Evolutionary Algorithms with Linkage Information for Feature Selection in Brain Computer Interfaces

Jason Adair, Alexander Brownlee and Gabriela Ochoa

Abstract Brain Computer Interfaces are an essential technology for the advancement of prosthetic limbs, but current signal acquisition methods are hindered by a number of factors, not least, noise. In this context, Feature Selection is required to choose the important signal features and improve classifier accuracy. Evolutionary algorithms have proven to outperform filtering methods (in terms of accuracy) for Feature Selection. This paper applies a single-point heuristic search method, Iterated Local Search (ILS), and compares it to a genetic algorithm (GA) and a memetic algorithm (MA). It then further attempts to utilise Linkage between features to guide search operators in the algorithms stated. The GA was found to outperform ILS. Counter-intuitively, linkage-guided algorithms resulted in higher classification error rates than their unguided alternatives. Explanations for this are explored.

1 Introduction

Brain Computer Interfaces (BCI) are of special benefit to individuals with motor difficulties; offering a range of devices including prosthesis and voice synthesisers. A typical BCI system works by converting the incoming neural signals into a set of numerical values known as features. These are passed to a classifier which then determines the appropriate action to take. However, this process is subject to a large amount of noise, and some features are irrelevant or meaningless. To overcome this, Feature Selection is required; selecting only the most relevant features from the dataset and discarding those that may have distorted the results, or slowed the system. When attempting to choose appropriate features, each additional potential feature causes exponential growth of the solution space, presenting a non-trivial problem, prohibiting the use of an exhaustive search. This is also known as the

Jason Adair, Alexander Brownlee and Gabriela Ochoa
Department of Computing Science and Mathematics, University of Stirling, Stirling, FK9 4LA, Scotland e-mail: {jad,sbr,goc}@cs.stir.ac.uk

Curse of Dimensionality. Due to this, intelligent search techniques must be applied. There are two primary divisions of feature selection techniques; filters and wrappers. Filters utilise an unsupervised ranking method and have no reliance on the classification stage of decoding, instead judging each individual feature on the basis of its relevance. Unlike filters, wrappers do not rank features, but instead evaluate subset effectiveness for classifier training. This allows the classifier to serve as a fitness function for an approach such as an EA. This results in a longer training process, but since BCIs are typically trained offline, the increase in classification accuracy takes precedence.

While many different algorithms have been applied to the problem of feature selection in BCI, they often lack the ability to specifically exploit relationships that may exist between features. A technique that has been useful in other feature selection problems has been to utilise linkage information [8, 22]. By measuring the distance in fitness between features, it has been demonstrated that it can be used as an indication of the overall fitness of a solution; therefore suggesting that 'linkage-aware' operators can increase the performance of an algorithm. This paper proposes a method in which operators in evolutionary algorithms can be guided using linkage to increase the classification accuracy of EEG data. To this end, we initially compare four base algorithms: hill-climber, Iterated Local Search (ILS), Genetic Algorithm (GA), and Memetic algorithm (MA). Thereafter we incorporate linkage into both hill-climbing and ILS. These techniques were applied to the dataset provided by the the second Berlin BCI competition; track three (motor-imagery). We also explored potential explanations for the behaviours observed. The main contribution of this paper is to assess the viability of guiding evolutionary algorithms for the feature selection phase of brain computer interfaces, using information extracted from pairwise interactions (linkage) within solutions.

2 Background and related work

In this section we discuss Brain Computer Interfaces (BCI) in more detail, elaborating on possible noise sources. We then describe other works involving evolutionary algorithms (EAs) in feature selection. Finally, we discuss the origins of *linkage* in EAs.

2.1 Brain Computer Interfaces

There are a range of different ways in which electrical activity from the brain can be recorded, but by far the most popular is Electroencephalography (EEG). EEG involves the placement of electrodes on the scalp surface, measuring the electrical fields of the neural matter below, and relaying it back to a computer for processing. This technique has become prominent over other more invasive methods due

to its ease of maintenance, its lack of invasive procedures (substantially safer), and relatively low cost. However, it does present some very non-trivial problems; as the electrodes that detect the electrical fields are placed on the scalp, the signal must be powerful enough to penetrate two to three centimeters of cranium, skin and other biological material. For this level of energy to be generated, approximately six square centimeters of neural matter must be active (in the region of one hundred million neurons [25]), resulting in low spatial resolution, contamination of signals between electrodes, and natural band passing of the frequencies when passing through the skull. The signal is further distorted by additional electrical signals being detected from eye movements (electro-oculography), muscle movements (electromyography), and environmental noise (the fifty hertz band often consists of electrical activity from nearby wall sockets [21]).

Despite measurements grouping substantial quantities of neurons together, we are still faced with a great deal of information for processing. Most of this information is redundant due to the aforementioned issues, and many of the activities we wish to decode are region specific, only activating a subset of the electrodes. This means that the classifier is presented with substantial quantities of extremely noisy and redundant data, causing slow and often poor results.

2.2 Evolutionary Algorithms for BCI and Feature Selection

Evolutionary Algorithms have proven highly successful in the feature selection field [5, 7]. The typical approach is to use classifier accuracy as the fitness function: the EA begins by generating solutions and splitting the training set into 2 subsets. The classifier is then trained using the first subset, and its ability to correctly identify the labels of the second subset is used to derive the solution fitness.

