Int. J. Mach. Learn. & Cyber. (2011) 2:125–134
DOI 10.1007/s13042-011-0030-3

ORIGINAL ARTICLE

Classifying cognitive states of brain activity via one-class neural networks with feature selection by genetic algorithms

Omer Boehm · David R. Hardoon · Larry M. Manevitz

Received: 22 April 2011 / Accepted: 22 June 2011 / Published online: 27 July 2011

Abstract It is generally assumed that one-class machine learning techniques can not reach the performance level of two-class techniques. The importance of this work is that while one-class is often the appropriate classification setting for identifying cognitive brain functions, most work in the literature has focused on two-class methods. In this paper, we demonstrate how one-class recognition of cognitive brain functions across multiple subjects can be performed at the 90% level of accuracy via an appropriate choice of features which can be chosen automatically. Our work extends one-class work by Hardoon and Manevitz (fMRI analysis via one-class machine learning techniques. In: Proceedings of the Nineteenth IJCAI, pp 1604–1605, 2005), where such classification was first shown to be possible in principle albeit with an accuracy of about 60%. The results of this paper are also comparable to work of various groups around the world e.g. Cox and Savoy (NeuroImage 19:261–270, 2003), Mourao-Miranda et al. (NeuroImage, 2006) and Mitchell et al., (Mach Learn 57:145–175, 2004) which have concentrated on two-class classification. The strengthening in the feature selection was accomplished by the use of a genetic algorithm run inside the context of a wrapper approach around a compression neural network for the basic one-class identification. In addition, versions of one-class SVM due to Scholkopf et al. (Estimating the support of a high-dimensional distribution. Technical Report MSR-TR-99-87, Microsoft Research, 1999) and Manevitz and Yousef (J Mach Learn Res 2:139–154, 2001) were investigated.

Keywords One-class classification · fMRI · fMRI-classification · Neural networks · Genetic algorithms

O. Boehm · L. M. Manevitz (✉)
Computer Science Department, University of Haifa ,
31905 Haifa, Israel
e-mail: manevitz@cs.haifa.ac.il

O. Boehm
e-mail: omerboehm@gmail.com

D. R. Hardoon
Department of Computer Science, University College London,
London, UK
e-mail: davidrh@me.com

D. R. Hardoon
School of Computing, National University of Singapore,
Singapore, Singapore

1 Introduction

In recent years, identifying cognitive activity from direct physiological data by using functional Magnetic Resonance Imaging (fMRI) studies as data and identifying the cognitive activity directly from the brain scans has become a real possibility. (See [1–6], to name a few.) This correspondence between physiological information and specific cognition lies at the very heart of the goals of brain science. Note that this work is, in a sense, the opposite of another area of central concern for brain science, specifically, the problem of identifying which areas of the brain are associated with various cognitive activity. However, there is a strong synergy between these two activities. While it might, in principle, be possible to identify the cognitive activity from full brain data, most researchers in this area, starting with [2, 1] have realized that the strong noise to signal ratio in brain scans requires aggressive feature selection.

This noise to signal ratio has several origins:

- The inherent noise in the technological scan;
- The variability within a single subject;
- The fact that a brain is actually performing many tasks simultaneously and one can not control for all of them;
- Brains are physically distinct across individuals and the mappings between them are only approximate [7];
- MRI technology has limited resolution, so in a sense the original data is always "smeared" in the spatial dimension.
- Activity levels are measured indirectly via blood oxygenation, so the data is also "smeared" with respect to time.

In addition, considering the dimensionality of the data, one always has very few data points. A typical scan has about 120 thousand voxels with real values, while the expense and difficulty in acquiring fMRI data of an individual means that the complete data set is in the order of a hundred samples. Thus, the problem being tackled has small data size, large dimensionality, and a large noise to signal ratio. A priori it would seem an unlikely endeavor. Nonetheless, the results reported (beginning with Cox and Savoy [1] and with Mitchell et al. [2]) show that it is possible.

In these works, methods to aggressively reduce non-relevant (noise) features were applied. Note that if one manages to reduce the number of features, one is essentially finding the voxels of the brain that are associated with the cognitive problem; i.e. the complementary problem.

In this work we decided to focus on one-class classification rather than two-class classification, for reasons that will be discussed below. (In our opinion it is often the appropriate setting for this application). (See [8, 9] for some further information on one-class approaches and [10, 11] for other interesting applications of one-class methods.) The one-class classification tools we used were a compression neural network and versions of one-class SVM due to [12] and [9]. Such a classifier was used as an evaluator in two different search approaches. We used a "wrapper approach" [13] to find the relevant features with partial success. While trying to automate and improve the search of the relevant features, we decided to embed the one-class evaluators into a genetic algorithm. This combination yielded extraordinarily good results which are actually comparable to the best two class classification reported results.

We were able to consistently find features that allow differential classification at about the 90% level which now makes this methodology applicable. (In contrast, results on this task without feature selection were about 60% which is similar to the reported results of [4] on a motor task).

