Development and Application of a Toolbox for Multivariate Pattern Analysis of Functional Magnetic Resonance Imaging Data

by

Alan Charles Meeson

A thesis submitted to the University of Birmingham
for the degree of Doctor of Philosophy.

December, 2014
School of Psychology
College of Life and Environmental Sciences
University of Birmingham

Abstract

The combination of functional magnetic resonance imaging (fMRI) and multivariate pattern analysis (MVPA) is a powerful method for investigating brain function, with multiple MVPA methods being applied to the task including Logistic Regression, Support Vector Machines, Neural Networks, and Gaussian Naive Bayes classifiers.

Careful review of application of these methods revealed a common process used in most studies; the majority of variations occurring in the implementation choices in key sections such as feature selection or classification algorithms being employed. Thus, it is possible to develop modularised tools for performing MVPA of fMRI data which can be applied in a variety of ways through selection of appropriate components.

Development of such a toolbox for use by the University of Birmingham Cognitive Neuroimaging Laboratory is described. The modular design allows for flexible application and provides a basis for development of novel methods, which is explored through implementation of a novel cross-validation method and development of a method for investigating the effects of learning on tuning of neural populations.

The development process has resulted in an efficient, robust and reliable toolbox, capable of performing a pre-implemented set of standard multi-variate pattern analyses and provides a basis for further development of novel methods.

Acknowledgements

I would first like to thank my supervisors Zoe Kourtzi and Andrew Welchman for their patience, guidance and support through the course of my studies.

I would also wish to thank Hiroshi Ban, Dorita Chang, Dicle Dövencioğlu, Shuguang Kuai, Sheng Li, Caro di Bernadi Luft, Matthew Patten and Jiaxiang Zhang for the neuroimaging data and time spent discussing neuro-imaging analysis which facilitated the development and testing of the toolbox. I would further like to thank Matthew Dexter for sharing his technical experience and his help talking through development issues.

Finally, I wish to thank my friends and family for bearing with me through the course of my PhD, particularly my parents Don and Lesley Meeson for their tireless support and patience which made the completion of this thesis possible.

Table of Contents

List of Figures

List of Tables

Table of Listings

List of Abbreviations

2D	Two-dimensional.
3D	Three-dimensional.
4D	Four-dimensional.
ANOVA	Analysis of Variance.
BOLD	Blood oxygenation level dependent.
CNIL	University of Birmingham Cognitive Neuroimaging Laboratory.
EEG	Electro encephalography.
EPI	Echo-planar imaging.
fMRI	Functional magnetic resonance imaging.
fSNR	Functional signal-to-noise ratio.
GLM	General linear model.
GPGPU	General-purpose computing on graphics processing units.
HRF	Haemodynamic Response Function.
ISI	Inter-stimulus Interval.
LCO	Leave-*c*-out.
LO	The lateral occipital region, a sub region of LOC.
LOC	The lateral occipital complex.
LORO	Leave-one-run-out.
LPO	Leave-*p*-out.
LR	Logistic Regression.
MRI	Magnetic Resonance Imaging.
MVPA	Multivariate Pattern Analysis.
PCA	Principle Component Analysis.
PET	Passive Emmision Tomography.
PSC	Percent Signal Change.
RFE	Recursive Feature Elimination.
ROI	Region of Interest.
RVM	Relevant Vector Machine.
SVM	Support Vector Machine.
TE	Echo time.
TR	Repetition time.
V3B/KO	Kinetic occipital region (anatomically defined as V3B).

CHAPTER 1 - GENERAL INTRODUCTION

The human brain and its role in cognition has been a topic of study for centuries. Over that time the methods employed in its study have developed greatly, from speculations by Renaissance philosophers such as Descartes to the mid 19th Century work of Broca localising regions involved with speech through study of lesions, through to more recent methods which measure neural activity such as Electro Encephlography (EEG), Positron Emission Tomography (PET) and functional Magnetic Resonance Imaging (fMRI) (Savoy, 2001; Huettel et al., 2009).

FMRI provides a timeseries of 3D images (referred to as volumes) composed of cuboid regions known as voxels. The value assigned to each voxel corresponds to the BOLD signal, a measure discussed later in this chapter, which correlates with neural activity.

Recently fMRI has been shown to be a particularly popular methodology, due to its provision of both a high spatial resolution (with voxel sizes on the scale of ~1mm) and decent temporal resolution (on the scale of ~1s) while maintaining a non-invasive nature which allows for application to a wide range of research questions. Thus there has been an exponential growth in the number of published papers utilising fMRI since the early 1990s, and it has largely taken over from PET, an earlier imaging methodology, which suffers in spatial and temporal resolution in comparison, and is limited in the number of samples which can be collected due to the risk of radiation toxicity (Huettel et al., 2009).

Corresponding with the rise in prevalence in imaging studies there has been a correlating rise in the use of machine learning techniques, particularly classification algorithms, to perform multivariate pattern analysis (MVPA) of imaging data (Pereira et al., 2009; O'Toole et al., 2007).

In MVPA studies classifiers are trained to use patterns of neural activity which arise from the small biases present in the response of each voxel collected during fMRI experiments to predict the experimental conditions during which the scans were taken. The classification performance of these models can then be used as an indicator of the degree to which a region of the brain, defined by the set of voxels used in the classification, encodes information relating to the chosen experimental conditions. By considering the patterns of neural activation across a set of voxels MVPA techniques achieve a greater degree of sensitivity and are capable of discriminating between experimental conditions which are encoded in the same regions of the brain.

The popularity of multivariate pattern classification for the analysis of fMRI data can in part be attributed to the advantages it provides over prior voxel-based inferential methods, such as Analysis of Variance (ANOVA) or General Linear Models (GLM), and multivariate exploratory methods, such as Principal Component Analysis (PCA).

Voxel-based inferential methods tend to consider each voxel independently which, while a viable method to detect involvement of regions in some instances, may fail to detect information which is encoded as patterns of activation across several voxels. The exploratory multivariate analysis methods may counter this issue, however they are limited in turn by the failure to provide the quantifiable link back to the experimental conditions which are provided with the voxel-based inferential methods. A pattern classification approach however, accounts for both factors by considering neuronal activation as patterns across voxels, and by providing a direct link back to the experimental design at the level of groups of or single stimulus presentations (O'Toole et al., 2007).

In addition to addressing limitations of previous methods, pattern classification based approaches allow for investigation of the manner in which information is encoded within the brain, rather than solely localisation. This has allowed for studies such as Ban et al. (2012) that investigates the manner in which visual depth cues are integrated.

An additional factor to the popularity of pattern classification approaches to the analysis of neuro-imaging data has been the wide range of pre-existing algorithms available from the field of machine learning, which has provided great scope for interdisciplinary collaboration. As such, a great number of machine learning algorithms have been applied to the problem, including Logistic Regression (LR) (Yamashita et al., 2008; Rayali et al., 2010), Support Vector Machines (SVM) (Preston et al., 2008; Li et al., 2007) and Relevance Vector Machines (DeMartino et al., 2011).

While classification algorithms are at the core of this type of analysis process, a wide variety of associated techniques have been applied to tackle the problems inherent in fMRI data, such as feature selection (DeMartino et al., 2008; Kriegeskorte et al., 2006) and sample estimation (Pereira et al., 2009).

Consideration of a number of machine learning based fMRI analysis techniques shows that most methods can be characterised as a sequence of abstract processes, each of which can be implemented with any one of a range of algorithms (Pereira et al., 2008). The selection of which algorithms to employ can determine the specific question asked by the analysis, e.g. while a normal analysis could determine whether a specific region encodes information on a specific cue, the choice to eliminate any univariate signal by employing normalisation across voxels could further refine the result to indicate whether the same region encoded the information in a multivariate manner.

While the correct application of machine learning techniques can provide a great deal of information regarding the localisation and function of neural processes, there are certain seemingly minor implementation choices which can greatly affect the meaning and accuracy of the results. For example, when performing feature selection using Recursive Feature Elimination (RFE) (DeMartino et al., 2008) neglecting to apply normalisation, or applying it at the wrong stage, may bias the selection of features towards those which provide less information.

There are a number of toolboxes already in existence aimed at performing MVPA on neuro-imaging data; three of note being:

- Princeton MVPA toolbox (Detre et al., 2006)
- PyMVPA (Hanke et al., 2009)
- CoSMoMVPA (Oosterhof and Connolly, 2014)

Despite the existence of two of these toolboxes at the time work was begun at the University of Birmingham Cognitive Neuroimaging Laboratory, it was decided to develop the toolbox presented in this thesis in order to meet the specific needs of the laboratory: a toolbox which implemented in Matlab a set of standardised pre-defined analyses which could be applied to the pre-processed data produced following initial analysis work performed using Brain Voyager QX (Brain Innovation B.V.)

In creating the toolbox described in this thesis a number of basic MVPA techniques were gathered into a single tool, which allows for automated processing of fMRI Data while protecting the user from some of the common pitfalls. This toolbox focuses on efficient and robust computation of a set of standard analyses, including basic Support Vector Machine classification, recursive feature elimination and searchlight analysis. This was achieved through a combination of repeated optimisation and testing, in addition to design with a view towards parallel processing using the Matlab Parallel Computing Toolbox (The MathWorks, Inc., Natick, Massachusetts, United States).

1.1 Functional Magnetic Resonance Imaging Data

In order to discuss the application of machine learning to fMRI data some background on the nature of the data in question is desirable. The techniques and underlying technology of fMRI is a large topic, and to discuss it in detail would require a book, such as those by Huettel (2009) or Moonen and Bandettini (1999). As such the discussion of fMRI in this chapter is limited to those aspects which are relevant to the methods discussed in later chapters of this thesis.

1.1.1 BOLD Signal

Unlike other methods such as EEG, fMRI does not directly measure neural activity but rather indirectly, via the associated haemodynamic changes. Localised increases in levels of neural activity are reliably accompanied by increases in blood flow; however since they are not accompanied by a change in oxygen utilisation, this results in a relative increase in the quantity of oxyhaemoglobin and decrease in the quantity of deoxyhaemoglobin near regions of neural activity (Raichle, 1998). The differing magnetic properties of oxyhaemoglobin (which is diamagnetic) and deoxyhaemoglobin (which is paramagnetic), result in differing distortions to the local magnetic fields. This allows for these changes in concentration to be measured using Magnetic Resonance Imaging (MRI), with an increase in neural activity and its corresponding decrease in the concentration of deoxyhaemoglobin resulting in an increase in the intensity of the MR signal. (Huettel, et al. 2009) This is referred to as the Blood Oxygen Level Dependent (BOLD) response (Savoy, 2001).

1.1.2 Haemodynamic Response

The change in BOLD signal which is triggered by the neuronal activity occuring as a response to a stimulus is referred to as the haemodynamic response. There is a degree of variability in the shape of this response, which caused by a number of factors including the properties of the stimulus itself and consequently the underlying neuronal activity. For example, a stimulus of longer duration would prompt neuronal activity of a corresponding increased duration, thus increasing the width of the haemodynamic response. (Huettel et al., 2009).

Despite this variability there is a typical haemodynamic response function, which can be characterised with some degree of consistency, comprising the following four stages: rise, peak or plateau, fall, and undershoot. This response function can be modelled nicely using a gamma, or two-gamma function – an example of which is shown in Figure 1.1.

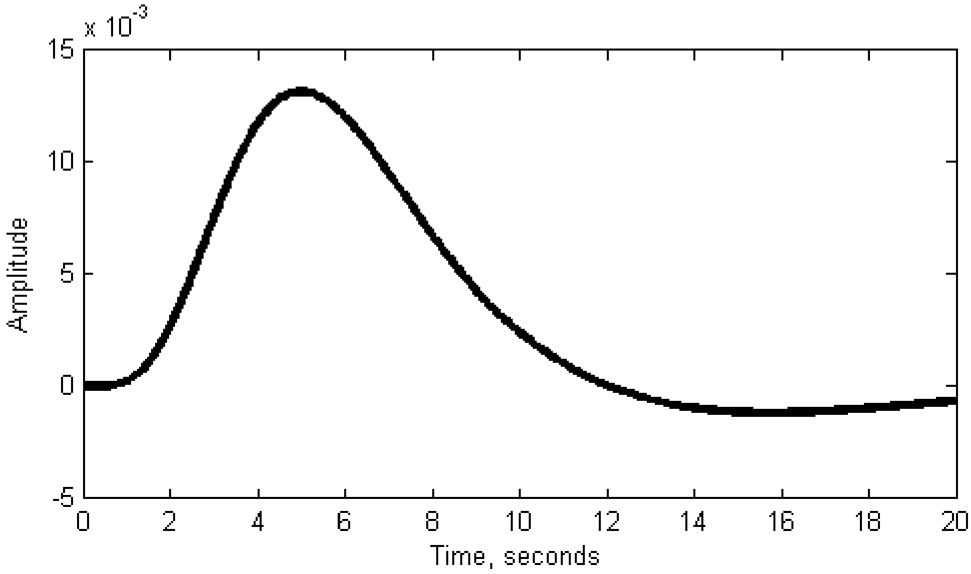


Figure 1.1: Example of a two-gamma model of a haemodynamic response function (HRF).

Due to the slow nature of the BOLD signal, there is typically a delay between stimulus presentation and the onset of the response of between 1 and 2 seconds, with the peak amplitude being reached at approximately 4.5 seconds. If the neural activity is extended - e.g. by repeated stimulus presentation - the peak of the response may be similarly extended, however the response to a single stimulus presentation typically returns to the baseline after between 16 and 20 seconds (Huettel et al., 2009; Meizin, 2000).

1.1.3 FMRI data acquisition

As mentioned previously, the physics underpinning fMRI is a large topic, detailed description of which is beyond the scope of this thesis, however a brief discussion of it is provided here.

Through measuring the haemodynamic response, described in Section 1.1.2, using an MRI scanner we can take three dimensional (3D) images, or volumes of the brain. Scans of the brain or regions thereof are usually taken as sequences of scans of individual slices; however, in some cases scans will be performed directly in 3D, typically for anatomical or high resolution fMRI acquisition. When imaging is performed using multiple 2D slices, the thickness of these slices and the resolution in the plane of the slice are affected by a number of factors, including the strength of the magnetic field used by the MRI scanner (Huettel et al., 2009). By taking repeated 3D scans of the brain at regular intervals it is possible to acquire a four dimensional (4D) representation, which can also be considered in terms of a timecourse of haemodynamic responses to activations for each voxel in the 3D representation.

The choice of the spatial resolution at which to scan is dependent on the research question being asked, as there are a number of trade offs to be considered. While opting for the highest resolution possible may seem the intuitive choice, since it potentially allows for investigation in greater detail, it can be undesirable. The BOLD signal, which fMRI measures, is dependent on the total change in deoxygenated haemoglobin within the voxel, and as such smaller voxels resulting from high resolution imaging tend to have lower signal-to-noise ratios.

In addition to this scanning at higher resolutions increases the time taken to acquire each slice, which can result in a lower temporal resolution. As such, when scanning at higher resolutions with lower slice thickness it is common practice to limit the region of brain being scanned to cover the particular regions of interest relevant to the study in order to allow for maintaining a sufficient frequency of images (Huettel et al., 2009).

1.1.4 fMRI protocol design considerations

Displaying a pre-determined sequence of stimuli while scanning allows for the linking of the neural activity to real world input. Careful selection of stimuli allows for the isolation of particular visual cues and the avoidance of confounding factors.

When designing the presentation of the stimuli the goal is to do so in a manner which allows for the gathering of as many samples of activation as possible while maximising the evoked change in the BOLD signal being recorded and avoiding the introduction of confounding factors. In order to do this the presentation method must account for the limitations imposed by the fMRI methodology.

As discussed previously the magnitude of the haemodynamic response to a stimulus can extend for up to 20 seconds before returning to the baseline levels (Huettel et al., 2009). It follows that if two stimuli are presented within a few seconds of each other, the activation recorded for the latter stimulus will be akin to a linear combination of the first and second stimuli, and so be correlated with the activation of the first stimulus. This correlation between the activation evoked by each stimuli can make it more difficult to distinguish between them, and may act as a confounding factor.

An intuitive solution to this would be to leave a gap of 20 seconds between stimulus presentations avoiding any overlap between the responses to said stimuli. This helps avoid the introduction of confounds by minimising the correlation between successive stimuli. However, this severely limits the number of samples which can be gathered due to time constraints, reducing the chances of obtaining interpretable results. Additionally the activation resulting from single stimuli often fails to provide sufficient signal strength to allow for reliable detection of activated regions. More useful approaches to this problem include those used by the two presentation designs discussed below. Block design, which presents several stimuli of the same category in series followed by a gap, to reduce the impact of the overlap between classes while evoking large changes in activation. Event related design, which varies the ordering so that and correlations are counter balanced, and in some studies applies deconvolution to the recorded signal to separate the contribution of different events (Hinrichs et al., 2000).

Two presentation designs frequently referred to in this thesis warrant further description, which follows.

Block design

In this method, sets of multiple stimuli from the same category are presented in blocks. This repeated presentation has the benefit of providing a strong signal as the repeated activation extends the peak of the HRF, while in designs where periods with no stimuli are presented between blocks the HRF has time to return to baseline levels. The downside to this approach is the limited number of discrete samples that can be recorded. Additionally as the stimuli in each block must be of the same category, this method may be less suited to experiments which aim to record the behavioural performance in stimulus categorisation as well as the performance as given by classification of the resulting fMRI data.

Event related design

In this method stimuli are presented individually with only a short inter stimulus interval (ISI). This allows for the collection of a greater number of samples, allowing for the potential for more categories of stimuli while maintaining a sufficient number of samples from each category. The downsides of this approach are that the response generated by a single stimulus will be weaker than a block of the same, and since the ISI is much shorter than the 20 seconds required for the haemodynamic response to fade there will be some overlap in the response of adjacent presentations.

In order to handle the problem of the overlapping responses it is possible to generate the sequence in which the stimuli are presented such that there are no repeated patterns of adjacent stimulus conditions. While this does not remove the interference caused by the overlap, it does serve to prevent the overlap from acting as a confounding factor in further analysis.

An issue common to both designs discussed above is the effect of fatigue and flagging attention in the subject being scanned, which may cause decreases in the magnitude of the BOLD responses over the course of a scanning session. This has the effect of not only reducing the magnitude of the BOLD responses – and potentially the difference in response between stimulus categories – but can also cause difficulties in subsequent analysis as the responses at different times within a scanning session may in effect appear to have been taken from a different distribution of responses, i.e. one with a reduced mean.

A common approach to dealing with this is to break the scanning session into a series of scans (or runs) approximately 5-8 minutes in duration with rest periods in between. This has the beneficial side effect of providing ready made independent sets of data which are useful for analysis techniques such as cross-validation.

1.1.5 fMRI data pre-processing

While careful design of the stimuli and presentation can account for some of the issues that arise due to the nature of the fMRI methodology, others are handled by post-collection processing. There is a wide range of methods, each intended to combat specific issues (Huettel et al., 2009). Three examples of this are provided here: linear trend removal, spatial alignment, and slice scan-time correction.

Following on from the previous discussion of the effects of fatigue and attention: while the experiment can be designed to minimise the effects of fatigue and attention, there are also post-processing methods which can be applied. One such method is trend removal, which can be used to model and adjust for fatigue induced linear trends in the BOLD signal.

When dealing with multiple scans of a single subject or when performing studies with multiple subjects, particularly when it is desirable to make generalisations across them, it is necessary to align all of the brains scanned to Talairach space (Talairach and Tournoux, 1988; Huettel et al., 2009). This space is defined such that the X and Y axes are aligned with the horizontal plane defined by the anterior and posterior commissures, with the Z axis perpendicular to them in the ventral-dorsal directions. By performing this alignment and further using interpolation into a 3D grid of uniform cube cells or voxels, a common frame of reference is established which allows for simple analysis across scans.

As mentioned in the previous section a single volume, or image of the brain, is not taken at one moment, but rather as a series of individual slices, each taken a small time apart (~50ms). When performing further analysis it is useful to be able to assume that all parts of a single volume have been imaged at a single time. To allow this a process called slice scan time correction is performed, in which an adjusted volume timecourse is created using interpolation between adjacent volumes.

In this thesis most of the post processing of the raw fMRI timecourses were performed using Brain Voyager QX (Brain Innovation B.V.), which provides implementations of the techniques mentioned above along with a number of other functions.

The end result of the post processing procedure is a timeseries for each voxel, such that all voxels can be assumed to have been recorded at the same time within a single volume, and where the voxels are aligned to a common coordinate system.

1.1.6 Data Following Post Processing

The end result of the post processing procedure is a timeseries for each voxel, such that all voxels can be assumed to have been recorded at the same time within a single volume, and where the voxels are aligned to a common coordinate system. This, along with the set of condition labels corresponding to each volume in the time series for the basis of the data being analysed throughout the thesis.

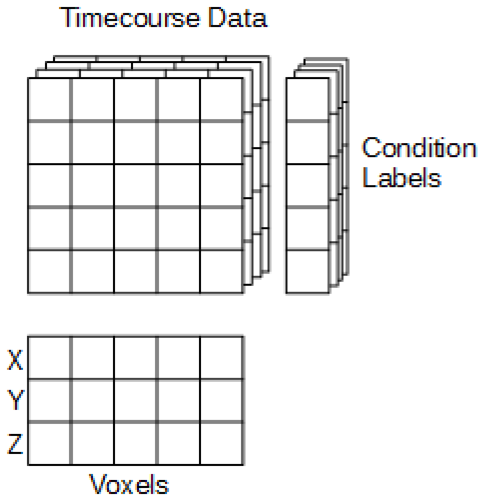


Figure 1.2: Overview of fMRI data following post-processing.

This data can be considered as a collection of matrices and vectors, which can be defined as follows:

Voxels

The set of voxels can be defined as a matrix, $V \in \mathbb{N}^{n \times 3}$, where *n* is the number of voxels, such that $v_i \in V$ is the three dimensional coordinate vector for the *i*-th voxel in set *V*.

Timecourse Data

At this stage there is a set of timecourses, one for each run. They can in general be defined as a matrix, $X \in \mathbb{R}^{m \times n}$, where m is the number of volumes, such that $x_{j,i} \in X$ is the recorded value for the *j*-th volume of the *i*-th voxel.

Condition Label

As with the timecourse data, there is a set of condition labels, one for each run. They can in general be defined as a vector, $Y \in \mathbb{N}^{m \times 1}$, such that $y_j \in Y$ is the condition label for the *j*-th volume.

1.2 Univariate Analysis

Univariate methods offer some of the simplest means to analyse fMRI timecourse data, and involve the individual analysis of each voxel timecourse. One of the most common and useful is the General Linear Model (GLM). This technique treats the timecourse of each voxel as the result of a linear combination of several predictors, which include the expected responses to the sequence of stimuli presented throughout the scan, and potentially confounding factors such as head motion. This is preferred over a straightforward t-test or Analysis of Variance (ANOVA), since it allows for regressing out of identified confounding factors.

Head motion predictors for translation and rotation can be taken directly from measured data. Predictors for expected responses from a given category of stimulus are obtained by first generating a box-car time course in which values are set to 1 at time points where stimuli of that category are presented, and 0 at all other times. This box-car time course is then convolved with a standard haemodynamic response function.

Considering this for an individual voxel, *j*; by defining a design matrix, $D \in \mathbb{R}^{n \times p}$, where *n* is the number of volumes in the time course, and *p* is the number of predictors, this can be expressed as

$$x_{ij} = b_0 + b_1 D_{i,1} + b_2 D_{i,2} + \dots b_p D_{i,p} + e_i \quad ,$$

where x_i is the BOLD signal response for the *i*-th volume in the time course, b_0 is the beta weights for each predictor and b_0 is a constant representing the baseline level and e is the error. A set of of optimal beta-weights equivalent to a least-squares estimate, which minimise the error can be obtained by the following equation:

$$b=(D'D)^{-1}D'y$$

After fitting this model to the data it is then possible to generate statistics such as the *F* statistic or t-value to indicate the degree to which a given voxel can differentiate between the experimental conditions. This allows for the identification of interesting voxels, such as those which respond more strongly to the stimuli than to the fixation periods or those which respond more strongly to some categories of stimuli than others. The results provided by this analysis, when applied to a standardised fMRI protocol of localiser stimuli, such as retinotopic localisers, can be used to identify standard regions of interest, in this case early visual areas including V1 and V2.

While univariate methods are capable of identifying individual voxels which respond to a stimulus with a change in activation, they are incapable of identifying groups of voxels which encode information regarding a stimulus as a pattern of relative activation.   Additionally when aggregating responses across groups of voxels to increase sensitivity or reduce noise, voxels which, while perhaps providing a clear signal, show a response lesser in magnitude may be overwhelmed by other less informative voxels which happen to have higher activation.

1.3 Multivariate Pattern Analysis

Unlike univariate based analysis which deals with voxels on an individual or averaged basis, Multivariate Pattern Analysis (MVPA) considers groups of voxels, looking not only at the behaviour of each individual voxel, but also its activation relative to the other voxels in the group. The actual voxels included within a group is determined by feature selection and can depend on a number of factors, including: anatomical or functionally defined region, spatial proximity and information content. This multivariate approach allows for investigation of not just where, but also how information is encoded in the brain (O'Toole et al., 2007).

While the core of the MVPA process is typically classification, the nature of fMRI data means that a classification algorithm on its own is insufficient. To perform any meaningful analysis a number of additional processes need to be applied beforehand, to prepare the data, or after the classification to extract further meaning.

Pereira et al. (2009) suggest that this collection of the classifier and the preceding and following processes can be considered as a series of decision points; namely which specific implementation of the generic process to employ at each stage. Through consideration of the various techniques to be included in the MVPA toolbox, the following stages were identified as those in which the definitive choices of the analysis are made.

Each of the steps listed below can be implemented in a number of different ways, and the choice of implementation can have a large effect on the meaning of the results provided by the analysis; as such the selection of which particular methods to use is a matter of importance.

1.3.1 Normalisation

As discussed earlier the fMRI data being analysed are typically collected in a series of scans, and a number of factors can cause drift in the BOLD signal between these. The key purpose of normalisation is to counteract these effects and ensure that the data in each scan can be considered to come from the same distribution, or at least have approximately the same mean and standard deviation, as this is a requirement for classification. Two typical methods for normalisation are normalising each row, and normalising each column.

When normalising each row (be they samples or volumes depending on when the normalisation is performed) the values of the voxels within the row are scaled to have a mean of 0 and a standard deviation of 1. Given a timecourse $X \in \mathbb{R}^{m \times n}$, where $x_{i,j}$ indicates the value for the *i*-th row and *j*-th voxel this is represented as:

$$x_{i,j} = \frac{x_{i,j} - E\, x_{i,\cdot}}{\sigma(x_{i,\cdot})}$$

The goal of this method is to reduce the effect of large volume wide signal changes (Pereira et al., 2009), and it is used implicitly when using certain classification methods such as the pattern-correlation classifier used by Haxby et al. (2001).

When normalising by columns each column (which corresponds to a voxel) is scaled to have a mean of 0 and a standard deviation of 1. Using the same notation as above, this is represented as:

$$x_{i,j}=\frac{x_{i,j}-E\,x_{\cdot,j}}{\sigma\left(x_{\cdot,j}\right)}$$

When performed individually on each run this can help to account for between run variations in the baseline or magnitude of BOLD response; however it can also be applied to ensure that the values of each voxel fall on the same scale (Pereira et al., 2009). Normally, linear classifiers can account for the difference in scale between voxels, however ensuring the voxels are on a common scale can be particularly useful when using a linear classification model and seeking to interpret the linear weights as indicators of the importance of each voxel to the classification, such as in recursive feature elimination (DeMartino et al., 2008).

There are additional situations in which normalisation is employed within MVPA, such as normalising the activation within volumes across a set of voxels in order to eliminate any univariate signal (Misaki et al., 2010), allowing for the determination of whether information regarding a given stimulus is encoded in a univariate or multivariate manner.

1.3.2 Cross-validation

Classification analyses require at least two sets of data to operate on: a set of training examples on which to build the model, and a set of testing examples on which to evaluate its performance. The process of dividing the available data into these training and testing sets can be as simple as arbitrarily selecting a set of volumes to serve as a hold out set (e.g. one run); however this poses the further challenge of selecting a testing set sufficiently large to give a good estimation of the classifier performance, while retaining a sufficiently large training set on which to build a robust classifier model. This task is further complicated by limited quantity of available data common in fMRI studies. As such in order to maximise the power of the analysis for the limited number of samples which can be collected in most fMRI studies, cross-validation is typically used for this purpose rather than a simpler hold-out-set method.

Of the various cross-validation methods available in the field of statistics and machine learning, a variant of the *k*-fold cross-validation method is the most suitable for most fMRI studies. In the original *k*-fold method, the available data is divided randomly into *k*-folds of equal size. *K* different analyses are then performed, with one of the fold forming the test set, and the remaining folds forming the training set. In each of these *k* analyses, a different fold is selected as the test set.

As it stands, the *k*-fold approach is unsuited to fMRI data. The *k*-fold method assumes that any given sample is completely independent from those adjacent to it, and with fMRI data this is not the case due to the temporally extended nature of the BOLD response. By replacing the random selection used in the *k*-fold cross-validation method with a direct mapping of the cross-validation folds to the runs of the fMRI data set, we arrive at leave-one-run-out cross-validation. Since each run is separated by rest periods, this ensure a complete separation of samples and the potential overlap is avoided.

Ensuring a clean separation between training and testing datasets, such as with the issue described above, is required in order to avoid overfitting and bias in the resulting classification models, and failure to do so can result in a circular analysis; a problem which is discussed in detail by Kriegeskorte et al. (2009).

While the Leave-one-run-out cross-validation method addresses the problem of ensuring there is no overlap between training and test data, it does not address the other key problem common to cross-validation methods: the overlap between training sets.

For all cross-validation methods where the number of folds is greater than 2, each training set will contain data which are shared with all of the other training sets. This results in an increased correlation between any models trained on these sets. Since the variance of correlated variables is the sum of their covariance matrix, this results in an increased variance of prediction error of the models. This effect increases with the number of folds used, with leave-one-out cross-validation having the largest variance. Conversely, the use of fewer folds can also raise the variance, due to the decrease in the size of the training set.

A good compromise is to use stratified 10-fold cross-validation, which has been found to offer the best trade off between these two factors (Kohavi, 1995). Designing fMRI experiments such that approximately 10 runs of data are collected, with a balanced number of examples of each class in each, helps minimise the effects of this overlap between training sets while still benefitting from the increased training set size.

1.3.3 Sample Estimation

In order to perform MVPA on fMRI data is is necessary to convert it from the 3D time-series of volumes into a 2D matrix in which the rows and columns correspond to samples and features respectively. This can be as simple as mapping each voxel to a column and each volume to a row, however this is usually a sub-optimal approach. Some views of the MVPA process combine the sample estimation and feature selection steps into a single pattern generation step (e.g. Pereira et al., 2009), which has the benefit of allowing for the use of features generated as a composite of several voxels, however in this thesis they are considered as separate steps. The process of feature selection is discussed later on.

The chief goal of sample estimation is to convert the fMRI data from the volume based representation to a sequence of samples which correspond to either a particular stimulus presentation or a block of stimuli of the same category as indicated by the experiment design. As discussed with regard to experiment design, each stimulus trial is typically composed of two or more volumes in the case of event related designs or entire blocks of volumes in the case of block designs. As such one effective approach is to simply take the mean of all volumes in each given trial or block (Pereira et al., 2009):

$$S_{t,j} = \frac{\sum_{i=1}^{|T|} x_{t_i,j}}{|T|} \quad ,$$

where X is the matrix of timecourse data, T is a set of the indices of volumes belonging to t-th trial or block, and $S_{t,j}$ denotes the resulting sample value for the t-th trial.

Another more complicated approach is to estimate response amplitudes for each trial or block using a linear model of the haemodynamic response predictor for each stimulus, with the resultant beta-values forming the sample value for each voxel. This approach, while more complex and not directly utilising the BOLD signal, allows for the consideration of other factors such as trends and head motion when calculating the samples (Misaki et al, 2010).

Whichever method is used, consideration must be given to the trade off between the number of samples produced and the signal robustness of the produced samples. This can be seen quite clearly when considering the methods which simply average sets of volumes. Trials containing few volumes will typically result in less robust, noisier signals than those of blocks containing a greater number of volumes. Conversely, averaging a larger number of volumes will result in a smaller number of samples being produced, which limits the performance of the subsequent classification algorithm. (Pereira et al., 2009).

1.3.4 Feature Selection

While fMRI data are relatively limited in the number of samples available, there is the opposite problem with features; even at relatively low resolutions (e.g. 3mm cubic voxels) there are tens of thousands of voxels available. It is necessary to perform feature selection both to reduce the number of features being considered and to focus on voxels which are relevant to the analysis being performed.

This can take the form of simply limiting the selection of voxels to an individual region of interest (ROI), selecting voxels based on some property of the voxel (DeMartino et al., 2008; Ryali et al., 2010; Yamashita et al., 2008), or the generation of composite features based combinations of the values of multiple voxels using methods such as Principal Component Analysis (PCA) (Hansen et al., 1999).

As mentioned previously this thesis does not cover the use of feature generation methods, such as PCA and Independent Components Analysis (Calhoun et al., 2003), but rather focuses on the simpler and more readily interpreted feature selection approaches, some relevent examples of which are presented below.

Of these methods, possibly the simplest is selection of voxels based on activity. This method selects voxels from within a specified set based on their response to the stimuli, such that the voxels which are selected show a significantly higher response to the test stimuli than to a baseline condition. This is achieved by thresholding or ordering voxels based on the t-value of a t-test or GLM contrasting these sets of test stimuli with the baseline condition. Since this approach merely selects voxels which are involved in processing a set of stimuli, but makes no further selection this can safely be performed using the entire data set; this is not the case with the following methods however.

Selecting voxels by activity results in a selection of voxels which are involved in processing the stimuli, yet it gives no consideration to the ability of these voxels to distinguish between conditions. In order to select for these characteristics voxels can be selected in a similar manner based on the results of an Analysis of Variance (ANOVA) with the null hypothesis that all task conditions share the same mean. This method increases the complexity of the feature selection however. Since a direct selection based on the voxels' ability to distinguish between test conditions is being made, the data on which the ANOVA is performed must be constrained to the training set in order to avoid bias and overfitting.

The preceding methods both consider the selection of voxels in the terms of individual voxels representing information. When performing a multi-variate pattern analysis it may be desirable to select voxels which contribute to the multi-variate encoding of information instead. One method for this is Recursive Feature Elimination (RFE) which selects voxels based on their contribution to the classification between test conditions as indicated by the magnitude of their linear weights. This selection is performed in an iterative fashion, with a classification model being trained on a set of training data, and the voxel or voxels with the lowest absolute linear weight being discarded at each step. (DeMartino et al., 2008). As with the ANOVA based method above, care must be taken to ensure the independence of the training data on which the RFE is performed from the testing data used in the subsequent classification analysis.