Genetic algorithms are often used as search methods used in feature selection in BCIs [24]. While they are somewhat more computationally demanding, offline learning of classifiers allows us to focus on improving accuracy at the expense of speed. During their earlier implementations, standard genetic algorithms reported results that produced classification accuracies of around 74-76% [17] but have since been refined to produce in excess of 90% classification accuracy [24] (in two class problems, such as Yes or No and Left or Right). This superior performance over filter methods is further supported by [10] who reported a substantially lower rate of classification error for GA than seen in Recursive Feature Elimination, Across-Group Variance and RELIEF, a trait that appears fairly consistent across the literature. The substantial increase in classification accuracy obtained from genetic algorithms has arisen largely from adapting the generalised operators to better suit the BCI arena. Rejer [24] notes that it is possible to modify a GA to lean towards improving accuracy of the classifications while minimising the number of features in the solution, finding a tradeoff between a slight decrease in accuracy and a significant decrease in the solution size and training time. To realise this, they modified the mutator function to behave in a similar fashion to forward selection; preserving

the GAs ability to explore the solution space while giving precedence to the smaller feature sets observed in the SFFS method.

2.3 Linkage in EAs

In evolutionary algorithms, *linkage* is a relationship or dependency between decision variables. As far back as 1975, Holland [15] suggested that operators aware of linkage information might be necessary for efficient GA search. The linkage model used by an EA can be implicit (e.g. linkage learning GA [11]) or explicit (e.g. multivariate Estimation of Distribution Algorithms [12, 20]). Interest in approaches that explicitly make use of the linkage and the structure that it imposes on the search space remains of current interest, for example [3, 9, 26].

However, it has also been shown [1, 4, 13] that some aspects of linkage are inessential for fully ranking all solutions to a problem and locating the global optima. Indeed, including such inessential dependencies in the problem model used by the algorithm can hamper performance [2, 19, 23]. This concurs with our findings for the linkage-aware ILS when applied to the feature selection problem studied in our experiments.

3 Methodology

In this section, we will introduce the methodologies used in our experiments. First, we will describe the dataset, the means by which the features were extracted from it, and how the classification model was constructed. Then we will introduce our definition of linkage for the feature selection problem, and how it was computed for our data set. Finally we will describe the search algorithms that we applied to the problem of Feature Selection, giving precedence to those that use less complex operators, emphasising the effects of utilisation of Linkage information. Additionally, Memetic and Genetic algorithms are included as base comparisons.

3.1 Dataset

The sample data used was provided by the second Berlin BCI competition, in which dataset three was selected (motor-imagery) [1] as previous studies have shown it to be a clearly defined, yet challenging dataset. The recorded task involved having a participant utilize motor imagery of right and left hand movements to control an on-screen bar. Over a nine second trial (T=9); the first two seconds had no required ac-

[1] `http://www.bbci.de/competition/iii/#datasets`

tions, at t=2 to t=3, an acoustic signal was given and a cross was displayed onscreen to focus the participants attention. At t=3, an arrow was immediately displayed indicating the hand which the participant was to visualise moving. This dataset was recorded from three EEG channels, placed on the scalp of a single participant and filtered between 0.5 and 30Hz. This was repeated over 7 runs, with 40 trials in each, which provided 140 training and 140 test trials.

3.2 BCI Model

In our experiments, a model BCI was created using MATLAB. The overall architecture for this is given in Figure 1. While the entire model was required for experimentation purposes, this paper's contributions reside within the 'Feature Selection' phase.

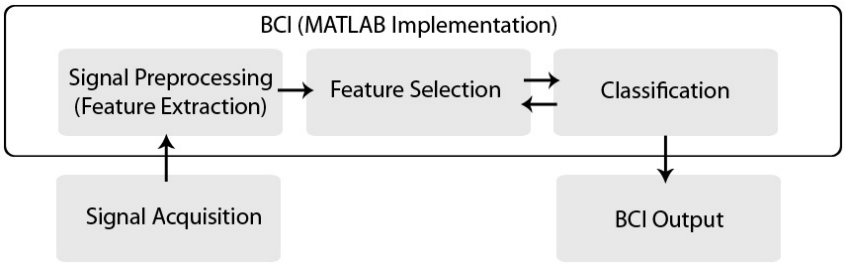

Fig. 1 Signal Acquisition was performed by the Department of Medical Informatics, Institute for Biomedical Engineering, University of Technology Graz. Feature extraction (Power Spectral Density), feature selection algorithms and classifiers were all implemented in MATLAB

3.2.1 Signal Preprocessing (Feature Extraction)

Rejer [24] achieved a high degree of success with this dataset using Power Spectral Density (PSD) and for this reason, the same Feature Extraction methodology was adopted. Each channel was decomposed into 12 further frequency bandings; 2 primary bandwidths of 8-13Hz (alpha) and 13-30Hz (beta), and 5 sub-bands within each (8-9, 9-10, 10-11, 11-12, 12-13Hz and 13-17, 17-20, 20-23, 23-26, 26-30Hz). To extract features for classifier training, the PSD was calculated for each second, of each frequency, in each channel. This resulted in 324 numerical features, which were referenced sequentially, as displayed in the appendix (Table 2).

