi-optimal" subsets. Many search strategies have been proposed in the literature, however most of them do not take into consideration relationships between features. Due to the fact that features usually have different degrees of dependency among each other, we propose in this paper a new search strategy that utilizes dependency between feature pairs to guide the search in the feature space. When compared to other well-known search strategies, the proposed method prevailed.

Key words: Feature selection, search strategy, dependency, mutual information

1. Introduction

The identification of optimal feature subsets plays an important role for a wide range of classification problems, as it can lead to better performance in terms of accuracy and computational cost. The selection of good feature subsets requires an evaluation measure to estimate the goodness of subsets and a search strategy to generate candidate feature subsets (1). Evaluation measures are broadly divided into two categories: filters and wrappers. A filter method uses a measure that is independent of the predetermined classification algorithm to estimate the goodness of candidate subsets. A wrapper method on the other hand estimates the goodness of a candidate subset using the classification accuracy obtained by feeding that particular subset to the adopted classification algorithm. Thus, wrappers are computationally more expensive than filters, however they are usually more accurate.

Searching for the optimal subset, which can achieve the best performance according to the defined evaluation measure, is quite a challenging task. The exhaustive search, which considers all possible subsets, is guaranteed to find the optimal solution. However, it is impractical to run, even with moderate size feature sets. A number of other search strategies that differ in their computational cost and optimality have been proposed in the literature. One of the early search strategies is the branch and bound (2), which requires the evaluation function to be monotonic. This method can be computationally expensive for large data sets. Sequential search methods, such as sequential forward selection and sequential backward elimination (3), have been widely used because of their simplicity and relatively low computational cost. The major drawback of the traditional sequential search methods is the nesting effect, i.e., in backward search when a feature is deleted, it cannot be re-selected, and in forward search when a feature is selected, it cannot be deleted afterwards. A slightly more reliable sequential search method is the plus-l-minus-r ($l - r$), which considers removing features that were previously selected and selecting features that were previously eliminated (4). Another trend of search strategies is the stochastic search, where it has been found that including some randomness in the search process makes it less sensitive to the dataset (1), and hence helps avoid local minimas. Some of the famous stochastic methods used in feature selection are: simulated annealing (5), Genetic Algorithm (GA) (6), Ant Colony Optimization (ACO) (7), Particle Swarm Optimization (PSO) (8) and Differential Evolution (DE) (9).

Despite the encouraging results achieved in certain cases, the main drawback of most of the above methods is that they do not take into consideration the importance of features and how they are related. This may have an effect on the performance, especially for complicated search problems. Some methods attempt to estimate the relevance of subsets of features as the basis for selection. It has been shown that estimating the relevance of individual features may not be difficult, however, the real challenge is to estimate the relevance of subsets of features. This issue has been studied by a number of researchers (10; 11; 12; 13; 14). Some of the interesting finding of Guyon et al. (1) are:

- a feature that is irrelevant by itself may become relevant when used with other features;
- a relevant feature may not be needed because of possible redundancies.

Because of the difficulty associated with estimating the relevance of subsets of features, the proposed search strategy adopts the wrapper approach and focuses on dependency between feature pairs as a mean to guide the search.

The paper is organized as follows: the next section explains the importance of dependency between features. Section 3 describes the proposed search strategy. Experimental results are presented in section 4, and a conclusion is given in section 5.

2. Dependency Between Features

One of the promising approaches in searching the feature space is population-based search, where the current population consists of a number of feature subsets. Each one of these subsets is modified, in a certain way, to produce the next generation of subsets. Examples of population-based search procedures include Genetic Algorithms (GA), Particle Swarm Optimization (PSO) and Differential Evolution (DE). All of these methods were not originally developed for feature selection, and hence, their original implementations do not take into consideration relationships between features when producing future populations.

Let's presume that $\mathcal{F}$ is the original feature set with $n = 20$ features, $\mathcal{S}_1$ and $\mathcal{S}_2$ are two subsets of the current population, each with $m = 4$ features. Let's also consider that $\mathcal{S}_1$ is one of the best subsets that has been tested so far and we would like to modify $\mathcal{S}_2$, with the help of $\mathcal{S}_1$, to produce a better subset for the next population. If $\mathcal{S}_1 = \{f_2, f_5, f_9, f_{15}\}$, $\mathcal{S}_2 = \{f_1, f_2, f_6, f_{17}\}$ and features f_1 and f_9 are highly dependent, then it will be logical to only consider replacing f_6 and/or f_{17}. Furthermore, instead of randomly choosing any feature from $\mathcal{F}$ as a replacement feature, the search space can be reduced using the dependency between features. For instance, if f_6 is to be replaced, we first find which of the two features f_5 and f_{15} is closer to it, i.e., has a higher dependency value. Let's presume that f_{15} is closer to f_6. Then, candidate replacement features are reduced to the ones that lie (in terms of dependency) between f_6 and f_{15}. The newly produced subset will not only have a chance of being a better subset than $\mathcal{S}_2$, but it might also outperform $\mathcal{S}_1$, if this replacement makes it closer to the optimal solution.