One further method worth mentioning is the Seachlight analysis (Kriegeskorte et al., 2006; Kriegeskorte and Bandettini, 2007). This approach, rather than selecting voxels from within a region of interest, as the above methods tend to be applied, analyses all of the voxels in the brain. This is achieved by performing a separate analysis for each voxel in which the voxels used are those which fall within a spherical "searchlight" centered on the current voxel. These selected voxels may then be further reduced by selection based on a grey matter mask, or by another voxel statistic. This analysis, while computationally expensive, produces a map of the brain indicating for each voxel the degree to which its local neighbourhood is capable of performing the classification in question. This result is interesting in and of itself, however it can further be applied to voxel selection in a similar manner to that of the ANOVA statistic, with the same limitations on independence of the dataset.

There are a number of other approaches to feature selection, such as noise perturbation (Hanson et al., 2004), automatic relevence determination (MacKay, 1992; Neal, 1996; Yamashita et al., 2008), identification of intersecting feature sets (Ban et al., 2012) and the various pre-existing methods from the field of machine learning which are yet to be applied, however a full review of this field is outside the scope of this thesis.

1.3.5 Classification

It is the classifier that lies at the heart of most MVPA analyses. The key idea behind this methodology is that a classifier is able to accurately predict the stimulus based on the neural response to it, then the region of brain from which said response is taken must encode some information relating the stimuli. A variety of classification algorithms have been applied to the task of analysing fMRI data, such as Logistic Regression (Yamashita, et al., 2008; Rayali et al., 2010), Neural Networks (Hanson et al., 2004; Polyn et al., 2005), Gaussian Naive Bayes classifiers (Mitchell et al., 2004), Support Vector Machines (Preston et al., 2008; Li et al., 2007), and Relevant Vector Machines (DeMartino et al., 2011).

Comparisons of these various methods, such as those performed by Pereira (2007) and Norman et al. (2006) have found that linear methods are most suited to this task, with logistic regression and linear support vector machines being particularly suitable (Misaki et al., 2010).

Of these, the linear methods – Logistic Regression, linear Support Vector Machines and Relevance Vector Machines – all approach the task as that of finding a separating hyperplane between two classes through the space defined by the feature vectors. Where these linear methods differ is the manner in which they determine the position of the hyperplane; for example Logistic Regression seeks to minimise the error of predictions, while Support Vector Machines seek to maximise the margin between the two classes.

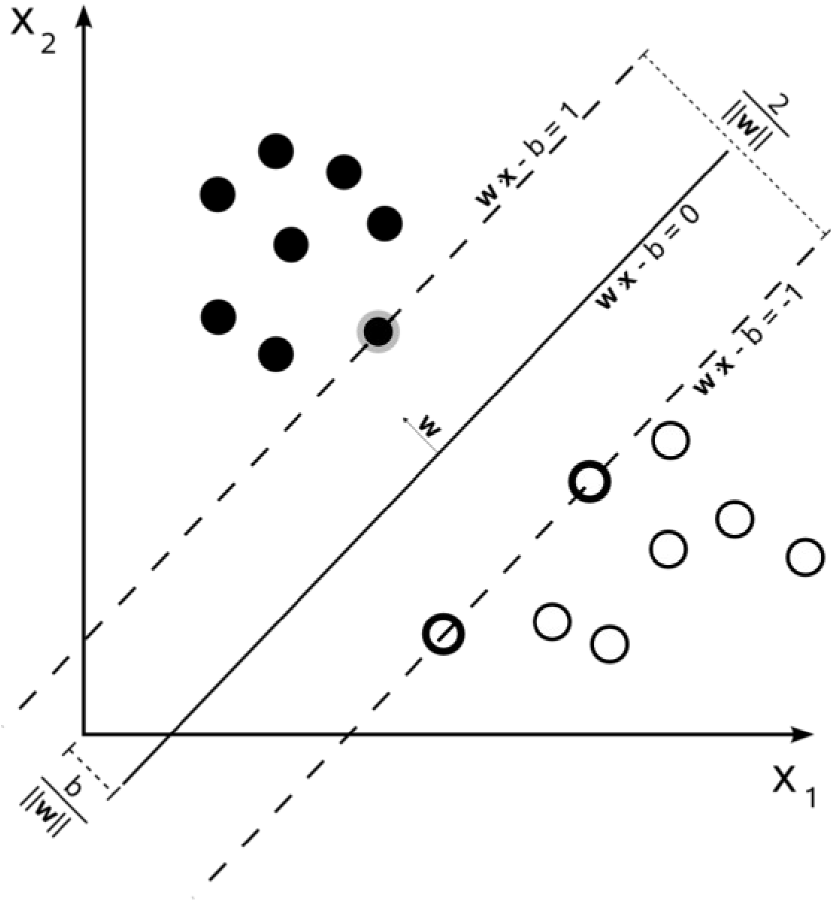


Figure 1.3: Illustration of a hard margin support vector classifier. The decision boundary is shown as a solid diagonal line, with the margins show as dashed lines either side. The samples which fall on the margins are the support vectors. The direction of the weight vector is shown as a small arrow perpendicular to the decision boundary, labelled with **w.** In this figure, **x** denotes the training data, **b** the offset and **w** the weight vector. (Figure from https://en.wikipedia.org/wiki/File:Svm_max_sep_hyperplane_with_margin.png, accessed on 27th October 2015).

The more interesting methods in the field of fMRI, regularised logistic regression and support vector machines, both include a means of reducing the effects of overfitting to which underdetermined systems like fMRI are prone. Regularised logistic regression does this by applying a penalty to the fitness function of models with non-zero weights for higher numbers of features, which results in models which use fewer features. Support Vector machines, instead of minimising prediction error seek to maximise the margin between the two classes. This results in a model which has more likelihood of being generalisable to new data.

1.3.6 Post-processing and visualisation

Though not strictly speaking part of the core analysis process, some consideration should be given to the methods of post-processing and visualisation of the results produced.

One of the key results provided by a classification analysis is the accuracy of the classification. By itself this measure can indicate the presence or absence of information in a selected region of voxels. While the accuracy of the classifier can indicate the presence or absence of information, it is when we apply further analysis to this that more interesting results can be obtained. For example the significance of the result can be established using methods such as shuffling or the binomial theorem (Pereira et al., 2009), while comparisons to behavioural performance can be made using methods such as psychometric functions (Patten, 2013). Furthermore the results of an analysis can serve as the basis for entirely new analyses, such as the case of using a searchlight analysis for feature selection (Kriegeskorte et al., 2009).

The classification accuracy, while important, is not the only result of use. The raw prediction values, and consequent confusion matrix can be useful, as is explored in Chapter 4 of this thesis, and by Zhang, et al. (2010).

CHAPTER 2 - DEVELOPMENT OF THE TOOLBOX

As discussed in the previous chapter, part of the work of this thesis comprises the collection of methods for Multivariate Pattern Analysis (MVPA) of functional Magnetic Resonance Imaging (fMRI) data, and their implementation into a cohesive package using Matlab (The MathWorks, Inc., Natick, Massachusetts, United States).   Though there are other popular projects available which provide tools for multivariate pattern analysis of fMRI data, such as PyMVPA (Hanke et al. 2009), the novel aspect of the application developed as part of the work of this thesis was to provide not only a collection of methods as a basis for further development, but also a collection of standard predefined analysis scripts which have been optimised for efficient computation and tested through application to a number of fMRI data sets in a variety of studies (Zhang et al, 2010; Ban et al 2012; Kuai and Kourtzi, 2011).

The main scripts provided by the toolbox perform multivariate pattern analysis on subsets of predefined regions of interest. Depending on the purpose of the analysis these may be commonly defined visual regions, or alternatively the entire brain. The key components of these scripts, shown in Figure 2.1 are implemented in a modular fashion which can be used to facilitate development of further analyses. The re-use of these common components ensures consistency of the data provided to the various analyses. In addition to these key scripts and components, a number of auxiliary scripts are provided for performing univariate analyses, visualisation and analysis of classification results.

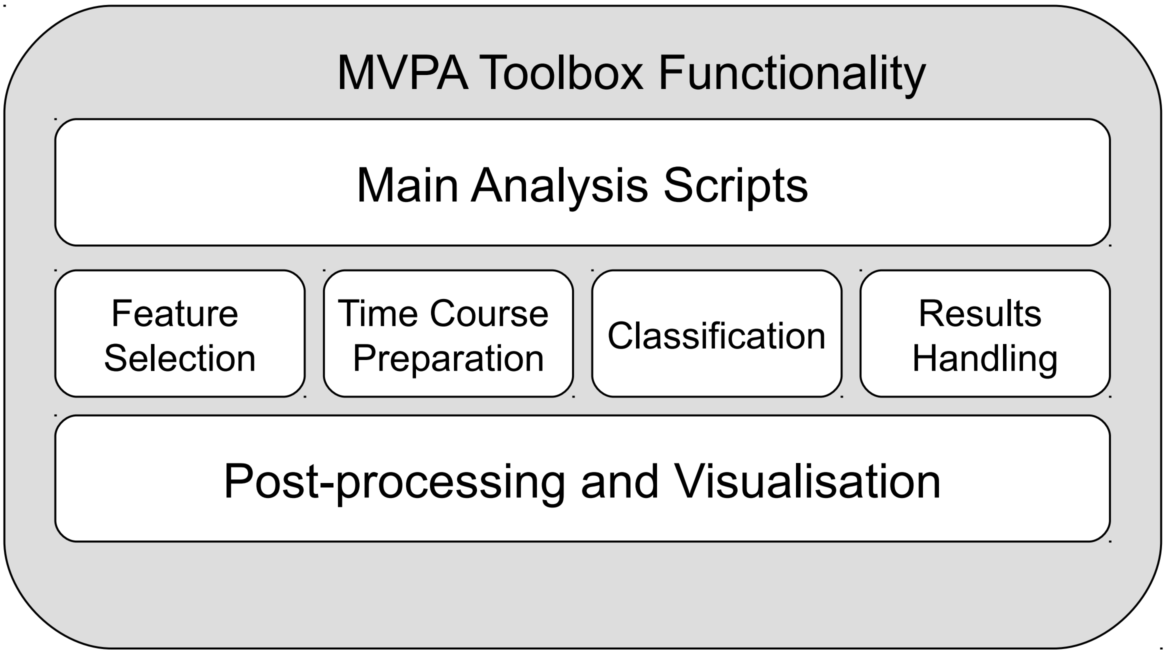


Figure 2.1: Key components of the toolbox

The standard process being used for MVPA was identified through a combination of examination of existing analysis scripts and discussion with members of the Cognitive Neuroimaging Laboratory. A comparison of several variants of multivoxel pattern analysis as used in the laboratory at that time was performed in order to identify the key decision points in the required MVPA analysis scripts, allowing for re-use of common elements. Similar to the work by Pereira et al. (2009) these processes revealed an overall structure common to the various MVPA scripts in use, with several options for the implementation of key sections, primarily sample estimation and feature selection.

From this point the implementation of the toolbox proceded in an iterative fashion. Initial versions concentrated on provision of a single method for each of the key sections in order to establish a working version which provided the most common analysis options. Subsequent iterations of the development cycle introduced further, more complicated methods, such as the mini-blocks sample estimation method and the searchlight feature selection method.

Additionally a system for configuration of the toolbox with regard to which methods to use at each stage and where to source the fMRI data from was implemented. This configuration system allowed for the use of a standard set of scripts to perform number of analyses without modification.

At each stage of development the toolbox was tested on data from studies being conducted in the laboratory, particularly those by Li et al., (2007), but also including those by Zhang et al., (2010), Ban et al., (2012), Kuai et al., (2013) and Patten (2013). Data from these studies were typically provided with initial pre-processing having already been performed using Brain Voyager QX (Brain Innovation B.V.). Testing on these data sets was chiefly conducted in order to verify the correctness of the results, with this being verified by manual inspection of the data at each stage of the analysis and through comparison of the final results with those of the existing study specific scripts where available.

With the addition of the more complicated methods having defined the required flexibility of the overall structure and the correctness of the results having been established through testing, focus was then given to optimisation of the execution speed and memory footprint of the analysis scripts. Execution of the scripts already developed was profiled using Matlab's profiler tool (The MathWorks, Inc., Natick, Massachusetts, United States) in order to identify bottlenecks in terms of both processing and data input/output. As a result of these steps the toolbox was then refactored to avoid duplicate processing, and caching was introduced to limit the number of occasions on which large amounts of timecourse data was read from the hard disk.

Following these optimisation steps the possibility of parallel processing was explored, which resulted in the establishment of a dedicated compute cluster using the Matlab Distributed Computing Server (The MathWorks, Inc., Natick, Massachusetts, United States), and the implementation of a version of the toolbox designed to handle parallel execution of multiple sections of the main MVPA.

The remainder of this chapter presents each of the analysis scripts and components provided by the toolbox.

2.1 Standard Multivariate Pattern Analysis

As previously discussed the toolbox provides four main multivariate pattern analyses. The first three analyses are region of interest based, focusing on determining the extent to which pre-defined regions encode information regarding a set of presented stimuli. These regions are typically identified based on a combination of anatomical location and the response to standard localiser stimuli (Sereno et al, 1995). The remaining analysis, referred to as the searchlight method, is based on the method described by Kriegeskorte et al (2006). This method generates a mapping of the multivariate information content similar to the univariate measures generated using a general linear model (GLM). This is achieved by performing a classification analysis for each voxel in the brain using neighbouring voxels which fall within a spherical 'searchlight'.

Since all these analyses are classification based, they follow the same general structure, as shown in Figure 2.2, with the key differences being in the manner of feature selection used. The greatest variation from the standard process is found in the searchlight analysis, due to the differing amounts of data which must be considered.

One of the reasons for implementing the toolbox in a modular fashion is to allow for the interchangeability of various methods of performing each stage of the analysis process. While the method of feature selection is one of the key factors of an anlaysis, as mentioned above, the options presented are not limited to feature selection, but also cover a number of variations in approach to preparation of the time course data, and definition of the classification analysis.

Where possible this interchangability has been facilitated through implementation of the different analysis methods as separate classes which implement a common interface. This design choice allows for ease of swapping between the various methods, and for future expansion through implementation of additional methods as new classes.

The selection of the particular method used to perform each stage is conducted in a configuration file, which is also responsible for providing the location of all the files containing the data required for the analyses. This separate configuration file provides an extra level of traceability of the analysis, providing a linked record of what specific methods were used and precisely what data they were applied to. Additionally, customising the analyses through a configuration file makes modification of the analysis scripts themselves unnecessary, facilitating performance of the analysis on compute clusters to which the scripts have already been deployed.

In the following sub-sections the key components the toolbox provides for the multivariate analysis scripts are discussed, along with the various methods which can be applied in each of them.

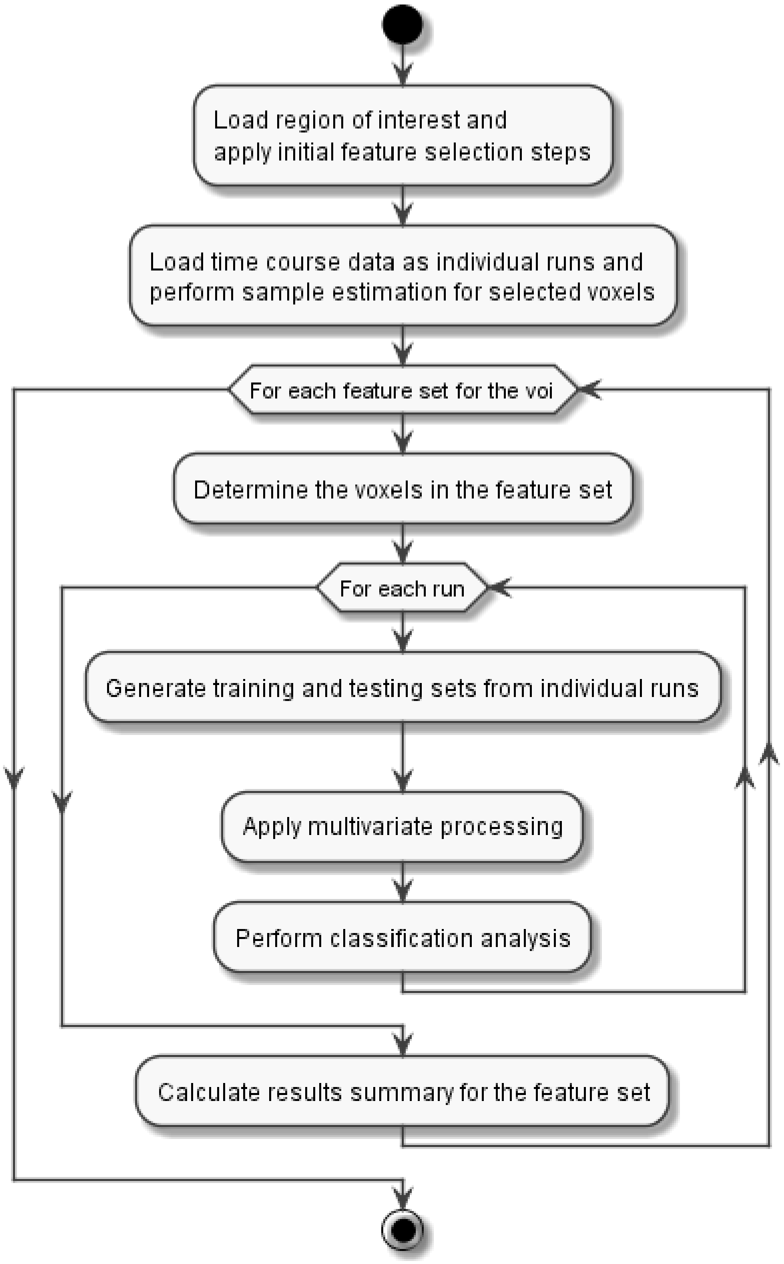

Load region of interest and apply initial feature selection steps
Load time course data as individual runs and perform sample estimation for selected voxels
For each feature set for the voi
Determine the voxels in the feature set
For each run
Generate training and testing sets from individual runs
Apply multivariate processing
Perform classification analysis
Calculate results summary for the feature set

Figure 2.2: Activity diagram showing the high level process for MVPA implemented in the toolbox.

2.1.1 Feature Selection

When performing multivariate pattern analysis on fMRI data sets the process of feature selection serves several purposes. The chief purpose of feature selection is to improve the performance of the classification models being applied. This is accomplished in part by addressing the high dimensionality typical of fMRI data; a single region of interest can contain hundreds or thousands of voxels, so it is often the case that the number of features greatly exceeds the number of samples available for the classification. It is this under determined nature which makes fMRI data particularly susceptible to the Hughes effect (Oommen et al, 2008; Hughes, 1968), in which increasing dimensionality for a fixed sample size reduces the predictive power of the model.

One method of avoiding the over fitting which often occurs in this situation would be to apply classification models which employ regularisation. Regularisation penalises complex models and depending on the specific regularisation method can produce sparse models, resulting in an implicit feature selection being performed within the learning algorithm.

An alternate approach, and the one used by the toolbox during the feature selection stage, is to apply reduce the number of features to which the classification models are applied, and thereby improve the performance of the models and reduce overfitting. This reduction of the number of feature is achieved by selecting features for inclusion in the classification based on properties of the feature. This can contribute to reducing the noise in the data by selecting voxels which are believed to encode information relevent to the classification being performed, and thus eliminating voxels which are likely to contain only noise.

While the chief purpose of feature selection is to reduce the dimensionality of the data set, it can also serve to help determine the meaning of the analyses by specifying the characteristics of the voxels on which they are performed. An example of this is selecting voxels which have already been shown to be involved in the processing of a particular visual cue as the feature set to be used for classification of a second set of stimuli which encode similar information by means of a different visual cue (Ban et al, 2012).

The implementation of feature selection in this toolbox is split into two stages. The first stage is primarily responsible for loading pre-defined regions of interest, and performing preliminary feature selection using univariate statistical methods. These portions of the toolbox are implemented in the `VoiList`, `RawVoiList` and `GreyMatterVoiList` classes which extend the abstract `RegionList` class to provide the resulting regions of interest in a linked-list style format.

The second stage is responsible for the more complex feature selection steps, such as the multivariate recursive feature elimination method, as well as providing subsets of the ROIs to allow processing of feature sets with different sizes and locations. These modules implement a common `FeatureSelector` interface, which provides a similar linked-list style access, with the addition of providing a cache specification to identify for which voxels time course data must be cached.

While Figure 2.3 shows that each of the `RegionList` class utilises a single `FeatureSelector` class, they are still implemented separately rather than being combined, as this allows for their use in other analyses which do not require the further feature selection steps, such as the univariate analysis discussed in Section 2.2, and also allows for further development where this correspondence may not hold.

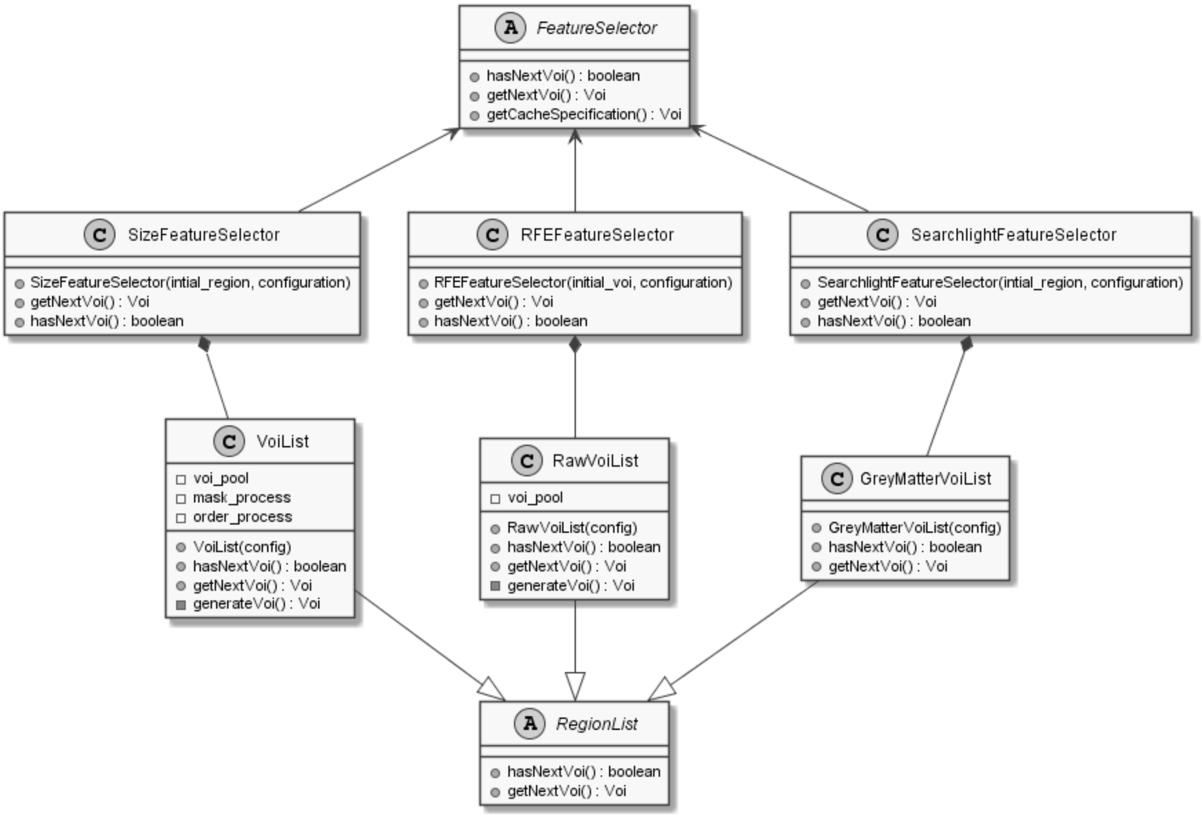


Figure 2.3: Simplified UML class diagram for feature selection classes.

2.1.1.1 Determining of Regions of Interest

As discussed earlier in Section 2.1 the toolbox implements four main analyses, which differ primarily in their approach to feature selection. These analyses as previously noted, fall into two groups: those focusing on the behaviour of specific regions of interest, and those considering the whole brain. It follows that similar divisions are found in the sections of the toolbox responsible for identifying the regions of interest, and selecting the relevent voxels from them.

The initial identification of regions of interest is performed manually outside the toolbox. For voi-based analyses this is typically done through the identification of standard regions of interest, using a combination of anatomical and functional localiser scans such as those described by Sereno et al (1995), while for whole brain analyses this is performed solely by use of the anatomical scans to generate a mask indicating which voxels are grey-matter, and thus of interest to our analysis.

Corresponding to the four main analyses, the toolbox also employs four main methods for the identification of voxels which are relevent to the analyses:

- **Raw region of interest** which uses no further processing after the initial definition of the region of interest, but rather relies on subsequent methods to identify the relevent voxels.
- **Grey Matter Masking** which limits the voxels to only those which are indicated as grey matter by a pre-defined mask.

- **Static Masking and Ordering** which selects voxels based on a single set of statistics per participant
- **Dynamic Masking and Ordering** which selects voxels based on a separate set of statistics per cross-validation set.

The first two of these methods are primarily used with recursive feature elimination (RFE) and searchlight feature selection methods respectively, and rely on these to make further selection of relevent voxels. The latter two methods – static and dynamic masking and ordering – are chiefly used by the voi-size feature selection method. These static and dynamic region of interest selection methods are similar, sharing a common processing structure as shown in Figure 2.4.

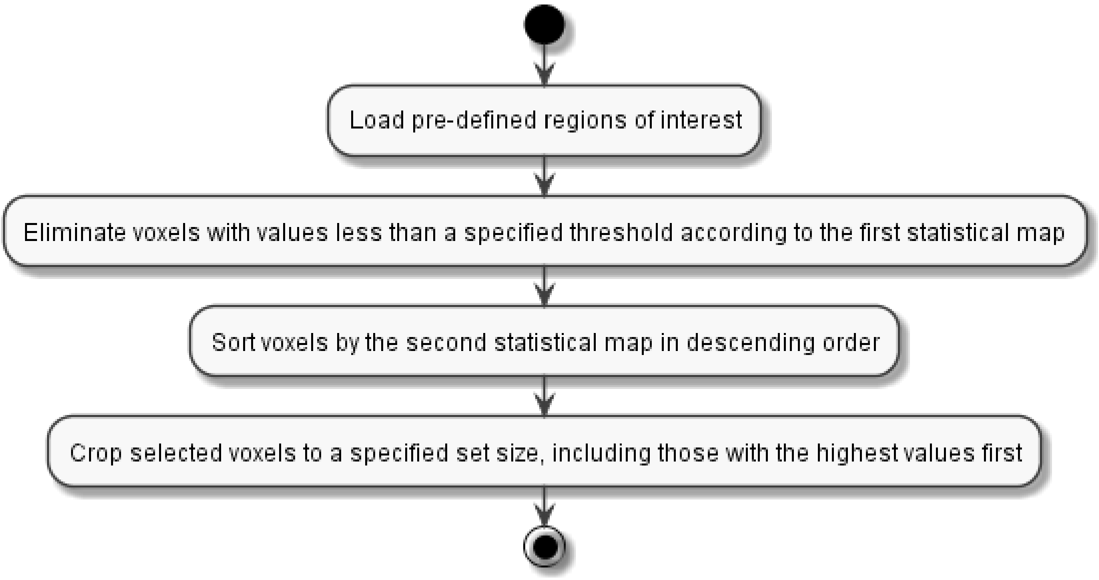

Load pre-defined regions of interest
Eliminate voxels with values less than a specified threshold according to the first statistical map
Sort voxels by the second statistical map in descending order
Crop selected voxels to a specified set size, including those with the highest values first

Figure 2.4: UML activity diagram for masking and ordering based feature selection.

```
voxels = load the set of voxels from the pre-defined region of interest.

mask_statistics = load the voxel statistics from the pre-calculated map.
voxels = voxels(mask_statistic > masking threshold)

order_statistics = load the voxel statistics from the pre-calculated map.
voxels = sort voxels in descending order by order_statistic
voxels = crop to first desired_num_voxels voxels
voxel_set = create StaticVoi(voxels)
```

Listing 2.1: Pseudo code for applying static masking and ordering to a region of interest.

```
voxels = load the set of voxels from the pre-defined region of interest.
statistics = load the voxel statistics from the pre-calculated map.
voxels = voxels(statistic > masking threshold)

for each test run
      dynamic_statistics = load voxel statistics for this test run
      run_voxels{run_index} = sort voxels in descending order by
            dynamic_statistics
      run_voxels = crop to first desired_num_voxels run_voxels
end for
voxel_set = create DynamicVoi(run_voxels)
```

Listing 2.2: Pseudo code for applying dynamic masking and ordering to a region of interest.

As shown in Figure 2.4 and Listings 2.1 & 2.2, the process consists of two key steps: first applying thresholding to the values in a pre-calculated statistical map to identify voxels which show a significant involvement in processing of the presented stimuli, then cropping the remaining voxels to a specified number of the voxels showing the highest involvement, as determined by sorting using a second satistical map, which may potentially be the same as the first.

The static and dynamic methods share the first masking stage in common; and while both also apply the sorting and cropping stage, the key difference between the two lies in the nature of the statistics which can be employed here.

The requirements of the static method are the same as those of the first step; since both use a single set of statistics for all cross-validation sets it is required that the statistics used are not directly related to the main contrast used in the analysis. A common approach, and one recommended for use with the toolbox, is to use the degree to which voxels respond to the set of all stimuli; by contrasting this activation with baseline levels using a t-test, a measure of the significance of the response of each voxel can be generated. Since this contrast does not pre-select voxels which show a difference in response between the various stimulus classes it is safe to include the entire dataset in the calculation.

The dynamic method allows for the use of statistics which are directly related to the main contrast, such as an ANOVA test of the activation evoked by each stimulus category, by using a separate set of statistics for each cross-validation. By using multiple statistical maps we can ensure that each is generated using only samples belonging to the training set of that cross-validation fold, and so avoid breaking the independence of training and testing data.

These processes for region of interest selection are implemented as a set of three classes as shown in Figure 2.3. The raw region and searchlight methods are each handled by their own class, while the static and dynamic masking and ordering processes are both implemented by the `VoiList` class, due to their common process. These modules are implemented separately from the feature selection methods which use them to allow for re-use in other scripts, such as the univariate analyses discussed in Section 2.2 which do not require further feature selection.

The resulting voxel sets produced by these region of interest selection processes are encapsulated in classes which extend the abstract class `Voi`. For processes which result in the same set of voxels for all cross-validation sets, such as static masking and ordering, this is the `StaticVoi` class, while for the dynamic masking and ordering process this is the `DynamicVoi` class. By encapsulating these voxel sets, the rest of the toolbox can retrieve the required set of voxels for the current cross-validation set using a common interface and without needing to know how they were selected.

2.1.1.2 Feature Set Selection

While the previous stage of the overall feature selection process is concerned with the identification of regions of relevent voxels, this stage is responsible for the generation of multiple sub-sets of the regions as determined by controlling the size or location of the desired feature set. Three methods for this were implemented, which correspond to the main MVPA scripts; voi-size selection, recursive feature elimination and searchlight.

2.1.1.2.1 Voi-Size

The voi-size feature selection method relies on the univariate masking and ordering processes as described in Section 2.1.1 for identification of voxels which are likely to encode information relating to the stimuli. This stage of the feature selection deals exclusively with controlling the size of the feature sets. Sub-sets of the identified region of interest are processed with increasing numbers of voxels. By performing the classification analysis on each of these feature sets a mapping between the number of voxels included and the resulting classification analysis is generated. This allows for the subsequent comparison of the regions of interest in terms of the accuracy of classification for a fixed size of feature set.

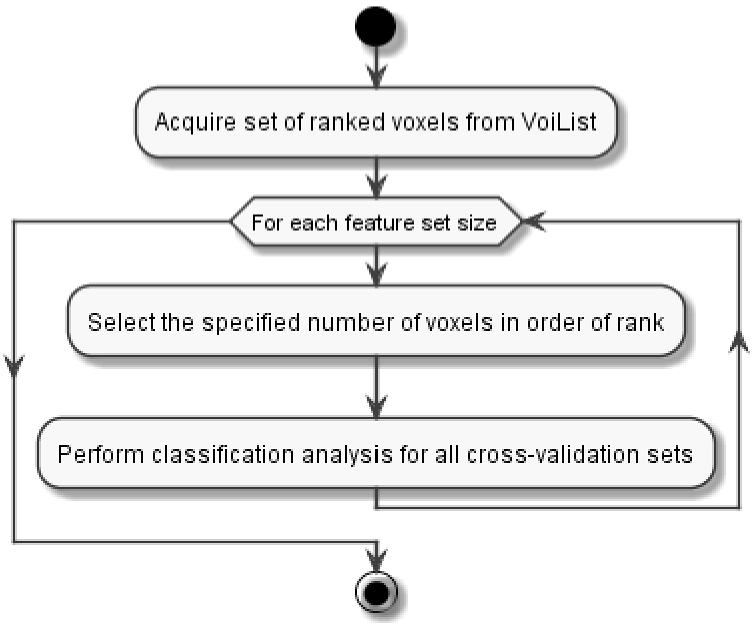

Acquire set of ranked voxels from VoiList
For each feature set size
Select the specified number of voxels in order of rank
Perform classification analysis for all cross-validation sets

Figure 2.5: Activity diagram for voi-size based feature selection.

2.1.1.2.2 Recursive Feature Elimination

The toolbox implements the recursive feature elimination (RFE) method detailed by DeMartino et al (2008). While the voi-size based feature selection method relies on univariate statistics to identify voxels which are involved in processing the stimuli being classified, the RFE method utilises a multivariate approach. The concept behind this method is that by examination of the weights given to each voxel through training the classification model, the degree to which each voxel contributes to the classification can be determined.

This is particularly apparent with linear classification models such as the linear SVM, where the single weighted sum model allows for a simple interpretation of the importance of each voxel as the absolute magnitude of the linear weight assigned to it. Naturally this only holds where all voxels are scaled to a common range, as otherwise the linear weight assigned to the voxel would not only be a measure of the voxel's importance, but also a scaling factor; this stipulation indicates the importance of normalisation for RFE based analysis.

Since the RFE process directly identifies voxels which are capable of distinguishing between stimuli, it is important that the test set is not involved in this feature selection. As such the RFE procedure is applied only to the training set of each cross-validation fold. By repeatedly identifying the voxel contributions in this manner and eliminating the voxels which contribute the least to classification at each repetition, a feature set can be identified which contains only the voxels which contribute to classification. In order to obtain a stable estimate of the contribution of the contribution of each voxel, the value used to determine which voxels to remove is calculated as the mean of the contribution of the voxel to the classification over a number of subsets of the training data. These subsets are generated by random sampling with replacement. As a compromise between the stability of the estimate and the computation time the default number of sets used is five, however this is a parameter which can be set by the user. As with the voi-size analysis, the performance of the classifier on each of the feature set sizes is evaluated on the test folds.