3.2.2 Feature Selection and Encoding

As displayed in Rejer's paper [24], 6 PSD features were sufficient for a high classification accuracy, therefore we used a fixed solution length of 6 integers, each representing one selected feature. Several different algorithms were implemented including: Hill-climbing, Genetic Algorithms, Iterated Local Search, Memetic Algorithms, and variations of each in which Linkage was used to influence the operators. These are described in more detail in Sections 3.4-3.6.

Figure 2 displays the data flow within the Feature Selection phase of the metaheuristics with Linkage. At **(1)**, the training data is used to create a mapping of all pairwise linkages within the featureset, which is then passed to the metaheuristic. The metaheuristic **(2)** then selects features and performs cross-validation using the training data. The fitness returned by the cross-validation is then used by the metaheuristic to guide the next iteration of feature selection. After stopping criteria have been met, the Feature Selection phase is ended, and the selected features at that point are passed on to be used on the testing data **(3)**. As this is a black box optimisation problem, the classifier accuracy was utilised as a fitness function.

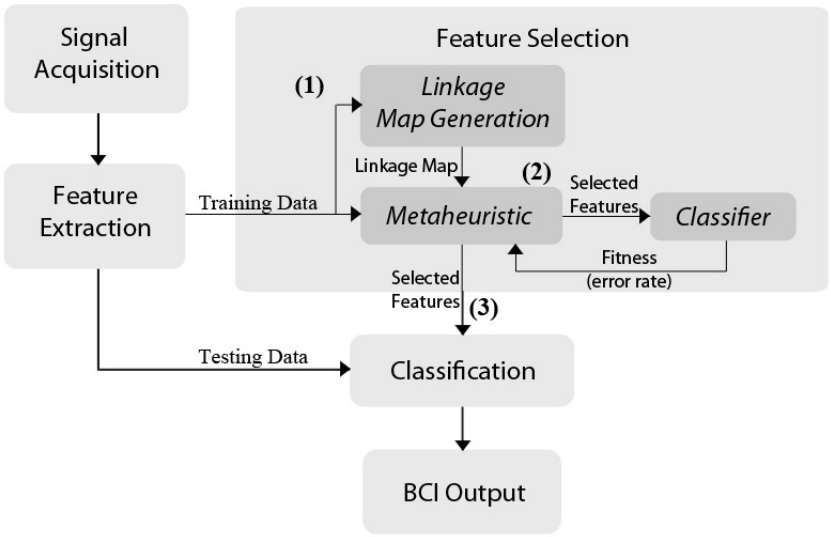

Fig. 2 Sequence diagram displaying the incorporation of Linkage in the Feature Selection phase

3.2.3 Classifier

Selected features were extracted from each of the trials and used to train a Support Vector Machine (SVM). The SVM was chosen after pretrial experimentation in which it consistently outperformed the K-Nearest-Neighbour classifier. A notable issue with SVMs are their tendency to over-fit data, leading to poor generalisability

- this was addressed by using 10 fold cross validation of the training set in the fitness function.

3.3 Feature Linkage

Linkage between features was determined by applying the Linkage Detection Algorithm [14] to the training data. The algorithm starts with no features selected: the classifier accuracy f_ϕ is determined. The accuracy f_a is calculated having selected only feature a. From this we have a change in accuracy from the baseline δ_a = $f_a - f_\phi$. This is then repeated to find δ_b when selecting only feature b, and δ_{ab} when selecting features a and b. For a pair of features a and b, the change in classifier accuracy is measured while selecting the two features separately δ_a, δ_b and both together δ_{ab}. We call the difference in these changes in accuracy the *Linkage Score*, $s_{ab} = \delta_{ab} - (\delta_a + \delta_b)$. If s is non-zero, there is deemed to be linkage between the variables. This method can be expanded to higher levels of interaction but its complexity grows rapidly with the level of interaction.

The Linkage Score was calculated for every pair of the 324 features. Dependencies (linkage) were classified as benign and malign in [16]. *Benign* linkage is that for which the combined change in fitness is in the same direction as the independent changes (i.e. the signs of $\delta_a + \delta_b$ and δ_{ab} are the same). *Malign* linkage shows a combined change in the opposite direction to the independent changes (i.e. the signs of $\delta_a + \delta_b$ and δ_{ab} are the opposite). We adopt these terms in the following way. If a pair should yield a positive Linkage Score, it reflects an increase in error rate over the combination of the individual scores and is deemed 'malign'. A negative score suggests that there is a reduction in error rate when the features are combined and is hence a 'benign' linkage. We would expect a 'good' solution to include low levels of malign linkage, and high levels of benign linkage. We designed our operators accordingly.

3.4 Iterated Local Search

Iterated Local Search (ILS) is a little explored algorithm in BCI and to the authors best knowledge, has not been tested on feature selection for EEG. It has been selected as it is less convoluted than other EA methods, lacking the need for a population or cross-over, which should help emphasise the effects of the guided mutation operator. In essence, it is a nested hill-climbing algorithm; in a traditional hill-climber, a small mutation, replacing a selected feature with an unselected one, is performed on the initial solution to create a new potential solution. This new solution is scored via a fitness function and then accepted if it is deemed to be fitter than the initial solution. This process is repeated to find a more optimal solution, but often becomes trapped in local optima. In an ILS, a 'kick' is performed by mutating

a large portion of the solution (3 of the 6 features in this case). A hill-climber is then performed on this new, heavily mutated solution, and the resulting solution from it is then compared to the original, pre-kicked featureset.