In addition to the above, dependency is also useful in identifying the K best subsets that are selected so far. For instance, we can not choose both $\mathcal{S}_1 = \{f_2, f_5, f_9, f_{15}\}$ and $\mathcal{S}_3 = \{f_1, f_2, f_5, f_{15}\}$ to be among the K best subsets, since they are almost identical, where f_1 and f_9 are highly dependent as mentioned earlier. This restriction will enhance the exploration capability, where diversity among the "good" subsets is quite important in avoiding local minimas.

A famous approach to estimate dependency between two random variables is based on the concept of mutual information (15). The mutual information between random variables X and Y is defined as:

$$I(X;Y) = \int P_{XY}(x,y) \log\left(\frac{P_{XY}(x,y)}{P_X(x)P_Y(y)}\right) dxdy \tag{1}$$

The entropy, which is a measure of uncertainty of random variables, is usually used to represent mutual information according to the following formula:

$$\begin{aligned} I(X;Y) &= H(X) + H(Y) - H(X,Y) && (2)\\ &= H(X) - H(X|Y) && (3) \end{aligned}$$

where $H(X)$ is the entropy of X, $H(X,Y)$ is the joint entropy of the two random variables, while $H(X|Y)$ is the conditional entropy, which represents the uncertainty in X after knowing Y.

When we deal with real data, the main problem for evaluating $I(X;Y)$ is the estimation of probabilities $P_X(x)$,$P_Y(y)$ and $P_{XY}(x,y)$. One possible solution is to subdivide the XY plane into boxes of size $\Delta x \Delta y$. By doing so, we are able to estimate the *discrete* values of P_X, P_Y and P_{XY}. An alternative approach was proposed in (16), which uses variable box size over the XY plane. The method presented in (17) estimates the MI by an adaptive partitioning of the observation space. A Parzen window method is proposed in (18) to estimate the input distribution. For simplicity, we used here a fixed box size implementation. Hence, the MI can be rewritten as:

$$I(X;Y) = \sum_{r_x} \sum_{r_y} P_{XY}(r_x, r_y) \log \left(\frac{P_{XY}(r_x, r_y)}{P_X(r_x) P_Y(r_y)} \right) \tag{4}$$

where, r_x and r_y are the discrete levels for X and Y respectively.

Let's consider the earlier example about subset $\mathcal{S}_2$ and suppose that we have the entropies shown in Fig. 1, where $H(C)$ represents the entropy of the target classes. The objective here is to select the optimal subset of features that would minimize the uncertainty in C. Features f_1, f_2 and f_{17} provide good information about the target classes, as indicated by the area covered in shaded vertical lines. The figure also shows that the *additional* information provided by f_{15} is more than that provided by f_6 (the shaded grid areas). However, instead of presuming that f_{15} is the best feature when combined with f_1, f_2 and f_{17}, why not searching the area that surrounds f_{15}, i.e., features that are closer to f_{15} than f_6. This limited space may include

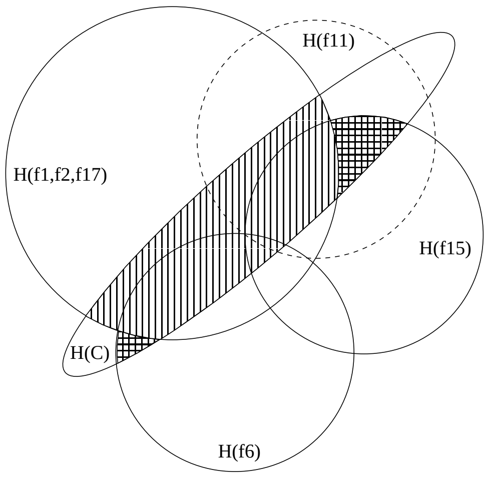

Figure 1: Importance of features in subset $\mathcal{S}_2$, and identifying a candidate replacement feature for f_6

good features, such as f_{11}, which when replacing f_6 will not only make $\mathcal{S}_2$ a better subset, but it will also make it better than the subset $\{f_1, f_2, f_{15}, f_{17}\}$.