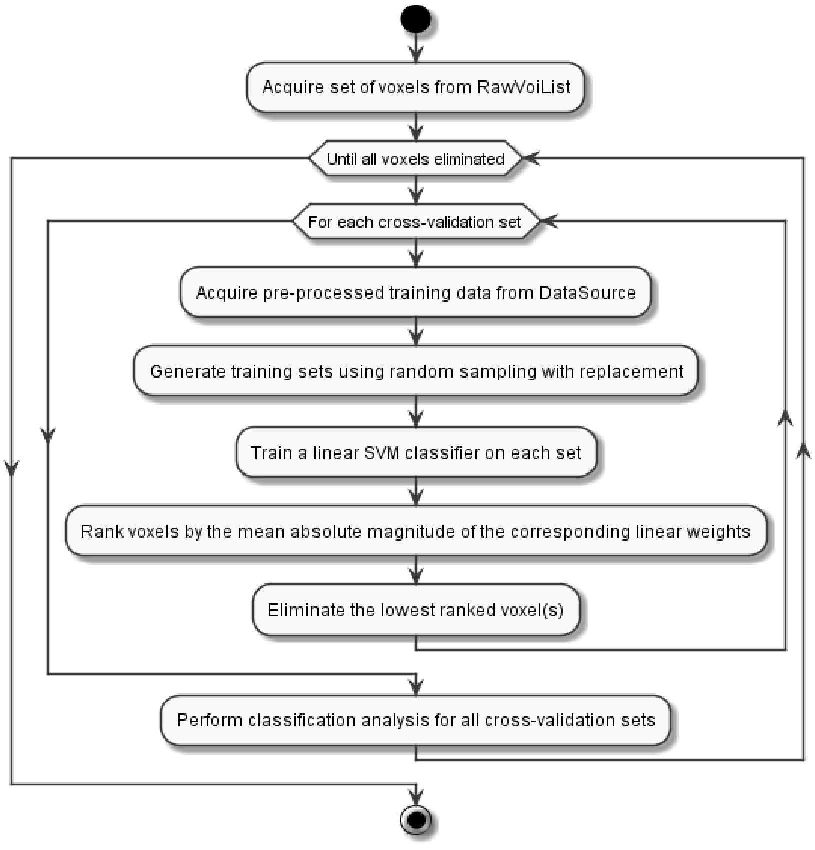

Acquire set of voxels from RawVoiList
Until all voxels eliminated
For each cross-validation set
Acquire pre-processed training data from DataSource
Generate training sets using random sampling with replacement
Train a linear SVM classifier on each set
Rank voxels by the mean absolute magnitude of the corresponding linear weights
Eliminate the lowest ranked voxel(s)
Perform classification analysis for all cross-validation sets

Figure 2.6: Activity diagram for recursive feature analysis.

```
voxels = the pre-existing set of voxels.
data = load timecourse sample data for voxels from cache.
labels = load timecourse class labels from cache.

for si = 1 .. num_subsets_to_use
      subset_data = randomly sample num_samples - num_to_exclude from data
      subset_labels = select labels corresponding to above.

      model = train linear svm classifier(subset_data, subset_labels).
      score(si,:) = abs(model.linear_weights).
end for

mean_score = mean of score across all subsets.
voxels = sort voxels descending by mean score.
voxels = voxels, with the lowest ranked num_to_discard eliminated.
```

Listing 2.3: Pseudo code for RFE voxel ranking an elimination.

2.1.1.2.3 Searchlight

The searchlight method was described in a paper by Kriegeskorte et al. (2006) as a means of generating an information based functional mapping. In contrast to activation based methods which localise function based on change in average activation for a region across conditions, the searchlight method considers changes in patterns of activation. In order to generate the map a separate multivariate analysis is performed for each voxel in the brain, or the selected subsection thereof. The set of voxels involved in each analysis is defined as those which fall within a sphere centred on the voxel for which the analysis is performed. In the original paper the analysis uses the Mahalanobis distance, however the method implemented in the toolbox performs a classification using a linear SVM classifier (Vapnik, 1995; Chang and Lin, 2011; Joachims, 1999).

It is for this method that the provision of a set of voxels to cache becomes relevant. While the other methods deal with a single region at once, the searchlight method considers the whole brain. Holding the timecourse data required for this in memory is not feasible, particularly with higher resolution scans. The toolbox limits the data cached to that which is required for the next arbitrary number of searchlight spheres, redetermining this cache once it has been exhausted.

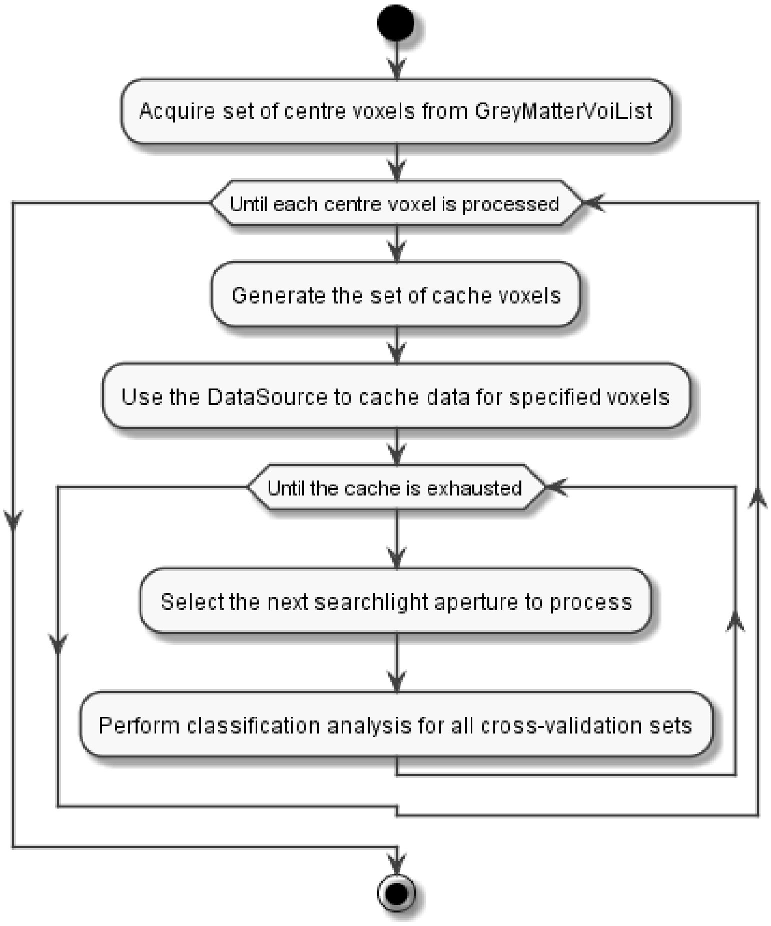

Acquire set of centre voxels from GreyMatterVoiList
Until each centre voxel is processed
Generate the set of cache voxels
Use the DataSource to cache data for specified voxels
Until the cache is exhausted
Select the next searchlight aperture to process
Perform classification analysis for all cross-validation sets

Figure 2.7: Activity diagram for searchlight feature selection.

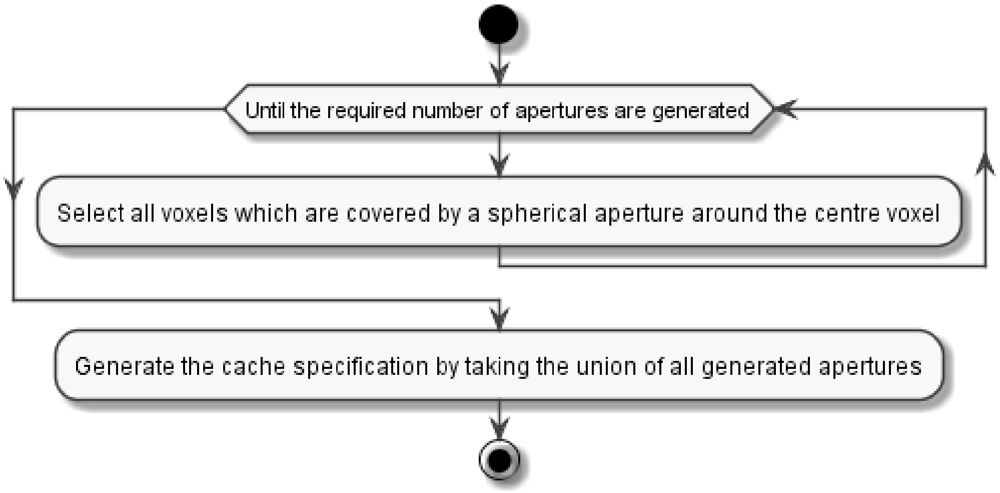

Until the required number of apertures are generated
Select all voxels which are covered by a spherical aperture around the centre voxel
Generate the cache specification by taking the union of all generated apertures

Figure 2.8: Activity diagram for generation of searchlight cache.

2.1.2 Timecourse data preparation

The timecourse data preparation components are responsible for every step in the process of converting the volume timecourse provided by Brain Voyager QX (Brain Innovation, Maastricht, Netherlands) into the samples which are presented to the classifier. This includes timecourse normalisation and sample estimation, caching and the use of cross-validation to separate the data into training and testing sets.

By encapsulating these processes into a set of components which are collated behind a single interface, the process of acquiring a set of timecourse data that has been prepared for classification is simplified to requesting the data for a specified cross-validation fold and set of features.

As with the feature selection components there are two main stages to the time course data preparation processing. The first is responsible for performing the preprocessing of run time courses on an individual basis; this includes the application of normalisation and sample estimation steps. Caching data at this point in the processing takes advantage of the independent nature of the runs, also by caching after the sample estimation there is a significant reduction in the quantity of data to hold in memory. This stage of the analysis is implemented in the `PreprocessedData` classes shown in Figure 2.9.

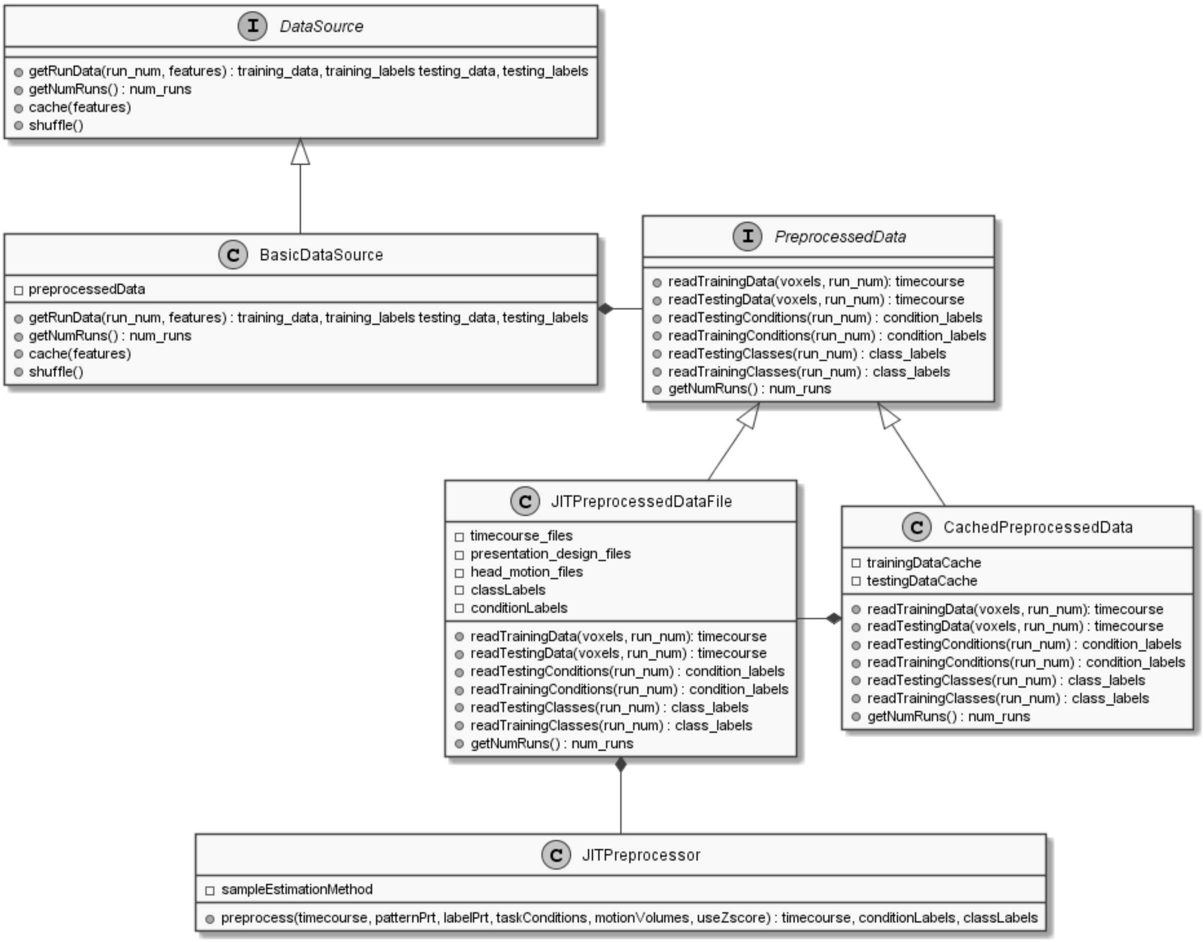

I DataSource
getRunData(run_num, features) : training_data, training_labels testing_data, testing_labels
getNumRuns() : num_runs
cache(features)
shuffle()
C BasicDataSource
preprocessedData
getRunData(run_num, features) : training_data, training_labels testing_data, testing_labels
getNumRuns() : num_runs
cache(features)
shuffle()
I PreprocessedData
readTrainingData(voxels, run_num): timecourse
readTestingData(voxels, run_num) : timecourse
readTestingConditions(run_num) : condition_labels
readTrainingConditions(run_num) : condition_labels
readTestingClasses(run_num) : class_labels
readTrainingClasses(run_num) : class_labels
getNumRuns() : num_runs
C JITPreprocessedDataFile
timecourse_files
presentation_design_files
head_motion_files
classLabels
conditionLabels
readTrainingData(voxels, run_num): timecourse
readTestingData(voxels, run_num) : timecourse
readTestingConditions(run_num) : condition_labels
readTrainingConditions(run_num) : condition_labels
readTestingClasses(run_num) : class_labels
readTrainingClasses(run_num) : class_labels
getNumRuns() : num_runs
C CachedPreprocessedData
trainingDataCache
testingDataCache
readTrainingData(voxels, run_num): timecourse
readTestingData(voxels, run_num) : timecourse
readTestingConditions(run_num) : condition_labels
readTrainingConditions(run_num) : condition_labels
readTestingClasses(run_num) : class_labels
readTrainingClasses(run_num) : class_labels
getNumRuns() : num_runs
C JITPreprocessor
sampleEstimationMethod
preprocess(timecourse, patternPrt, labelPrt, taskConditions, motionVolumes, useZscore) : timecourse, conditionLabels, classLabels

Figure 2.9: Simplified UML class diagram of data handling classes.

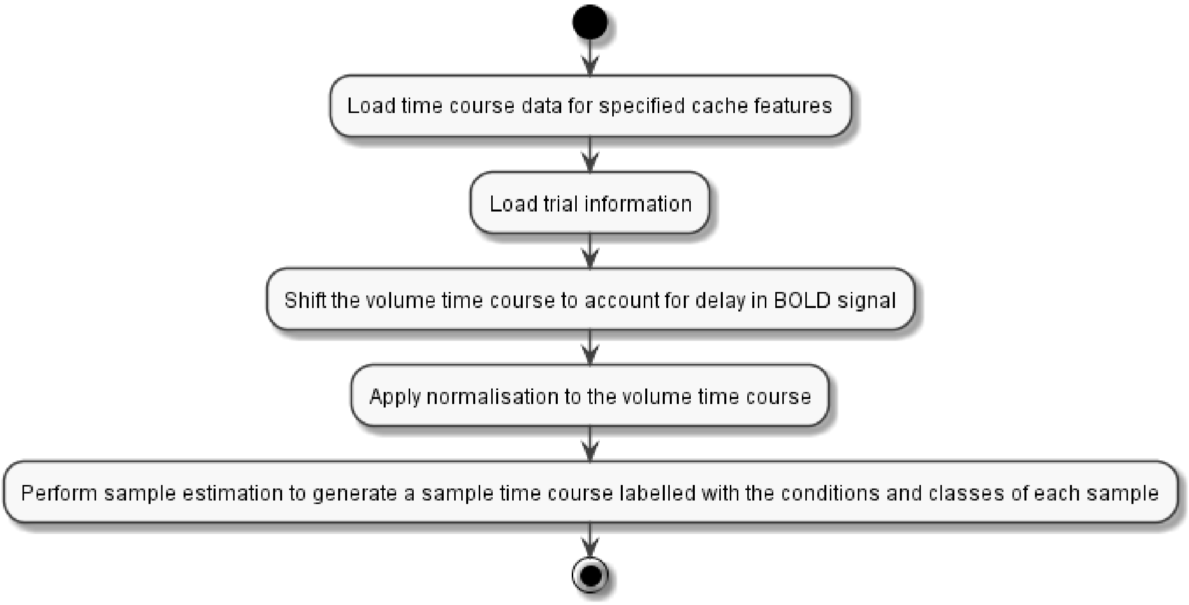

Load time course data for specified cache features
Load trial information
Shift the volume time course to account for delay in BOLD signal
Apply normalisation to the volume time course
Perform sample estimation to generate a sample time course labelled with the conditions and classes of each sample

Figure 2.10: UML activity diagram for run level timecourse pre-processing.

The second stage of the analysis performs the generation of cross-validation training and testing sets. The standard analyses implemented in the toolbox use leave-one-run-out cross-validation, and as such each run is used as a single fold. In addition to the cross-validation the this stage of the analysis applies the optional steps for removing the univariate signal from the timecourse data to enable the classification to focus on the multivariate signal. This stage corresponds to the `DataSource` classes shown in Figure 2.9.

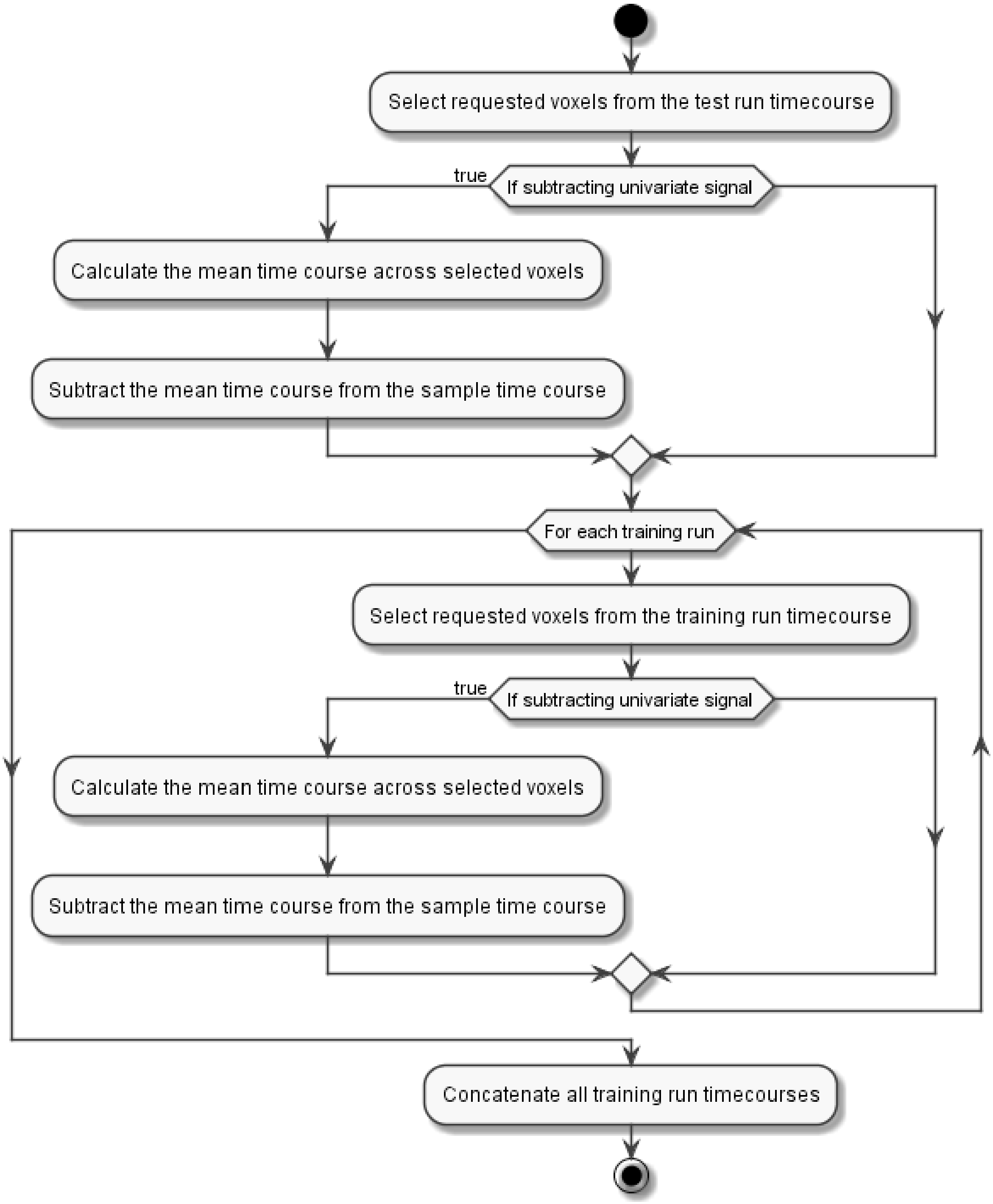

Select requested voxels from the test run timecourse
true
If subtracting univariate signal
Calculate the mean time course across selected voxels
Subtract the mean time course from the sample time course
For each training run
Select requested voxels from the training run timecourse
true
If subtracting univariate signal
Calculate the mean time course across selected voxels
Subtract the mean time course from the sample time course
Concatenate all training run timecourses

Figure 2.11 UML activity diagram for cross-validation and univariate signal subtraction.

```
timecourse = timecourse data loaded from datasource
for each volume, vi in timecourse
      volume_tc = timecourse(vi, :)
      volume_mean = mean(volume_tc)
      timecourse(vi,:) = volume_tc - volume_mean.
end for
```

Listing 2.4: Pseudo code for univariate signal subtraction.

2.1.2.1 Normalisation

The typical design of fMRI studies results in a set of runs collected in series, and as mentioned earlier in the thesis there are factors which change the magnitude of the BOLD response to the stimuli over the course of the analysis. Additionally the degree of the BOLD response varies with the location of the voxels being measured due to factors such as blood flow and depth from surface. The key application of normalisation in MVPA is to ensure that the distribution formed by these BOLD responses falls on a common scale.

The normalisation step of the time-course preparation module applies *z*-score normalisation for each voxel across all volumes in the run, resulting in time course data that is scaled to a standard deviation of one and centred around the mean. This ensures that runs can be considered to come from the same distribution and thus allows prediction to be made across runs.

```
timecourse = timecourse data loaded from datasource
for each voxel, vi in timecourse
      voxel_tc = timecourse(:, vi)
      voxel_mean = mean(voxel_tc)
      voxel_std = standard deviation(voxel_tc)
      timecourse(:, vi) = (voxel_tc - voxel_mean) / voxel_std.
end for
```

Listing 2.5 Pseudo code for z-score normalisation.

2.1.2.2 Sample Estimation

At the sample estimation stage of the time-course pre-processing the time-course is a sequence of volumes, the activations for which have been normalised to a standard scale. In order to perform a classification analysis on these data it is desirable to have a single pattern representing the response generated by the trial or block as indicated by the design, however at this stage there are multiple volumes per trial, typically 2, and even more per block in block design studies. The sample estimation step is responsible for generating a pattern from these volumes and doing so in a way which minimises noise.

There are a number of ways of doing this ranging from selecting the signal intensity from a single volume (Polyn et al., 2005), to using a GLM to deconvolve the overlapping responses and using the betas of the resulting model (Worsley and Friston, 1995; Kriegeskorte et al., 2007). However, this toolbox implements an effective, yet simple method; that of taking the mean response of all volumes in each trial/block (Kamitani and Tong, 2005). This simple method has the benefit of being particularly efficient in terms of computation.

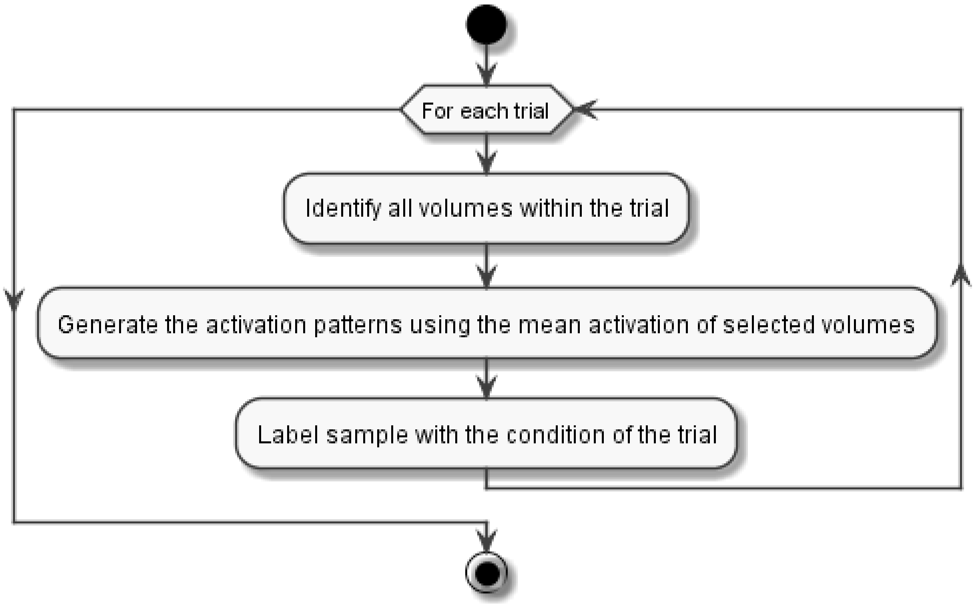

For each trial
Identify all volumes within the trial
Generate the activation patterns using the mean activation of selected volumes
Label sample with the condition of the trial

Figure 2.12: Activity diagram for sample estimation.

```
trial_indices = index of each volume within the trial
trial_volumes = timecourse(trial_indices, :)

for each voxel, vi in timecourse
      voxel_tc = volumes(:, vi)
      sample_values(vi) = mean(voxel_tc)
end for
sample_label = timecourse label(trial_indices(1))
```

Listing 2.6: Pseudo code for trial averaging.

As shown in Figure 2.10, prior to the generation of samples the time-course needs to be shifted relative to the onset and offset times indicated by the trial information to account for the delay in the BOLD signal, since the trial information provided by the design files refer to the stimulus presentation times, rather than the peak response.

By taking the mean of the volumes for a sample the noise in the signal is reduced. In the case of block design experiments, in which a single block will contain multiple presentations of variants on a class of stimuli, taking the average of the responses reduces the effect of the individual variations and results in a measure more representative of the common properties of the class of stimuli.

Event related designs do not have this advantage, and as such their timecourses are signficantly more noisy. A step towards countering this is the 'mini blocks' method, in which having applied the averaging method, the resulting samples are then grouped together into a specified number of sets, the contents of which are then averaged.

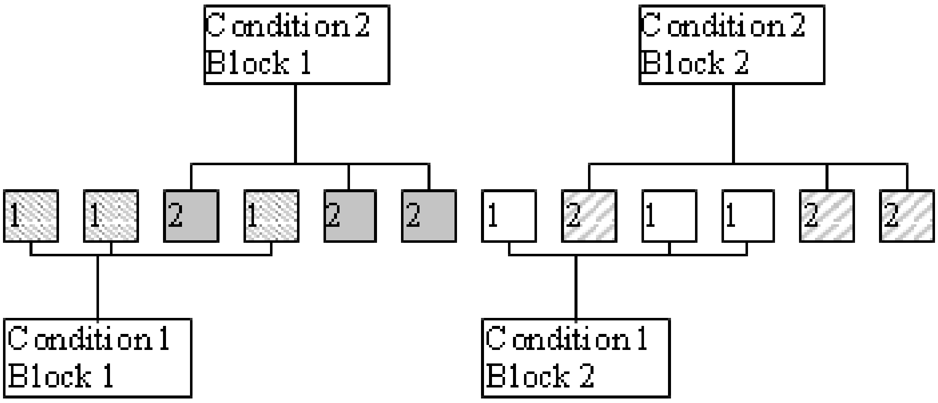


Figure 2.13: Composition of mini-block samples.

It is not uncommon for the trials in each run not to divide evenly into mini blocks, resulting in the final mini block produced being smaller than the others. When the difference in size is only one or two trials this does not make too great a difference, however when the final block is left containing only one or two trials it is likely too noisy to be useful. In this implementation, when the final block is less than half the size specified it is merged in with the penultimate block.

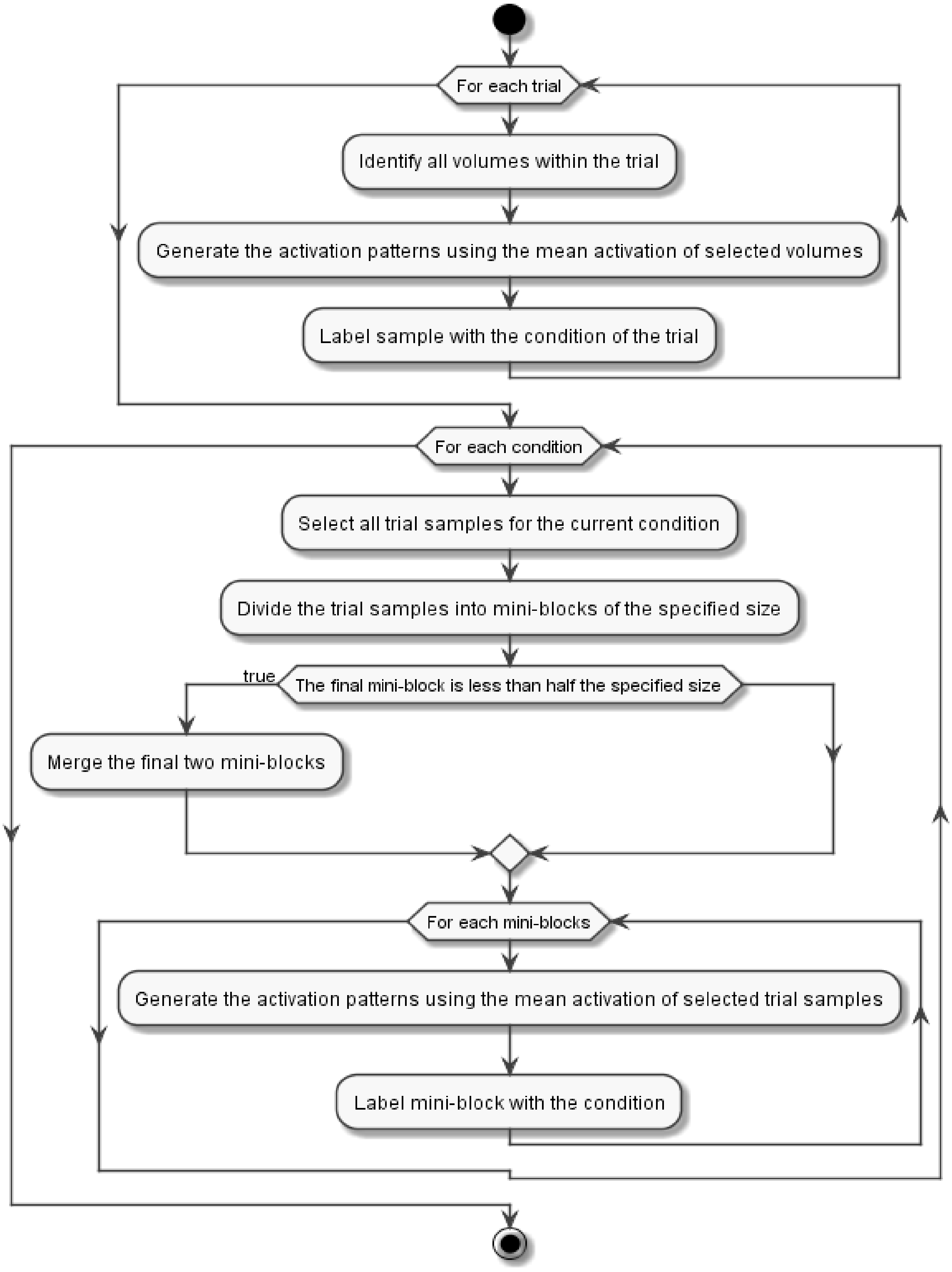

For each trial
Identify all volumes within the trial
Generate the activation patterns using the mean activation of selected volumes
Label sample with the condition of the trial
For each condition
Select all trial samples for the current condition
Divide the trial samples into mini-blocks of the specified size
true
The final mini-block is less than half the specified size
Merge the final two mini-blocks
For each mini-blocks
Generate the activation patterns using the mean activation of selected trial samples
Label mini-block with the condition

Figure 2.14: Activity diagram for mini-block sample estimation.

```
trial_samples = previously generated trial samples.
trial_labels = previously generated trial labels.

for each condition
  cond_trial_labels = trial_labels(trial_labels == current condition).
  cond_trial_samples = trial_samples(trial_labels == current condition).

  number of miniblocks = number of condition trials / desired size.
  remainder = number of condition trials modulo desired size.
  miniblock_indices = group trials into sets of desired size.
  if remainder < desired size / 2
    combine the miniblock_indices for the final two miniblocks.
  end if

  for each miniblock
    miniblock_data = mean(cond_trial_samples(in this miniblock)).
    miniblock_label = cond_trial_samples(first trial in this miniblock).
    append miniblock_data to sample_data.
    append miniblock_label to sample_labels.
  end
end
```

Listing 2.7: Pseudo code for mini-block composition.

2.1.2.3 Caching

The classification analyses implemented in this toolbox are typically applied multiple times to the same data, varying in the set of selected voxels used. Rather than re-loading and processing the time-course data each time, which would impose a significant overhead, the time-course data are cached.

The caching is handled by the `CachedPreprocessedData` class, which wraps the standard `PreprocessedData` classes, as shown indicated Figure 2.9. The `CachedPreprocessedData` class maintains a cache of the timecourse data for both training and testing sets, along with a set of voxels to which the cached data relates. All requests to instances of the `CachedPreprocessedData` class for time-course data are first checked to see if the requested data is already in the cache; the only data loaded is that which relates to voxels not already present in the cache, as shown in the following algorithm:

```
If (the cache has no data for one or more requested voxels),
      Use the PreprocessedData object to load/preprocess missing data.
      Add the data to the cache.
End If
Return the data for the requested voxels.
```

Listing 2.8: Pseudo code algorithm for accessing timecourse data via a `CachedPreprocessedData` object.

By caching at this stage, after the sample estimation has been performed, the necessity of reapplying the sample estimation is also avoided and the quantity of data which are to be cached is reduced. A further benefit arises from the fact that the toolbox primarily relies on leave-one-run-out cross-validation. Since each run, and thus cross-validation fold, is sample estimated independently; caching at this stage simplifies the generation of training and testing sets to the concatenation of the already prepared training runs. Since the decision of which voxels to cache data for is dependent on the feature selection method used, the `CachedPreprocessedData` class focuses solely on storing and sourcing the data, relying on the `FeatureSelection` classes to specify the required voxels.

2.1.2.4 Cross-validation

In the typical fMRI study there are a limited amount of data available due to the cost and difficulty of collection. To avoid the need to dedicate a relatively large proportion of the data to testing it is common to employ cross-validation. By holding out a relatively small portion of the data, but performing multiple analysis while rotating which portion is held out, it is possible to generate a good measure of the performance of the classification algorithm while retaining a large training set for building the models.