3.5 Genetic Algorithm

Genetic Algorithms (GA) are powerful tools in optimisation problems and have demonstrated considerable results in feature selection for BCIs [17]. An initial population of potential solutions is (typically randomly) generated with each solution consisting of a chain of features known as genes. After initialisation, genetic algorithms utilise three operators; selection, crossover and mutation [17]. The selection operator is modelled on the principle of natural selection in which the fittest organisms will survive to pass on their genes. This is achieved by selecting the fittest individuals within the population (best solutions generated) via an objective function and using their components to create the next generation. The crossover operator then recombines the selected solutions to form the next generation, often a single point is randomly selected and pairs of individuals swap their genes after this point, allowing the search space to be explored. The limitation here, is that only the original randomly selected elements can be combinatorially explored, ignoring the rest of the search space. To combat this, a mutation operator is introduced; one or more genes in the solution are randomly selected and replaced with alternative genes selected at random from the entire feature space. This not only widens the scope of the exploration, but also helps prevent the algorithm becoming trapped in local optima [21].

3.6 Memetic Algorithms

Memetic algorithms have recently been used, and proven to be a viable technique in a range of feature selection problems [18]. One of the caveats with genetic algorithms is that they lack a mechanism which allows exploration of the immediate search space surrounding the currently selected solutions, something that memetic algorithms have sought to overcome by integrating a local search technique into the overall meta heuristic. This is achieved through a hybridised genetic algorithm, in which a random mutation hill-climbing search is performed on each of the newly created offspring before returning them to the population [6].

3.7 Linkage Integration

For the purposes of exploring how linkage could be exploited by the algorithms, 100 repeat experiments were carried out for each of 6 variations of a 1000 iteration hill-climbing algorithm with single point mutation. Each repeat experiment began by randomly generating a single solution; all algorithms were seeded with this same solution. Each of the proposed operators considers linkage among the selected features in a solution, and whether replacing a feature increases or decreases this.

3.7.1 Hill-Climbing Algorithm with Linkage Integration

H1. Basic Hill-climbing Algorithm - A simple hill-climber in which the mutation point and a replacement feature were both randomly selected was required as a control.

H2. Selection of Mutation Point - Target Most Malign Feature Pair - All pairs of selected features within the current solution are compared. One of the features in the pair that reflects the largest malign linkage score is selected at random for deselection and replacement with another feature chosen at random.

H3. Selection of Mutation Point - Target Most Malign Feature - Both features of the pair with the largest malign linkage score in the solution are compared with the other selected features in the solution. The feature with the most malign linkages is deselected and replaced with an unselected feature chosen at random.

H4. Selection of Mutation Point - Spare the Most Benign Pair - The mutation point is chosen at random, but the feature pair within the solution that have the largest benign linkage score are excluded from possible mutation.

H5. Selection of Replacement - Good Mutation - A feature is chosen for deselection, and 20 features are chosen at random from the unselected features as potential replacements. Each of these potential replacement features are paired with the remaining solution features, and the one with the highest benign linkage score is selected.

H6. Selection of Replacement - Best Mutation - As in the 'Good Mutation' condition, but *all* unselected features are assessed as potential replacement candidates.

H7. Selection of Mutation Point - Target Most Benign Feature - To ensure that we were using the linkage information appropriately, a counter-intuitive method which deselected the feature with the most benign linkage scores with other selected features was also used.

3.7.2 Iterated Local Search with Linkage Integration

To explore the exploitation of linkage information in a more sophisticated algorithm, Iterated Local Search was selected for modification. ILS has a two tiered iterative structure, from which we chose to provide guidance to the 'kick' function.

For each selected feature in the solution, we calculate its mutual linkage (ML); the mean linkage score between that feature and the other selected features in the solution. The 3 features with the highest ML were retained in the solution, and the remaining 3 were removed and replaced with randomly selected features.

Two variations of this method were tested:

I1 - Benign-preservation - The ML was computed using only benign linkage scores between features.

I2 - Malign-preservation - The ML was computed using only malign linkage scores between features.

3.8 Experimental Parameters

In this section we outline the parameters that were used to govern the execution of the experiments.

3.8.1 Termination Criteria

All algorithms were restricted to 100,000 evaluations of the classifier.

Hill-Climbing Algorithm - As each iteration requires only one evaluation of the solution, 100,000 iterations were used.

Iterated Local Search - Preliminary tests suggested that 'kicks' were only required when the inner hill-climber became stuck in a local optimum. This was demonstrated by higher performance in experiments which used 1000 hill-climb iterations and 100 kicks, compared with 100 iterations and 1000 kicks. Hence, the former was selected for comparison.

Genetic Algorithm - This used an initial population of 20 random solutions, and Tournament Selection with a tournament size of 2. Random single point crossover and single point mutation were used to generate offspring. A steady state model was used: a pair of offspring replacing the losing solutions in each tournament. They were then scored using the classifier before being added to the population for future tournaments - this resulted in only 2 runs of the classifier per iteration and therefore

the GA ran for 50,000 iterations.