3. The Proposed Search Strategy

The proposed search strategy is similar to GA, ACO, PSO and DE in the sense that they are all population-based methods. However, unlike other methods, the proposed search strategy utilizes dependency between feature pairs to guide the search. We will refer to it as DSS (Dependency-based Search Strategy), and it is implemented using a wrapper approach as follows:

1. Randomly initialize the subsets of the first population and sort them according to their fitness values
2. For each subset, $\mathcal{S}_i$, of the current population
    - Make a copy of $\mathcal{S}_i$

- Randomly choose one of the K best subsets, $\mathcal{S}_k$
- Identify candidate features for replacement, which are features of $\mathcal{S}_i$ that are not present in $\mathcal{S}_k$. Place those features in $\mathcal{S}'_i$
- Initially assign 0 to the accumulated difference between the original and newly generated subset, $AcDiff = 0$
- do
    - Randomly choose one of the features of $\mathcal{S}'_i$, $f_{i,j}$ and find the feature $f_{k,l}$ in $\mathcal{S}_k$ that maximizes $I(f_{i,j}; f_{k,l})$
    - identify the candidate features to substitute $f_{i,j}$ according to the following equation:

$$\mathcal{S}_m = \arg_{f_m} \Big(I(f_{k,l}; f_m) \geq I(f_{k,l}; f_{i,j}) \Big) \tag{5}$$

    - Randomly choose one of the features of $\mathcal{S}_m$, to substitute $f_{i,j}$, $\mathcal{S}'_i \leftarrow \mathcal{S}'_i \setminus \{f_{i,j}\}$
    - Calculate the accumulated difference between the replaced features and their replacements, as follows:

$$AcDiff = AcDiff + (H(f_m) - I(f_m; f_{i,j})) \tag{6}$$

- while $AcDiff$ is less than a certain constant and $\mathcal{S}'_i \neq \emptyset$
- Repeat the same procedure of step 2 for the remaining subsets

3. Evaluate the newly generated subsets
4. Select the K best subsets from the original and newly generated subsets, given that there is at least a certain ratio of difference between any two subsets. This is measured by averaging the mutual information between

each feature from a given subset and its closest counterpart in another subset

5. Select the remaining subsets of the population in a similar procedure as in the previous step
6. Discard the unselected subsets
7. If the stopping criterion is not met, goto step 2 to generate another population

The rationale behind Eq. 5 is that if $f_{i,j}$ is the closest feature to $f_{k,l}$ and the two features are highly dependent on each other, then $\mathcal{S}_m$ will only consist of $f_{i,j}$ and $f_{k,l}$. Hence, the replacement of $f_{i,j}$ will have little or no effect on the performance of $\mathcal{S}_i$. On the other hand, if there is only a small degree of dependency between $f_{i,j}$ and $f_{k,l}$, then $\mathcal{S}_m$ will consist of other features that are closer to $f_{k,l}$ than $f_{i,j}$, and in this case the replacement will most likely have an effect on the performance of $\mathcal{S}_i$. In other words, $\mathcal{S}_m$ will have an adaptive size, where the number of the candidate replacement features depends upon the degree of dependency between $f_{i,j}$ and $f_{k,l}$. Eq. 6 is used to specify when to stop replacing features. If features f_m and $f_{i,j}$ are highly dependent on each other, then $I(f_m; f_{i,j})$ will approach $H(f_m)$ according to Eq. 3 (the uncertainty in one of them after knowing the other will be close to 0) and $AcDiff$ will almost stay unchanged. Accordingly, this replacement is not expected to make a noticeable difference on the performance of $\mathcal{S}_i$, and hence, the algorithm will consider replacing another feature. On the other hand, if f_m and $f_{i,j}$ are not highly dependent, then $AcDiff$ will have a higher value due to this replacement. When $AcDiff$ exceeds a certain threshold, there will be some difference between the original and newly generated subsets. Note that we do not want the original and newly generated subsets to be very different, as this may lead to big jumps in the search space and will hinder the convergence of the algorithm.

Identifying the K best subsets also plays an important role in guiding the search, as it is important to choose good and diverse subsets. Concentrating on the goodness of subsets without considering how similar those subsets are may cause the search to be trapped in local minima. Thus, steps 4 and 5 aim at selecting good and yet diverse subsets to better explore the search space.

4. Experimental Results

Three different classification problems are considered. In the first problem the Madelon dataset, which is obtained from the UCI repository, is used. EEG data and speech segments are used in the second and third problems respectively. The following methods were implemented to select feature subsets: genetic algorithms, particle swarm optimization, differential evolution and the proposed dependency-based search strategy.

A GA-based feature selection solution would typically be a fixed length binary string representing a feature subset, where the value of each position in the string represents the presence or absence of a particular feature. A traditional GA algorithm was used here with probability of crossover = 0.8, and probability of mutation = 0.05. The obtained strings are constrained to have the number of '1's matching a predefined number of desired features. Feature selection using PSO is also implemented using binary strings. It uses particles' best and global best values to guide the search. A DE-based feature selection utilizes a real-valued optimizer and applies a rounding and uniform crossover operators.