However, importantly, as discussed below, the evaluation of a candidate collection of features using this method requires the use of test data from both classes. While this is necessary and standard for testing one-class methods, from one point of view, this contaminates the "one-class" philosophy because one has to perform such evaluation many times during the genetic algorithm process and bias the feature selection with the second class.

As a secondary point, we expected to see that the selected features would be focused in specific and contiguous areas of the brain in visual cortex. (For example, "faces" features are expected to be in an area of the temporal lobe known as the fusiform gyrus [14]). Surprisingly, this was not the case. In fact, no voxels were found that were persistent between runs. Our interpretation is that, the information needed for classification has percolated and it suffices to only sample these dimensions, and the genetic algorithm picks out specific samples which can vary.

The paper is organized as follows: sect. 2 discusses one-class versus two-class classification; Sect. 3 briefly describes the data set the experiments were performed on; Sect. 4 discusses feature selection and our manual search; Sect. 5 describes how we used the genetic algorithm to this task; Sect. 6 discusses the drawback of using data from both classes for evaluation during the genetic algorithm run; Sect. 7 discusses issues related to the "converse problem" of finding areas associated with these tasks and finally, Sect. 8 includes a summary and our intended future directions;

2 One-class versus two-class classification

The problem of classification is how to assign an object to one of a set of classes which are known beforehand. The classifier which should perform this classification operation (or which assigns to each input object an output label), is based on a set of example objects. This work focuses on the problem of one-class classification. In this case, an object should be classified as an object of the class or not. The one-class classification problem differs in one essential aspect from the conventional classification problem. In one-class classification it is assumed that only information of one of the classes, the target class, is available. This means that just example objects of the target class can be used and that no information about the other class of outlier objects is present during training. The boundary between the two classes has to be estimated from data of only the normal, genuine class. The task is to define a boundary around the target class, such that it accepts as much of the target objects as possible, while it minimizes the chance of accepting other objects.

When one is looking for a two-class (or n-class with $n \geq 2$) the assumption is that one has representative data for each of the classes and uses them to discover separating

manifolds between the classes. While the most developed machine learning techniques address this case, this is actually a very unusual situation.

While one may have invested in obtaining reasonably representative data addressing one-class, it is unusual to have a representative sample of its complement in two-class learning. Similar problem can be exhibited in the information retrieval field e.g. querying some search engine for 'houses' will probably yield reasonable results, but looking for anything other than a house i.e. search for 'not houses' would probably yield poor results. The same is true for the n-class case.

A significant weakness of n-class filters is that they must be re-created as data for each class is obtained, and divisions between sub-classes must all be trained separately. Furthermore, essentially, one can never have sufficient data to distinguish between class A and "anything else". Thus, while one may initially have data representing class A, B and C, one must then use two-class methods to find a filter distinguishing between class A and B, class A and C, and class B and C; or alternatively one must find a filter between class A and class (B or C) and class B between (A or C); etc. two-class classification then becomes overly specific to the task at hand. The assumption in using these filters will be that the data comes from one of these classes. Should one wish to add class D, then existing filters must be retrained, and many additional filters distinguishing D from the rest of the above classes must be trained.

It is more natural to imagine a scenario where data is gathered for a particular kind of cognitive task and then, when data for another task is gathered, a different filter is made for the new class. Thus one can incrementally build up a library or "battery" of classification filters; and then test a new data point by this battery. Of course, it would then be possible for a data point to pass several such filters.

However, as expected, in earlier results by [4] the results for two-class classification were superior to those of one-class classification. Their work showed that while one-class classification can be done in principle, for this fMRI task, their classification results (about 60%) were not sufficient for an actual application.

In this work, we have remedied this problem, by showing that one can obtain, automatically, filters with accuracy close to their two-class cousins. The main methodology was finding the appropriate features. This was a reasonable hope given the large dimension of features given by the fMRI map (which were all used in [4]) and since, as described above, most of these features can be thought of as "noise" for this task.

To do this we proceeded with the following main tools:

1. A choice of a one-class classifier approach. The two that we considered were
   (a) The compression neural network [15, 8].
   (b) Different versions of one-class SVM [9, 16]
2. The use of the wrapper approach [13] to judge the quality of features.
3. A manual ternary search proceeding by a ternary dissection approach to the brain (at each stage using the one-class wrapper as an evaluator).
4. Finally, the use of a genetic algorithm [17] to isolate the best features.

The one-class learning method was used to perform the evaluation function in the manual search and the genetic algorithm.

Each of these steps has its own complications and choices. For example: Step 1a requires choosing an appropriate compression ratio for the one-class neural network and, of course, choosing the training method. Step 1b has many variants; we did not exhaust all of them, but we found the results too sensitive to the choices and so in the end used a version of 1a almost exclusively.

Step 3, being manual took too long; we used its results to help decide on the initial conditions of the genetic algorithm.

In both step 3 and step 4, there is a need to evaluate the quality of the features for discrimination. While it is standard in one-class to use the second class data to evaluate the classification results, in this case, the results of this evaluation implicitly affects the choice of features for the next step, and so distorts the pure one-class learning method.