Memetic Algorithm - The conditions for the MA were identical to the Genetic Algorithm, with allowances made to incorporate a hill-climbing phase. After the initial population was generated, a 100-iteration-hill-climber was applied to each solution, as were subsequent offspring generated. This resulted in 200 evaluations per generation, with an additional 1980 evaluations to initialise the population (100 iterations for each member, minus the evaluations of the initial population).

4 Results and Discussion

In this section we report on the results found from the comparison of the hill-climber, ILS, Genetic and Memetic Algorithms. This is followed by a visualisation of the linkage found in the feature sets and the results of their application to both the Hill-climber and ILS algorithms. The resulting solutions are then analysed for explanations of the trends observed.

4.1 Algorithm Performance Comparison

For the purposes of selecting an appropriate algorithm for modification to exploit linkage information, preliminary tests were performed. After 30 runs of each algorithm, genetic algorithms were found to produce consistently lower error rates, closely followed by Iterated Local Search (Figure 3). The hill-climber performed inconsistently, producing inferior solutions to the other techniques.

We deemed Iterated Local Search to be the preferable choice for modification as it produces solutions that are competitive with those of the Genetic Algorithm, but does does not require a cross-over operator which might imply linkage.

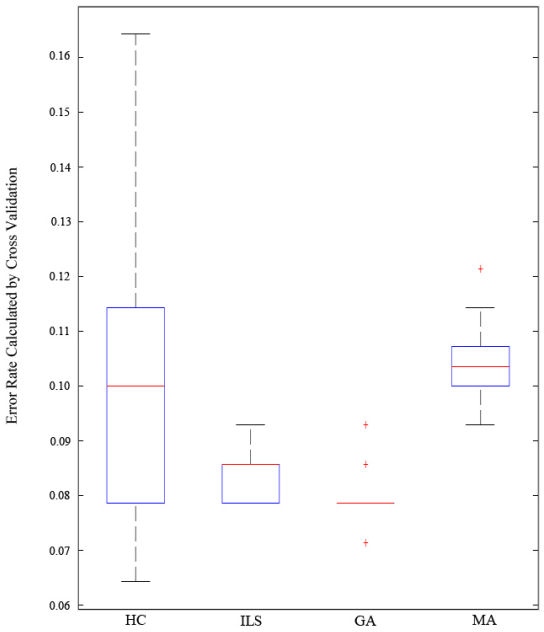

Fig. 3 Box plots comparing the error rates of solutions found by each algorithm over 30 runs.

4.2 Feature Interaction

In the algorithm selection experiment, 104 solutions with an error rate of less than 10% were found. As these are high quality solutions, they were analysed for indications of feature linkage by comparing the number of times each feature was selected among the good solutions, with the accuracy of the classifier when presented with only that feature. A plot of feature selection rates for the features, sorted by descending classifier accuracy is given in Figure 4. Given that the most predictive features were the most commonly selected, it would suggest that individual abilities are highly important for feature selection in this problem. It should be noted however, that there are significant gaps in the selected feature space, suggesting some feature linkage and that simply choosing the most predictive individual features would be a less than optimal approach. It is this interaction that we sought to detect and exploit in the following experiments.

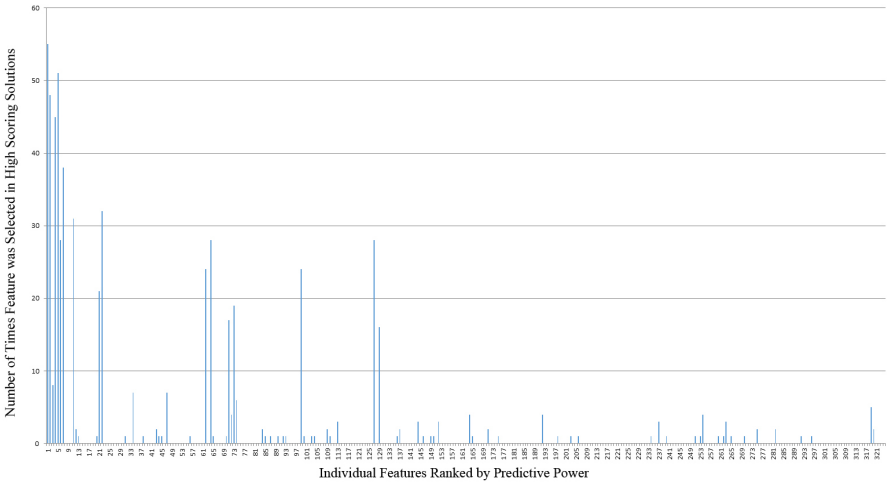

Fig. 4 Each feature was independently tested as a single feature solution to train the classifier and cross validation was performed. This allowed features to be 'ranked' according to their individual power. 104 solutions with $<10\%$ error rate were detected in the earlier experimental phase and the occurrence of each feature was tallied.

4.3 Linkage

The linkage score (as described in Section 3.3) was calculated for all pairings of the 324 features and is illustrated in the heat map in Figure 5. Heat maps showing only benign and malign linkage scores are also provided (Figures 6 and 7). Darker regions represent strong levels of linkage, lighter regions being weakly linked. Linkage scores were more pronounced in the broader frequency ranges, as seen in Figure 5: features 1 to 27 and 163 to 189 have clear bonds, showing that these features are strongly linked. This was especially noticeable in the malign linkage scores map, Figure 7. The information presented by these maps was then provided to the linkage exploitation algorithms in the following experiments.