When dealing with fMRI data the most commonly used method is leave-one-run-out cross-validation. The leave-one-run-out cross-validation method is derived from *k*-fold cross-validation, in which data sets are divided into *k* discrete folds. Each of these *k* folds is used as the test set in turn, with the remaining sets being combined to form the training sets. The leave-one-run-out method makes use of each run of the fMRI data as a single fold since they are already separated into independent sets of samples. This method is further helped by the fact that design of the experiment can be tailored to ensure certain properties which are beneficial to the cross-validation. As discussed by Kohavi (1995), a *k*-fold cross-validation will benefit from having approximately 10 folds, particularly if the folds are stratified. By designing experiments such that the runs number in the region of 8-10, and contain an equal number of presentations of the stimuli from each class, these properties can be provided.

Naturally there are some situations in which providing 8 runs can be problematic, such as cases where one or more runs have been excluded due to excessive noise or head motion during the scan. Also ensuring an equal number of stimuli from each class may not be possible, for example if the categories being classified are the participant's responses to stimuli. In situations such as these it may be desirable to use other cross-validation methods which can ensure greater numbers of balanced folds themselves, such as leave-*p*-out cross-validation; however using such methods is by no means as simple as using leave-one-run-out cross-validation. The overlapping nature of the haemodynamic response to stimuli means that adjacent trials are not independent; while this can be accounted for when said trials belong to the same fold, it can violate the requirement of independence of training and testing sets if adjacent trials are placed in separate folds. Further discussion on the application of alternative methods of cross-validation is provided in Chapter 3.

The cross-validation step is implemented in the toolbox as part of the `DataSource` class, and performed following the sample estimation. The independent nature of the time course runs allows for pre-processing up to the sample estimation stage to be performed separately on each run. By dealing with the runs independently up to this stage the data can be cached rather than needing to perform sample estimation separately for each cross-validation step. After this, the cross-validation process merely comprises the concatenation of the training folds.

2.1.2.5 Removal of univariate signal

The purpose of most classification based multivariate pattern analysis is to determine whether an identified region of the brain encodes any information with regards to the contrast between a set of stimuli; however when performing classification the algorithm makes no distinction between information of a univariate or multivariate nature.

In order to identify regions which specifically encode information in a multivariate manner the toolbox provides the option of subtracting the mean univariate activation time-course from the time-course data. This serves to eliminate whole region univariate responses to specific stimuli, and ensure that any ability to classify stimuli based on the remaining data is due to a multivariate signal.

This is performed as the final step before classification, since in order to calculate the mean signal for the time-course of a set of voxels, the set of voxels must first have been determined and this only occurs following the initial sample estimation step.

```
timecourse = timecourse data loaded from datasource
for each volume, vi in timecourse
      volume_tc = timecourse(vi, :)
      volume_mean = mean(volume_tc)
      timecourse(vi,:) = volume_tc - volume_mean.
end for
```

Listing 2.9: Pseudo code for univariate signal subtraction.

2.1.3 Classification

It is the underlying principle of MVPA that if a classifier can be trained to predict with significant accuracy the stimuli which correspond to specific response patterns, then the regions from which these response patterns originate must encode some information about the stimuli in question.

It follows that the selection of the classifier model to use is important. A number of different classification algorithms have been applied to the task of analysing fMRI data, with varying results. Comparisons of the various methods have found that due to the high dimensionality and low sample sizes found in fMRI data sets the more complex algorithms are prone to overfitting; as such the simpler linear models are preferred (Mizaki et al, 2010; Norman et al, 2006). An additional benefit of the linear models is that, when trained on data whose features have been normalised to a uniform scale, the resulting weights of the model can be readily interpreted as an indication of the importance of the feature to the overall classification. This property proves useful for feature selection (De Martino et al, 2008).

The classifiers which have been implemented to date are variants of the linear SVM model; specificly SVM light (Joachims, 1999) and libSVM (Chang and Lin, 2011). These implementations were chosen due their speed of execution and the availability of MatLab compatible versions. An additional benefit to these implementations of the SVM model is that when applied using the same paramters they produce identical results, allowing for interchangeable use of either implementation.

In order to facilitate interchangability of classifiers and thus allow for the addition of further classification algorithms at a later date, the classifiers used in the toolbox are encapsulated by a class in order to provide a standard interface as shown in Figure 2.15. The wrappers implement a standard interface which provides separate methods for training the classifier and applying it to test data. The training methods are responsible for converting the configuration settings into the command strings which the classification algorithms expect, while the testing methods are responsible for collating the classification results into a common format. These results include summary statistics, such as the prediction accuracy, and also the detailed output including the predicted categories, the actual categories, the number of voxels used and the linear weights of the classifier model.

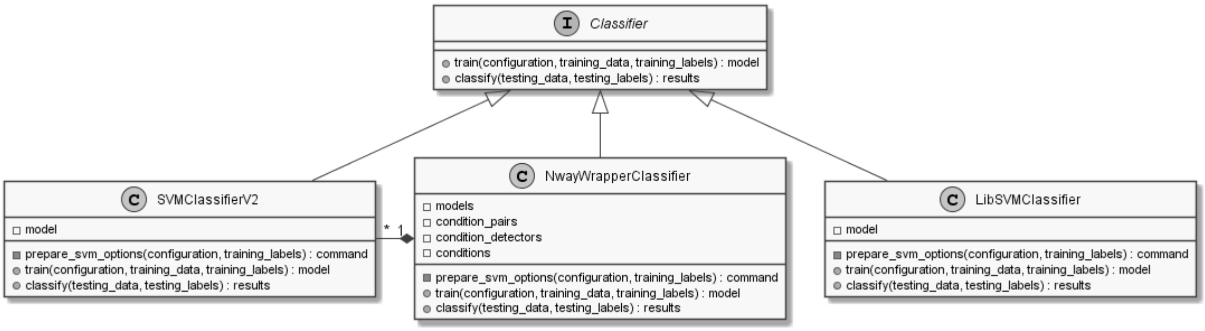

I Classifier
train(configuration, training_data, training_labels) : model
classify(testing_data, testing_labels) : results
C SVMClassifierV2
model
prepare_svm_options(configuration, training_labels) : command
train(configuration, training_data, training_labels) : model
classify(testing_data, testing_labels) : results
* 1
C NwayWrapperClassifier
models
condition_pairs
condition_detectors
conditions
prepare_svm_options(configuration, training_labels) : command
train(configuration, training_data, training_labels) : model
classify(testing_data, testing_labels) : results
C LibSVMClassifier
model
prepare_svm_options(configuration, training_labels) : command
train(configuration, training_data, training_labels) : model
classify(testing_data, testing_labels) : results

Figure 2.15: Simplified UML class diagram for classifiers.

In cases where classification of multiple categories is required, rather than the two-category binary classification of which linear models are capable, an ensemble of binary classifiers is used along with a negative voting system to implement an N-way classifier. This approach is used rather than a direct n-class classification model in order to allow for the potential of interchangeable use of models which do not support direct n-class classification. As shown in Figure 2.16 a binary classifier is trained for each combination of two categories, resulting in an ensemble of classifiers.

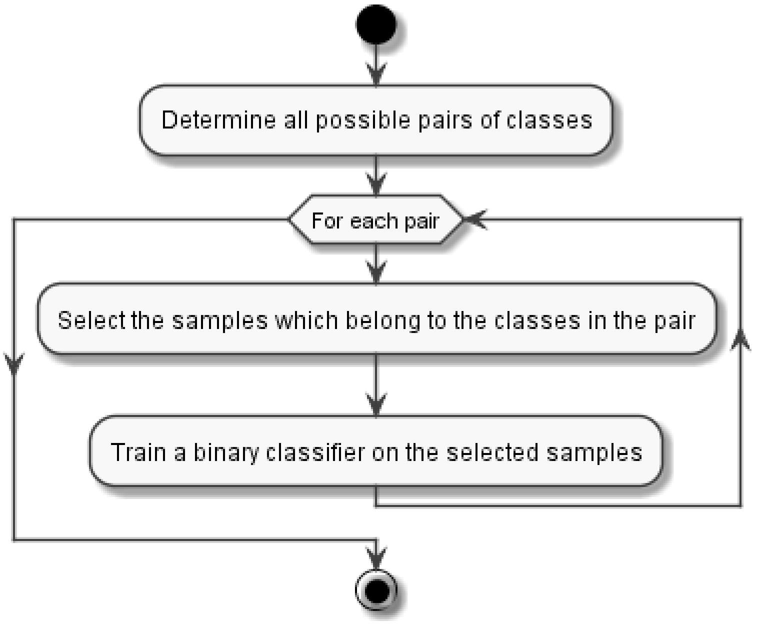

Determine all possible pairs of classes
For each pair
Select the samples which belong to the classes in the pair
Train a binary classifier on the selected samples

Figure 2.16: Activity diagram showing training process for N-way classification.

```
Conditions = set of all condition numbers

for ci = 1 .. number of conditions
  for c2i = ci+1 .. number of conditions
    condition_pairs = append [conditions(ci) conditions(c2i)]
  end
end
```

Listing 2.10: Pseudo code for determination of condition pairs

```
data = prepared timecourse samples
condition_labels = condition labels for the above samples
condition_pairs = determine all possible pairs of conditions

for i = 1 .. num condition_pairs
  class_labels = zeros(num samples, 1).
  class_labels(condition_labels == condition_pairs(i,1)) = 1.
  class_labels(condition_labels == condition_pairs(i,2)) = -1.
  models(i) = train linear svm(data, class_labels).
end
```

Listing 2.11: Pseudo code for training N-way classification model.

Classification of samples is performed by applying each of the models, with the final prediction being determined using negative voting. That is, each classifier casts a vote for the category to which it believes the sample does not belong. This approach has been used in previous work within the Cognitive Neuro-imaging Laboratory, for whom this toolbox was created, and so is used here for consistency and standardisation. Additionally, it does not place any classifier in the position of only being able to make an incorrect vote. For example: a binary classifier which has been trained to distinguish between classes A and B cannot place a correct vote for a sample of class C being in class B, but it is perfectly capable of placing a vote against it being in class A. Once all classifiers have been applied, the category with the fewest votes is selected; in cases of ties, a random selection between the categories with the fewest votes is made.

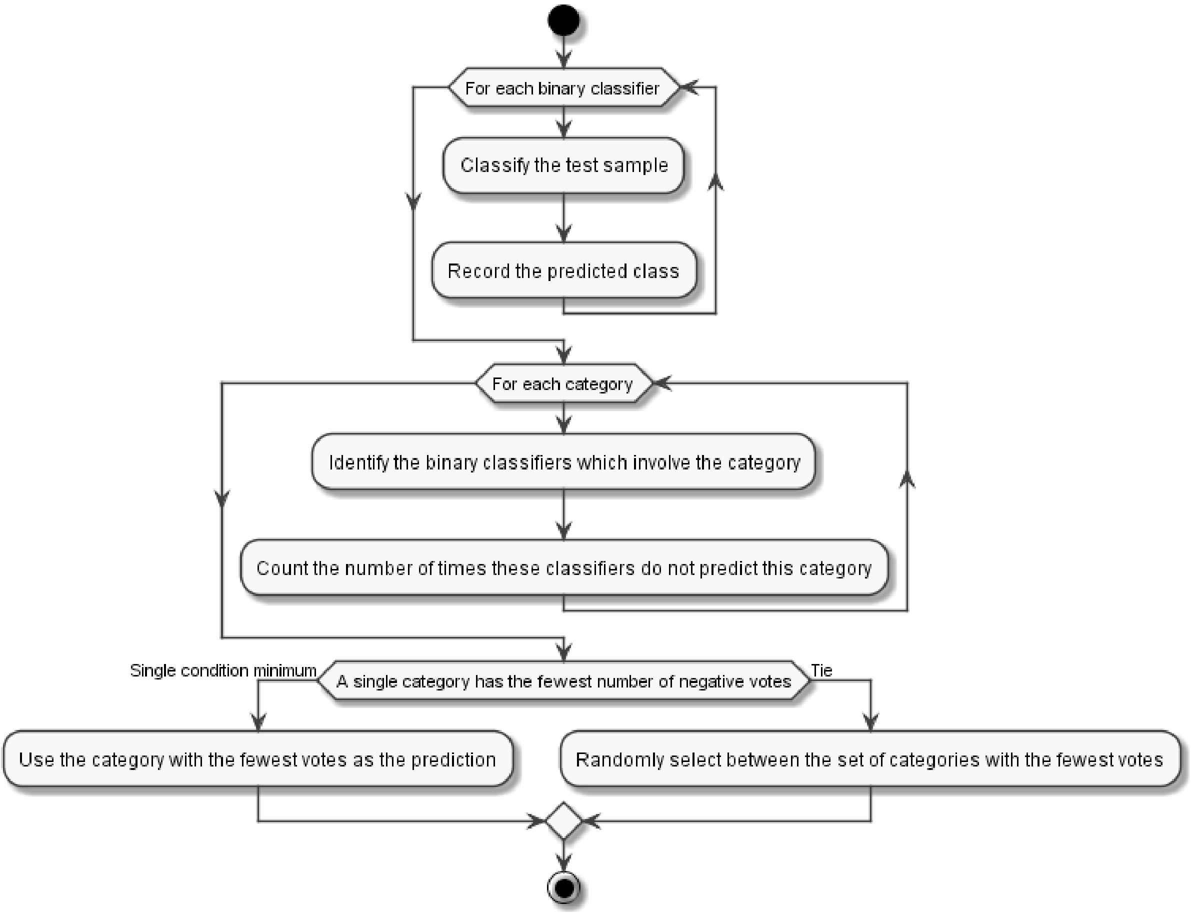

For each binary classifier
Classify the test sample
Record the predicted class
For each category
Identify the binary classifiers which involve the category
Count the number of times these classifiers do not predict this category
Single condition minimum
A single category has the fewest number of negative votes
Tie
Use the category with the fewest votes as the prediction
Randomly select between the set of categories with the fewest votes

Figure 2.17: Activity diagram showing the testing process for N-way classification.

```
data = prepared timecourse sample
condition_labels = condition label for the above sample
condition_pairs = determine all possible pairs of conditions

for i = 1 .. num condition_pairs
  predicted_label(i) = model(i).classify(data).
end

votes = zeros(num conditions,1)
for ci = 1 .. num conditions
  for i = 1 .. num condition_pairs
    if (condition_pairs(i,1) == conditions(ci) or
        condition_pairs(i,2) == conditions(ci)) and
        predicted_label(i) != conditions(ci) then
      votes(ci) = votes(ci) + 1.
    end if
  end
end

predicted_index = index of min(votes).
predicted_condition = conditions(predicted_index).
```

Listing 2.12: Pseudo code for N-way voting classification.

2.1.4 Results Handling

The main output of the classification analysis is the prediction accuracy, since this provides a direct indication of the degree to which a set of voxels encodes information related to the contrast being classified. This is not the only interesting result of the classification however; values which may be dismissed as intermediate steps towards the classification accuracy can also provide useful insight. This can be easily seen with the linear weights assigned to each voxel, which as discussed in terms of recursive feature elimination in Section 2.1.1.2.2 can be seen as an indication of the importance of each voxel to the classification. These are not the only values of interest; the predicted class of each sample can also be useful for further analyses, as shown in Section 2.3.2 in reference to prediction curves.

As a result of the usefulness of these many facets of the results, and the need for traceability in terms of the data on which the classification was performed, the results to be recorded from a classification analysis can rapidly become quite large. It is for this reason that the handling of results is performed by a dedicated component rather than merely saved as an afterthought. Following a classification the standard analysis scripts handle the calculation of mean and standard error of accuracy across the cross-validation runs; however from here they are passed on to the results handler component appropriate to the analysis, as shown in Figure 2.18.

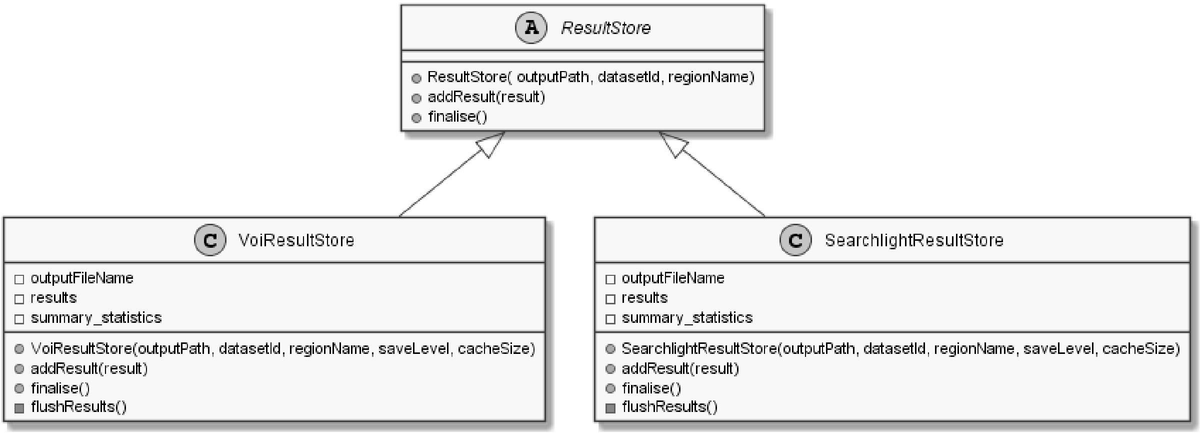


Figure 2.18: Simplified UML class diagram for Results Stores.

The results handler is responsible for storing these for each feature set, collating of a summary of results containing only the key statistics, and writing out the results when they become too large to hold in memory. Since this can produce several detailed results files for each analysis the other side of the results handling component is to provide a standard interface for retrieving the detailed results for visualisation and post-classification processing as shown in Figure 2.19.

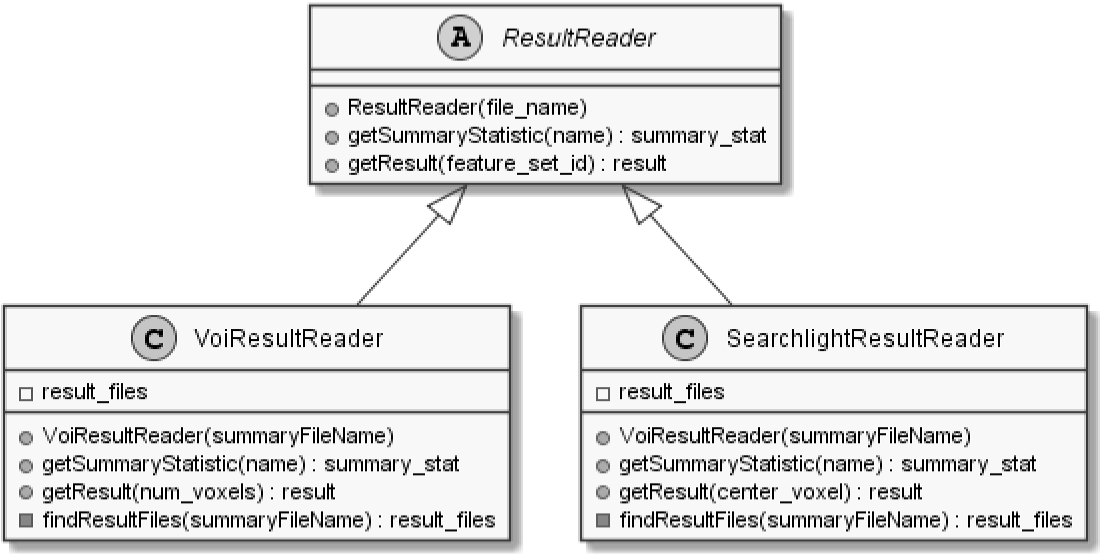

A
ResultReader
ResultReader(file_name)
getSummaryStatistic(name) : summary_stat
getResult(feature_set_id) : result
C
VoiResultReader
result_files
VoiResultReader(summaryFileName)
getSummaryStatistic(name) : summary_stat
getResult(num_voxels) : result
findResultFiles(summaryFileName) : result_files
C
SearchlightResultReader
result_files
VoiResultReader(summaryFileName)
getSummaryStatistic(name) : summary_stat
getResult(center_voxel) : result
findResultFiles(summaryFileName) : result_files

Figure 2.19: Simplified UML class diagram for Results Readers.

2.2 Univariate Analyses

While the multivariate analyses provided above allow for the detection of information encoded in patterns of activation which univariate analysis methods are unable to detect, these univariate analyses are still capable of providing useful information, as evidenced by the use of GLMs to generate univariate voxel statistics for use in feature selection. While providing a GLM analysis model was unnecessary, the toolbox does provide three additional univariate analyses in addition to the standard MVPA scripts; percent signal change (PSC), functional signal to noise ratio (fSNR), and haemodynamic response function (HRF) modelling.

2.2.1 Percent Signal Change and functional Signal to Noise Ratio

The percent signal change (PSC) provides an intuitive measure of the degree to which the BOLD signal changes for a given condition with comparison to a baseline level. Functional Signal to Noise Ratio (fSNR) provides a measure of the ratio between the insensity of the signal associated with a particular condition and the variabilitty in the data due to sources of noise (Huettel et al., 2009). Both of these measures have been used in studies within the Cognitive Neuro-imaging Laboratory to determine whether detected differences between classes were a result of changes in the univariate signal, or changes in multivariate encoding (Patten, 2013; Zhang et al., 2010).

The implementation provided in the toolbox calculates these values individually for each stimulus condition, which also allows for identification of different responses to different stimuli. The implementation of the the PSC and fSNR analyses is largely shared, since they require the same preprocessing steps as shown in Figure 2.20.

The scripts make use of the standard feature selection components described in Section 2.1.1.1 to load and prepare the regions of interest. The difference in requirements for timecourse preparation – namely the lack of normalisation and sample estimation – preclude the use of the standard time-course components.

Following the preparation steps shown in Figure 2.20 the PSC and fSNR differ only in the formula used, as shown below:

$$PSC = \frac{mean\,stimulus\,activation - mean\ fixation\,activation}{mean\ fixation\,activation} \times 100$$

$$fSNR = \frac{mean\,stimulus\ activation - mean\ fixation\ activation}{standard\ deviation\ of\ all\ activation}$$

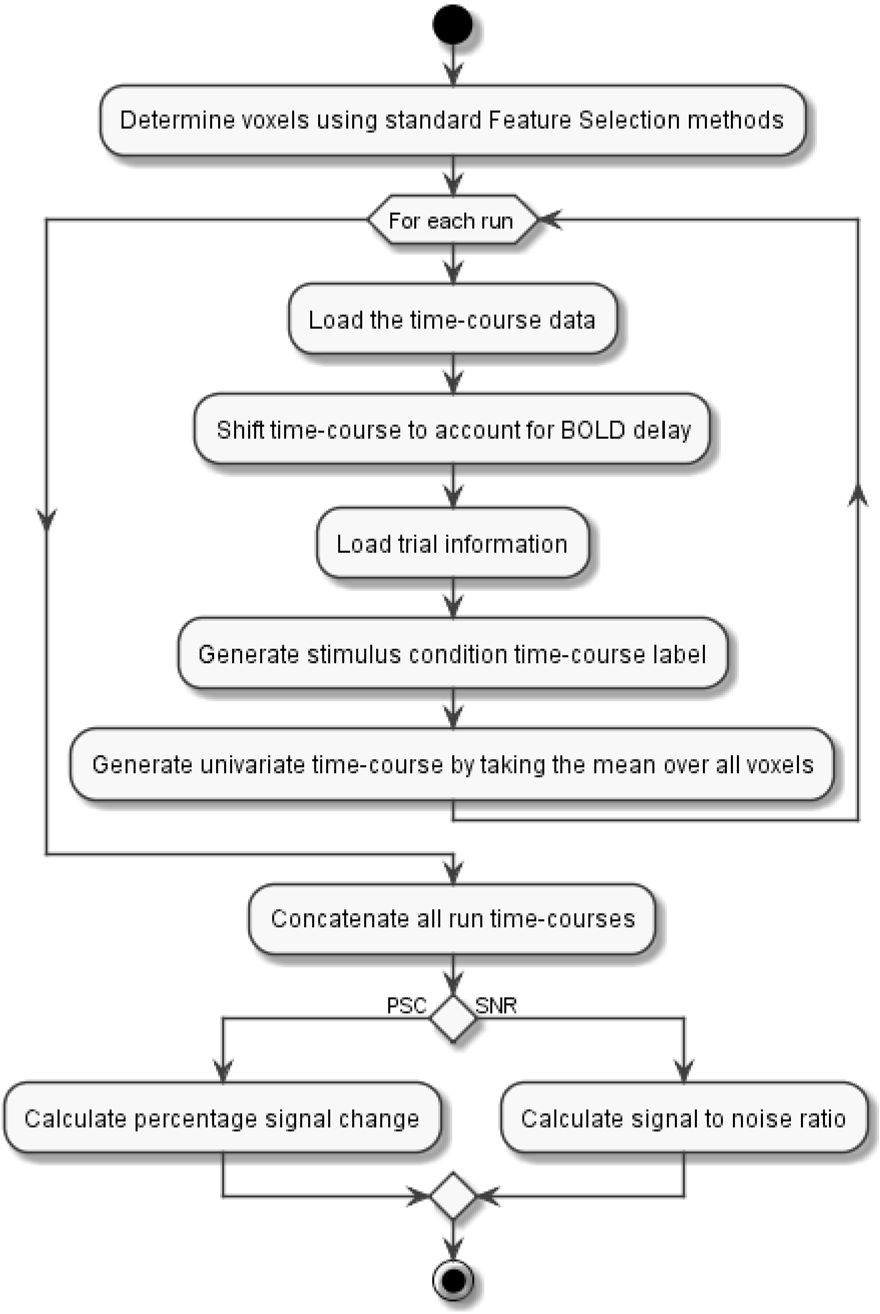

Determine voxels using standard Feature Selection methods
For each run
Load the time-course data
Shift time-course to account for BOLD delay
Load trial information
Generate stimulus condition time-course label
Generate univariate time-course by taking the mean over all voxels
Concatenate all run time-courses
PSC
SNR
Calculate percentage signal change
Calculate signal to noise ratio

Figure 2.20: Activity diagram for calculation of PSC and fSNR.

2.2.2 Haemodynamic Response Function Modelling

A number of analyses require the use of a haemodynamic response function, for example: the general linear model requires the convolution of a HRF with the design matrix to generate the predicted response time-course. While in most cases a standard model can be used, it may be preferable to estimate the actual HRF of the participant and use this. As such a script for estimation of the haemodynamic response functions from functional data is provided in the toolbox.

Estimates are based on the mean of an 18 second window of the responses for each trial in the time-course. The models are then generated by using a least squares algorithm to fit a two-Gaussian function, based on that used by Kruggel and von Cramon (1999), to these response windows. In the two-Gaussian function *g* and *d* refer to amplitude and standard deviation respectively, *t* refers to the time from onset, and *p* refers to the delay from onset to peak activation.

$$y(t)=\frac{g_a}{d_a}\times e^{\frac{-(t-p_a)^2}{(2\times d_a^2)}}+\frac{g_b}{d_b}\times e^{\frac{-(t-p_b)^2}{(2\times d_b^2)}}$$

From an intuitive viewpoint, this approach uses one Gaussian function to model the rise and peak of the haemodynamic response, and the second Gaussian to model the following undershoot.

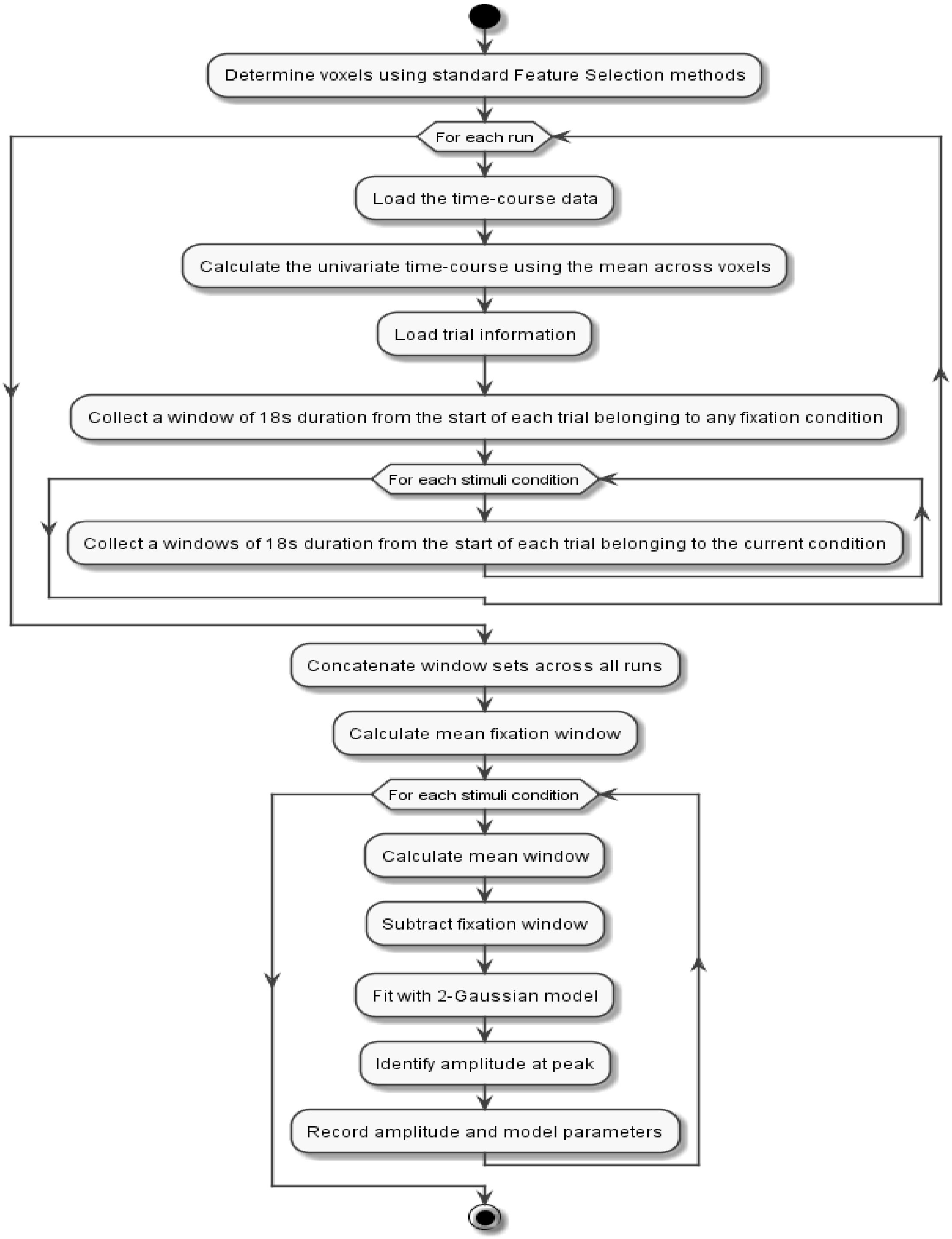

Determine voxels using standard Feature Selection methods
For each run
Load the time-course data
Calculate the univariate time-course using the mean across voxels
Load trial information
Collect a window of 18s duration from the start of each trial belonging to any fixation condition
For each stimuli condition
Collect a windows of 18s duration from the start of each trial belonging to the current condition
Concatenate window sets across all runs
Calculate mean fixation window
For each stimuli condition
Calculate mean window
Subtract fixation window
Fit with 2-Gaussian model
Identify amplitude at peak
Record amplitude and model parameters

Figure 2.21: UML acivity diagram for estimation of HRF from data.

2.3 Visualisation and post-classification analysis

In addition to the multivariate and univariate scripts, the toolbox also provides some tools for visualisation and further analysis of the results of MVPA. The simplest of these generates graphs of the classification accuracy to show at a glance the regions which exhibit a significant ability to classify stimuli, and to aid in identifying the number of voxels beyond which no further information is gained from adding more. The more interesting methods provided however, look deeper into the results of the classifications, utilising information such the predicted categories of each sample. The following two sub-sections describe the methods which were included in the toolbox.

2.3.1 Accuracy graphs

Accuracy provides a simple measure of the performance of a classification model in terms of the number of correctly categorised samples. The accuracy of a given model when applied to a test set is given by the formula:

$$accuracy = \frac{true\ positives + true\ negatives}{true\ positives + true\ negatives + false\ positives + false\ negatives}$$

Provided in the toolbox are some tools for generating graphs of the accuracy results of the standard analyses. Examples of these are provided below in Figures 2.22 and 2.23. The types of graphs can be generated both for each individual participant, and to show the average performance across all subjects.

The graph shown in Figure 2.22 can be used to identify at what point the inclusion of additional voxels in the classification ceases to show a benefit, and also provide an indication of the stability of the result as the feature set changes. While it is ill suited for the comparisons required by most analyses it does prove a useful to identify potential problems with the analysis.

The graph shown in Figure 2.23 shows the accuracy for each region at a specified feature set size. In cases where there are insufficient voxels in a given region, the accuracy of the feature set with the maximum number of voxels available for that region is used. The graphs of this style can give an initial indication of whether each region encodes relevent information. Provision is made for the optional addition of plotting binomial test threshold marks such that a classification accuracy above this level can be considered significant.

The binomial distribution can be used to determine the probability of making a given number of correct predictions, given the expected prediction performance and the total number of predictions made. By taking the inverse cumulative distribution function of the binomial distribution specified by the expected chance performance of the classification and the number of test trials, it is possible to then extract the expected number of correct predictions for a given probability; in this case p-values of 0.95 and 0.99.

These expected numbers of correct predictions, divided by the total number of test samples, give the significance thresholds plotted on the graph, which are analogous to those of a binomial significance test. This binomial theorem based significance threshold has the benefit of being quite easy to calculate, however it might be considered to provide overly conservative thresholds. Pereira et al (2009) provide a more detailed discussion of significance testing of classification accuracy for fMRI studies.

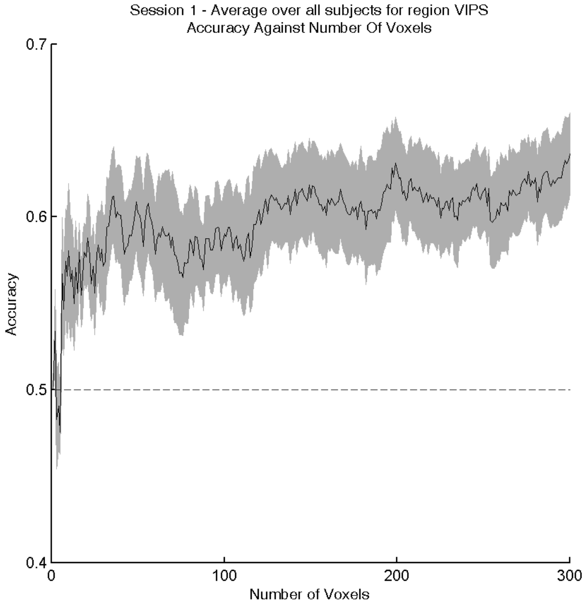


Figure 2.22: Example plot of classification accuracy by numbers of voxels. The black line show the mean accuracy across all subjects, while the shaded area shows the standard error of the mean. The dashed line at 0.5 indicates chance performance.