However, since the results seem robust to the choice of the second class data we concluded that the impact of the second class is low. Below, in Sect. 6, we elaborate further about this problem and suggest a method that we hope will eliminate this problem in the future.

3 Task and data description

The experimental data of the visual task were generously provided by Malach [18, 19].

In the experiment that provided the data analyzed here, four subjects, inside a MRI-scanner, were passively watching images belonging to five different semantic categories as follows : human faces, houses, patterns, objects, blank image. The blank image is considered as 'null', as if nothing is viewed. Normalization between individuals were carried as suggested in [7, 18].

The time-course reference of the experiment is built from each subject viewing a sequence of the first four categories separated by the "blank" category i.e. blank, face, blank, house, blank, pattern, blank, object, blank. 147 fMRI scans are taken over this sequence per subject;

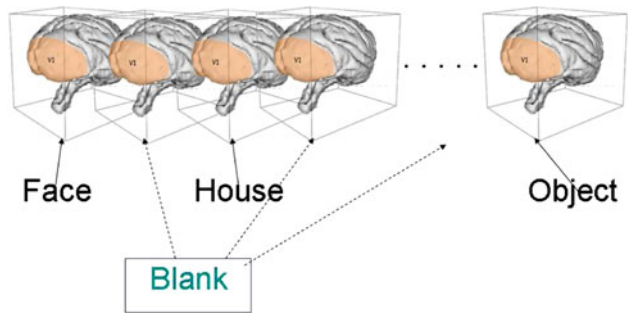

Fig. 1 Illustration of the fMRI scans taken during the experiment

thus the 'raw' data consists of 21 data points for the first four semantic categories and 63 data points for the blank image (Fig. 1).

The individual fMRI images are dicom format (58 image slices) of size 46 × 46 overall consisting of 122,728 real-valued voxels.

4 Feature selection and manual search

4.1 Results without feature selection

In some preliminary work, we ran this task without feature selection, but because of computational limitations at the time, we used every 5th slice out of the 58 available. Thus the data was represented by 13,800 features. The one-class task was run both with a compression neural network (60% compression) and with a version of one-class SVM on the cross individual data. In this experiments we used 38 positive samples for training and 25 positive and 25 negative samples for testing repeated for 10 random runs. Table 1 shows the success rate when trained on each category versus blank for the neural network approach while Table 2 shows the results for one class SVM.

We see that we were unable to produce results above random using the one-class SVM methodology. On the other hand, the compression neural network produced significant results but only in the 60% level. Tests for trained category versus other categories were similar.

Table 1 Combined individuals—bottleneck neural network with 60% compression

	Face	Pattern	House	Object
Blank	56.6 ± 3.8%	58 ± 3.7%	56.2 ± 3.1%	58.4 ± 3.1%

This is comparable to results reported in [4] on identifying the fMRI correlate of a motor task ("finger flexing") using one-class learning (about 59% obtained using either a compression neural network or a one-class SVM).

4.2 Feature selection via manual search

To recapitulate, our approach reduces the question of finding the features, to a search amongst the subsets in the space of features. In this work, we have examined both one-class SVM and compression neural networks as the machine learning tool. These were investigated in [4, 6] where it was found that the neural network approach worked somewhat better. This is not so surprising when considering the work of [16], where it was shown, in a comparative study in a textual classification task, that while both seem to have similar capabilities; the SVM was much more sensitive to the choice of parameters.

The main emphasis of this work is the feature selection, using the wrapper approach and the genetic algorithm approach. We followed two paths: initially we worked by hand and did a primitive, greedy search on the subsets as follows:

- First, start with a "reasonable" area of the data scan; i.e. all background dead area cropped out; the most external levels of the brain discarded. That is, the raw scan had about 120,000 real valued voxels; after reduction we had about 70,000 voxels.
- Second, divide (various options to do that) the brain into overlapping two or three geometrically contiguous boxes (by selecting along one dimension)—run the classifier and discard the lowest returns; continue with the best box as long as it classifies better than the previous loop.
- When all boxes do worse; consider either (1) perform a different division of the boxes along the same dimension as before, but now of different sizes that overlaps the previous chosen boxes or (2) select the boxes by slicing in a different dimension. (i.e. if the search was on boxes defined by the row indices, now use the best row indices found and try to create boxes with different column indices).
- Cease when no improvement is found.