The proposed DSS is implemented as explained in the previous section. Reasonable values of K and the ratio of difference between subsets were found to be around 5 and 0.1 respectively. For all experiments described below, a population size of 50 was used by the four feature selection methods. In addition, all of the four methods started with the same initial population and they all have the same stopping criterion, which is reaching a pre-defined maximum number of iterations.

4.1. The Madelon Dataset

The Madelon dataset was designed for the NIPS 2003 variable selection benchmark. It is a two-class classification problem with sparse binary input features. There are 500 features, only 5 of which are useful and the rest are either redundant or irrelevant. 2000 patterns were used for training and 600 for validation. This dataset represents quite a useful benchmark for feature selection, as the optimal subset of features is already known.

For all of the four methods, classification accuracy obtained using the k-nearest neighbor (k-NN) classifier with $k = 5$ was used as a fitness function, the desired number of selected features was set to 5 and the maximum number of iterations was set to 100. The experiment was repeated 30 times and the averaged classification accuracy values are shown in Fig. 2. The results indicate that the DSS not only achieved a higher accuracy by the end of the 100 iterations, but also required smaller number of iterations to find the optimal subset. PSO was found to be the second best, where it managed to find the optimal solution in one run out of the 30 repetitions. Despite being good optimization algorithms, the GA and DE did not perform very

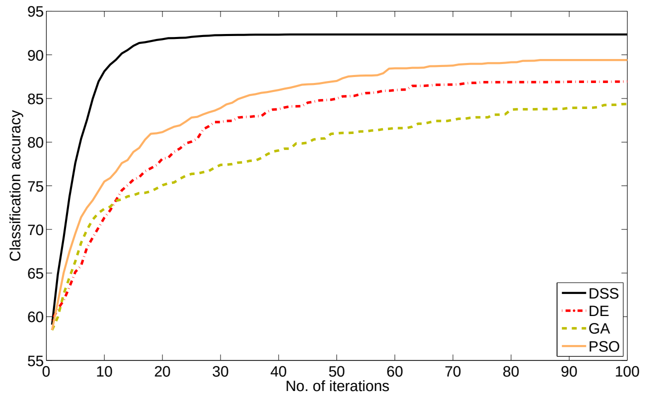

Figure 2: Number of iterations vs. classification accuracy for the selection of 5 features

well in this experiment, as they did not find the optimal solution in any run. This is mainly due to the large number of irrelevant and redundant features. The standard deviation values of the classification accuracy over the 30 runs for DSS, DE, GA and PSO are: 0.0, 3.23, 5.40 and 2.30 respectively. This indicates that GA has the highest fluctuation among the four methods, and it probably means that its performance is sensitive to the initial population when having large feature sets.

4.2. EEG Classification

The second problem involves the classification of Electroencephalogram (EEG) signals into one of pre-determined mental tasks. The data used here was obtained from the Department of Medical Informatics, University of Technology, Graz, Austria. EEG signals were recorded from three right-handed females with 56 Electrodes. The subjects were asked to imagine right or left finger movements according to stimuli on screen. The wavelet packet transform was used to extract 3 features from each channel. Hence, the total number of features extracted was 168 (56 channels $\times$ 3 features/channel). 406

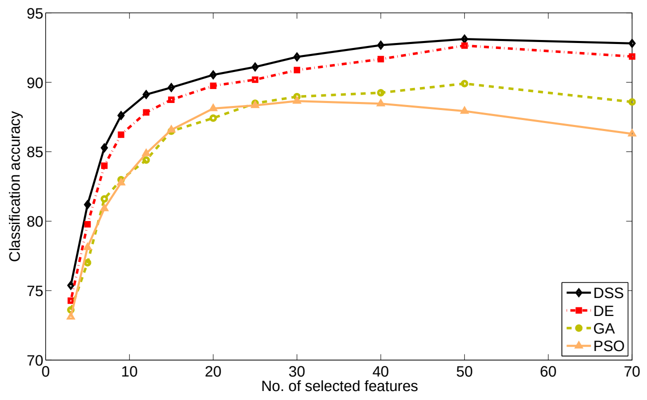

Figure 3: Number of selected features vs. classification accuracy - EEG data

trials were recorded, 208 for left movement and 198 for right. 300 patterns were used for training and the remaining 106 for testing. The desired number of selected features was varied from 3 to 70. The selection of feature subsets was performed using the four methods mentioned earlier, and the number of iterations was set to 400. The experiment was repeated 30 times and a Linear Discriminant Analysis (LDA) classifier was used. The averaged classification accuracy values of the selected subsets are shown in Fig. 3.