Fig. 5 Linkage scores between all potential feature pairings (a_m,b_n) extracted from the Berlin BCI II Competition IV Dataset. Darker points represent stronger linkage between pairs of features. Each axis represents a concatenation of all 324 features; each of the 12 frequency bands (f_i) consists of the Power Spectral Densities extracted from 9 time points (t_j) simultaneously recorded over 3 channels (c_k). The lower section of the image demonstrates the breakdown of the features within a frequency band. For a detailed list of feature properties, see Table 2 (appendix).

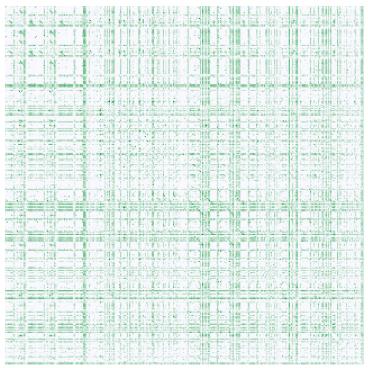

Fig. 6 Figure 5 filtered to display only benign linkage

Fig. 7 Figure 5 filtered to display only malign linkage

4.4 Hill-Climbers with Linkage

Figure 8 shows boxplots for the error rates of the final solutions found by the hillclimber, using the seven different mutation operators (with and without linkage guidance - as described in 3.7.1). Counter-intuitively, using the linkage-guided mutation operators appears to hinder the performance of the simple hill-climbing algorithm in all conditions. Notably, operator H6 (ensuring that the mutated feature is replaced by one that causes the most benign linkage within the solution) returns, by a large margin, the worst results. Two operators that produced solutions competitive with those of the unguided algorithm (H1) were both related to the target of the mutation operator; selecting the most benign (H7) and most malign (H3) features from within the solution. This suggests that high degrees of linkage, albeit benign or malign, may be harmful to the fitness of a solution.

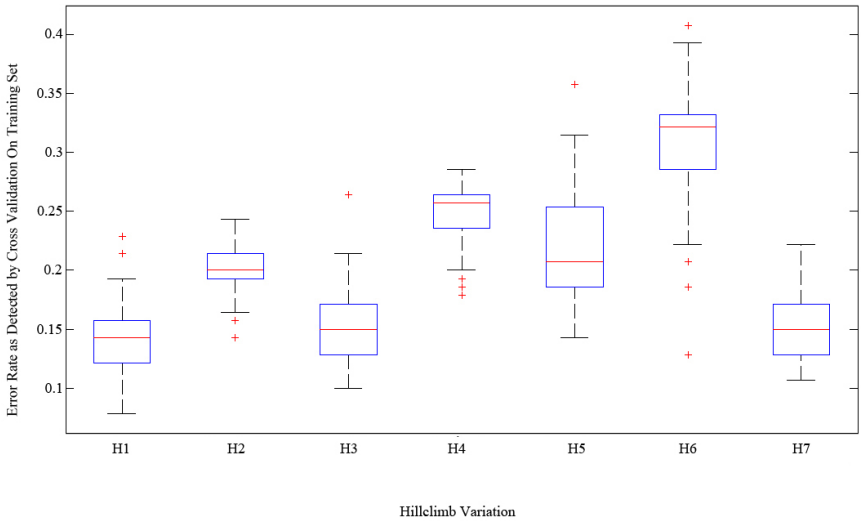

Fig. 8 Preliminary testing of different methods of linkage guidance in hill-climbing algorithms

4.5 ILS with Linkage

When considering the preliminary testing phase in which linkage was used to exploit hill-climbing algorithms, it appears that selection of the mutation targets in a solution may be beneficial, and that interfering with the selection of their replacements is detrimental. This led us to choosing a modified 'kick' phase, in which only the targets for mutation were manipulated. The results of these tests are displayed in Figure 9. Performance of the guided and unguided ILSs were not found to be statistically significantly different (analysis performed by a two-tailed t-test, $p > 0.05$).

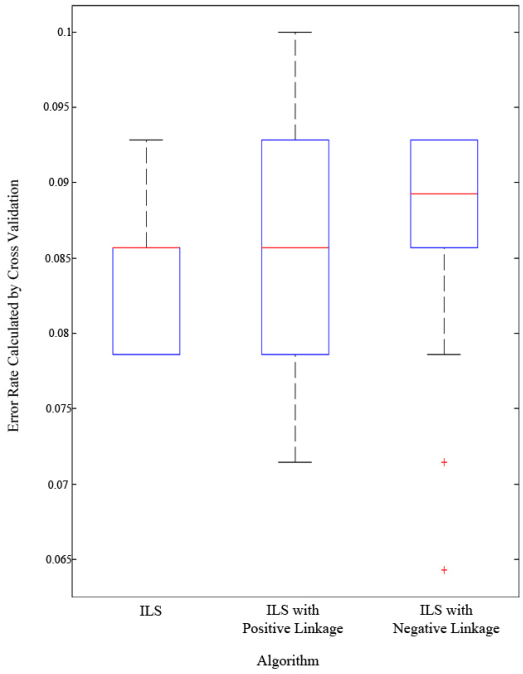

Fig. 9 Comparison of error rates obtained by Iterated Local Search, and Iterated Local Search with guidance via positive and negative linkage

5 Analysis

To further explore the reasons as to why linkage exploitation did not prove effective in the previous experiments, we performed further analysis on the most optimal solutions found over the course of this paper. For each solution, 3 scores were calculated; the Cross Validation Error (CVE) from the training set, the predictive accuracy from the testing set and what we term the 'intra-solution linkage' score. This score quantifies the strength of the linkages between features within a solution by summing the mutual linkage scores for each selected feature, as described in section 3.7.2. It is a measure of how much linkage is present between the selected features in a solution.