Figure 2 illustrates a sample process of the manual greedy binary search. The assumption was that the task 'relevant' features reside in a relatively small contiguous chunk in the brain. The entire data point which we started

Table 2 Combined individuals—one-class SVM parameters set by subject *A*

	Face	Pattern	House	Object
Blank	51.4 ± 2.55%	52.20 ± 3.49%	53.7 ± 3.77%	52.4 ± 2.9%

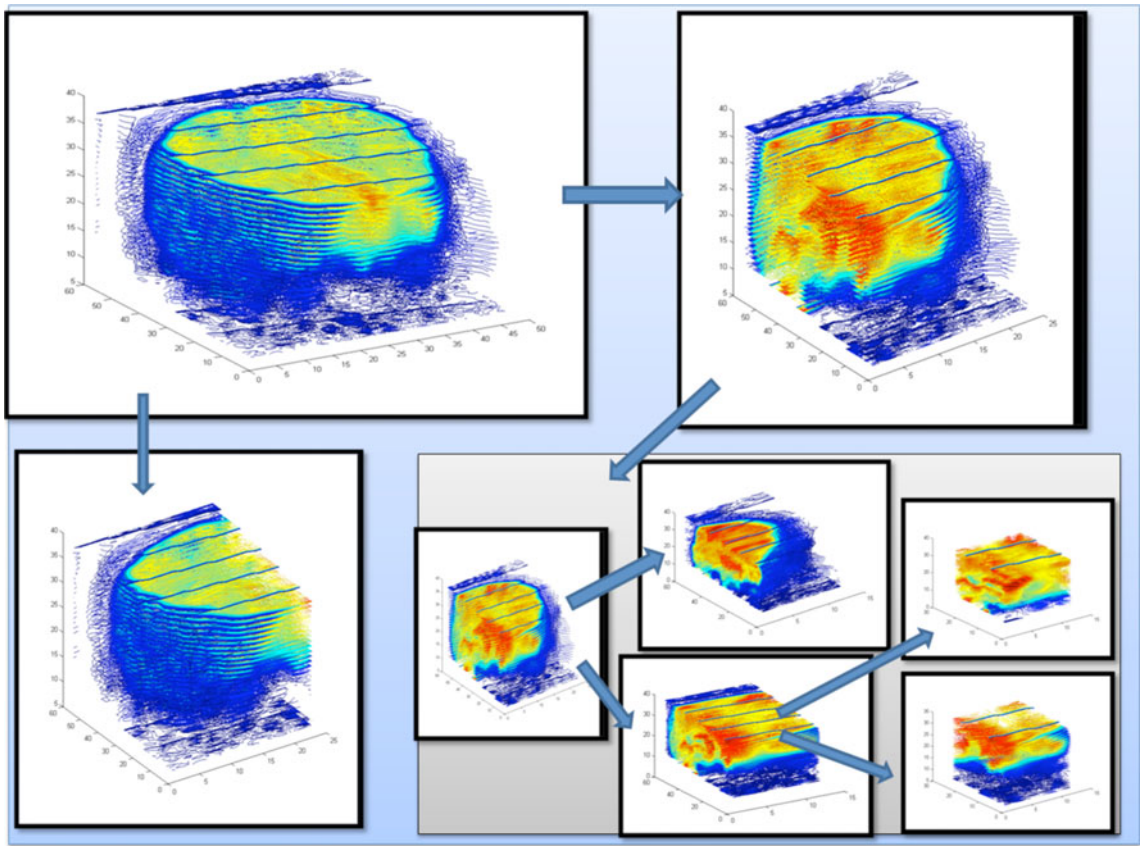

Fig. 2 Conceptual manual binary search via the one-class bottleneck neural network

with is then divided into two parts. Each part is tested separately and the one which shows superior performance (right top corner) is kept while the other part (left bottom corner) is dropped. The same process is iterated for the remaining part until no improvement is found.

Following this work, we were able to produce the following Table 3 of results: (obtained on one of the possible search paths) Each data point (3D matrix) which originally contained about 120,000 features was reduced as explained above into about 70,000 features. ($[58 \times 46 \times 46] \rightarrow [48 \times 39 \times 38]$).

Table 3 represents the results in a specific run for the 'faces' (fMRI data acquired for subjects while viewing images of faces). We used a bottleneck neural network, with compression rate of 60%, which was trained solely for the 'faces' data and then tested against the rest of the categories. This was averaged over fivefolds. The decision how to continue was according to the average over all categories. As can be seen, this method brought us up to 80% accuracy on blank data, and 72% on average.

For a control and comparison, we also considered random selection of about the same proportion of features; and the results were not much above random.

5 Feature selection and the genetic algorithm

It is clear that this way of work is very tedious and there are many possible intuitive choices. In an attempt to automate it, we decided, to apply a genetic algorithm [17] approach to this problem, although the computational requirements became almost overwhelming.

During experimentations we implemented and tested a variety of configurations for the genetic algorithm. In general, each gene representation serves as a "mask" where a "1" indicates that a feature is chosen. This "mask" is actually a set of indices. In order to evaluate the performance of a given mask, the evaluator will have to extract the data elements which are specified in the mask from each and every data point, and perform classification over this "filtered" data.

Figure 3 illustrates an intersection between a single slice of one of the data points and an arbitrary mask.

We used population sizes in the range of 30–50. In the initial generation, the creation function typically set "1" for about 10% of the features selected randomly.