It is clear that DSS outperformed the other three methods in all cases, and hence it achieved excellent performance regardless of the size of the selected number of features. The second best algorithm is DE followed by GA and finally PSO. The detailed results for the selection of 3, 30 and 70 features are shown in Figs. 4, 5 and 6 respectively. For the selection of 3 features, the results indicate that DSS is the only method that converged to the optimal solution in most runs (29 out of 30). The DE, GA and PSO managed to achieve the optimal solution in 11, 10 and 1 runs respectively. The standard deviation values of the classification accuracy for DSS, DE, GA and PSO

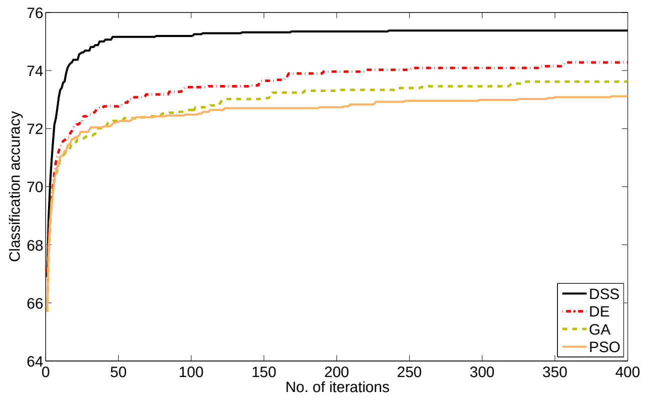

Figure 4: Number of iterations vs. classification accuracy for the selection of 3 EEG features

are: 0.52, 1.11, 1.63 and 0.98 respectively.

The search became more complicated as the number of selected features increased. The results obtained for the selection of subsets of 30 features indicate that more time was needed for the methods to find good solutions. When evaluating the runs individually, DSS managed to outperform the other three methods in 21 runs, and it shared the highest accuracy with another method in four other runs. However, DE was a bit more consistent in its performance, where the standard deviation was 0.94, 0.67, 2.05 and 1.17 for DSS, DE, GA and PSO respectively.

Searching for subsets of 70 features was even harder. On average, higher standard deviation values were recorded; 1.12, 1.20, 1.45 and 1.33 for the four methods in order, which indicate higher fluctuations in performance. As with the previous case, DSS achieved higher accuracies in more runs than any other method (19 runs as a clear winner and shared the highest with another method in seven other runs). Fig. 6 shows an interesting trend,

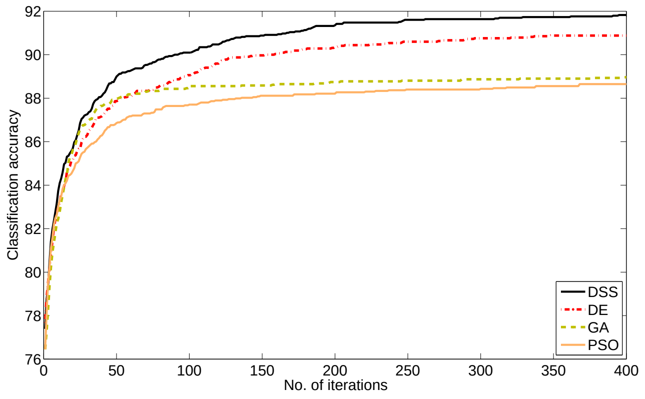

Figure 5: Number of iterations vs. classification accuracy for the selection of 30 EEG features

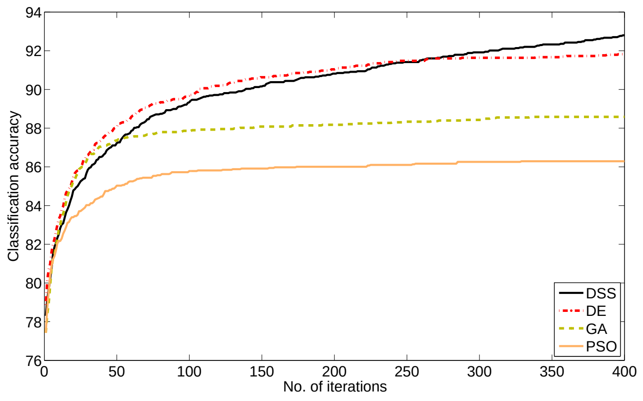

Figure 6: Number of iterations vs. classification accuracy for the selection of 70 EEG features

where unlike the other three methods that in many runs got trapped in local minimas, DSS continued to improve as the number of iterations increased. In fact, DE was a bit better than DSS in the first 250 iterations, but unlike DSS, the performance of DE has hardly improved after that. This trend of continuous improvement in the performance of DSS is mainly due to its enhanced exploration capability, as explained in sections 2 and 3.