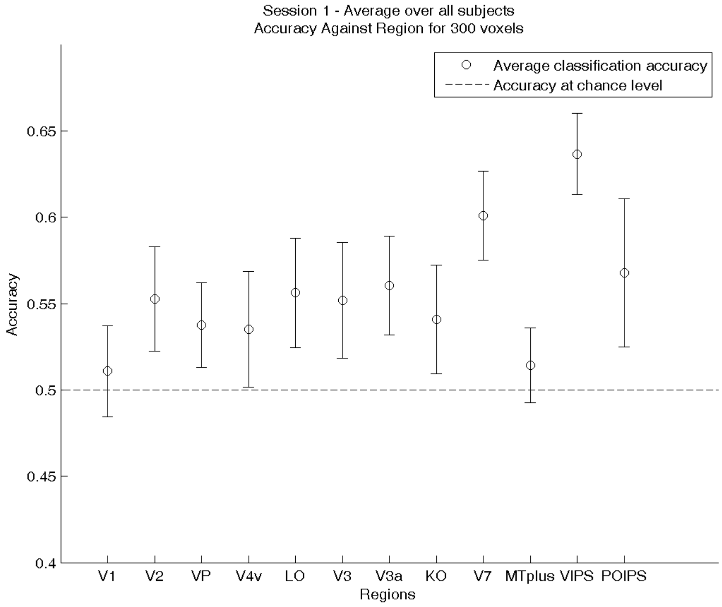


Figure 2.23: Example plot of classification accuracy by region. Values plotted are the average accuracy across all subjects, error bars show the standard error. The dashed line at 0.5 indicates chance performance.

2.3.2 Psychometric curves

In studies in which the stimuli are linked to some parameter, such as a linear space between two extremes or the degree of noise included in the stimuli, the results of the classification can be used to identify thresholds of perception. The toolbox provides scripts for the visualisation of this nature of study, as show in Figure 2.24.

The graphs produced by these scripts plot different values depending on the design of the study to which they are applied. For analyses in which stimuli are classified as belonging to either of two extremes of a space defined by the parameter, the results plotted are the proportion of samples at each parameter level which are predicted as the second extreme.

For analyses in which the stimuli are classified as belonging to one of two classes with the parameter corresponding to varying levels of noise in the stimuli, the results plotted are the prediction accuracy of classification analyses performed only on the stimuli at each level of noise.

Regardless of which type of values are to be plotted, they are fitted with a cumulative Gaussian function, using psignifit version 2.5.6 for MatLab (see: http://bootstrap-software.org/psignifit/) which implements the maximum-likelihood method described by Wichmann and Hill (2001a and 2001b). In addition to the graphs, the scripts record values for the goodness of fit, perception thresholds and their confidence intervals.

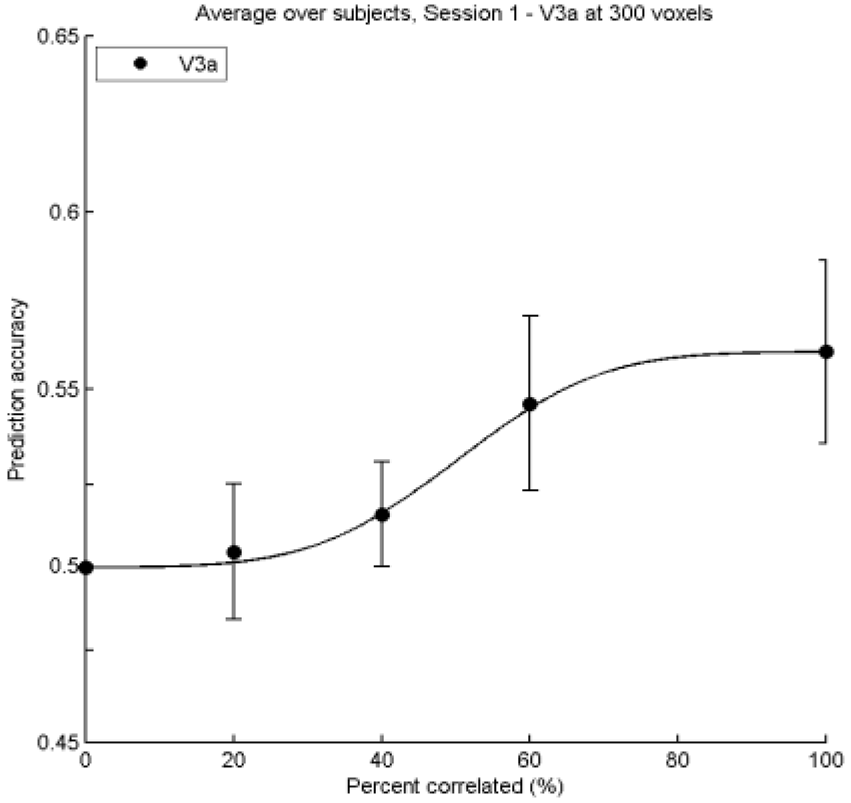


Figure 2.24: Example psychometric function plot showing the prediction accuracy of binary classifications of stimuli with varying levels of signal. Error bars show standard error or the mean.

2.4 Parallel Computation

While the analysis of the data pertaining to a single participant can be performed in a matter of minutes when using a simple analysis, more complex analyses, such as RFE, or higher resolution scans (which result in more data) can increase this time to be closer to an hour. When this is multiplied by the number of participants involved, it is apparent that a reduction in the time needed for processing is desirable, particularly when it may be necessary to perform multiple separate analyses to investigate different properties of the data.

The analysis scripts provided by the toolbox which have been discussed so far have been considered in terms of being used on a single machine. The nature of the data produced by most fMRI studies, however, lends itself particularly well to parallel processing. This is particularly apparent with analyses which focus on individual regions of interest, in which it is possible to analyse individual subjects and scanning sessions as well as individual regions of interest.

Due to the toolbox being implemented in MatLab is it a relatively easy task to deploy the relevent scripts to a compute cluster and parallelise the computation over a number of nodes. Additionally, due to the largely separate nature of the analyses, the speed up due to the parallelisation is only limited to the competition for the common storage medium of the timecourse data, which due to the caching implemented in the data preparation modules, is minimised.

A number of approaches to parallelisation using the Matlab Parallel Computing Toolbox and Distributed Compute Server (The MathWorks, Inc., Natick, Massachusetts, United States) were considered, including the use of `parfor`, and batch processing methods. Early attempts were made using `parfor`, since this required the least adaptation to the existing codebase; merely requiring the replacement of a `for` statement with `parfor`, and the provision of a matlab pool on which to execute. This approach was met with some success, however due to the embarassingly parallel nature of the task further experimentation found that a batch processing approach resulted in better performance and fault tolerance, with the failure of an individual task not requiring the entire analysis to be re-run.

```
Create a new Job on the Matlab Cluster.

For each subject in the experiment,
      For each session of the subject,
            For each region of interest,
                  Create a task to perform the desired MVPA analysis
                        on the current region of interest
                        of the current session
                        of the current subject.
                  Add the new task to the job.
            End For
      End For
End For

Submit the job to the cluster to begin processing.
```

Listing 2.13: Pseudo code for creating and submitting a voi based analysis job to a MatLab compute cluster.

The exception to this, as in other instances, is the searchlight analysis, since it processes the entire brain, rather than discrete subsections thereof. A simple approach to this would be to merely parallelise at the subject and session level; however, this makes less than optimal use of most clusters as it is likely that there would be more compute nodes available than the number of subjects. The method implemented in the toolbox is that of partitioning the set of voxels around which searchlights will be centred. While this results in some overlap of the data loaded in each set, it does mean that the processing of a single whole brain map can be broken down into an arbitrary number of processes allowing for rapid searchlight analysis of a single participant if required.

```
Create a new Job on the Matlab Cluster.

For each subject in the experiment,
      For each session of the subject,
            For i = [1 .. desired_number_of_sets],
                  Determine set of voxel centers.
                  Create a task to perform the searchlight MVPA analysis
                        on the current region of interest
                        of the current session
                        of the current set of voxel centers.
                  Add the new task to the job.
            End For
      End For
End For

Submit the job to the cluster to begin processing.
```

Listing 2.14: Pseudo code for creating and submitting a searchlight analysis job to a MatLab compute cluster.

2.5 Validation

While the key features provided by the toolbox are the analyses and standard components it provides, another important feature is the degree to which it has been tested to ensure the correctness of implementation and the results produced.

Since the toolbox has been developed in part to replace existing analysis scripts, early stages of development and thereafter the simpler analyses were validated by comparison of the results they produced on a test dataset to those produced by the existing scripts. Since these simpler analyses are deterministic in nature they could be expected to produce exactly the same results for a given set of data, so it follows that any discrepancies could be traced back to the cause and corrected.

Additional validation was performed on both these simple analyses and the more complex ones, by taking advantage of the interpreted nature of the MatLab language, which allowed execution of the analysis scripts to be observed step by step, and the resulting intermediate values compared to those calculated manually, or provided by older implementations.

In addition to this testing on smaller test data sets the toolbox has been applied in a number of published studies conducted at the University of Birmingham (Zhang et al, 2010; Ban et al 2012; Dövencioğlu et al, 2013; Kuai et al, 2013).

This thorough unit and integration testing, and subsequent deployment in the University of Birmingham Cognitive Neuroimaging Lab, has resulted in a robust tool for performing MVPA which has been shown to produce reliable results.

2.6 Summary

To summarise the toolbox provides a set of standard analyses and components for the analysis of fMRI data, which covers both univariate and multivariate analysis. The modular design of the toolbox provides re-usable components for feature selection and preparation of time course data in order to facilitate development of further analyses, such as those presented in later chapters. The implementation in MatLab and the easily separable nature of fMRI data enables easy parallel processing through use of the the MatLab parallel compute toolbox and Distributed Compute Server (The MathWorks, Inc., Natick, Massachusetts, United States). The validity and reliability of the scripts provided have been assured through thorough testing on validation datasets, and application in a number of published studies.

This well tested, modular toolbox provides an efficient means of performing multivariate pattern analysis on fMRI data and a strong basis for the implementation of further analysis methods.

CHAPTER 3 - LEAVE-*C*-OUT CROSS-VALIDATION

3.1 Introduction

Most fMRI studies, particularly those involving human subjects, are limited in the quantity of samples which can be collected. This limited quantity of available data restricts the complexity of the models which can be used for its analysis; an example of this is shown in Chapter 2, where the suitability of linear and non-linear classification methods is discussed. As a result, further limiting the data available for training models by using a dedicated hold-out set for testing is undesirable.

The solution to this problem used in most MVPA methods is to employ cross-validation. The most commonly used method, leave-one-run-out (LORO) cross-validation, is a variation of the standard *k*-fold cross-validation method used in machine learning and statistics. In the *k*-fold method the available data are divided into *k* sets and each of these sets is held out in turn as the test set, with the remaining sets combining to serve as the training dataset. Whereas the *k*-fold cross-validation method artificially divides the datasets, the leave-one-run-out method takes advantage of the fact that fMRI datasets are naturally split into runs. This is a popular approach due to its simplicity in terms of implementation and understandability, and its effectiveness in ensuring that the resulting training and testing datasets are completely independent.

The leave-one-run-out cross-validation method makes certain assumptions about the data to which it is applied. It is assumed there is a sufficient number of runs to make a good estimate of the accuracy and variance of the classification model. In terms of the more general *k*-fold cross-validation, Kohavi (1995) suggests *k*=10 is usually sufficient to produce a stable estimate of the performance of the classification model. A typical fMRI study will have around 8 runs per session, which is sufficient; however, adverse factors such as excessive head motion may necessitate the discarding of affected runs, which may have an adverse impact on the stability of the estimation of the model's performance.

It is also assumed that in each run there will be an approximately equal number of samples from each class. Kohavi (1995) found that this property, referred to as stratification, has a beneficial effect on the estimation of the bias and variance of the analysis. The typical fMRI experiment design is such that each run will naturally have a balanced representation of each experimental condition. However, there are certain situations in which this balance may not be preserved, such as when dealing with participant responses rather than presented stimuli.

In situations where one or more of these assumptions are broken it may be desirable to look into employing one of the variety of other methods for cross-validation which the fields of machine learning and statistics provide. One readily apparent method would be to use an exhaustive approach such as leave-*p*-out cross-validation, which involves selecting *p* observations as a testing set and using the remaining observations as a training set, with this process being repeated for every possible combination of *p* observations.

Leave-one-run-out cross-validation is typically preferable over this method in machine learning studies, since leave-p-out cross-validation involves a considerably larger number of cross-validation folds, and a consequent increase in the degree of overlap between training sets. As discussed in chapter 1, an increased overlap between training sets results in an increased correlation between the resulting models and a consequent increase in the variance of the prediction error of the models. While reducing the number of training sets would seem like an intuitive solution to this problem, doing so also reduces the quantity of data available on which to train the models and can also raise the variance of the prediction error. As mentioned above Kohavi (1995) finds that using stratified k-fold cross-validation with *k*=10 provides a good trade off between these two factors. However when the data available contains too few runs or is not stratified it may be worth applying leave-p-out cross-validation.

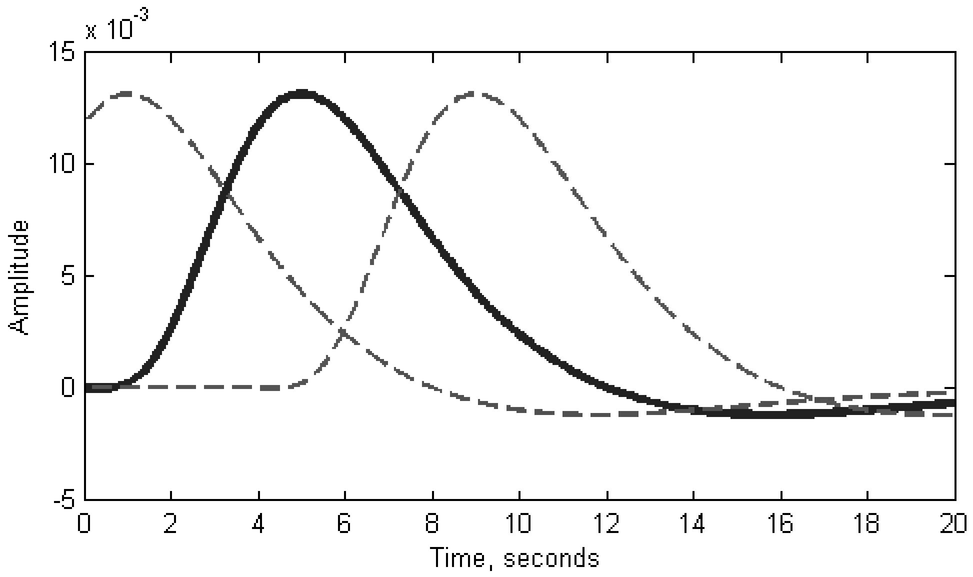


Figure 3.1: Illustration of the overlap in haemodynamic response function between adjacent trials, using HRFs generated using a two-gamma model and spaced at 4 second intervals.

While using leave-p-out cross-validation may be beneficial in situations where using leave-one-run-out cross-validation is less suitable, there are a number of difficulties which must be addressed in order to implement such an approach. The most serious of these is ensuring the independence of the training and testing datasets. Due to the nature of the BOLD signal, the activation in response to a stimulus being presented can extend for up to 20 seconds before completely returning to baseline levels. As shown in Figure 3.1 this results in an overlap of the responses to time-adjacent stimulus presentations, resulting in observations which are not entirely independent.

In the course of most fMRI analysis this issue is handled by careful experimental design; by selecting the presentation order such that no permutation of adjacent stimuli is repeated the effect of this overlap is minimised. When dealing with leave-one-run-out cross-validation methods this has proven to be sufficient, however when dealing with situations in which observations for both the training and testing datasets will be taken from within the same run this may introduce an unacceptable dependency between the two.

In this chapter a method for applying cross-validation to fMRI data is presented, which is designed specifically to address the problem of imbalanced test sets and also allow for an arbitrary number of cross-validation sets. This method is derived from leave-*p*-out (LPO) cross-validation, with the key modification being that each test set comprises one sample from each class, as such this method is referred to as leave-*c*-out (LCO) cross-validation. This method is limited, in that it does not address the issue of increased overlap between training sets, however it is intended for use in situations where this factor is secondary to that of the limited number of samples available for training.

3.2 Methods

The leave-*c*-out (LCO) cross-validation method differs conceptually from the more common leave-one-run-out (LORO) method in two key points: the selection of the test set and of the training sets. In the LORO method this is an almost trivial task, simply selecting one run to be the test set, and then concatenating the rest for a training set, and performing it requires little to no information regarding the latter stages of the multivariate pattern analysis (MVPA) process. This simplicity comes from being able to rely on the experiment design to ensure certain factors such as a clean separation of training and testing sets, and a good balance of samples in each run. The LCO method however, is more involved since these factors must be accounted for by the method rather than being able to rely on the experiment design.

In addition to the differences in selection of training and testing sets, there are certain modifications to the sample estimation and normalisation procedures which are required as a consequence. In this section the leave-*c*-out cross-validation method will be discussed covering both the two key changes in set selection and the further modifications which they make necessary.

3.2.1 Modifications to Sample Estimation

One of the goals of the LCO method is to ensure that the test sets have a balanced selection of each of the conditions included in the latter classification. This requires the knowledge of which samples will be available after the sample estimation step.

In the processing sequence used with LORO cross-validation, sample estimation is performed separately for each run, with the samples and the patterns which correspond to them being processed at the same time. Due to the fact that both training and testing data potentially may be sourced from within the same run, some aspects of the sample estimation process, namely the normalisation, are now dependent on the division of data resulting from the cross-validation. Fortunately, while the LCO method requires knowlege of which samples are available it does not require the actual patterns. As such the implementation of the sample estimation is split into two stages: first determining which samples will result from the sample estimation step and second performing the actual sample estimation process to generate the patterns.

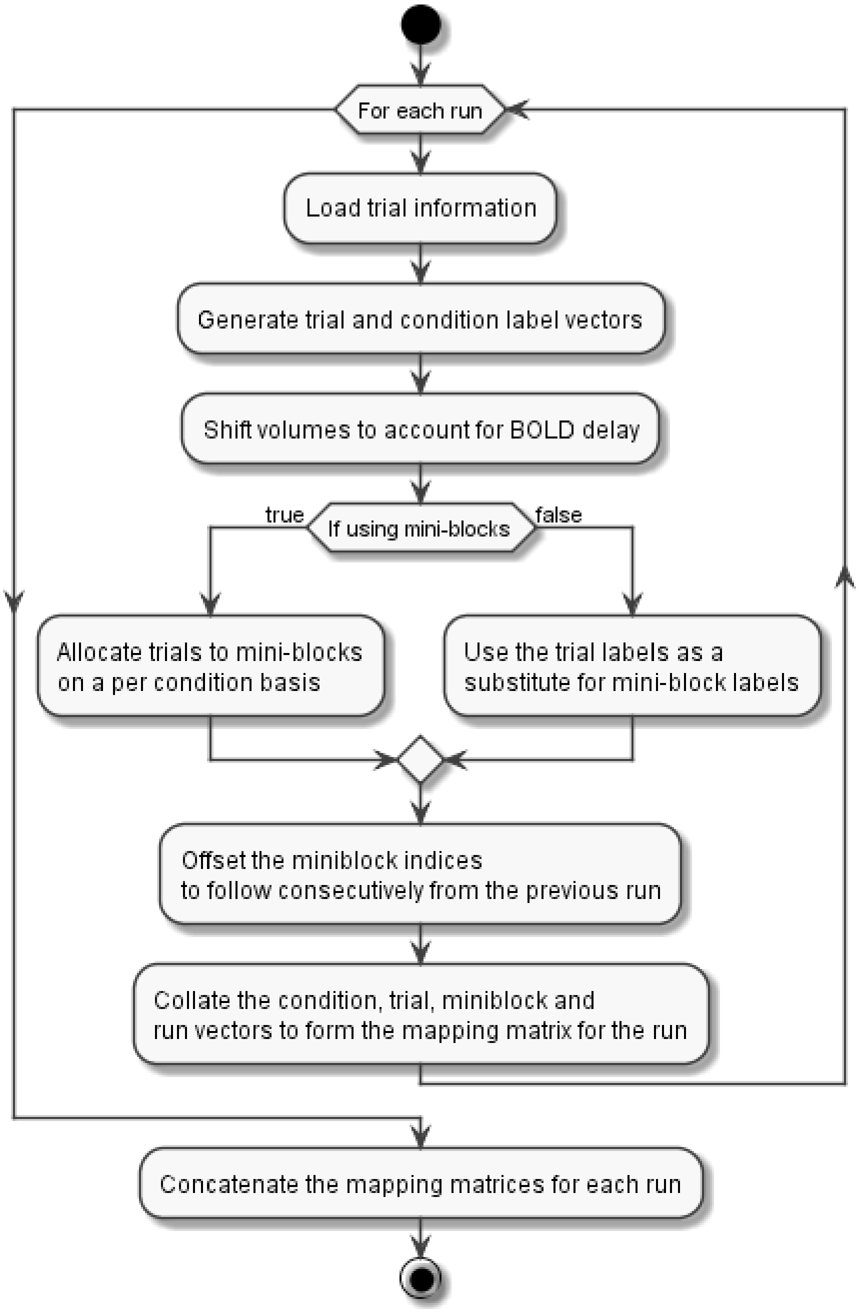

For each run
Load trial information
Generate trial and condition label vectors
Shift volumes to account for BOLD delay
If using mini-blocks
true
false
Allocate trials to mini-blocks on a per condition basis
Use the trial labels as a substitute for mini-block labels
Offset the miniblock indices to follow consecutively from the previous run
Collate the condition, trial, miniblock and run vectors to form the mapping matrix for the run
Concatenate the mapping matrices for each run

Figure 3.2: Activity diagram showing the process to generate the mapping between volumes and mini-blocks.

For the block/trial averaging and miniblock methods discussed in this thesis this takes the form of generating a mapping for each volume to the trial and miniblock to which it will belong. The activity diagram in Figure 3.2 shows an overview of this process.

This process results in a matrix with a row for each volume in the concatenated time course of all runs. Each row comprises the following four properties: the condition number, the trial number, the mini-block number and the run number. Each condition is assigned an arbitrary number for ease of processing, the mapping between numbers and conditions is common across all runs. The trial number is assigned according to the order in which they appear in each run. The mini-block number is assigned in a similar fashion; however, the numbers for each run are offset such that each mini-block has a number which is unique through out the entire time course.

Generating this mapping facilitates the later application of exclusion windows for determining the training set as discussed in Section 3.2.3 by allowing decisions to be made at the mini-block level and then be mapped back to the volume level. Additionally if recorded in the analysis results, this mapping maintains traceability from the classification results back to the stimuli which are being classified.

3.2.2 Selection of the Test Set

The selection of the test set is the part of the leave-*c*-out cross-validation method which both allows for an arbitrary number of test sets and ensures an equal representation of all conditions in each. This is accomplished by first collecting lists of the mini-blocks for each condition. When generating the test sets one mini-block from each condition is selected; however, there may be an unequal number of mini-blocks for each condition, and there may also be more cross-validation sets to generate than there are mini-blocks in each condition. As such the pool of samples is expanded by replicating the list of mini-blocks until the number of mini-blocks is greater than the number of required cross-validation sets, and then cropped such that the number of mini-blocks for each condition is equal to the number of cross-validation sets.

Testing sets are then generated by randomly sampling without replacement from these lists, such that one sample from each condition is selected. This method of replicating the lists and random selection is preferred over random sampling with replacement as it ensures that each mini-block appears in a test set at least once.

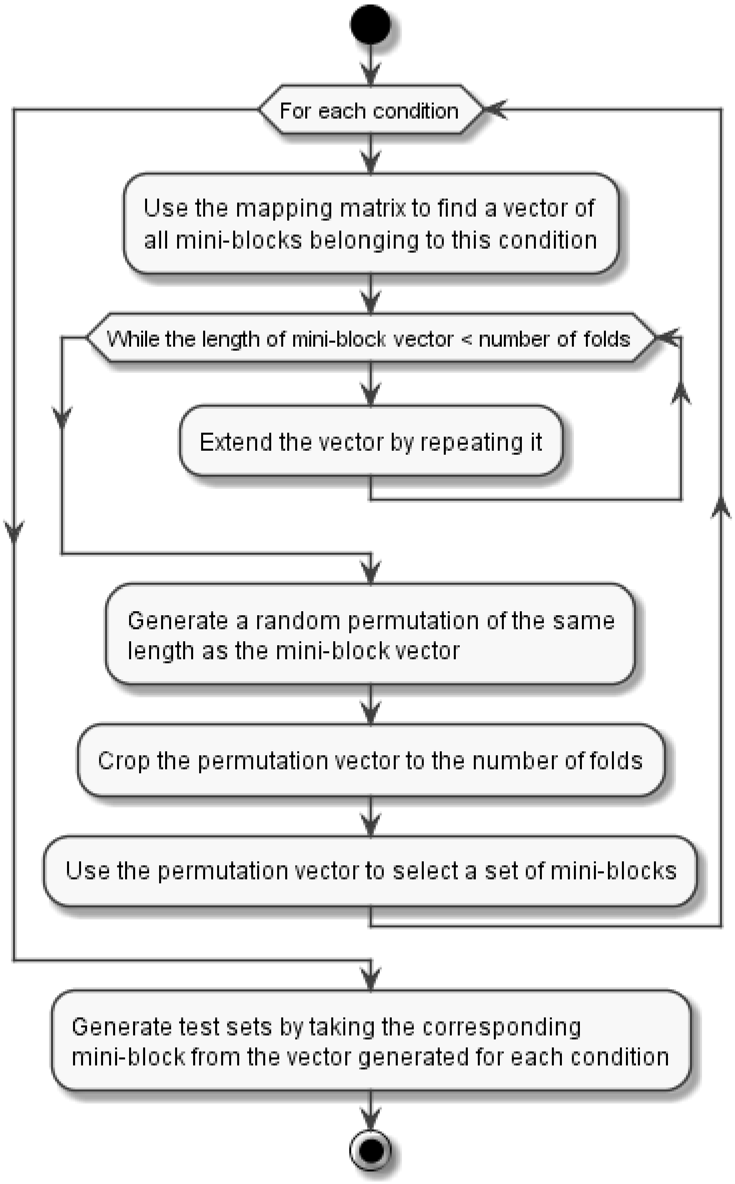

For each condition
Use the mapping matrix to find a vector of all mini-blocks belonging to this condition
While the length of mini-block vector < number of folds
Extend the vector by repeating it
Generate a random permutation of the same length as the mini-block vector
Crop the permutation vector to the number of folds
Use the permutation vector to select a set of mini-blocks
Generate test sets by taking the corresponding mini-block from the vector generated for each condition

Figure 3.3: Activity diagram showing the process used for selecting the test sets in leave-c-out cross-validation.

3.2.3 Selection of the Training set using Exclusion Windows

The selection of the training set is the portion of the LCO method which is responsible for ensuring a clear separation between the training and the testing datasets. It is the temporally extended nature of the haemodynamic response, that can extend for up to 20 seconds, which causes the problematic overlap of adjacent samples. As mentioned in Chapter 1, this problem is often addressed during the experiment design by presenting stimuli in blocks, or by permuting the combinations of adjacent stimuli to balance out the effects of the overlap.

One other method is to simply leave a sufficiently long gap between stimulus presentations to allow for the response to return to baseline activation levels. This method is sub-optimal since leaving such gaps causes a great reduction in the number of observations which can be collected. However, if this method is used with cross-validation in a manner which limits the reduction of samples to those adjacent to members of the test set, the loss caused by it is more acceptable.

Following this idea the selection of the training set is performed not only as the negation of the testing set, but rather as the negation of the combination of the testing set with the addition of a window of volumes to either side of the peak response of each observation in the testing set as shown in Figure 3.4.

Figure 3.4 also shows a large part of the duration of the haemodynamic response is taken up with the undershoot, which has a much lower difference in amplitude relative to the baseline. It follows that it may not be necessary to exclude a window equivalent to the entire duration of the HRF; as such the method allows for the specification of the size of the exclusion window in order that multiple sizes may be evaluated.

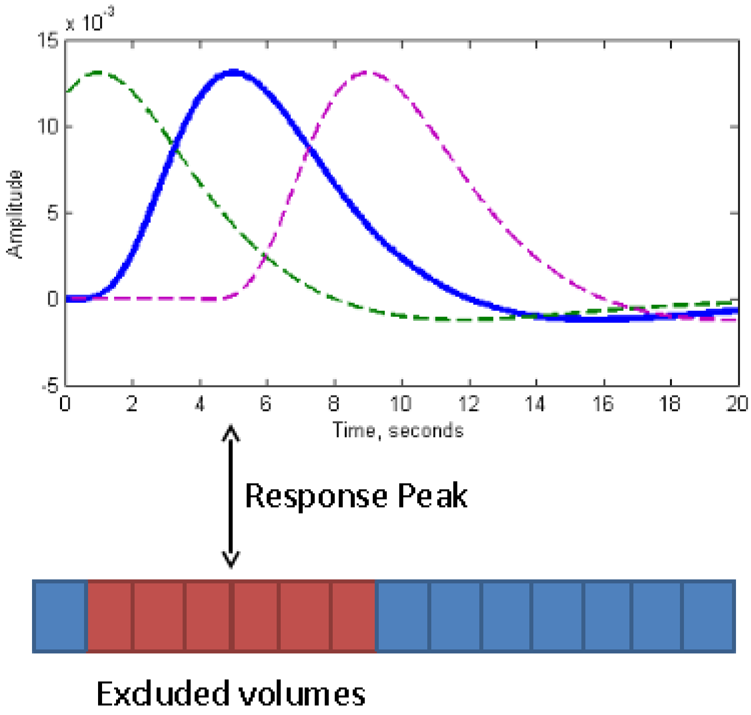


Figure 3.4: Volumes excluded by a 6 second window surrounding the peak response of a trial.

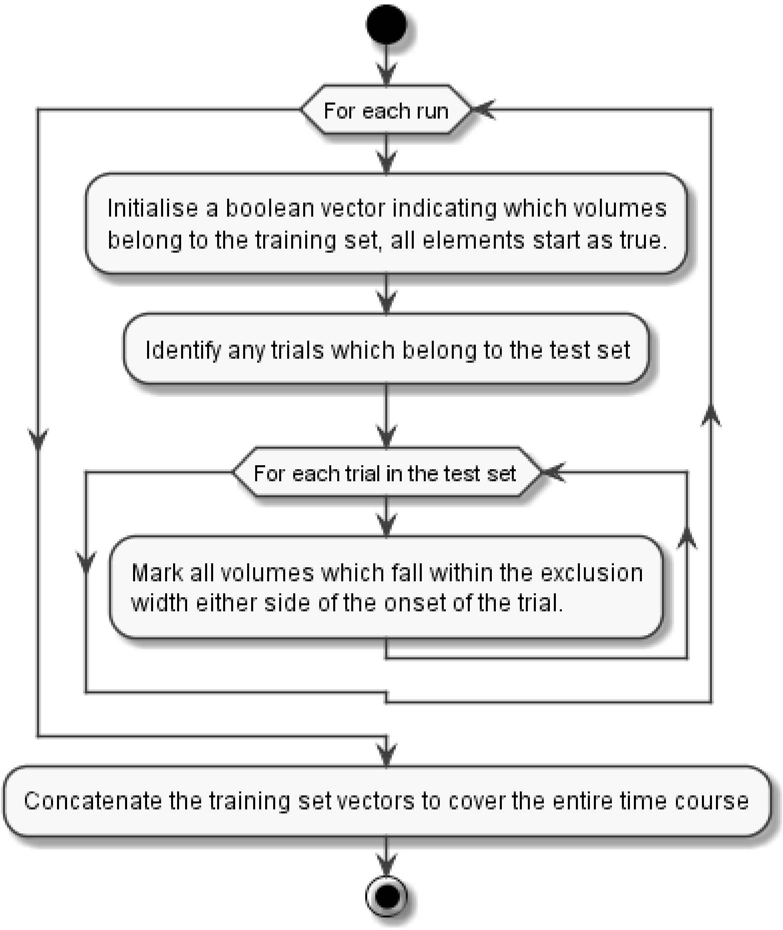

For each run
Initialise a boolean vector indicating which volumes belong to the training set, all elements start as true.
Identify any trials which belong to the test set
For each trial in the test set
Mark all volumes which fall within the exclusion width either side of the onset of the trial.
Concatenate the training set vectors to cover the entire time course

Figure 3.5: Activity diagram showing the process for determining the training set using exclusion windows.

3.2.4 Selection of the Training Set using Random Windows

The method described in Section 3.2.3 seeks to avoid the issues arising from overlap of training and testing samples by excluding a windows of volumes around the selected testing volumes. As a result any change in accuracy when compared to a LORO cross-validation analysis can be attributed to either eliminating the overlap or to the reduction in the number of samples in the training set. In order to control for the possibility that the reduction is due to the reduction in available samples the following method performs the LCO cross-validation with the exception that instead of excluding volumes in a window around the selected test samples, an equal number of trials are excluded from random points in the timecourse, as shown in Figure 3.6.

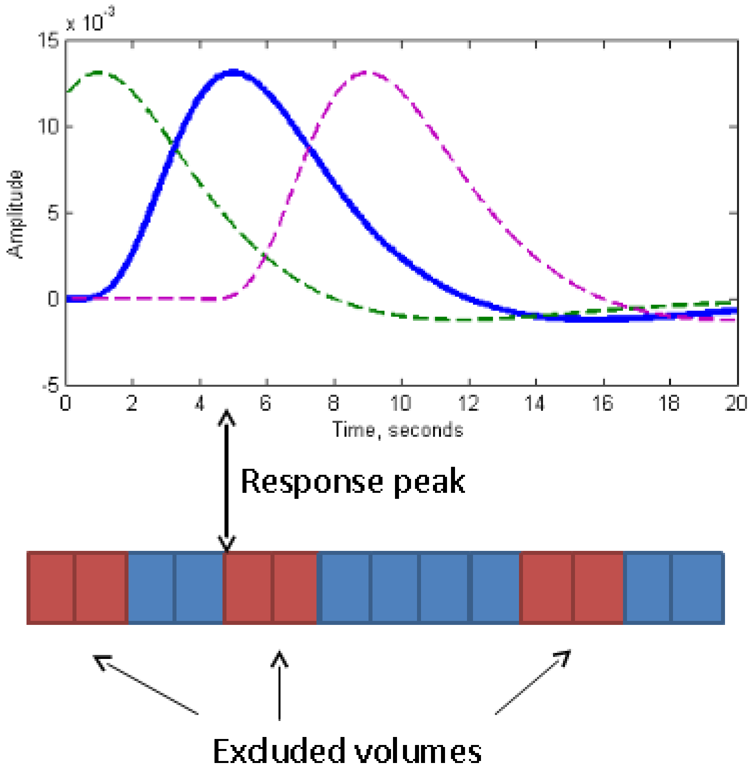


Figure 3.6: Random trials excluded equivalent to a 6 second window surrounding the peak response of a trial.

This exclusion of random trials is accomplished by first performing the exclusion window process described in Section 3.2.3 to determine the number of trials which must be excluded, then randomly selecting an equal number of trials from elsewhere within the run to exclude instead. In addition to these randomly excluded trials, the trials belonging to the test set are always excluded. An overview of this process is given in Figure 3.7.