A typical configuration included

Table 3 Manual ternary search via the one-class bottleneck neural network for 'faces' data

Iteration	[Rows, columns, height]	# Features	Houses (%)	Objects (%)	Patterns (%)	Blank (%)	Avg (%)
1	[1–17, 1–39, 1–38]	25,194	58	56	55	60	57
	[15–33, 1–39, 1–38][a]	28,158	62	55	64	65	62
	[30–48, 1–39, 1–38]	28,158	55	52	50	60	54
2	[15–33, 1–39, 1–15]	11,115	61	63	55	60	60
	[15–33, 1–39, 13–30][a]	13,338	69	68	72	70	70
	[15–33, 1–39, 27–38]	8,892	58	57	60	60	59
3	[15–23, 1–39, 13–30]	6,318	63	69	68	62	66
	[20–26, 1–39, 13–30][a]	4,914	70	67	**76**	79	73
	[25–33, 1–39, 13–30]	6,318	60	67	70	75	68
4	[20–23, 1–39, 13–30][a]	2,808	**74**	70	71	73	72
	[22–25, 1–39, 13–30]	2,808	65	**73**	60	**80**	71
	[24–26, 1–39, 13–30]	2,106	70	69	69	68	69
5	[20–21, 1–39, 13–30]	1,404	67	65	74	63	67
	[21–22, 1–39, 13–30]	1,404	60	63	70	64	64
	[22–23, 1–39, 13–30]	1,404	65	63	72	68	67
6	[20–23, 1–18, 13–30]	1,296	67	66	70	72	69
	[20–23, 19–39, 13–30]	1,512	67	70	72	78	72

The bold values indicate the largest value in a column

[a] Indicates the chosen part. (If no part is chosen, the current path is terminated, and a different division step is performed. See text)

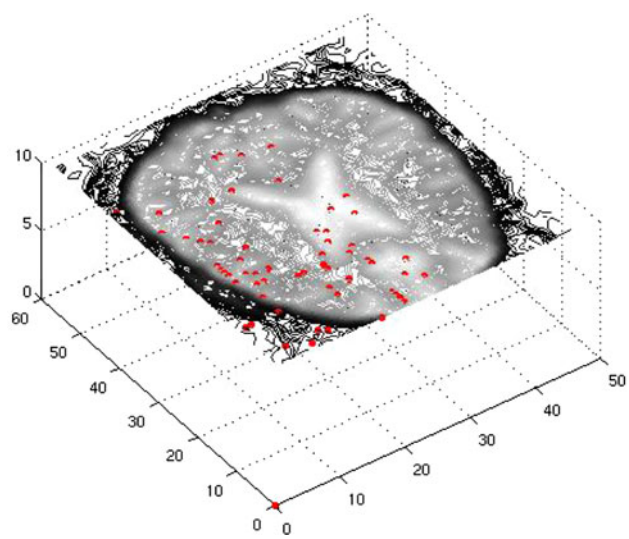

Fig. 3 An example of a "mask" (*red dots* indicate "1"s) intersected with slice #10 of one of the data points

- The gene representation e.g. bit strings, three dimensional matrices. (the matrix dimensions were set to be same as the "trimmed" data points).
- A selection function e.g. stochastic uniform, remainder, uniform and roulette.
- A reproduction method e.g. considered different amounts of elite members, different crossover fractions and various crossover options i.e. two-point crossover for bit string representation, or two planes crossing a cube for three dimensional matrix representation.
- A mutation function e.g. Gaussian, uniform, adaptive feasible etc.

The evaluation methods are the heart of the genetic algorithm. Each implementation included similar steps, i.e. similar pseudo code, and the differences were in the classifier type and data manipulations due to the different representations. The evaluation method works as follows: Given a gene, recreate the data by masking the gene (mask) over each one of the data points. The newly created data set after this action is a projection of the original data set and should have a significantly smaller dimension in each generation, due to the genetic pressure resulting from choosing precise features for classification. This smaller dimension also results in much faster classifier runs. Divide the new data into three parts:

- Training data (60%)—taken from one class.
- Threshold selection and testing data (20%)—taken from two classes.

Train the one-class classifier (either a bottleneck neural network or one-class SVM) over the training data (of all subjects).

Use threshold selection dedicated data and the trained classifier, to determine the best separating threshold.

Finally test using the remaining testing dedicated data and calcluate a success rate. The final evaluation function

Table 4 Genetic algorithm results

	Faces (%)	Houses (%)	Objects (%)	Patterns (%)
Faces	–	84	84	92
Houses	84	–	83	92
Objects	83	91	–	92
Patterns	92	85	92	–
Blank	91	92	92	93

The genetic algorithm was able to find a filter for each class with a success rate almost similar to the ones produced for the two-class classifiers

Table 5 Comparison between one-class results without feature selection and with feature selection via the genetic algorithm between trained classes and blank

	Faces (%)	Houses (%)	Objects (%)	Patterns (%)
Neural network without feature selection	57	58	56	58
Neural network with genetic feature selection	91	92	92	93

of the gene uses a weighted average of the success rate i.e. the number of the data points which were correctly classified and the variance of each testing error from the threshold. That is, the evaluation function tries to take into account the level of the certainty of each test answer.