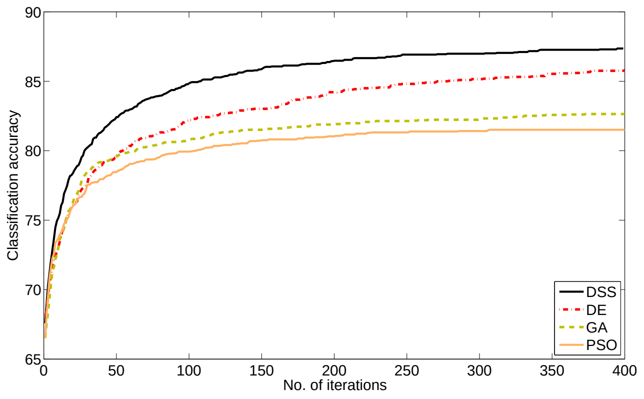

Figure 7: Number of iterations vs. classification accuracy for the selection of 30 EEG features using a kNN classifier

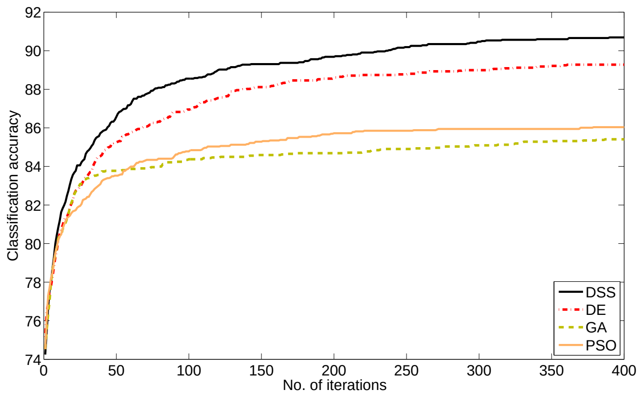

Figure 8: Number of iterations vs. classification accuracy for the selection of 30 EEG features using a SVM classifier

In addition to the LDA classifier, the k-NN and Support Vector Machine (SVM) were also considered. Results are shown in Figs. 7 and 8 for the selection of subsets of 30 features. These figures indicate that DSS clearly outperformed the other three methods, even when considering different classifiers. It is worth mentioning that k-NN and SVM did not perform as good as LDA, due to the limited number of available patterns.

4.3. Speech Segment Classification

In the third experiment, speech segments are classified according to their manner of articulation. Nine classes were considered: vowel, semi-vowel, nasal, fricative, stop, closure, lateral, rhotic, and silence. We used speech signals from the TIMIT database, where segment boundaries were identified. Three different sets of features were extracted from each speech frame: 16 log mel-filter bank (MFB), 12 linear predictive reflection coefficients (LPR), and 10 wavelet energy bands (WVT). A context dependent approach was adopted to perform the classification. So, the features used to represent each speech segment, Seg_n, were the average frame features over the first and second halves of Seg_n and the average frame features of the previous and following segments (Seg_{n-1} and Seg_{n+1} respectively). Hence, the baseline feature sets based on MFB, LPR, and WVT consisted of 64, 48 and 40 features respectively. An LDA classifier was used to classify the features of each baseline set into one of the nine manner-of-articulation classes. 4000 segments were used, the first 2000 for training and the last 2000 for validation. The obtained classification accuracy for MFB, LPR and WVT were 75.40%, 63.15% and 72.80% respectively. It is clear that MFB achieved the best performance among the three baseline sets, however, it used more features than the other two baseline sets. The three baseline feature sets were concatenated to form a new set of 152 features. The four feature selection algorithms were used to select from these features. The desired number of selected features was varied from 7 to 70. 300 iterations were used by each method, and the experiment was repeated 20 times.

The first thing that can be noticed from the averaged classification results,

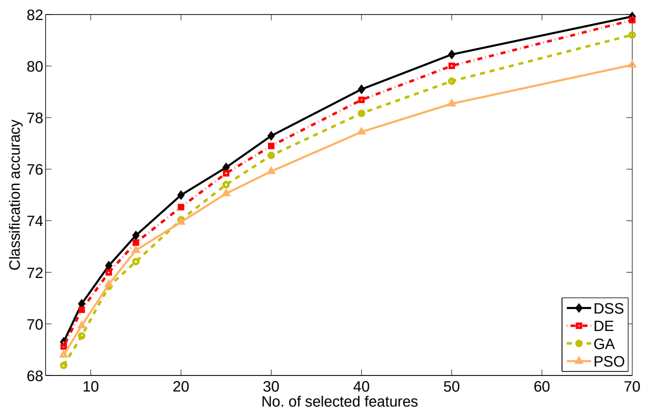

Figure 9: Number of selected features vs. classification accuracy - speech data

shown in Fig 9, is the enhanced performance that was achieved in comparison to the three baseline feature sets. For instance, with 40 features DSS managed to achieve an average accuracy of 78.82%, which is not only higher than the accuracy of the 40 wavelet feature, but it is also higher than the accuracy of the 64 MFB features. The figure also indicates that the difference in performance between the four feature selection methods is not large, however, DSS still managed to outperform the other three methods in all cases. DE achieved quite good results in this experiment, as the number of irrelevant and redundant features was not big. The detailed results for the selection of 7 features are shown in Fig. 10. As with last experiment, DSS managed to converge faster than the other methods and managed to achieve the highest accuracy in 14 runs and share the highest accuracy with another method in 5 other runs. The standard deviation values of the classification accuracy for DSS, DE, GA and PSO are: 0.10, 0.20, 0.52 and 0.47 respectively.