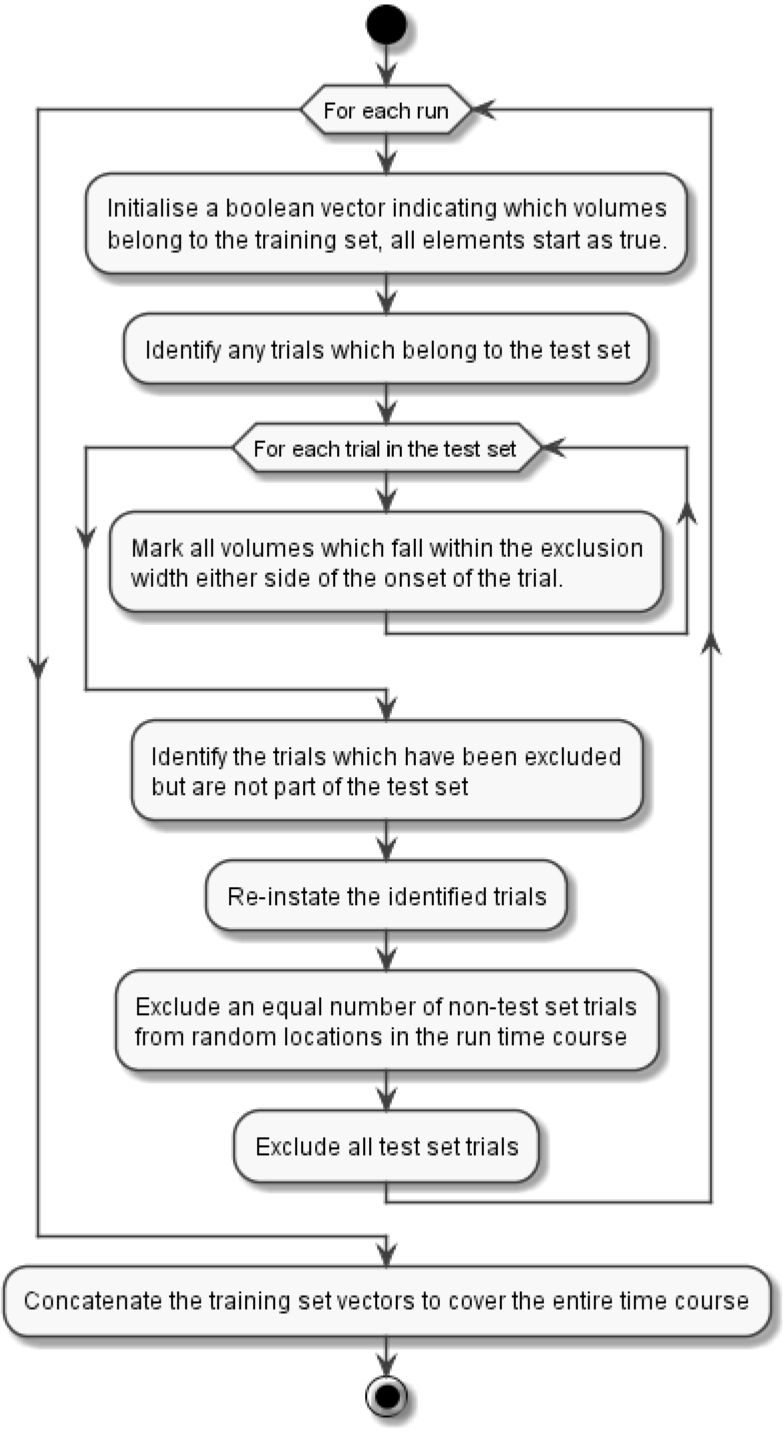

For each run
Initialise a boolean vector indicating which volumes belong to the training set, all elements start as true.
Identify any trials which belong to the test set
For each trial in the test set
Mark all volumes which fall within the exclusion width either side of the onset of the trial.
Identify the trials which have been excluded but are not part of the test set
Re-instate the identified trials
Exclude an equal number of non-test set trials from random locations in the run time course
Exclude all test set trials
Concatenate the training set vectors to cover the entire time course

Figure 3.7: Activity diagram showing the process for determining the training set using random exclusions.

3.2.5 Modifications to Normalisation

In the standard analysis implementations using leave-one-run-out cross-validation the timecourse normalisation step is performed separately on each run, and includes the entire run in the normalisation. When performing normalisation in analyses which use LCO cross-validation some consideration must be given to the split between training and testing sets to avoid circular analysis, as there will be cases in which samples from both the training and testing datasets appear in the same run. Therefore when performing the normalisation the parameters of the normalisation must be determined solely on the training set data, before being applied to the entire run. Naturally, in any runs which do not contain any test samples the normalisation can be performed using the entire run time course as standard.

3.3 Experimental Procedure

The methods described in the section above propose the means of avoiding the issue of overlapping training and test sets in a manner which, when using a sufficiently large exclusion window, should eliminate the regions where the extended nature of the haemodynamic response may cause overlapping information. A number of analyses were performed using data collected for a study on the effects of learning on decision templates in the visual cortex carried out by Kuai et al (2013). In this section a brief overview of the data collected and pre-processing performed by Kuai et al (2013) is presented, followed by the MVPA and control analyses performed to evaluate the validity of the leave-*c*-out cross-validation method and determine the size of exclusion window necessary to eliminate overlap.

3.3.1 Observers

Nine healthy students (age=21.1+-0.75) from the University of Birmingham volunteered for the study. They participated in fMRI scans both before and after a set of behavioural training sessions, and seven of the participants took part in a further post-training scan involving stimuli which differed in size from those they had trained with.

3.3.2 Stimuli

The stimuli presented were pentagons formed from 30 Gaussian dots. Two classes of stimuli were generated by varying the length of the sides of the pentagons. Multiple levels of stimuli between Class I and Class II were generated by use of linear morphing.

3.3.3 fMRI Design

All fMRI sessions comprised seven or eight runs which used an event related design. Seven conditions were presented in each run; six were stimulus conditions with stimuli levels taken from the range provided by the linear morphing, with the seventh being a fixation condition consisting of a fixation point at the centre of the screen. In each run 14 trials per condition were presented. In addition to the 98 trials of fixation and stimuli conditions, an additional three 9s blocks of fixation were included at the beginning, middle and end of each run.

Trial presentations were designed to align with the volumes acquired with a duration of 2 volumes (3s). The first volume covered the 200 ms stimulus presentation and a 1300 ms blank. The second volume of each trial covered the participant's response; a colour cue was presented for 1200 ms followed by a fixation for 300 ms. During this time participants were expected to press a key indicating the class to which they believed the stimulus belonged. To control for correlations between the participant's behavioural and neural responses the colour of the cue indicates the finger to use to indicate to which class they believe the stimulus belongs. A green cue indicates the index finger should be used for Class I and the middle finger should be used for Class II, while a red cue reverses this mapping.

The order in which trials were to be presented differed across runs and observers, and was generated such that despite the overlap in haemodynamic responses of adjacent trials there would be no statistical correlation between a trial and the one preceding.

3.3.4 fMRI Data Acquisition

The fMRI scanning sessions were performed at the Birmingham University Imaging Centre using a 3T Achieva Philips scanner with an eight-channel head coil. For localisation and visualisation of the functional data anatomical images were obtained for each participant using a sagittal three-dimensional T1-weighted sequence (voxel size=1 x 1 x 1 mm, slices-175). Functional EPI images were acquired using a high-resolution gradient echo-pulse sequence (20 slices at 1.5 x 1.5 x 2mm resolution; TR: repetition time, 1500 ms; TE: time to echo, 34 ms; 4 dummy scans and 216 volumes acquired per run). The volume encompassed by these images was positioned to cover the occipital and posterior temporal cortex .

3.3.5 fMRI Data Preprocessing

Preprocessing was performed using Brain Voyager QX (Brain Innovation, Maastricht, Netherlands). For the functional data this pre-processing included slice scan time correction, three-dimensional motion correction, linear trend removal, and temporal high-pass filtering. The resulting functional images were aligned and transformed into Talairach space. The first functional session was aligned to the anatomical data, and subsequent sessions scans were aligned to the first functional volume of the first session.

3.3.6 ROI Localisation

The regions of interest used for analysis were those identified in Kuai et al (2013) as involved in shape processing; these were the early regions V1 & V2, and the higher ventral regions V3v, hV4 and LO. Kuai et al (2013) identified the retinotopic visual regions used standard retinotopic mapping procedures as described by Sereno et al (1995) and Wandell et al (2007), while LO was defined as a sub region of LOC which showed stronger activation for intact images than scrambled images (Kourtzi & Kanwisher, 2001).

3.3.7 Multivariate Pattern Analysis

Eight multivariate pattern analyses were performed on the dataset described above. Four analyses were performed using the leave-*c*-out cross-validation procedure described in Section 3.2 both with no exclusion window and with exclusion window sizes of 6, 12 and 18 seconds. A further four analyses were performed using the same LCO cross-validation procedure and exclusion window sizes, however volumes were excluded using the random exclusion method described in Section 3.2.4 in order to control for the effects of the reduction in trials available in the training set with the increase of exclusion window sizes. All leave-c-out crossvalidation analyses were performed with 100 folds and, barring the difference in exclusion methods, all eight of these analyses use the method described below.

Feature selection was performed with the threshold and ordering method described in Section 2.5.1. Voxels were first thresholded to include only those which showed significantly ($p < 0.05$, uncorrected) more activation during stimulus conditions than fixation. Next voxels were sorted in descending order of t-value with the first 150 voxels for each ROI being selected; in regions with fewer than 150 voxels available after thresholding all voxels would be included.

Normalisation was performed using the z-score method described in Sections 2.3 with the modifications described in Section 3.2.5. Sample estimation used the miniblocks method described in Section 2.4 with the modifications described in Section 3.2.1. The haemodynamic delay was accounted for by shifting the fMRI time series by 3 volumes (4.5 s). Samples for each trial were generated by taking the mean of the values of the two volumes of the trial. These samples were then grouped into blocks of six trials of the same condition, with the a value for each block then being calculated as the mean of the six trials.

Classification was performed using a linear SVM model. The classifier was trained to categorise blocks as either being stimuli from Class I or Class II as indicated by the participant in each trial. The potentially unequal number of trials from Class I and Class II in the training set was controlled for using the inbuilt cost-factor in SVMLight (Joachims, 1999) which weights the cost of mis-categorising training samples.

3.4 Results

As described in Section 3.3.7 the first set of analyses applied the leave-*c*-out cross-validation method to the data from Kuai et al (2013) using both no exclusion window and exclusion windows of sizes 6, 12, and 18 seconds either side of the response peak. The results for these analyses are presented below in Figure 3.8 and Table 3.1.

Table 3.1: Comparison of the effects of not excluding volumes and excluding volumes with various window sizes using leave-c-out cross-validation.

Exclusion Window Width (seconds)		**Regions of Interest**				
		V1	**V2**	**V3v**	**hV4**	**LO**
6	*t*	4.48	4.43	4.63	6.85	3.89
	p	**0.000222**	**0.000244**	**0.000163**	**0.00000275**	**0.000730**
12	*t*	3.51	3.82	5.68	7.74	4.01
	p	**0.00156**	**0.000839**	**0.0000217**	**0.000000647**	**0.000562**
18	*t*	4.05	4.63	5.06	8.84	3.87
	p	**0.000527**	**0.000162**	**0.0000700**	**0.000000122**	**0.000758**

Degrees of freedom = 15. Bold values indicate significance to P < 0.05 after Bonferroni correction (P < 0.0033).

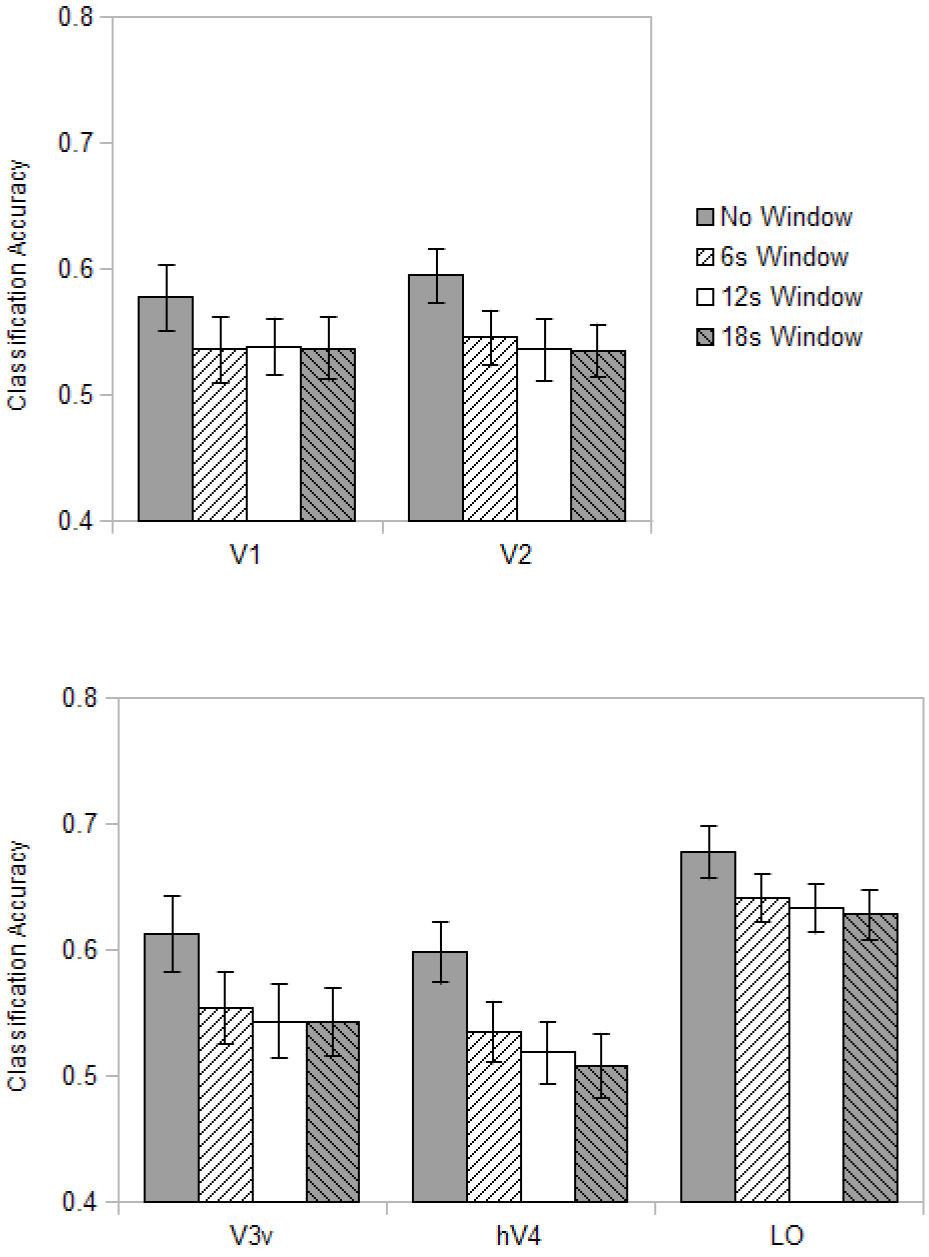


Figure 3.8: Comparison of exclusion window widths when applying n-fold with exclusion windows. Acccuracy is given as a proporition of correct predictions, with chance performance being at 0.5.

The second set of analyses applied the leave-*c*-out cross-validation method to the data from Kuai et al (2013) using both no exclusion window and the exclusion of random trials equivalent in number to exclusion windows of sizes 6, 12, and 18 seconds either side of the response peak. The results for these analyses are presented below in Figure 3.9 and Table 3.2.

Table 3.2: Comparison of the effects of not excluding trials and excluding random trials with various window sizes using leave-c-out cross-validation.

Exclusion Window Width (seconds)		**Regions of Interest**				
		V1	**V2**	**V3v**	**hV4**	**LO**
6	*t*	-0.198	1.02	1.19	2.01	0.819
	p	0.577	0.163	0.127	0.0312	0.213
12	*t*	-0.195	2.45	1.86	4.02	1.71
	p	0.576	0.0135	0.0414	**0.000552**	0.0541
18	*t*	-0.0366	2.12	2.47	3.23	2.50
	p	0.514	0.0255	0.0130	**0.00278**	0.0122

Degrees of freedom = 15. Bold values indicate significance to P < 0.05 after Bonferroni correction (P < 0.0033).

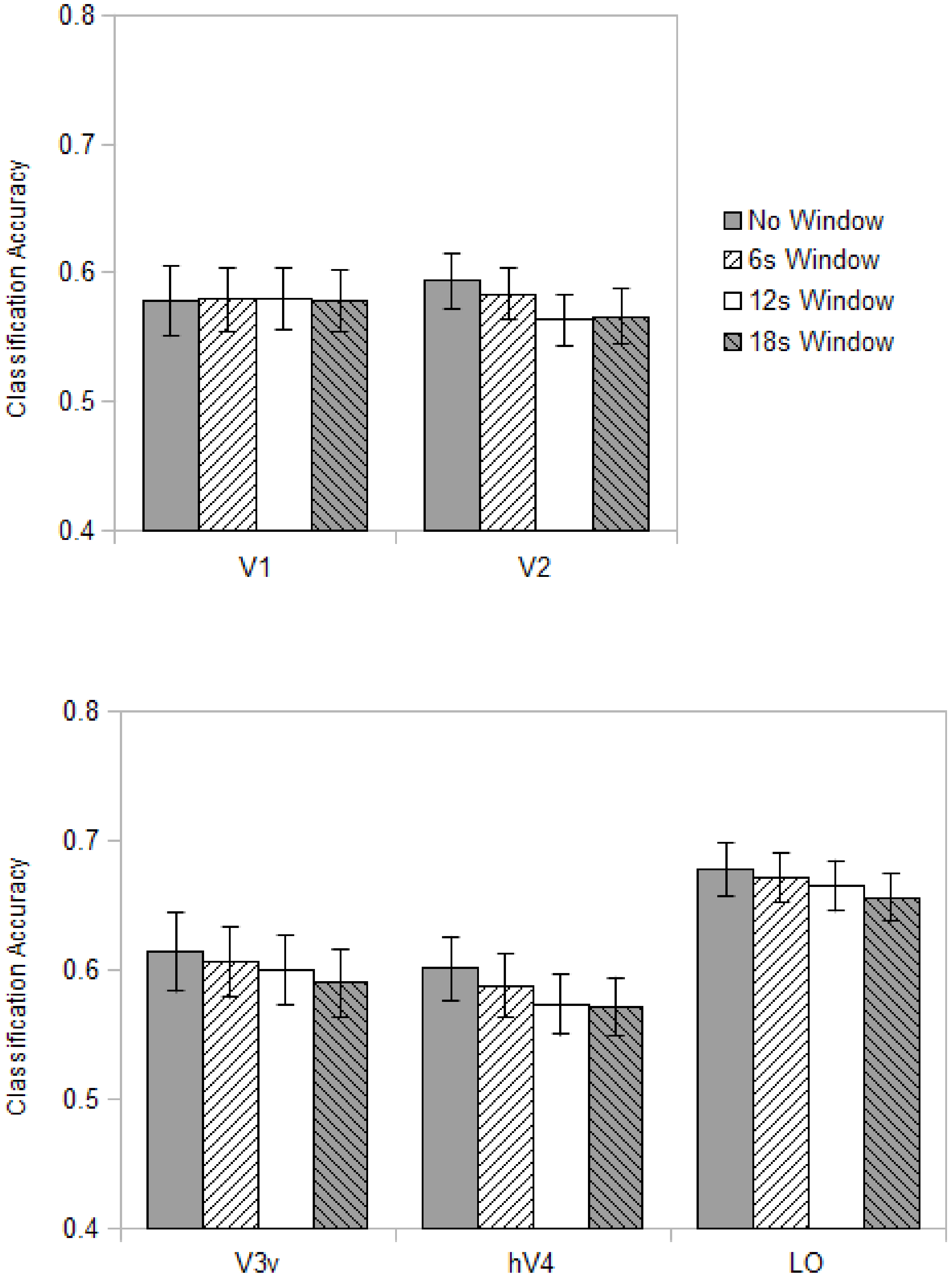


Figure 3.9:Comparison of exclusion window widths using the random exclusion method. Acccuracy is given as a proportion of correct predictions, with chance performance being at 0.5.

The final analysis considers the difference in accuracy which results from the application of the two previous methods by calculating the distribution of paired differences in accuracy of each subject included in the analyses. The results for this analysis are presented below in Figure 3.10 and Table 3.3.

Table 3.3: Comparison of the excluding windows of trials and excluding random trials.

Exclusion Window Width (seconds)		**Regions of Interest**				
		V1	**V2**	**V3v**	**hV4**	**LO**
6	*t*	7.63	5.58	5.20	7.63	5.88
	p	**0.00000154**	**0.0000525**	**0.000109**	**0.00000154**	**0.0000301**
12	*t*	6.23	2.92	6.56	5.62	4.23
	p	**0.0000162**	0.0105	**0.00000905**	**0.0000488**	**0.000735**
18	*t*	5.39	4.90	4.57	7.28	3.16
	p	**0.0000756**	**0.000192**	**0.000366**	**0.00000268**	**0.00645**

Degrees of freedom = 15. Bold values indicate significance to P < 0.05 after Bonferroni correction (P < 0.0033).

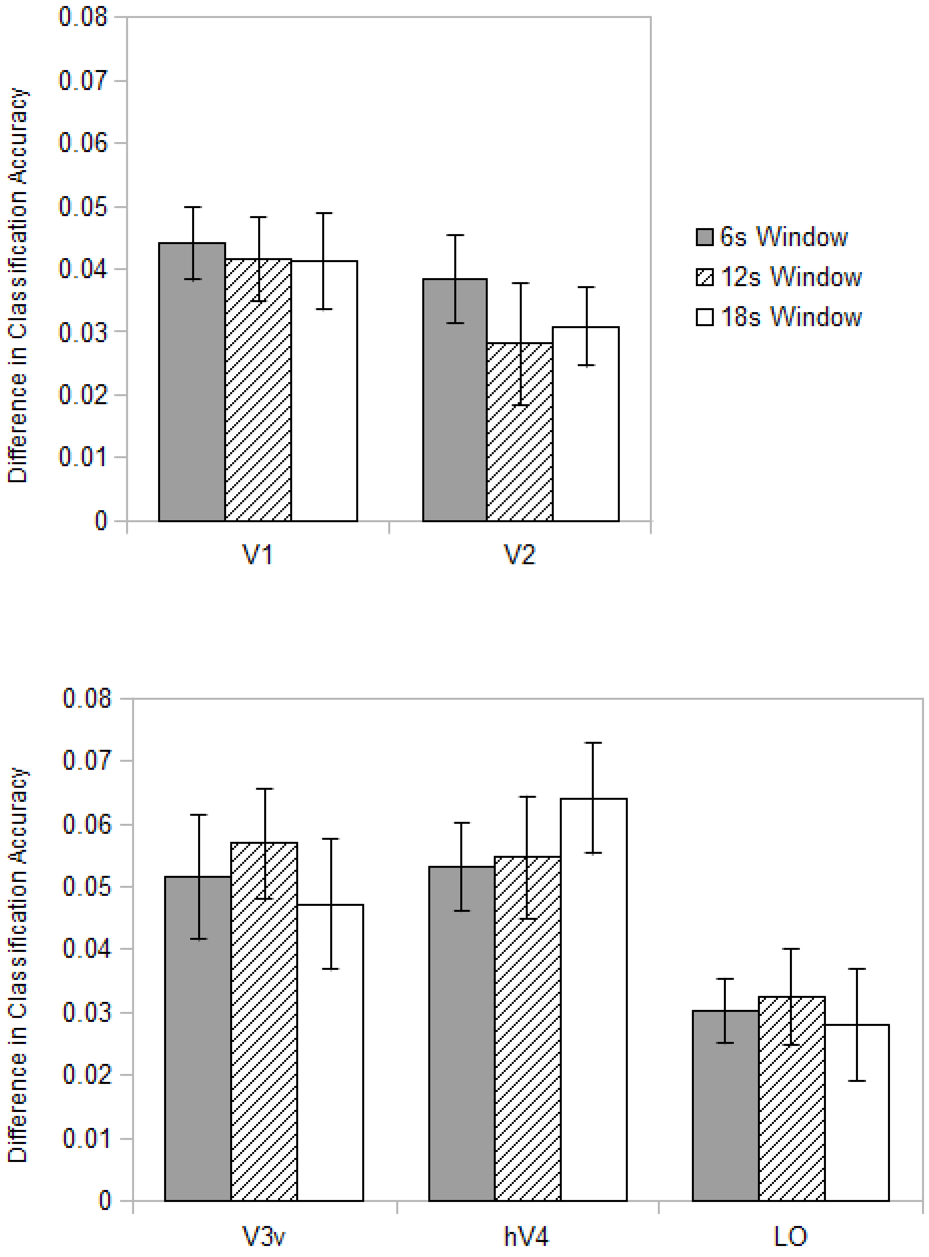


Figure 3.10: Difference in classification accuracy between exclusion window and random exclusion methods. The difference in accuracy was calculated on a per subject basis, with the values plotted showing the mean and standard error across all subjects.

3.5 Discussion

As explained earlier in this chapter, in order to calculate a stable estimate of the performance of a classification model using cross-validation it is desirable to have a sufficient number of stratified cross-validation folds. The popular leave-one-run-out (LORO) method relies on the fMRI data to ensure these properties, and as such may struggle in situations where it does not.

The leave-*c*-out (LCO) method described in this chapter is designed to provide a means of performing cross-validation which does not require the data on which it is applied to have these properties, but rather provides a user selectable number of cross-validation folds with well balanced test sets regardless of the underlying data.

As described in Section 3.2.2 the method by which test sets are generated ensures that each contains one sample from each condition involved in the classification, while minimising the number of samples excluded from the training set by not including those conditions which are not involved in the analysis being applied. Also, by design the number of cross-validation sets can be controlled as a user specified parameter, with the constraint that in order to ensure each sample appears in at least one test set, the number of cross-validation sets must be equal to or greater than the total number of samples belonging to the most populated condition.

While the method by which test sets are generated ensures a user specifiable number of balanced test sets, the fact that it results in both training and testing samples being drawn from the same run introduces the possiblity of an overlap of information between the training and testing sets. Inspired by event related designs, which limit interference between adjacent trials by leaving a gap to allow for the haemodynamic response to return to baseline, the method described in Section 3.2.3 is designed to address the issue of adjacent trials causing an overlap between testing and training sets by excluding from the training set any trials which fall within a specified window either side of the trials in the test set. As can be seen in Figure 3.4 a window of 6 seconds either side of the peak activation covers the majority of the response while a window of 18 seconds covers almost the entirety of the haemodynamic response function, therefore the appropriate window size should fall in this range.

In order to evaluate the LCO cross-validation method it was applied with both no exclusion window and exclusion windows of size 6, 12 and 18 seconds to the dataset described in Section 3.3. As can be seen in Figure 3.8, excluding a window of volumes around each test sample eliminates the boost in accuracy characteristic of overlapping testing and training sets, with a significant difference between analyses which exclude a window and those which do not being seen in all regions regardless of window size with a $P < 0.05$ after correction for multiple comparisons (Table 3.1).

While a significant effect of excluding a window of volumes around the test samples has been shown, the effect of overlapping training and testing sets is not the only explanation of the difference in accuracy. As increases in window size eliminate more of the haemodynamic response caused by the test trials, these increases also eliminate more of samples from the training set which can be expected to lead to a reduction in the classification performance of models trained on it.

To control for this confound a separate set of multivariate pattern analyses was performed in which, rather than excluding samples which fall within a window either side of the test samples, an equal number of samples were eliminated randomly from the training dataset with no regard to their position. The results of these analyses (Figure 3.9), as expected, show a trend towards a reduction in accuracy as the number of excluded trials increases; however, at no point is there a significant difference in accuracy between the analyses which exclude trials and those which do not as was found in the previous set of analyses (Table 3.2). This suggests that the significant difference between analyses which do and do not exclude window of trials around the test samples shown in Figure 3.8 is indeed the result of eliminating overlap.

To further support this observation, the difference between the control analysis and the standard LCO cross-validation analysis is examined. Since both analyses were applied to the same data the difference in accuracy between the two methods can be directly calculated; the mean of this difference over all participants is shown in Figure 3.10. As shown in Table 3.3 there is a significant difference shown in almost all regions for all three exclusion window sizes, which further supports the method.

Finally, the typical duration of the haemodynamic response to a stimulus suggests that a window size of 18 seconds should ensure that any information relating to that trial is excluded, since the haemodynamic response will have returned to baseline levels. The results shown in Figures 3.8 and 3.10 suggest that window sizes as small as 6 seconds may be sufficient, which is not unexpected as a 6 second window either side of the peak response covers the majority of the activation as can be seen in Figure 3.4.

In conclusion the leave-*c*-out cross-validation method presented in this chapter allows for the provision of an arbitrary number of cross-validation sets and ensures a balanced representation of conditions within each test set, while avoiding any issue of overlap between training and test sets as discussed above.

CHAPTER 4 - LEARNING AND THE TUNING OF FUNCTIONAL MAGNETIC RESONANCE IMAGING PATTERNS

4.1 Introduction

The ability to quickly locate and recognise objects, particularly in cluttered scenes, is both a useful important skill and a complicated, involved process. Not only must a robust model of the object be derived from characteristic visual features, but these features must be combined in a fashion which is capable of distinguishing between similar features belonging to different objects, and which is robust when faced with variations in properties such as position, apparent visual size and orientation. The task of object recognition is further complicated by the distractors present in cluttered scenes, which necessitates the selection of relevant visual features for the model which are not disrupted or confounded by the clutter.

Learning has been shown to improve the ability to perform this process, and it has been suggested that this improvement takes the form of tuning neural selectivity to behaviourally relevant processes (for review see Kourtzi and DiCarlo, 2006). The learning mechanisms by which this neural selectivity for visuals forms is shaped however, remains essentially unknown.

When considering the behaviour of neurons or groups of neurons in terms of the response evoked by each member of a set of visual stimuli of various categories, we can see that each group will have a preferred stimulus category to which they are most responsive, and consequently a set of non-preferred stimuli to which they are less responsive. Considered from this viewpoint, changes in neural selectivity occurring due to learning would take one of three forms: enhanced response to preferred stimuli, decreased response to non-preferred stimuli, or a combination of the two.

A team at the University of Birmingham Cognitive Neuroimaging Laboratory (CNIL) conducted research to ascertain which of these three potential mechanisms mediated changes in neural selectivity resulting from learning, using a combination of behavioural and fMRI measurements.

A morphing stimulus space was used, which was generated by varying the spiral angle between radial and concentric, which resulted in stimuli which vary in similarity (see Figure 4.1). Observers were presented with stimuli with differing amounts of background noise (i.e. high-signal and low-signal stimuli) and were tasked with categorising these stimuli as being either radial or concentric patterns. Observers' choices (i.e. the proportion of stimuli predicted as concentric patterns) were measured for high-signal stimuli and compared with those recorded for low-signal stimuli, both before and after training.

The sensitivity of high resolution fMRI measures were exploited to investigate fine learning dependent changes in the preferences of large neural populations. The author of this thesis was brought into the team specifically to implement, adapt and develop methods to characterise these preferences, both in terms of the general behaviour of populations of voxels and in terms of the multivariate patterns typically analysed using multi-voxel pattern classification methods.

A method for the characterisation of the average tuning of populations of voxels was adapted from a method employed by Serences et al. (2009) to investigate the effects of attention on tuning of neural preferences. This method generated tuning functions for each voxel which mapped the voxel's activity in terms of BOLD signal to the offset from the voxel's preferred stimuli. By pooling these tuning functions across multiple voxels it was possible to fit them with a Gaussian function, and thus characterise the tuning of the pool of voxels in terms of the amplitude and width of the fitted Gaussian function.

Having implemented this voxel-based tuning method and adapted it to the requirements of this specific study, the author of this thesis then proceeded to develop a method for the investigation of the tuning of multi-variate patterns which characterised tuning in a similar manner to that of the voxel-based tuning analysis. Rather than forming tuning functions for pools of voxels based on BOLD signal, a multi-voxel pattern classification analysis was used to perform a six-way categorisation of the stimuli based on fMRI activation. The predictions generated by the classification were then used to generate tuning functions as mappings from the offset from the correct category to the proportion of samples classified with that offset from the correct category. As with the voxel-based tuning functions these pattern-based tuning functions were fitted with Gaussian functions, allowing for their characterisation in terms of amplitude and width, with meanings analogous to those of the voxel-based tuning functions.

Through consideration of the distribution of mis-classified stimuli rather than just the classification accuracy, this pattern-based analysis serves to examine the changes to the multi-variate encoding of stimuli (particularly when used in conjunction with voxel-based tuning methods of Serences et al.), while retaining the increased sensitivity of the existing MVPA methods (Norman et al., 2006).

Both the voxel-based and the pattern-based tuning functions developed by the author of this thesis were used in conjunction with the behavioural measures to investigate the link between behavioural and fMRI based learning changes through comparison of observers' performance before and after training.

The outcome of this research (Zhang et al. 2010) is described in this chapter, with a particular focus on the multi-variate analysis methods, the development of which were the primary contribution of the author of this thesis.

4.2 Materials and Methods

While this chapter primarily focuses on the contributions of the author of the thesis to this study, it is necessary to describe the full method employed in order to provide the required context in which to discuss the multivariate analysis methods. The experiment design, data collection, pre-processing and analysis of results were conducted by the other members of the team. The author of this thesis was responsible for the development of the multi-variate analysis methods and their corresponding control analysis methods described in Section 4.2.7.

4.2.1 Participants

Ten observers participated in the study; two males and eight females, with an age range of 19 to 37 years. All observers had normal or corrected-to-normal vision, gave written informed consent, and were paid for their participation. The data from two observers were excluded, one as a result of low behavioural performance after training and the second as a result of poor fMRI signal quality. The study was approved by the local ethics committee.

4.2.2 Stimuli

Stimuli were Glass patterns (Glass, 1969) generated using previously described methods (Li et al, 2009). In particular, stimuli were defined by white dot pairs (dipoles) displayed within a square aperture (7.7°× 7.7°) on a black background (100% contrast). For each dot dipole, the spiral angle was defined as the angle between the dot dipole orientation and the radius from the centre of the dipole to the centre of the stimulus aperture. Each stimulus comprised dot dipoles that were aligned according to the specified spiral angle (signal dipoles) for a given stimulus and noise dipoles for which the spiral angle was randomly selected. The proportion of signal dipoles defined the stimulus signal level.

Glass patterns were generated at varying levels of spiral angle between radial (0°) and concentric (90°) (Figure 4.1, Figure 1A from Zhang et al., 2010). Half of the observers were presented with clockwise patterns (0° to 90° spiral angle) and half with counter clockwise patterns (0° to -90° spiral angle). A new pattern was generated for each stimulus presented in a trial, resulting in stimuli that were locally jittered in their position.