We produced the results in Table 4 after 100 generations during a run where the population size was 30, 1% mutation rate, selection was roulette, bit string of length 120,000, 2 point cross-over randomly chosen (one point also works similarly); the number of features in generation 0 was 10% or 12,000; On one specific run the chosen gene decreased to 2,843 1's. In general the chosen gene had about 3,000 1's. The final number of features was, of course, not the same on each run.

During the run, we kept track of 'good' genes whose evaluation rate exceeded the weighted average 80%, and then used the best ones.

In Table 5, we reproduce the results from Table 1 and the corresponding row of Table 4. The dramatic increase in accuracy is evident. Similar increases can be seen in all the other rows of Table 4.

6 Pure one-class classification

In this section we discuss the drawback of our implementation which involves a second class data for the evaluation of the genes quality during the genetic algorithm run, and also suggest an alternative, pure one-class classification idea for future work.

Genetic algorithms have a lot of potential different configurations. However in our case, our investigations showed that while various genes representations, selection methods, elite counts, reproduction methods and population sizes etc. are improving the algorithm run time, they do not have a major effect on the results. Nonetheless this improvement is extremely important when dealing with data of such high dimensionality. We found that the actual quality of the results is determined mainly by the evaluation function. The evaluation function which is described in detail above, is the quality of a classification process, where the classifier is trained using data from a single class/category. The trained classifier needs to label (for a predefined testing set out of all classes/categories) a class/category for each data point in the test set. After this classification, the actual evaluation must take place. Note that in order to tell how good a classification is, it is of course necessary to use other class data in the testing phase. The other class data is never used during classifier training though. As a result of using the other class in the testing phases, the genetic algorithm implicitly learns to find the best differentiating genes of the classes used in the evaluation method. This is where the purity of our one-class approach is violated. In work currently underway, we hope to replace the other class data by synthetic data which also evolves.

7 Location of areas of brain associated with cognitive tasks

Having discovered features appropriate for classification; it is interesting to enquire whether or not these features are local, i.e. presented in a particular area of the brain, or distributed. Of course, this can only be asked up to the resolution of the fMRI data themselves.

To get a feel for this, we used Matlab visualization tools. We can show in Figs. 4 and 5 the location of the features (of one of the best genes found by the genetic algorithm) unified with a high resolution contour brain slices.

Surprisingly, although we have not yet quantitatively analyzed all of these results, a visual analysis does not indicate, contrary to expectations, a strong locality of the features. Thus we can not at this stage state which areas of the brain are important for the classification of each task. It is not inconsistent with our results that the best feature selection requires a non-trivial combination of areas. Another possibility, as mentioned above, is that areas of importance in the cortex need only be sampled to provide sufficient classification information and the genetic algorithm just converges in each run to a different such sample. A clarification of this issue awaits further analysis.