Table 1 Table comparing the correlation between the predictive accuracy of solutions on the test set, with the cross validation error rates and intra-solution linkage scores generated from the training set

		Score Derived from Training Set	
		Predictive Accuracy	**Intra-Solution Linkage Score**
Cross Validation Error on Training Set	**15-50%**	0.7543	-0.2263
	<15%	0.2411	-0.4296

Table 1 shows the Pearson's correlation coefficients between the solutions' predictive accuracy on the test set, and the measurements derived from the training set; intra-solution linkage and cross-validation error. The solutions are divided into 2 groups; low quality solutions (15-50% error rates on the test set - drawn from all stages of the runs) and high quality solutions (>15 % error rates on the test set - solutions found in the final stages of the runs). For the low quality solutions, the correlation between CVE and predictive accuracy is 0.7543. This drops to 0.2411 in the higher quality solutions, which we suspect is due to over-fitting of the test data. The correlation between the intra-solution linkage score and predictive accuracy scores for low quality solutions is low (-0.2263), but unlike CVE, the correlation increases in the higher quality solutions (-0.4296). It appears that intra-solution linkage scores may be a better indicator of the generality of solutions than CVE in higher quality solutions (later stages of search algorithms).

In summary, 'good' solutions (that is, with a low error rate on the validation data) have no, or, weak linkage between their selected features. Overfitted solutions (this is, those with low error rate on the training data but high error rate on the validation data), tend to have stronger linkage between their selected features.

6 Conclusion

While Iterated Local Search found slightly less fit solutions than the genetic algorithm, it has a number of advantages; as it is not a 'population-based approach', it requires less memory. This is especially relevant when considering Brain Computer Interfaces are often portable devices where size and power requirements are paramount. It is also comparatively simple, requiring less fine tuning of parameters.

The integration of linkage information in the evolutionary algorithms described in this paper provided no significant improvement in the results. When we consider the computational load required to calculate the linkage scores in advance, we would not recommend this form of implementation in real world systems. This is not to say that linkage should be dismissed as a form of guidance in BCI; while this paper failed to find a successful application, it was based on only one dataset. It should be noted that further analysis on solutions found by the evolutionary algorithms shows that the correlation between the training set's cross validation error rate, and prediction accuracy, declines in the higher scoring solutions. While this is something that we fully expect as over-fitting occurs, more interestingly, the negative correlation between the solutions predictive accuracy on the test set and the linkage scores within these solutions (derived from the training set) actually increases. This makes sense: we might expect that the classifier would be able to gain more information from features that are not linked (or correlated with each other) than those that are. This suggests that it may be possible to mitigate some of the effects of over-fitting by developing a multi-objective fitness function that gives increasing weight to the solutions that minimise linkage, while also continuing to minimise cross validation error rates.

Acknowledgements Work funded by UK EPSRC grant EP/J017515 (DAASE).

Appendix

Table 2 Table displaying each feature referenced by the channel, second and frequency from which it was extracted

Channel:		**1 (C3)**									**2 (Cz)**									**3 (C4)**								
Epoch (Second):		**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
Frequencies (Hz)	**8-13**	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
	8-9	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54
	9-10	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81
	10-11	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100	101	102	103	104	105	106	107	108
	11-12	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	128	129	130	131	132	133	134	135
	12-13	136	137	138	139	140	141	142	143	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159	160	161	162
	13-30	163	164	165	166	167	168	169	170	171	172	173	174	175	176	177	178	179	180	181	182	183	184	185	186	187	188	189
	13-17	190	191	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207	208	209	210	211	212	213	214	215	216
	17-20	217	218	219	220	221	222	223	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239	240	241	242	243
	20-23	244	245	246	247	248	249	250	251	252	253	254	255	256	257	258	259	260	261	262	263	264	265	266	267	268	269	270
	23-26	271	272	273	274	275	276	277	278	279	280	281	282	283	284	285	286	287	288	289	290	291	292	293	294	295	296	297
	26-30	298	299	300	301	302	303	304	305	306	307	308	309	310	311	312	313	314	315	316	317	318	319	320	321	322	323	324