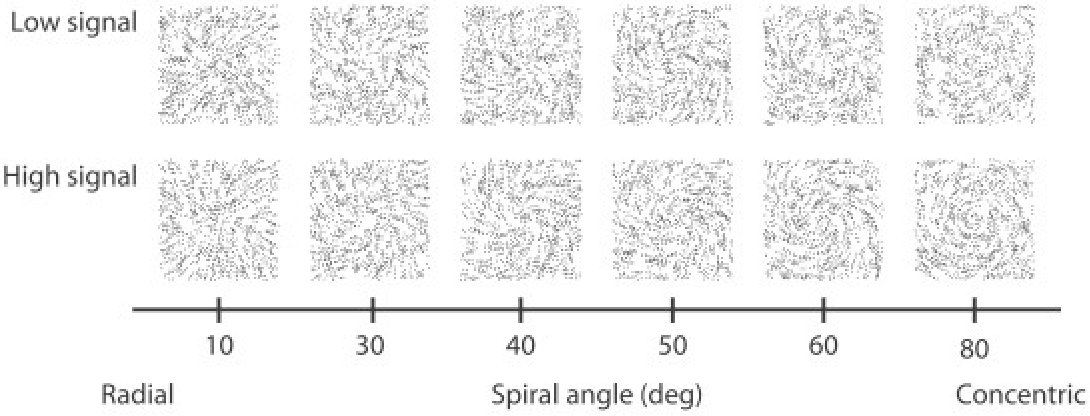


Figure 4.1: Glass pattern stimuli. (Figure 1A from Zhang et al., 2010).

4.2.3 Psychophysical Training

Initially, observers were familiarised with the task with a short (20 trial) practice session, then a baseline of their behavioural performance was measured by performing a pre-test session comprising one complete run, without feedback. Following this, observers participated in three training sessions conducted on different days, which consisted of five training runs with audio feedback on errors and one test run without feedback.

All psychophysical runs used for training or measuring behavioural performance comprised 160 trials, 16 from each stimulus condition. Stimuli were presented at 10 different spiral angles: 5°, 15°, 25°, 35°, 42°, 48°, 55°, 65°, 75° and 85°. Each trial lasted 1.5s and the stimulus was presented at 45% signal level for 200ms.

Observers were instructed to indicate whether each presented stimulus was more similar to a radial Glass pattern or a concentric Glass pattern by pressing one of two buttons on a mouse. The buttons for different stimulus categories were counterbalanced across observers.

4.2.4 fMRI sessions

Observers participated in three fMRI sessions; one following the psychophysical pre-test but before any training, and one following the last training session. Both of these session used stimuli presented at 45% signal level. A third fMRI session was performed after the post-training fMRI sessions using high-signal stimuli (80% signal level).

During scanning, observers performed the same categorisation task as during the psychophysical sessions. Each scanning session comprised eight experimental runs, each of which lasted 364s. Each run comprised eighteen stimulus blocks each with a duration of 18s. A 10s fixation block — in which only a fixation square was presented on the screen — was presented after every six stimulus blocks, as well as at the beginning and end of each run.

Each stimulus block was repeated three times in each run. The order of the blocks was randomised within each run, and each block was presented only once between two fixation blocks. Each stimulus block comprised 12 trials, including target and distractor stimuli, such that 10 trials contained target stimuli presented at one of six conditions (i.e. spiral angles), whereas two trials contained distractor stimuli from another condition. Possible combinations of spiral angles (target/distractor stimuli) presented in a block were 10°/50°, 30°/60°, 40°/80°, 50°/10°, 60°/30° and 80°/40°.

The presentation order of target and distractor stimuli within each block was randomised; one of the distractors was presented in the first six trials and the other in the last six trials. Each trial lasted 1.5s; stimuli were presented for 200 ms each and separated by a 1300 ms inter-stimulus interval, during which observers made their response to the stimulus by pressing one of two keys. The colour of the fixation square, which was presented during fixation blocks and throughout each trial, served as a cue for the motor response. If the cue was red, observers used the same key category matching as during the psychophysical training sessions (e.g. left key for concentric patterns), whereas if the cue was green, observers switched finger key matching (e.g. left key for radial patterns). The colour of the fixation square changed after every six stimulus blocks (i.e. before each fixation block) and was counterbalanced across runs.

4.2.5 fMRI data acquisition

The experiments were conducted at the Birmingham University Imaging Centre using a 3 T Philips Achieva MRI scanner. T2*-weighted functional and T1-weighted anatomical (1 × 1 × 1mm resolution) data were collected with an eight-channel head coil. Echo planar imaging data (gradient echo-pulse sequences) were acquired from 28 slices (repetition time, 2000 ms; echo time, 34ms; 1.5 × 1.5 × 2 mm resolution). Slices were oriented near coronal covering the entire occipital and posterior temporal cortex.

During scanning observers eye movements were recorded using an ASL 6000 Eye-tracker (Applied Science Laboratories, Bedford, MA). The collected data were then pre-processed using the Eyenal software (Applied Science Laboratories, Bedford, MA) and further analyzed using custom Matlab (Mathworks, MA) scripts.

4.2.6 fMRI Data Analysis

4.2.6.1 Data Pre-processing

MRI data were processed using Brain Voyager QX (Brain Innovation B.V.). T1-weighted anatomical data were used for co-registration, three-dimensional cortex reconstruction, inflation and flattening. Pre-processing of the functional data involved slice-scan time correction, three-dimensional head movement correction, temporal high-pass filtering (three cycles), and removal of linear trends. No spatial smoothing was performed on the functional data used for the multivariate analysis. The functional images were aligned to anatomical data under careful visual inspection, and the complete data were transformed into Talairach space (voxel size of 1 × 1 × 1 mm, nearest-neighbour interpolation). Transforming the data into Talairach space ensured that the coordinates of the selected regions of interest (ROIs) for each individual subject were comparable with previous studies. When aligning the functional data to the anatomical scans, a nearest-neighbour interpolation method was used to resample the data at high resolution (1 × 1 × 1 mm). While this caused some duplication of voxels, these duplicate voxels were removed during later voxel selection. For each participant, the functional imaging data between sessions were co-aligned, registering all volumes of each observer to the first functional volume. This procedure ensured a cautious registration across sessions.

4.2.6.2 Mapping regions of interest

For each individual observer, retinotopic visual areas (V1, V2, V3d, V3a, V7, V3v and V4v) were identified based on standard mapping procedures (DeYoe et al. 1996; Sereno et al. 1995; Engel et al. 1994). Two additional scans were used to identify V3B/KO (kinetic occipital area) and the lateral occipital complex (LOC). Area V3B/KO was defined as the set of contiguous voxels anterior to V3a that showed significantly stronger activation ($p < 0.005$) for kinetic boundaries than transparent motion (Dupont et al. 1997). LOC was defined as set of contiguous voxels in the ventral occipitotemporal cortex that showed significantly stronger activation ($p < 0.005$) for intact than scrambled images of objects (Kourtzi and Kanwisher, 2000).

4.2.7 Multivariate Pattern Analysis

4.2.7.1 Voxel Selection

Sets of voxels were defined for each observer and each ROI (retinotopic areas, V3B/KO and LOC). Initially these consisted of the voxels defined by the standard ROI mapping procedures, these selections were further refined based on the difference in strength of response between stimulus conditions and fixation; only voxels with a significantly ($P < 0.05$) stronger response to stimulus conditions than fixation were retained. Following this, any voxels which contained duplicate time-courses caused by the nearest neighbour interpolation used to transform the data to Talairach space were removed. In ROIs which had more than 250 voxels remaining, all voxels bar the 250 with the most significant difference in strength between stimulus condition and fixation were discarded. Finally, to avoid introducing a bias in favour of one session over the other, the union of the selected voxels from corresponding ROIs in the pre- and post-training sessions were taken resulting in a final set of voxels for each ROI of each observer.

4.2.7.2 Sample Estimation

The time course of each voxel was z-score normalised for each run and shifted by 4s to account for the haemodynamic delay. In order to reduce noise, fMRI responses were averaged across all trials per block, resulting in 24 patterns per session for each condition. To ensure that any effects arose from multivariate patterns and not univariate differences across conditions, the grand mean response across voxels was subtracted from each voxel's timecourse.

4.2.7.3 Pattern Classification

Linear support vector machine (SVM) classifiers were used to perform this pattern classification analysis. To allow for the categorisation of multiple classes the linear binary classification model was generalised to perform multi class classification by use of a negative voting algorithm as described in section 2.1.3.

Separate binary classification models were trained for each pair of classes, resulting in a set of 15 trained classifiers. These models were trained using only the samples from the training set which belonged to the classes involved in the corresponding contrast; i.e. the training of the classification model for the pair of classes corresponding to the stimuli with 15 and 75 degree spiral angles included only samples recorded from stimuli with spiral angles of 15 or 75 degrees.

When performing classification each of the 15 classification models were applied to the test sample, with the final predicted class being determined by means of a process of negative voting, in which each of the classifier models casts a vote for the class to which it believes the test sample does not belong.   The predicted class is then determined as the class with the fewest votes, with ties being resolved by random selection between the competing classes.

In order to account for the possibility of the classification methods being biased or overpowered, a separate analysis was performed which replicates the 6-way classification process with the exception of shuffling the sample labels. This shuffled label analysis is repeated for 1000 iterations with a different shuffling pattern for each iteration.

4.2.7.4 Pattern-Based Tuning Functions

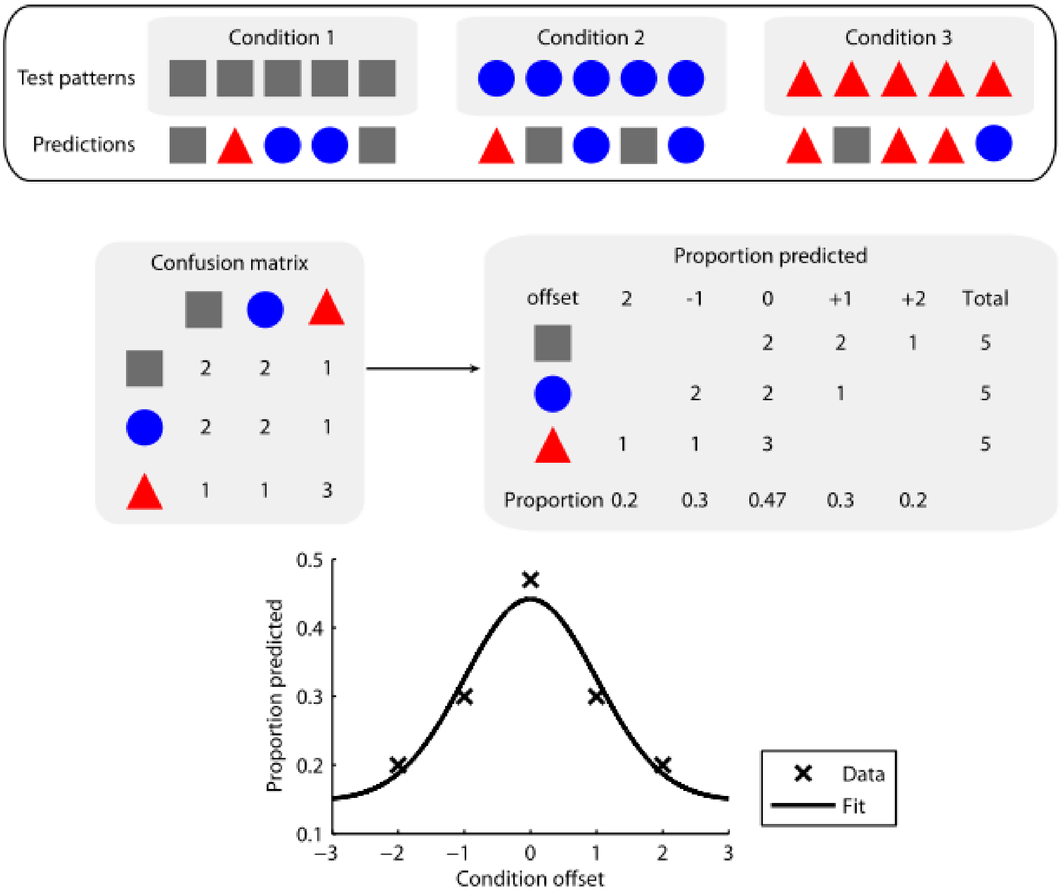
The effect of learning on the fMRI responses of the voxels was examined on a multivariate pattern based level through the calculation of pattern-based tuning functions using the results of the 6-way classification analysis. The 6-way classification model was trained and applied to the set of test samples as described in section 4.2.8.3, resulting in a predicted category for each test sample.

From these predictions a confusion matrix was calculated in which each row corresponds to the actual stimulus condition, each column corresponds to the predicted stimulus condition and the values indicate the number of samples of the actual condition which were classified as the predicted condition. This type of matrix is used in machine learning as a means of visualising the performance of supervised learning algorithms. These values were then mapped into the space of offsets from desired stimulus in terms of difference in spiral angle, as shown in Figure 4.2.

The proportion predicted values which are indicated at the bottom of the proportion predicted matrix and plotted in the tuning curve in Figure 4.2, are calculated using the following formula:

$$P(i)=\frac{n(i)}{N(i)}$$

in which *n(i)* denotes the number of patterns predicted to have distance i from the actual stimulus condition, and *N(i)* indicates the total number of patterns which could possibly have been predicted such that they had this offset *i*.



	Condition 1	Condition 2	Condition 3
Test patterns	■ ■ ■ ■ ■	● ● ● ● ●	▲ ▲ ▲ ▲ ▲
Predictions	■ ▲ ● ● ■	▲ ■ ● ■ ●	▲ ■ ▲ ▲ ●

Confusion matrix

	■	●	▲
■	2	2	1
●	2	2	1
▲	1	1	3

Proportion predicted

offset	2	-1	0	+1	+2	Total
■			2	2	1	5
●		2	2	1		5
▲	1	1	3			5
Proportion	0.2	0.3	0.47	0.3	0.2	

Figure 4.2: Illustration of the generation of pattern-based tuning functions using MVPA. The top panel provides an example of 15 patterns divided into three categories and the categories to which they are predicted by a classifier. These categories correspond to points on a linear stimulus space, and are denoted by squares, circles and triangles to facilitate easy identification by shape and colour. The confusion matrix shows for each condition (presented in rows) to which categories its stimuli are predicted (corresponding to the columns), while the matrix to the right shows these values aligned by the offset from the actual condition. The lower plot shows the calculated tuning profile as data and a fitted Gaussian function. (Figure S1B from Zhang et al., 2010).

For example: in Figure 4.2 only the 5 stimuli belonging to condition 1 could be predicted to offset +2, giving a value for *N(+2)* of 5. Only one pattern from category 1 was predicted to the offset +2, giving a *n(+2)* of 1; as such the proportion predicted value for offset +2 is *P(+2) = 0.2.*

As a result of using this proportion predicted measure the values being plotted are no longer constrained to sum to unity, which weakens the direct link between amplitude and standard deviation of the tuning functions. Additionally the interpretation of results is simplified, since accounting for the number of patterns which could be predicted to each offset results in a flat model when applied to a chance classifier.

An average pattern-based tuning function was then generated by fitting the proportion predicted, *P(i)*, values for all observers using a Gaussian function (shown below) in which α is the scaling parameter, μ the mean, s the standard deviation, and β denotes the baseline. The Gaussian function was fitted to the proportion predicted values using iterative least squares estimation. When fitting the Gaussian function to generate this tuning function the data points for offsets of ±70 degrees were excluded; since these offsets were the result of a single prediction (±10 degrees vs ±80 degrees), the small number of samples resulted in outlier values.

$$\gamma = \frac{\alpha}{\sqrt{2\pi s^2}} exp\left(-\frac{(x-\mu)^2}{2s^2}\right) + \beta$$

To quantify the pattern-based tuning functions for further analysis, values for amplitude and width were measured, by taking the value at x=0 and the standard deviation respectively, of the functions based on 1000 bootstrap samples.

4.2.7.5 Voxel Tuning Functions

Voxel tuning functions were generated based on the fMRI responses from individual voxels using a method derived from the one described by Serences et al. (2009). To ensure comparability with the MVPA and fMRI pattern-based tuning functions which result, the voxel tuning functions were generated using the same set of 250 voxels per region of interest which were previously identified. Additionally the same z-score normalisation and 4 second shift to account for the haemodynamic delay was applied to the fMRI timecourse of responses.

As with the method described by Serences et al. (2009) the first step applied was to determine the preferred stimulus of each voxel. This was defined as the stimulus condition which evoked the largest mean response across all trials in the training set as determined using the same leave-one-run-out cross-validation used by the MVPA; that is, all experimental runs bar the one run held out as the test set. Separately from this process a response profile for each voxel was produced using only the data from the test set. This response profile is formed from the mean response evoked by each stimulus condition.

To generate an overall tuning function for the ROI, the response profiles for each voxel were mapped into the space of all offsets in spiral angle from that of the preferred condition previously identified. This results in a space ranging from -70° to +70° with the preferred condition of each voxel being placed at 0°. These response profiles for each voxel in the common space of offsets from preferred condition were then pooled, with the mean across voxels forming the voxel-based tuning function for the entire ROI. This process was repeated for each fold of the leave-one-run-out cross-validation, and the resulting tuning function for each fold averaged.

To quantify these functions the average function across subjects was then fitted with the same Gaussian function as used for the pattern-based tuning functions. Robust estimates of the amplitude and width of the tuning functions were generated from 1000 bootstrap samples. This differed from the circular Gaussian method used by Serences et al. (2009) due to the difference in stimulus definition. Whereas the stimulus space used by Serences et al. (2009) ranged from 0° to 180° and consequently wrapped around, the stimuli used here range only from 10° to 80° and do not require a circular space.

4.2.7.6 Linear modelling of MVPA mis-predictions

As a result of the use of the proportion predicted measure, the area under the pattern-based tuning function is not constrained to a fixed value, and the amplitude and width of the tuning functions are not strictly linked. However, there may still be some element of coupling between the amplitude and width of the distribution, as naturally an increase in correct predictions (and thus, the amplitude) will result in a reduction in number of mis-classified samples. To account for this an additional separate analysis of the pattern of mis-predictions is performed.

This analysis looks for correlations between the magnitude of the offset in spiral angle from the preferred stimulus condition. Since this analysis excludes the correctly predicted trials, and does not consider the sign of the offset, the regression analysis performed fits a linear model to a set of values at each non-zero absolute offset value which correspond to the pooled values from both positive and negative offsets. For example, the value used for an offset of 10° would be the average of the values used for -10° and +10° in the pattern-based tuning analysis. As with the pattern-based tuning analysis, the slope measure used to quantify the linear functions was generated based on 1000 bootstrap samples.

Since the proportion predicted measure accounts for the inherent bias towards smaller offsets induced by the greater set of potential samples to predict to them, a classifier model operating at chance performance would show a flat line – i.e. no correlation between the offset and number of mis-predictions – while tendencies to mis-predict to stimulus conditions closer to the actual stimulus category would show as a line with a negative slope.

4.3 Results

4.3.1 Behavioural Performance

The observer's ability to categorise the global form patterns of stimuli as either radial or concentric were tested on high signal stimuli and before and after training with low signal stimuli. The results, as reported in Figure 4.3 (Figure 1B from Zhang et al., 2010), showed an improvement in observers' sensitivity in discriminating between conditions after training, with the 78% threshold performance for low-signal stimuli reducing after training.

Standard deviation measures were estimated from cumulative Gaussian fits on individual subject data, and shown using a repeated-measures ANOVA test to have higher values for low-signal than high-signal stimuli before training (F (1,7) = 14.35, P < 0.01), which decreased significantly following training (F(1,7) = 10.24m p < 0.05).

Additional testing in the laboratory produced similar results (Figure 4.4; Figure S2 from Zhang et al., 2010); the 78% threshold being lower after training than before, and a repeated measures ANOVA test showed significant differences in the standard deviation of the cumulative Gaussian fits for individual subjects data across sessions (F(1,7) = 32.42, p < 0.001).

The change in behavioural performance shown during the fMRI sessions and during testing in the laboratory following training sessions indicate that the training enhanced observers' sensitivity to stimulus category.

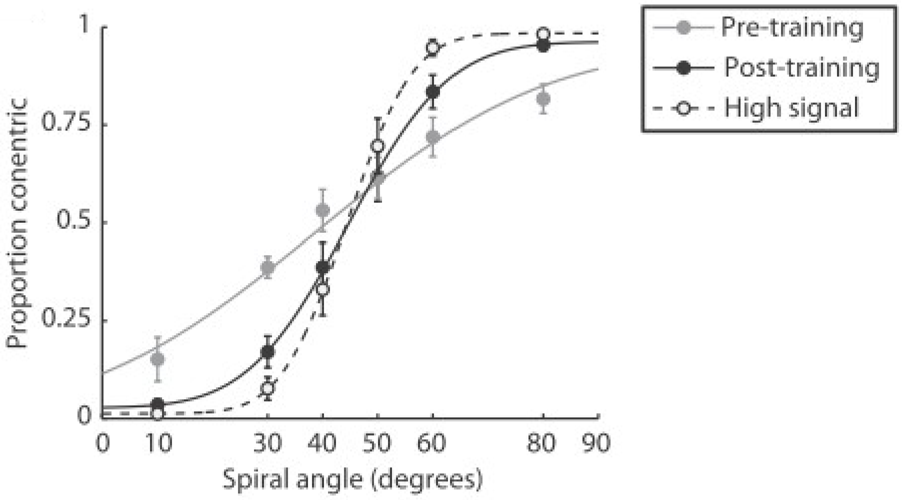


Figure 4.3: Behavioural performance during scanning. The curves indicate the best fit of the cumulative Gaussian function, while error bars indicate the standard error of the mean. (Figure 1B from Zhang et al., 2010).

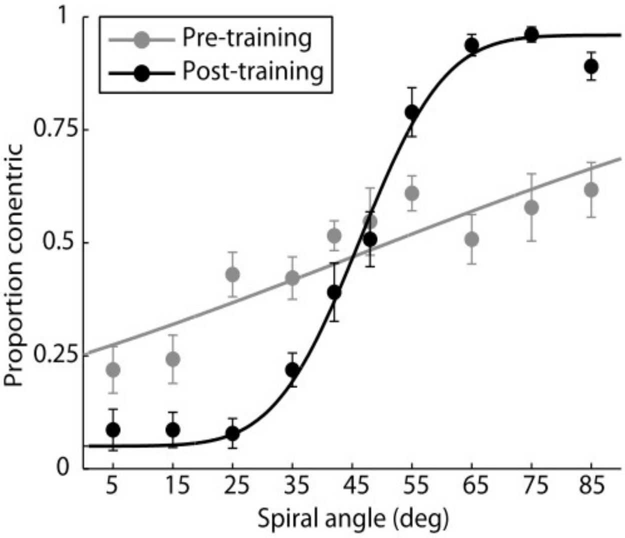


Figure 4.4: Behavioural performance during lab testing sessions. The curves indicate the best fit of the cumulative Gaussian model, while error bars indicate the standard error of the mean. (Figure S2 from Zhang et al., 2010).

4.3.2 fMRI Pattern-based Tuning Functions

Multivoxel pattern classification analysis was performed as described in Section 4.2.7.3 to determine which visual regions encoded information related to the global shapes present in the concentric and radial glass pattern stimuli. This analysis utilised an ensemble of linear Support Vector Machines (SVMs) and a negative voting algorithm to perform a six-way classification, whereby each stimulus was categorised into one of the 6 stimulus classes based on the fMRI responses which they evoked.

The predicted stimulus categories were then used to generate pattern-based tuning functions across spiral angle for each region of interest (ROI), as shown in Figure 4.5 (Figure 2 from Zhang et al. 2010). As can be seen in the resulting tuning functions a tuned response was present in all visual regions when applied to high-signal stimuli. This indicates that the classification model mis-predicted stimuli to similar spiral angles (those with a smaller offset from the actual spiral angle of the stimulus) more frequently than to dis-similar sprial angles. This suggests fMRI selecive responses for visual forms differ in local orientation signals.

To facilitate assessment of the effect of learning on fMRI selectivity, the pattern-based tuning functions were characterised in terms of their amplitude and width (i.e. standard deviation of the Gaussian fit). The amplitude of the pattern-based tuning functions corresponds to the accuracy of the six-fold classification analysis, which is shown in Figure 4.6; however since it is the result of a fitted model there is also some effect of the mis-predicted categories, particularly those with small offsets in spiral angle. The width of the pattern-based tuning functions provides indication as to the manner in which mis-classified stimuli are distributed. Larger widths correspond to more distributed mis-predictions, while narrower tuning functions occur when mis-predictions are more frequently made to smaller offsets.

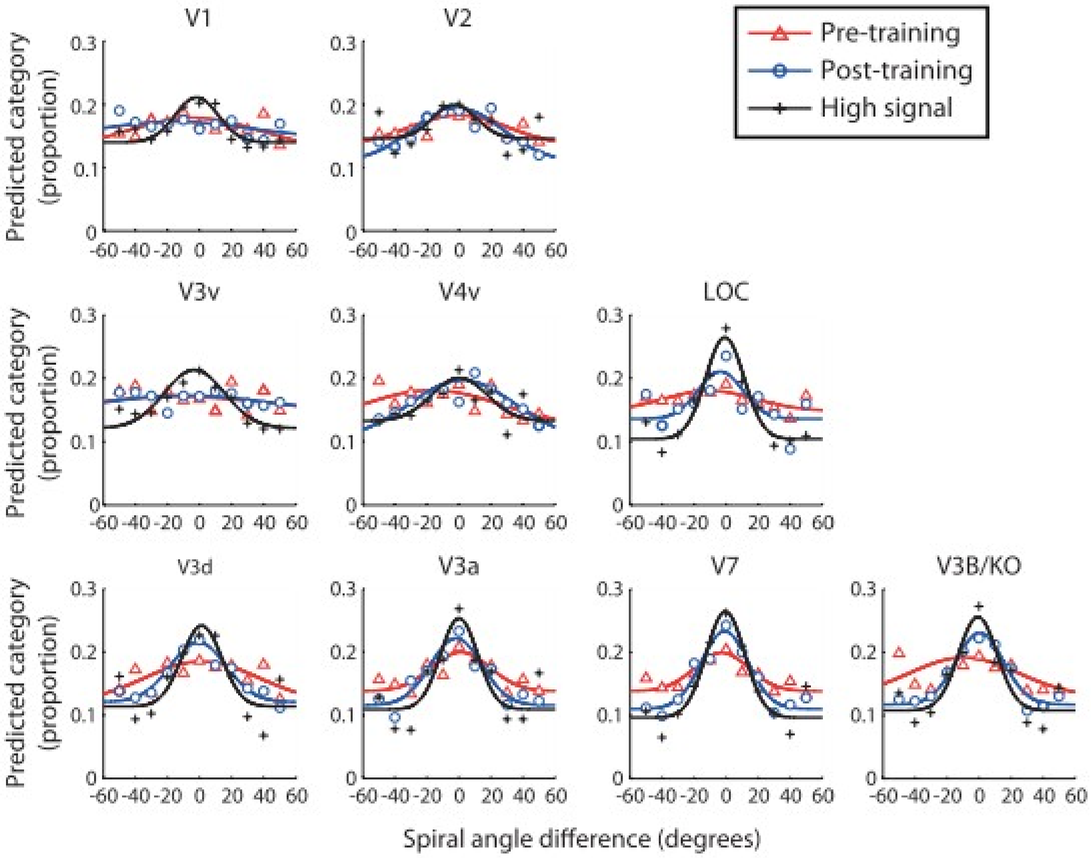


Figure 4.5: fMRI pattern-based tuning functions. The proportion of predictions made to each stimulus condition in terms of the difference in spiral angle between the presented stimulus and the prediction. Symbols indicate average data across observers, solid lines indicate the best fit of a Gaussian function to the data as calculated using 1000 bootstrap samples. (Figure 2 from Zhang et al. 2010)

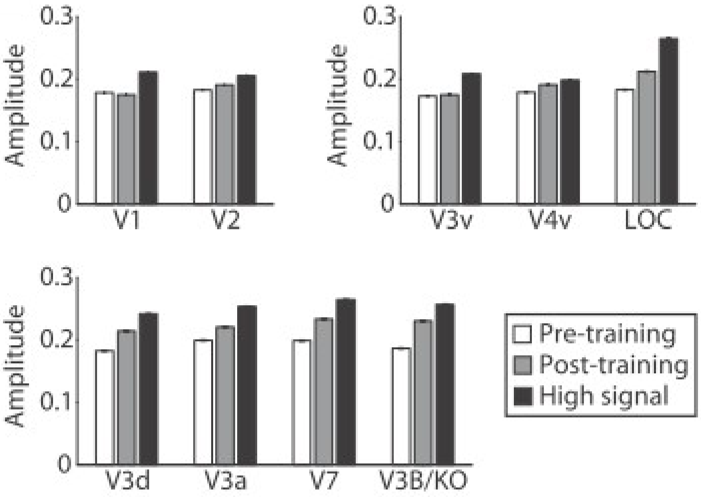


Figure 4.6: Amplitude of fMRI pattern-based tuning functions. Error bars indicate 95% confidence intervals as calculated from 1000 bootstrap samples. (Figure 3A from Zhang et al., 2010).

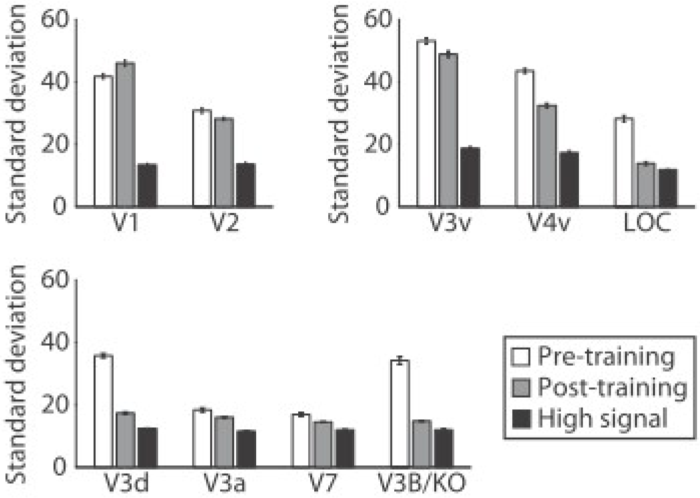


Figure 4.7: Width (standard deviation of Gaussian fit) of fMRI pattern-based tuning functions. Error bars indicate 95% confidence intervals as calculated from 1000 bootstrap samples. (Figure 3B from Zhang et al., 2010).

Using these measures comparisons were performed between the tuning functions of high-signal stimuli and those of the low-signal stimuli before and after training.

Comparison of the amplitude measures of the pattern-based tuning functions revealed that high-signal stimuli exhibited significantly higher amplitude than that of the low-signal stimuli before training in the higher visual areas, as shown in Figure 4.6 (Figure 3A from Zhang et al., 2010), which is consistent with the behavioural results reported in section 4.3.1.

Following training a significant increase in amplitude of the tuning functions of low-signal stimuli was shown in higher dorsal and ventral regions. A repeated-measures ANOVA test showed significant differences across sessions in dorsal ($F(2,14) = 15.33$, $p < 0.001$) and ventral areas ($F(2,14) = 27.26$, $p < 0.001$), but found no significant difference in the early visual regions V1 and V2, ($F(2,14) = 2.02$, $p = 0.17$).

More specifically in the dorsal visual areas the amplitude was found to be significantly higher for high-signal than low-signal stimuli before training ($F(1,7) = 64.49$, $P < 0.001$), while after training a significant increase in the amplitude of the low-signal stimuli was seen ($F(1,7) = 7.84$, $p < 0.05$). LOC exhibited similar results, however the earlier ventral areas (V3v, V4) showed no great difference, as indicated by the significant interaction between session and ventral regions ($F(4,28) = 3.13$, $p < 0.05$).

Examination of the widths of the pattern-based tuning functions - as shown in Figure 4.7 (Figure 3B from Zhang et al., 2010) - showed differences in the tuning widths across sessions, with a repeated measures ANOVA test showing significant differences in the tuning width across sessions for dorsal (F(2,14) = 20.09, P < 0.001) and ventral (F(2,14) = 16.90, p < 0.001) visual areas, though not the early (V1, V2) visual regions (F(2,14) = 1.15, p < 0.27).

The width of the pattern-based tuning functions before training was narrower for high-signal than for the low-signal stimuli for both dorsal (F(1,7) = 64.05, p < 0.001) and ventral (F(1,7) = 31.99, p < 0.01) regions. The dorsal (F(1,7) = 6.02, p < 0.05) and ventral (F(1,7) = 12.27, p < 0.01) areas also showed a significant decrease in width of the tuning for low-signal stimuli following training.

4.3.3 Control Analyses

The results of the pattern-based tuning function analysis reported in section 4.3.2 indicate that learning results in changes to tuning which form a combination of increased amplitude and decreased tuning width. However, since these tuning functions are based on a Gaussian function fitted to the distribution of a finite number of samples, it is possible that the amplitude and tuning width measures may have some element of coupling. That is, an increase in amplitude might cause a corresponding decrease in width, and vice-versa. To control for this and other confounds a number of control analyses were performed, as discussed below.

4.3.3.1 Six-way Classification Analysis

The amplitude of the pattern-based tuning functions reported previously were calculated as the amplitude of the Gaussian function with which the proportion predicted profile was fitted at an offset of zero, and as a result may have been influenced by the contribution of the nearby mis-predictions to the definition of the Gaussian function. This coupling, however, would have no effect on the accuracy of the six-way classification analysis upon which the pattern-based tuning functions were based.

The results of this six-way classification analysis, as shown in Figure 4.8 (Figure S3 from Zhang et al., 2010) are consistent with the trends shown by the amplitude of the pattern-based tuning functions shown in Figure 4.6.

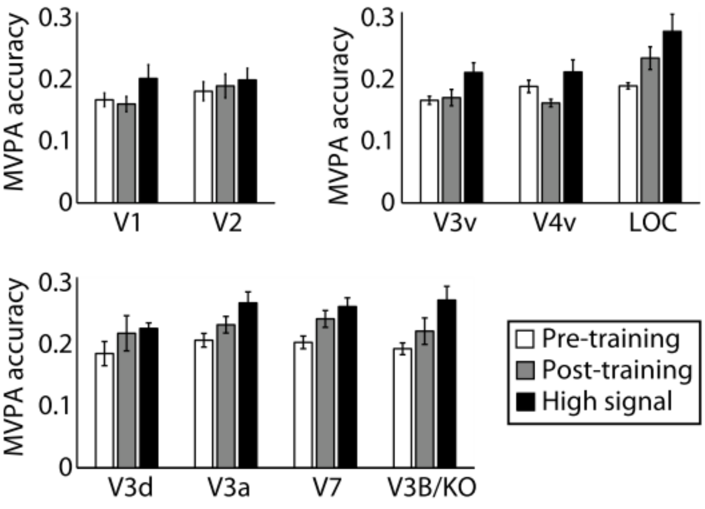


Figure 4.8: MVPA Six-way classification results. Error bars denote the standard error of the mean across observers. Chance level performance is 0.167. (Figure S3 from Zhang et al., 2010).

4.3.3.2 Linear regression of mis-categorised stimuli

In order to control for the possibility of coupling between width and amplitude of the tuning functions acting as a confound on the characterisation of changes in mis-predicted stimuli, a separate analysis of the mis-predicted samples was performed.

This analysis, as described in section 4.2.7.6, uses linear regression to characterise the distribution of mis-predicted samples in terms of the slope of the line with which they are fitted. This takes advantage of the properties of the proportion predicted metric, specifically that classification at chance performance would result in a uniform level for all offset categories. Following from this, a tendency towards mis-predictions to similar categories (and thus smaller offsets) would manifest as a negative slope, while conversely mis-predictions to dis-similar categories would result in a positive gradient.