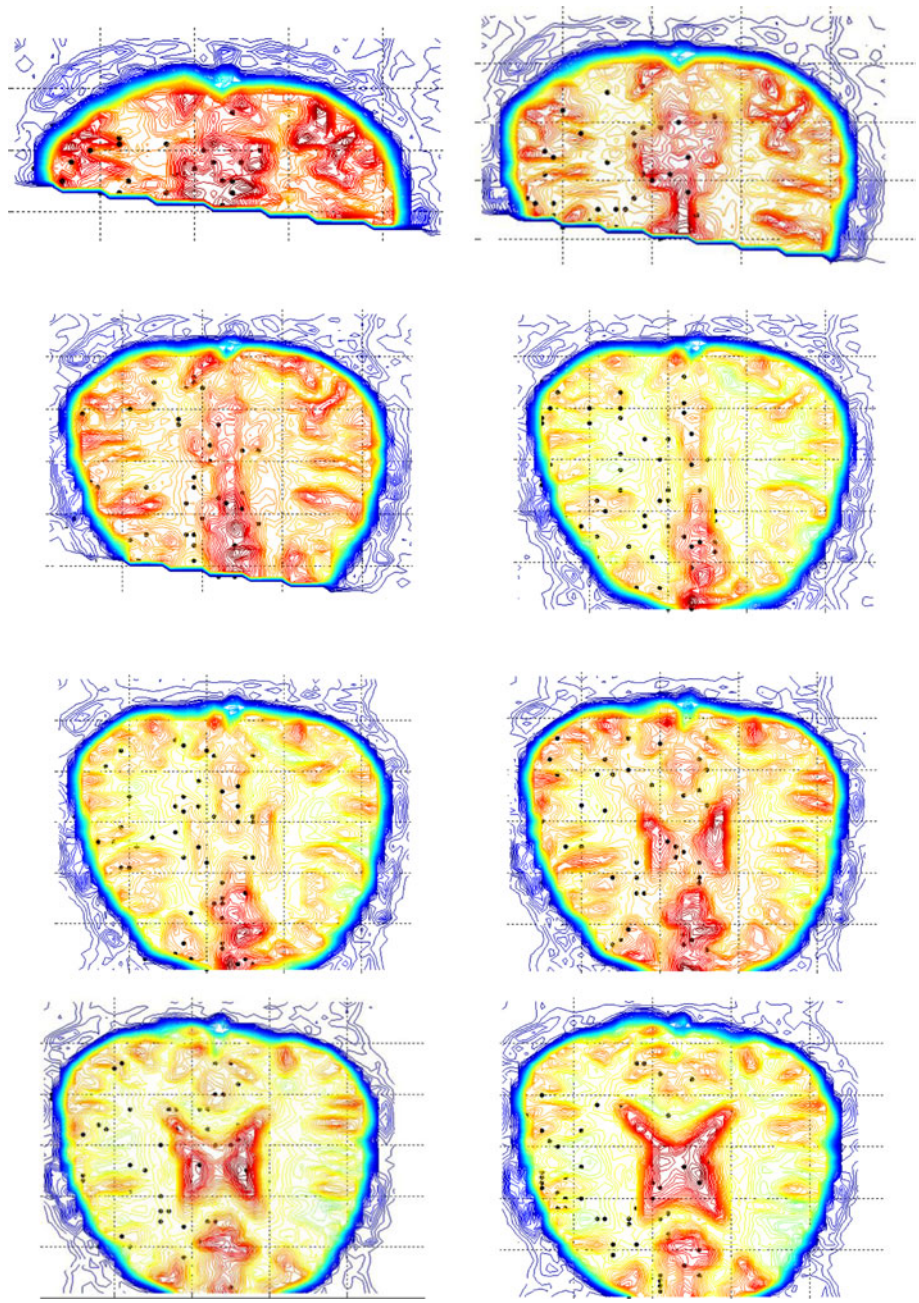

Fig. 4 A visualization of half of the contour slices of a 'house' data point and a chromosome (set of features) which was able to show 91% separation success rate. The *large rounded black dots* indicate selected features

8 Conclusions and future work

Recognizing cognitive activities from brain activation data is a central concern of brain science. The nature of available data makes this application, in the long term, a one-class activity; but until now only two-class methods have had any substantial success. This paper successfully solves this problem in the sample visual task experimented on.

- We have shown that classifying visual cognitive tasks can be done by one-class training techniques to a high level of generalization.
- We have shown that genetic algorithms; together with the one-class neural network compression network can be used to find appropriate features that, on the one hand, increase the accuracy of the classification to close to that obtainable from two-class methods.
- Preliminary results show that this method may indicate non compact areas of the brain must cooperate in order to be critically associated with the cognitive task.

This work needs to be extended to other (non-visual) cognitive tasks; and it needs to be seen to what resolution the work can be carried out. Can specific styles or specific faces of people be identified from these kind of