DSS also outperformed the other three methods when selecting 30 features. It achieved the highest accuracy in 17 runs and shared the highest

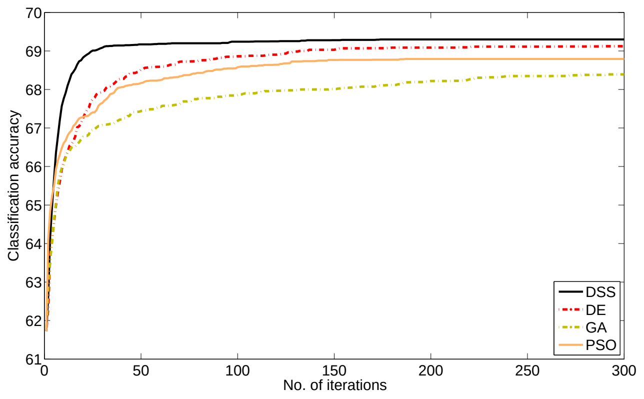

Figure 10: Number of iterations vs. classification accuracy for the selection of 7 speech features

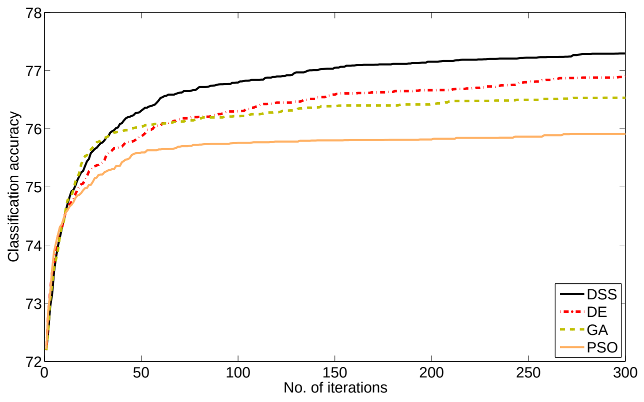

Figure 11: Number of iterations vs. classification accuracy for the selection of 30 speech features

accuracy with another method in 2 other runs. The figure also shows that unlike DE and DSS that managed to improve their performance as the number of iterations increased, GA and PSO got trapped in local minimas. The standard deviation values of the classification accuracy for the four methods in order are: 0.24, 0.26, 0.61 and 0.33.

Similar to the EEG experiment, results for the selection of 70 features,

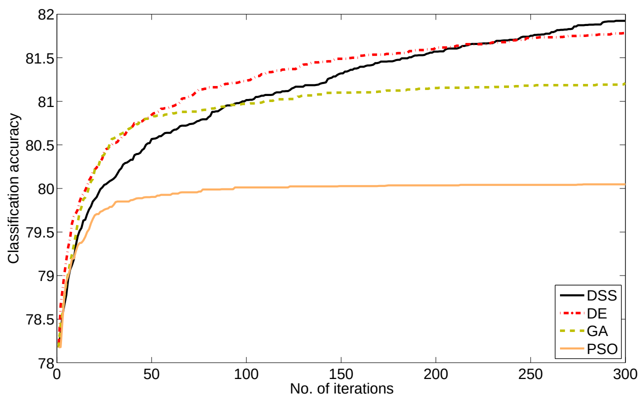

Figure 12: Number of iterations vs. classification accuracy for the selection of 70 speech features

shown in Fig. 12, reflect the enhanced exploration capability of DSS, where despite its slow start, compared to DE and GA, it continued to improve, while other methods trapped in local minimas at different locations of the search space. The figure also shows that the second best algorithm is DE, which outperformed GA and PSO with a clear margin. The standard deviation values are: 0.22, 0.22, 0.42 and 0.24 for DSS, DE, GA and PSO respectively.

The reason behind the lower standard deviation values in this experiment is the fact that there are less variations in the relevance of available features in comparison to the previous two experiments. This can be noticed when comparing the classification accuracy ranges of the different figures, for example Figs. 6 and 12.