References

1. A. E. I. Brownlee, J. A. W. McCall, and L. A. Christie. Structural coherence of problem and algorithm: An analysis for EDAs on all 2-bit and 3-bit problems. In *Proc. IEEE CEC*, pages 2066–2073, Sendai, Japan, 2015. IEEE Press.
2. A. E. I. Brownlee, M. Pelikan, J. A. W. McCall, and A. Petrovski. An application of a multivariate estimation of distribution algorithm to cancer chemotherapy. In *Proc. GECCO*, pages 463–464, Atlanta, GA, USA, 2008. ACM Press.
3. A.E.I. Brownlee, J.A.W. McCall, and Q. Zhang. Fitness modeling with Markov networks. *IEEE T. Evolut. Comput.*, 17(6):862–879, 2013.
4. Alexander E. I. Brownlee, John A. W. McCall, and Martin Pelikan. Influence of selection on structure learning in Markov network EDAs: An empirical study. In *Proc. GECCO*, pages 249–256. ACM Press, 2012.
5. Alexander E. I. Brownlee, Olivier Regnier-Coudert, John A. W. McCall, Stewart Massie, and Stefan Stulajter. An application of a GA with Markov network surrogate to feature selection. *Int. J. Syst. Sci.*, 44(11):2039–2056, 2013.
6. E. K. Burke, J. P. Newall, and R. F. Weare. *Practice and Theory of Automated Timetabling: First International Conference Edinburgh, U.K., August 29–September 1, 1995 Selected Papers*, chapter A memetic algorithm for university exam timetabling, pages 241–250. Springer Berlin Heidelberg, Berlin, Heidelberg, 1996.
7. Girish Chandrashekar and Ferat Sahin. A survey on feature selection methods. *Comput. Electr. Eng.*, 40(1):16–28, January 2014.
8. Benhui Chen and Jinglu Hu. *Exploitation of Linkage Learning in Evolutionary Algorithms*, chapter Protein Structure Prediction Based on HP Model Using an Improved Hybrid EDA, pages 193–214. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.
9. Francisco Chicano, Darrell Whitley, and Andrew M. Sutton. Efficient identification of improving moves in a ball for pseudo-boolean problems. In *Proceedings of the 2014 Conference on Genetic and Evolutionary Computation*, GECCO '14, pages 437–444, New York, NY, USA, 2014. ACM.
10. N. S. Dias, L. R. Jacinto, P. M. Mendes, and J. H. Correia. Feature down-selection in brain-computer interfaces dimensionality reduction and discrimination power. *2009 4th International IEEE/EMBS Conference on Neural Engineering, NER '09*, pages 323–326, 2009.
11. Georges R Harik and David E Goldberg. Learning linkage. In *FOGA*, volume 4, pages 247–262, 1996.
12. Mark Hauschild and Martin Pelikan. An introduction and survey of estimation of distribution algorithms. *Swarm Evol. Comput.*, 1(3):111 – 128, 2011.
13. Mark Hauschild, Martin Pelikan, Claudio F. Lima, and Kumara Sastry. Analyzing probabilistic models in hierarchical BOA on traps and spin glasses. In *Proc. GECCO*, pages 523–530. ACM Press, 2007.
14. Robert B. Heckendorn and Alden H. Wright. Efficient linkage discovery by limited probing. *Evol. Comput.*, 12(4):517–545, 2004.
15. John H. Holland. *Adaptation in natural and artificial systems*. University of Michigan Press, Ann Arbor, 1975. by John H. Holland.; Includes index.; Bibliography: p. 175-177.
16. L. Kallel, B. Naudts, and R. Reeves. Properties of fitness functions and search landscapes. In L. Kallel, B. Naudts, and A. Rogers, editors, *Theoretical Aspects of Evolutionary Computing*, pages 177–208. Springer, 2000.
17. Marcin Ko, Andrzej Majkowski, and Remigiusz Rak. Implementation of genetic algorithms to feature selection for the use of brain-computer interface. (5):71–73, 2011.
18. Jaesung Lee and Dae-Won Kim. Memetic feature selection algorithm for multi-label classification. *Information Sciences*, 293:80–96, 2015.
19. Claudio F. Lima, Fernando G. Lobo, Martin Pelikan, and David E. Goldberg. Model accuracy in the Bayesian optimization algorithm. *Soft Comput.*, 15(7):1351–1371, 2010.
20. J. A. Lozano, P. Larrañaga, I. Inza, and E. Bengoetxea. *Towards a New Evolutionary Computation: Advances on Estimation of Distribution Algorithms (Studies in Fuzziness and Soft Computing)*. Springer-Verlag, 2006.

21. Luis Fernando Nicolas-Alonso and Jaime Gomez-Gil. Brain computer interfaces, a review. *Sensors (Basel, Switzerland)*, 12(2):1211–79, jan 2012.
22. Joshua L. Payne, Casey S. Greene, Douglas P. Hill, and Jason H. Moore. *Exploitation of Linkage Learning in Evolutionary Algorithms*, chapter Sensible Initialization of a Computational Evolution System Using Expert Knowledge for Epistasis Analysis in Human Genetics, pages 215–226. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.
23. Elizabeth Radetic and Martin Pelikan. Spurious dependencies and EDA scalability. In *Proc. GECCO*, pages 303–310, 2010.
24. Izabela Rejer. Genetic algorithm with aggressive mutation for feature selection in BCI feature space. *Pattern Analysis and Applications*, 18(3):485–492, 2014.
25. Andrew B Schwartz, X Tracy Cui, Douglas J Weber, and Daniel W Moran. Brain-controlled interfaces: movement restoration with neural prosthetics. *Neuron*, 52(1):205–20, oct 2006.
26. Darrell Whitley and Wenxiang Chen. Constant time steepest descent local search with lookahead for NK-landscapes and MAX-kSAT. In *Proceedings of the 14th Annual Conference on Genetic and Evolutionary Computation*, GECCO '12, pages 1357–1364, New York, NY, USA, 2012. ACM.