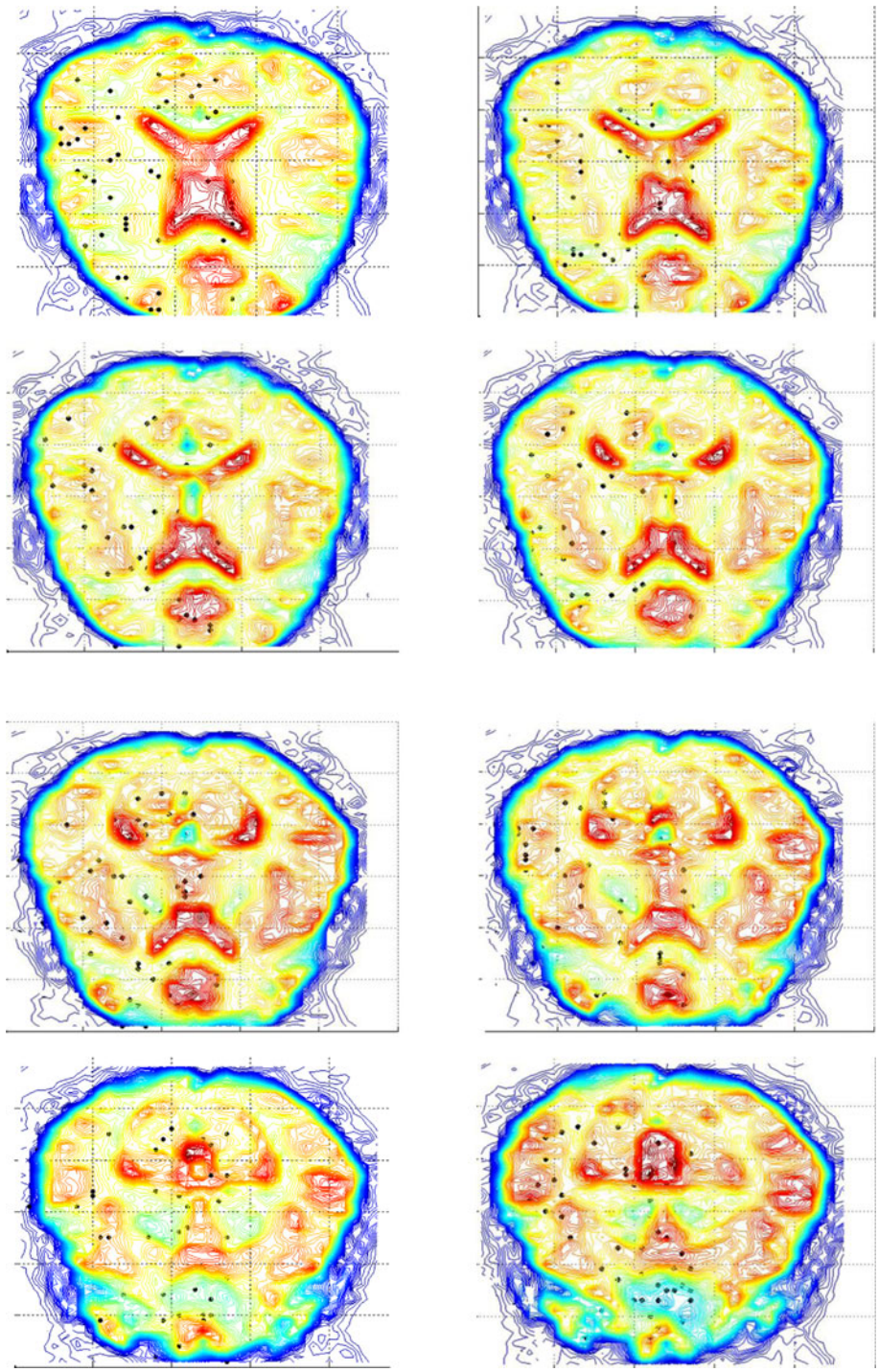

Fig. 5 A visualization of the second half of the contour slices and the chromosome described above

mechanisms? Is this a theoretical limit on either the accuracy or the resolution?

References

1. Cox D, Savoy R (2003) Functional magnetic resonance imaging (fmri) "brain reading": detecting and classifying distributed patterns of fmri activity in human visual cortex. NeuroImage 19:261–270
2. Mitchell TM, Hutchison R, Niculescu RS, Pereira F, Wang X, Just M, Newman S (2004) Learning to decode cognitive states from brain images. Mach Learn 57:145–175
3. Mourao-Miranda J, Reynaud E, McGlone F, Calvert G, Brammer M (2006) The impact of temporal compression and space selection on svm analysis of single-subject and multi-subject fmri data. NeuroImage. doi:10.1016/j.neuroimage.2006.08.016
4. Hardoon DR, Manevitz LM (2005) fMRI analysis via one-class machine learning techniques. In: Proceedings of the Nineteenth IJCAI, pp 1604–1605

5. Carlson TA, Schrater P, He S (2004) Patterns of activity in the categorical representations of objects. J Cog Neurosci 15(5):704–717
6. Pereira F, Mitchell T, Botvinick M (2009) Machine learning classifiers and fMRI: a tutorial overview. NeuroImage 45:S199–S209 (Mathematics in Brain Imaging)
7. Talarich J, Tournoux P (1988) Coplanar stereotaxic atlas of the human brain. Thieme Medical, p 122
8. Japkowicz N, Myers C, Gluck MA (1995) A novelty detection approach to classification. In: International Joint Conference on Artificial Intelligence, pp 518–523
9. Manevitz L, Yousef M (2001) One-class svms for document classification. J Mach Learn Res 2:139–154
10. Sato J, da Graca Morais Martin M, Fujita A, Mourao-Miranda J, Brammer M, Amaro E Jr (2009) An fMRI normative database for connectivity networks using one-class support vector machines. Human Brain Mapp 30:1068–1076
11. Yang J, Zhong N, Liang P, Wang J, Yao Y, Lu S (2010) Brain activation detection by neighborhood one-class svm. Cogn Syst Res 11:16–24 (Brain Informatics)
12. Scholkopf B, Platt J, Shawe-Taylor J, Smola A, Williamson R (1999) Estimating the support of a high-dimensional distribution. Technical Report MSR-TR-99-87, Microsoft Research
13. Kohavi R, John GH (1997) Wrappers for feature subset selection. Artificial Intelligence
14. Kanwisher N, McDermott J, Chun MM (1997) The fusiform face area: a module in human extrastriate cortex specialized for face perception. J Neurosci 17:4302–4311
15. Cottrell GW, Munro P, Zipser D (1988) Image compression by back propagation: an example of extensional programming. Adv Cogn Sci 3
16. Manevitz L, Yousef M (2007) Document classification via neural networks trained exclusively with positive examples. Neurocomputing 70:1466–1481
17. Goldberg DE (1989) Genetic algorithms in search, optimization and machine learning. Addison-Wesley Publishing company, Inc.
18. Hasson U, Harel M, Levy I, Malach R (2003) Large-scale mirror-symmetry organization of human occipito-temporal objects areas. Neuron 37:1027–1041
19. Levy I, Hasson U, Avidan G, Hendler T, Malach R (2001) Center-periphery organization of humand object areas. Nat Neurosci 4(5):533–539