Results for the selection of subsets of 30 features when using a k-NN classifier are shown in Fig. 13. One can notice that the k-NN classifier managed to achieve better results that LDA in this experiment. The results also show that DSS clearly outperform the other three methods, where it

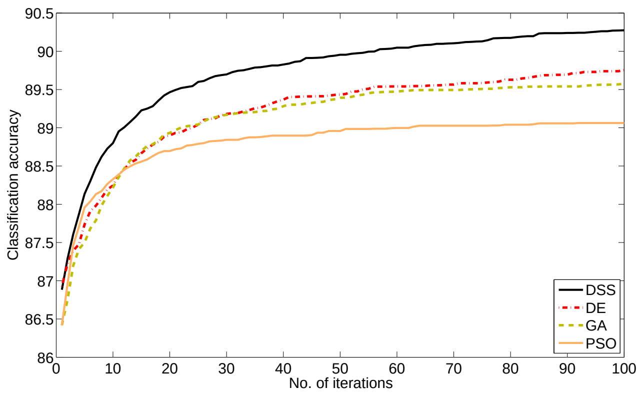

Figure 13: Number of iterations vs. classification accuracy for the selection of 30 speech features using kNN

achieved the highest accuracy in 17 out of 20 runs.

5. Conclusion

This paper presented a novel search strategy for feature selection. The proposed method, which utilizes dependency between feature pairs to guide the search, proved to be very successful in exploring the feature space and avoiding local minimas. Three different problems were used to compare the proposed method with other population-based optimization search methods. The proposed method managed to achieve a consistently high-quality performance, and outperformed the other methods in all cases. It proved to be particularly useful when dealing with high number of redundant and irrelevant features.

References

[1] I. Guyon, S. Gunn, M. Nikravesh and L.A. Zadeh *Feature Extraction: Foundations and Applications*, Springer-Verlag New York, 2006.

[2] P.M. Narendra and K. Fukunaga, "A branch and bound algorithm for feature subset selection," *IEEE Trans. Computers*, vol. 26, pp. 917-922, 1977.

[3] R. Kohavi and G. H. John, "Wrappers for feature subset selection," *Artificial Intelligence, special issue on relevance*, vol. 97, pp. 273-324, 1997.

[4] P.A. Devijver and J. Kittler, *Pattern Recognition: a statistical approach*, Prentice-Hall, Englewood CliEs, NJ, 1982.

[5] S-W. Lin, T-Y. Tseng, S-Y. Chou and S-C. Chen, "A simulated-annealing-based approach for simultaneous parameter optimization and feature selection of back-propagation networks", *Expert Systems with Applications*, vol. 34 , no. 2, pp. 1491-1499, 2008.

[6] H. Frohlich, O. Chapelle and B. Scholkopf, "Feature Selection for Support Vector Machines by Means of Genetic Algorithms," Proc. IEEE Intl. Conf. Tools with Artificial Intelligence (ICTAI'03), pp. 142-148, 2003.

[7] A. Al-Ani, "Feature Subset Selection Using Ant Colony Optimization", *Int. Journal of Computational Intelligence*, vol. 2, pp. 53-58, 2005.

[8] H.A. Firpi and E. Goodman, "Swarmed Feature Selection", Proc. Applied Imagery Pattern Recognition Workshop, pp. 112–118, 2004.

[9] R. Khushaba, A. Al-Ani and A. Al-Jumaily, ”Differential Evolution based Feature Subset Selection,” Proc. Intl. Conf. Pattern Recognition (ICPR’08), 2008.

[10] H. Peng, F. Long and C. Ding, ”Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 27, no. 8, pp. 1226-1238, August 2008.

[11] L. Yu, and H. Liu, ”Efficient feature selection via analysis of relevance and redundancy,” *Journal of Machine Learning Research*, vol. 5, pp. 1205-1224, Oct. 2004.

[12] G. Qu, S. Hariri and M. Yousif, ”A new dependency and correlation analysis for features,” *IEEE Trans. Knowledge and Data Engineering*, vol. 17, no. 9, pp. 1199-1207, Sep. 2005.

[13] A. Jakulin and I. Bratko, ”Analyzing attribute dependencies,” Proc. PKDD 2003, Lecture Notes in Artificial Intelligence, pp. 229-240, 2003.

[14] Z. Zhao and H. Liu, ”Searching for interacting features,” Proc. Intl. Joint Conf. on Artificial Intelligence (IJCAI-07), pp 1156-1161, 2007.

[15] T.M. Cover and J.A. Thomas, *Elements of information theory*, John Wiley & Sons, 1991.

[16] A.M. Fraser and H.L. Swinney, ”Independent coordinates for strange attractors from mutual information,” *Physical Review A*, vol. 33, pp. 1134-1140, 1986.

[17] G.A. Darbellay and I. Vajda, "Estimation of the mutual information by an adaptive partitioning of the observation space," *IEEE Trans. Information Theory*, vol. 45, pp. 1315-1321, 1999.

[18] N. Kwak and C-H. Choi, "Input feature selection by mutual information based on Parzen window," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 24, no. 12, pp. 1667-1671, 2002.