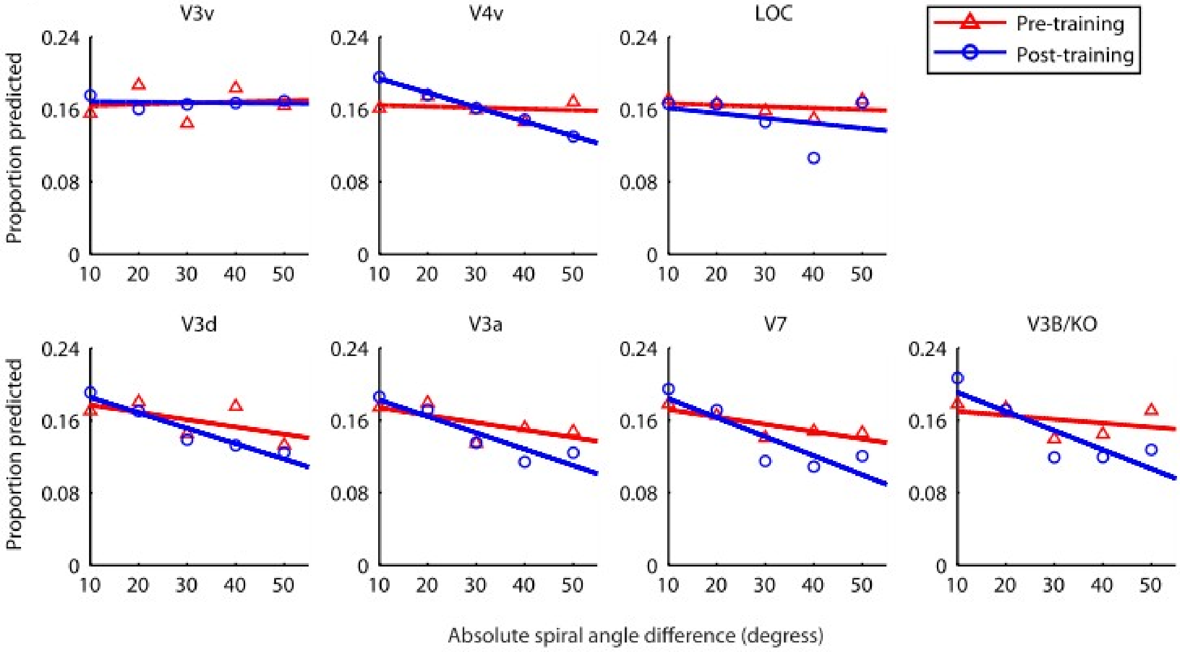


Figure 4.9: Linear Regression of mis-predicted stimuli. (Figure S4A from Zhang et al., 2010).

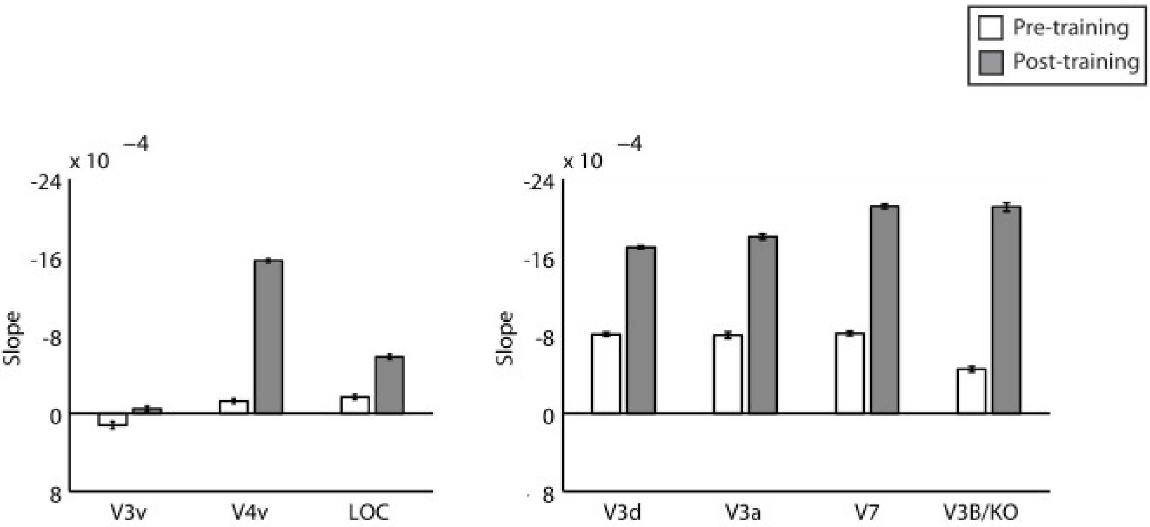


Figure 4.10: Slope of Linear Regression. Error bars indicate 95% confidence intervals calcualted from 1000 bootstrap samples. (Figure S4B from Zhang et al., 2010).

As shown in Figure 4.9 (Figure S4a from Zhang et al., 2010) and Figure 4.10 (Figure S4b from Zhang et al. 2010), following training the slope of the linear regression models increases in magnitude. This indicates that training alters the distribution of mis-prediction of the low-signal stimuli patterns such that mis-predictions to offsets closer to the actual condition are more common than mis-predictions to further offsets.

4.3.3.2 Voxel-based tuning functions

To further control for the potential coupling between amplitude and width of the pattern-based tuning functions, an additional analysis was performed to characterise the learning-dependent changes in individual voxels preferences using a method based on that employed by Serences et al. (2009). As this method does not rely on the distribution of a fixed set of samples, but rather the actual BOLD signal, this produces voxel-based tuning functions which have no coupling between the amplitude and standard deviation.

The results provided by this analysis, as shown in Figure 4.11 (Figure 4a from Zhang et al., 2009), show trends which agree with those found by the pattern-based tuning function analysis. That is, higher dorsal and ventral visual areas exhibited increases in amplitude of response, and decreases in standard deviation for low-signal stimuli following training.

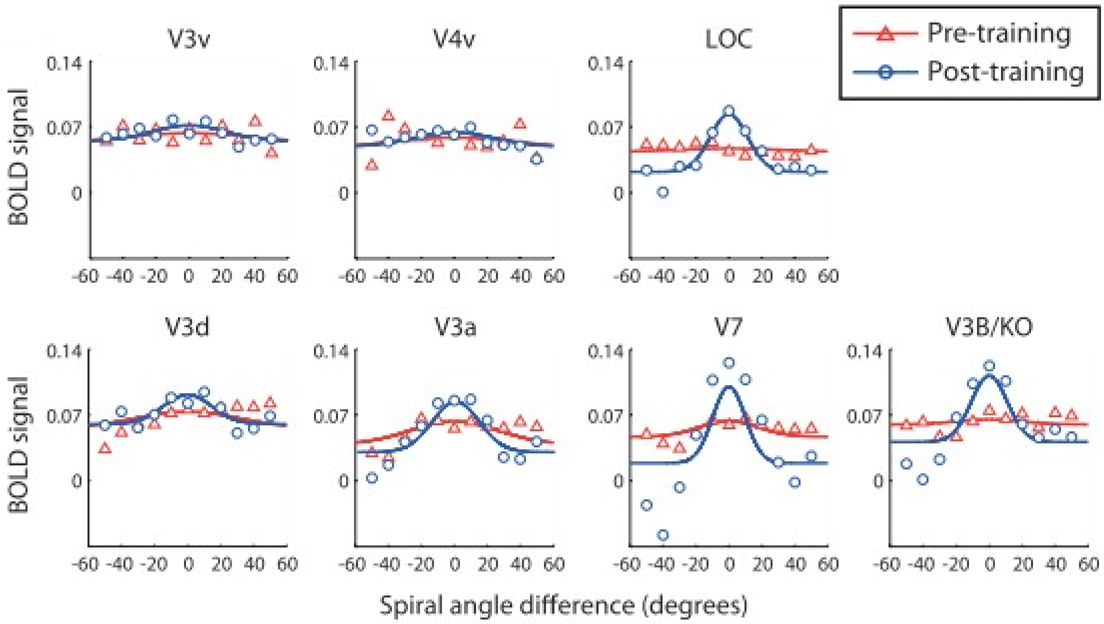


Figure 4.11: Voxel-based Tuning Functions. Data points show the mean BOLD signal across voxels plotted against the offset in spiral angle, while the solid lines show the fitted Gaussian model. (Figure 4A from Zhang et al., 2010).

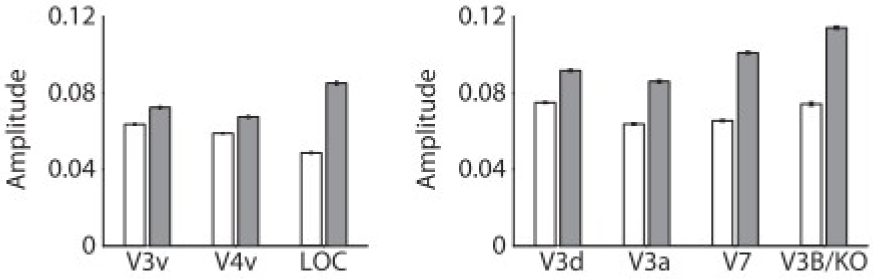


Figure 4.12: Amplitude for voxel-based tuning functions. Error bars indicate the 95% confidence intervals calculated from 1000 bootstrap samples. (Figure 4B from Zhang et al., 2010).

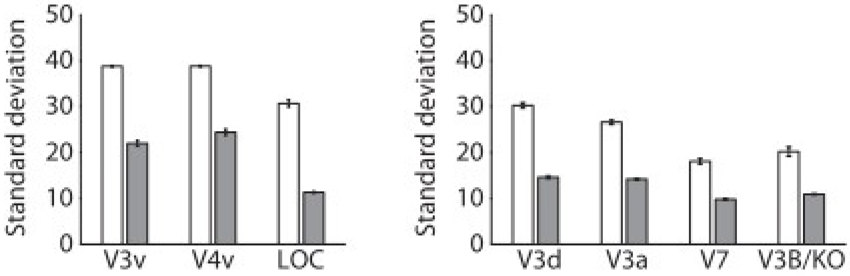


Figure 4.13: Standard Deviation for voxel-based tuning functions. Error bars indicate the 95% confidence intervals calculated from 1000 bootstrap samples. (Figure 4C from Zhang et al., 2010).

4.3.3.3 Confound controls

A series of further analyses were performed to exclude further confounds and validate the appropriateness of the data analysis methods. These analyses investigate the overall fMRI responsiveness, patterns in observers' eye movements, and the power of the MVPA classification.

The analysis of the fMRI responsivness was conducted to identify differences across sessions which could result from differences in the difficulty of the task. The results of an analysis of the percent signal change from a baseline of the fixation condition across cortical regions suggested that the differences in tuning found were not due to differences in overall fMRI signal, since no significant difference in the fMRI responsiveness across sessions was found (F(2,14) = 1.08, p = 0.37).

To account for the possibility of the effects of eye-movement on the results, eye-tracking was performed during the scanning sessions. Analysis of the collected data, shown in Figure S5 from Zhang et al. (2010), did not show a significant difference between sessions, suggesting that eye movement differences did not contribute significantly to the difference in tuning found between sessions.

Finally, to validate the classification methodology and ensure that it was neither biased or over-powered, a shuffling analysis was performed, in which the six-way classification analysis was applied to shuffled data labels. This was performed for 1000 permutations of the data labels, and the mean prediction accuracies from these 1000 iterations were found not to differ significantly from the expected chance level of 0.167, as shown in Table S1 from Zhang et al. (2010) which is included in Appendix 1.

4.4 Discussion

The findings reported in section 4.3 demonstrate that observers' sensitivity to visual forms are altered by training, with corresponding changes to fMRI selectivity in higher dorsal and ventral visual regions being apparent. Analysis of the fMRI responses using pattern-based tuning functions showed in particular that training on low-signal visual stimuli increases the amplitude while decreasing the width of these pattern-based tuning functions in the higher dorsal and ventral visual regions.

The increase in amplitude of the pattern-based tuning functions following training indicates a higher discriminability of multivoxel representations of stimuli which may relate to enhanced neural responses to preferred stimulus categories at the level of neural populations across voxels. This finding is supported by further voxel-based tuning analysis which, while looking at the pooled behaviour averaged over the significant voxels within each visual region, finds similar increases in amplitude of the BOLD response to preferred stimuli after training.

The decrease in the width of the pattern-based tuning functions after training indicates fewer mis-predictions being made when classifying the stimuli. Further analysis of the mis-predictions made show that this reduction manifests particularly at larger offsets from the preferred stimuli. This reduction in mis-predictions suggests learning decreases neural responses to non-preferred stimuli, which is further supported by the decrease in tuning width shown by the voxel-based tuning functions which directly model the BOLD response.

Our findings suggest that learning of visual patterns in the human visual cortex is implemented by enhancing the response to preferred stimulus categories, while reducing the response to non-preferred stimulus categories.

The findings presented here, and covered in Appendix 1 by Zhang et al. (2010), advance understanding of the mechanisms which mediate learning in two main respects. First, while previous imaging studies (Kourtzi et al., 2005; Sigman et al., 2005; Op de Beeck et al., 2006; Mukai et al., 2007; Yotsumoto et al., 2008) have found evidence for changes in overall fMRI responsiveness to trained stimuli, the results discussed here provide evidence for learning dependent changes related to neural selectivity. Second, the results presented here demonstrate learning-dependent changes in fMRI selectivity in dorsal visual areas, which is consistent with the findings of previous work by members of the team which showed that these dorsal visual regions are involved in the integration of local orientation signals into global forms (Ostwald et al., 2008).

The fMRI analyses performed as a part of the study discussed in this chapter were facilitated by the use of the Matlab MVPA Toolbox described in Chapter 2 of this thesis. The toolbox provided analysis scripts for calculating the percentage signal change, performing 6-way pattern classification and performing the shuffling variant of the classification out of the box. The detailed results provided by these analyses, particularly the classification analysis, provided all the details necessary for the development and application of the pattern-based tuning analysis, and the subsequent analysis of mis-predicted stimuli.

In addition to the pre-existing analyses implemented as part of the toolbox, the components provided by the toolbox for feature selection and time course pre-processing served to form the basis of the voxel-based tuning analysis which was implemented based on the description provided by Serences et al (2009).

This use of standardised analyses and analysis components, which have been employed in other published studies (Ban et al., 2012; Dövencioğlu et al., 2013) served to save in development time, allowing for a greater focus on development of novel analysis methods, while also providing the benefit of increased reliability due to repeated use and testing.

As a result the author of this thesis developed a novel method for the analysis of the tuning of the multi-variate encoding of visual stimuli based on the voxel tuning functions described by Serences et al. (2009), a method which allows for the investigation of the underlying mechanisms involved in altering tuning while retaining the increased sensitivity provided by multi-voxel pattern analysis.

In summary, the study discussed in this chapter serves to increase understanding of the role of tuning in learning, provides a tool for the investigation of tuning at the level of multivariate patterns encoded by pools of voxels and gives an example of the utility of the MVPA toolbox presented in this thesis and its scope for use as a basis of further development.

CHAPTER 5 - CONCLUSION

The toolbox, the development of which was described in this thesis, was created to meet the need for a standardised tool for Multivariate Pattern Analysis (MVPA) within the University of Birmingham Cognitive Neuroimaging Laboratory.

Prior to the development of this toolbox MVPA was performed using Matlab (The MathWorks, Inc., Natick, Massachusetts, United States) scripts developed on an ad hoc basis, with each project having its own custom variants of the scripts. While this approach was successful in producing a number of high quality MVPA imaging studies (Li et al., 2007; Preston et al., 2008; Ostwald et al., 2008), it also highlighted a number of disadvantages.

As these analysis scripts would be created with only a single project in mind, they would frequently be hard coded to refer to files belonging to a specific project, or a specific location on a machine; a practice which limited their re-usability for other projects and portability to other computers. When attempting to re-use one of these scripts as the basis for an analysis on another project, it would be necessary to re-write large portions from the beginning in order to ensure that the correct files were used and that any customisations specific to the previous project were removed.

The need for frequent re-implementation and modification by users whose primary focus was neuro-science rather than software development meant that the resultant tools, while functional and capable of performing the analyses required, were lacking in terms of efficency and optimisation.

The author of this thesis was tasked with the development of a toolbox of standardised scripts for performing multivariate pattern analysis of functional magnetic resonance imaging data, with a focus on re-usability, repeatability and efficient execution.

Identification of requirements and techniques to include in the toolbox was achieved through inspection of existing analysis scripts and discussion with members of the Cognitive Neuroimaging Laboratory. Following this, development of the toolbox was conducted using an iterative process of development, testing and optimisation to ensure reliability of results and efficient execution. To further enable rapid analysis, options for parallel processing were explored and implemented. Having been implemented and tested, this toolbox was employed to perform MVPA in a number of studies in the lab, such as those by Dövencioğlu et al. (2013), Dövencioğlu et al. (2012), Dövencioğlu (2013), Kuai et al., (2013) and Patten (2013).

The toolbox was developed in a modular fashion with a view to facilitating further expansion to include new methods and use of the toolbox as a basis for development of novel methods.

These properties were explored in Chapter 3 in which modifications to the standard leave-one-run-out cross-validation method were designed to allow for the application of leave-*C*-out cross-validation to fMRI data, a method derived from the classical leave-N-out cross-validation method common in the field of statistics.

The ability of the toolbox to be used as the basis for the development of novel methods was further explored in Chapter 4, in which a novel method for the investigation of tuning of populations of neurons at the voxel level was developed based on previous work by Serences et al., (2009). This method was then applied to the investigation of the effects of learning by Zhang et al., (2010).

The toolbox provides a robust, reliable and efficient means for performing a standard set of MVPA analyses and a basis for further development of novel methods. This provides the basis for a toolbox suitable for use by the neuro-science community at large. With some further development and refinement this toolbox could become a publishable tool suitable for easy application by novice users.

Beyond the scope of the thesis is the possibility of further expansion of the toolbox to include novel developments in the field of computer science, such as recent advancements in parallel processing using graphics cards (GPGPU) to speed up existing analyses and make previously computationally prohibitive analyses feasible.

Also the modular nature of the toolbox provides scope for more domain specific developments such as novel methods for the analysis of fMRI data, as shown by the modifications made to perform leave-*C*-out cross-validation, and the methods developed on top of the toolbox for pattern-based and voxel-based tuning function analysis.

In conclusion, as intended the toolbox developed for the University of Birmingham Cognitive Neuroimaging Laboratory addresses the need for a standardised set of analysis tools for common methods of MVPA. It provides these tools in a form suitable for re-use across multiple studies and with the capability to reproduce results exactly, provided the same configuration and version of the toolbox are employed. Through careful optimisation and the potential for parallel processing it enables the rapid execution of these multivariate pattern analyses, allowing for researchers to experiment with various analysis methods in a reasonable timescale. Finally the toolbox serves to provide a basis for further development of novel methods.

REFERENCES

Ban, H., Preston, T. J., Meeson, A., & Welchman, A. E. (2012). The integration of motion and disparity cues to depth in dorsal visual cortex. *Nature neuroscience*, *15*(4), 636-643.

Calhoun, V. D., Adali, T., Hansen, L. K., Larsen, J., & Pekar, J. J. (2003). ICA of functional MRI data: an overview.

Chang, C. C., & Lin, C. J. (2011). LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, *2*(3), 27.

De Martino, F., Valente, G., Staeren, N., Ashburner, J., Goebel, R., & Formisano, E. (2008). Combining multivariate voxel selection and support vector machines for mapping and classification of fMRI spatial patterns. *Neuroimage*, *43*(1), 44-58.

De Martino, F., De Borst, A. W., Valente, G., Goebel, R., & Formisano, E. (2011). Predicting EEG single trial responses with simultaneous fMRI and relevance vector machine regression. *Neuroimage*, *56*(2), 826-836.

Detre G, Polyn SM, Moore C, Natu V, Singer B, Cohen J, Haxby JV, Norman KA. The multi-voxel pattern analysis (MVPA) toolbox. Poster presented at the Annual Meeting of the Organization for Human Brain Mapping; Florence, Italy. 2006.

Dovencioglu, D. N., Ban, H., Schofield, A. J., & Welchman, A. E. (2012). The integration of disparity and shading cues to 3D shape in dorsal visual cortex. *Journal of Vision*, *12*(9), 1192-1192.

Dövencioğlu, D. N., Welchman, A. E., & Schofield, A. J. (2013). Perceptual learning of second order cues for layer decomposition. *Vision research*, *77*, 1-9.

Dövencioğlu, D. N. (2013). *Estimation of 3D shape from shading and binocular disparity* (Doctoral dissertation, University of Birmingham).

Glass, L. (1969). Moire effect from random dots. *Nature*, *223*(5206), 578-580.

Hanke, M., Halchenko, Y. O., Sederberg, P. B., Hanson, S. J., Haxby, J. V., & Pollmann, S. (2009). PyMVPA: A python toolbox for multivariate pattern analysis of fMRI data. *Neuroinformatics*, *7*(1), 37-53.

Hansen, L. K., Larsen, J., Nielsen, F. Å., Strother, S. C., Rostrup, E., Savoy, R., ... & Paulson, O. B. (1999). Generalizable patterns in neuroimaging: How many principal components?. *NeuroImage*, *9*(5), 534-544.

Hanson, S. J., Matsuka, T., & Haxby, J. V. (2004). Combinatorial codes in ventral temporal lobe for object recognition: Haxby (2001) revisited: is there a "face" area?. *Neuroimage*, *23*(1), 156-166.

Haxby, J. V., Gobbini, M. I., Furey, M. L., Ishai, A., Schouten, J. L., & Pietrini, P. (2001). Distributed and overlapping representations of faces and objects in ventral temporal cortex. *Science*, *293*(5539), 2425-2430.

Hinrichs, H., Scholz, M., Tempelmann, C., Woldorff, M. G., Dale, A. M., & Heinze, H. J. (2000). Deconvolution of Event-Related fMRI Responses in Fast-Rate Experimental Designs: Tracking Amplitude Variations. *Journal of Cognitive Neuroscience*, *12*(2), 76-89.

Huettel, S.A., Song A.W., McCarthy G. (2009). Functional Magnetic Resonance Imaging. 2nd ed. Sunderland, MA.: Sinauer Associates, Inc..

Hughes, G. (1968). On the mean accuracy of statistical pattern recognizers. *Information Theory, IEEE Transactions on*, *14*(1), 55-63.

Joachims, T. (1999). Making large scale SVM learning practical.

Kamitani, Y., & Tong, F. (2005). Decoding the visual and subjective contents of the human brain. *Nature Neuroscience*, *8*(5), 679.

Kohavi, R. (1995, August). A study of cross-validation and bootstrap for accuracy estimation and model selection. In *IJCAI* (Vol. 14, No. 2, pp. 1137-1145).

Kourtzi, Z., & Kanwisher, N. (2000). Activation in human MT/MST by static images with implied motion. *Journal of cognitive neuroscience*, *12*(1), 48-55.

Kourtzi, Z., & Kanwisher, N. (2001). Representation of perceived object shape by the human lateral occipital complex. *Science*, *293*(5534), 1506-1509.

Kourtzi, Z., Betts, L. R., Sarkheil, P., & Welchman, A. E. (2005). Distributed neural plasticity for shape learning in the human visual cortex. *PLoS biology*, *3*(7), e204.

Kourtzi, Z., & DiCarlo, J. J. (2006). Learning and neural plasticity in visual object recognition. *Current opinion in neurobiology*, *16*(2), 152-158.

Kriegeskorte, N., Goebel, R., & Bandettini, P. (2006). Information-based functional brain mapping. *Proceedings of the National Academy of Sciences of the United States of America*, *103*(10), 3863-3868.

Kriegeskorte, N., & Bandettini, P. (2007). Combining the tools: activation-and information-based fMRI analysis. *Neuroimage*, *38*(4), 666-668.

Kriegeskorte, N., Simmons, W. K., Bellgowan, P. S., & Baker, C. I. (2009). Circular analysis in systems neuroscience: the dangers of double dipping. *Nature neuroscience*, *12*(5), 535-540.

Kruggel, F., & von Cramon, D. Y. (1999). Modeling the hemodynamic response in single-trial functional MRI experiments. *Magnetic Resonance in Medicine*, *42*(4), 787-797.

Kuai, S. G., & Kourtzi, Z. (2011). Learning optimizes visual shape templates in the human brain. *Journal of Vision*, *11*(11), 1010-1010.

Kuai, S. G., Levi, D., & Kourtzi, Z. (2013). Learning optimizes decision templates in the human visual cortex. *Current Biology*, *23*(18), 1799-1804.

Li, S., Ostwald, D., Giese, M., & Kourtzi, Z. (2007). Flexible coding for categorical decisions in the human brain. *The Journal of Neuroscience*, *27*(45), 12321-12330.

Li, S., Mayhew, S. D., & Kourtzi, Z. (2009). Learning shapes the representation of behavioral choice in the human brain. *Neuron*, *62*(3), 441-452.

MacKay, D. J. (1992). Bayesian interpolation. *Neural computation*, *4*(3), 415-447.

Miezin, F. M., Maccotta, L., Ollinger, J. M., Petersen, S. E., & Buckner, R. L. (2000). Characterizing the hemodynamic response: effects of presentation rate, sampling procedure, and the possibility of ordering brain activity based on relative timing. *Neuroimage*, *11*(6), 735-759.

Misaki, M., Kim, Y., Bandettini, P. A., & Kriegeskorte, N. (2010). Comparison of multivariate classifiers and response normalizations for pattern-information fMRI. *Neuroimage*, *53*(1), 103-118.

Mitchell, T. M., Hutchinson, R., Niculescu, R. S., Pereira, F., Wang, X., Just, M., & Newman, S. (2004). Learning to decode cognitive states from brain images. *Machine Learning*, *57*(1-2), 145-175.

Moonen, C. T. W., & Bandettini, P. A. Functional MRI, 1999. *Spingler, Berlin*.

Mukai, I., Kim, D., Fukunaga, M., Japee, S., Marrett, S., & Ungerleider, L. G. (2007). Activations in visual and attention-related areas predict and correlate with the degree of perceptual learning. *The Journal of Neuroscience*, *27*(42), 11401-11411.

Neal, R. M. (1996). Bayesian Learning for Neural Networks Springer. *New York*.

Norman, K. A., Polyn, S. M., Detre, G. J., & Haxby, J. V. (2006). Beyond mind-reading: multi-voxel pattern analysis of fMRI data. *Trends in cognitive sciences*, *10*(9), 424-430.

Oommen, T., Misra, D., Twarakavi, N. K., Prakash, A., Sahoo, B., & Bandopadhyay, S. (2008). An objective analysis of support vector machine based classification for remote sensing. *Mathematical geosciences*, *40*(4), 409-424.

Oosterhof, N. N., and Connolly, A. C. (2014). CoSMoMVPA – CoSMo Multivariate Pattern Analysis toolbox 0.1alpha documentation. Retrieved November 30, 2014, from http://www.cosmomvpa.org.

Op de Beeck, H. P., Baker, C. I., DiCarlo, J. J., & Kanwisher, N. G. (2006). Discrimination training alters object representations in human extrastriate cortex. *The Journal of Neuroscience*, *26*(50), 13025-13036.

Ostwald, D., Lam, J. M., Li, S., & Kourtzi, Z. (2008). Neural coding of global form in the human visual cortex. *Journal of Neurophysiology*, *99*(5), 2456-2469.

O'Toole, A. J., Jiang, F., Abdi, H., Pénard, N., Dunlop, J. P., & Parent, M. A. (2007). Theoretical, statistical, and practical perspectives on pattern-based classification approaches to the analysis of functional neuroimaging data. *Journal of cognitive neuroscience*, *19*(11), 1735-1752.

Patten, M. L. (2013). *The neural basis of binocular depth perception* (Doctoral dissertation, University of Birmingham).

Pereira, F. (2007). *Beyond brain blobs: machine learning classifiers as instruments for analyzing functional magnetic resonance imaging data*. (Doctoral dissertation, Carnegie Mellon University)

Pereira, F., Mitchell, T., & Botvinick, M. (2009). Machine learning classifiers and fMRI: a tutorial overview. *Neuroimage*, *45*(1), S199-S209.

Polyn, S. M., Natu, V. S., Cohen, J. D., & Norman, K. A. (2005). Category-specific cortical activity precedes retrieval during memory search. *Science*, *310*(5756), 1963-1966.

Preston, T. J., Li, S., Kourtzi, Z., & Welchman, A. E. (2008). Multivoxel pattern selectivity for perceptually relevant binocular disparities in the human brain. *The Journal of Neuroscience*, *28*(44), 11315-11327.

Raichle, M. E. (1998). Behind the scenes of functional brain imaging: a historical and physiological perspective. *Proceedings of the National Academy of Sciences*, *95*(3), 765-772.

Ryali, S., Supekar, K., Abrams, D. A., & Menon, V. (2010). Sparse logistic regression for whole-brain classification of fMRI data. *NeuroImage*, *51*(2), 752-764.

Savoy, R. L. (2001). History and future directions of human brain mapping and functional neuroimaging. *Acta Psychologica*, *107*(1), 9-42.

Sereno, M. I., Dale, A. M., Reppas, J. B., Kwong, K. K., Belliveau, J. W., Brady, T. J., ... & Tootell, R. B. (1995). Borders of multiple visual areas in humans revealed by functional magnetic resonance imaging. *Science*, *268*(5212), 889-893.

Sigman, M., Pan, H., Yang, Y., Stern, E., Silbersweig, D., & Gilbert, C. D. (2005). Top-down reorganization of activity in the visual pathway after learning a shape identification task. *Neuron*, *46*(5), 823-835.

Talairach, J., & Tournoux, P. (1988). Co-planar stereotaxic atlas of the human brain. 3-Dimensional proportional system: an approach to cerebral imaging.

Vapnik, V. N. (1995). The nature of statistical learning theory.

Wandell, B. A., Dumoulin, S. O., & Brewer, A. A. (2007). Visual field maps in human cortex. *Neuron*, *56*(2), 366-383.

Wichmann, F. A., & Hill, N. J. (2001a). The psychometric function: I. Fitting, sampling, and goodness of fit. *Perception & psychophysics*, *63*(8), 1293-1313.

Wichmann, F. A., & Hill, N. J. (2001b). The psychometric function: II. Bootstrap-based confidence intervals and sampling. *Perception & psychophysics*, *63*(8), 1314-1329.

Worsley, K.J., Friston, K.J. (1995). Analysis of fMRI time-series revisited–again. *Neuroimage*, 2, 173–81.

Yamashita, O., Sato, M. A., Yoshioka, T., Tong, F., & Kamitani, Y. (2008). Sparse estimation automatically selects voxels relevant for the decoding of fMRI activity patterns. *NeuroImage*, *42*(4), 1414-1429.

Yotsumoto, Y., Watanabe, T., & Sasaki, Y. (2008). Different dynamics of performance and brain activation in the time course of perceptual learning. *Neuron*, *57*(6), 827-833.

Zhang, J., Meeson, A., Welchman, A. E., & Kourtzi, Z. (2010). Learning alters the tuning of functional magnetic resonance imaging patterns for visual forms. *The Journal of Neuroscience*, *30*(42), 14127-14133.

APPENDIX 1

Learning alters the tuning of functional magnetic resonance imaging patterns for visual forms.

Zhang, J., Meeson, A., Welchman, A. E., & Kourtzi, Z.

2010

Published in: The Journal of Neuroscience, Issue 30, Vol. 42, pp 14127-14133.

Behavioral/Systems/Cognitive

Learning Alters the Tuning of Functional Magnetic Resonance Imaging Patterns for Visual Forms

Jiaxiang Zhang,[1,2] Alan Meeson,[1] Andrew E. Welchman,[1] and Zoe Kourtzi[1]

[1]School of Psychology, University of Birmingham, Edgbaston, Birmingham B15 2TT, United Kingdom, and [2]Medical Research Council, Cognition and Brain Sciences Unit, Cambridge CB2 7EF, United Kingdom

Learning is thought to facilitate the recognition of objects by optimizing the tuning of visual neurons to behaviorally relevant features. However, the learning mechanisms that shape neural selectivity for visual forms in the human brain remain essentially unknown. Here, we combine behavioral and functional magnetic resonance imaging (fMRI) measurements to test the mechanisms that mediate enhanced behavioral sensitivity in the discrimination of visual forms after training. In particular, we used high-resolution fMRI and multivoxel pattern classification methods to investigate fine learning-dependent changes in neural preference for global forms. We measured the observers' choices when discriminating between concentric and radial patterns presented in noise before and after training. Similarly, we measured the choices of a pattern classifier when predicting each stimulus from fMRI activity. Comparing the performance of human observers and classifiers demonstrated that learning alters the observers' sensitivity to visual forms and the tuning of fMRI activation patterns in visual areas selective for task-relevant features. In particular, training on low-signal stimuli enhanced the amplitude but reduced the width of pattern-based tuning functions in higher dorsal and ventral visual areas. Thus, our findings suggest that learning of visual patterns is implemented by enhancing the response to the preferred stimulus category and reducing the response to nonpreferred stimuli in higher extrastriate visual cortex.

Introduction

Detecting and identifying meaningful objects in clutter is a critical skill for interactions and survival in complex environments. Although these processes appear fast and effortless, the computational challenges of visual recognition are far from trivial. For example, the recognition of coherent objects entails segmentation of relevant features from clutter and discrimination of highly similar features belonging to different objects. Learning has been suggested to facilitate these processes by tuning neural selectivity to behaviorally relevant visual features (for review, see Gilbert et al., 2001; Kourtzi and DiCarlo, 2006). However, the learning mechanisms that shape neural selectivity for visual forms in the human brain remain essentially unknown. In particular, learning-dependent changes in neural selectivity (i.e., sharpening of neuronal tuning to a visual stimulus) may result from three different possible mechanisms: enhanced response to the preferred stimulus, decreased response to the nonpreferred stimulus, or a combination of the two. Here, we combine behavioral and fMRI measurements to test which of these neural mechanisms mediate learning of forms in the human visual cortex. We exploit the sensitivity of high-resolution fMRI and multivoxel pattern classification analysis (MVPA) methods to investigate fine learning-dependent changes in neural preference at the level of large neural populations in the human visual cortex as revealed by fMRI.

In particular, we used a morphing stimulus space that is generated by varying the spiral angle between radial and concentric patterns resulting in stimuli that vary in their similarity (see Fig. 1*A*). Observers were presented with stimuli at different amounts of background noise (i.e., high-signal vs low-signal stimuli) and judged whether they resembled a concentric or radial visual pattern. We measured the observers' choices (i.e., proportion concentric) with high-signal stimuli and compared them with those with low-signal stimuli before and after training. Similarly, we measured the choices of a pattern classifier, that is, the proportion of patterns on which the classifier predicted each stimulus from fMRI activity pooled across voxels. We investigated the link between behavioral and fMRI learning changes by comparing psychometric functions and fMRI pattern-based tuning functions before and after training.

Our findings demonstrate that training alters the observers' sensitivity to visual forms and fMRI selectivity in higher dorsal and ventral visual areas. Specifically, training on low-signal stimuli enhanced the amplitude but decreased the width of pattern-based tuning in higher dorsal and ventral visual areas. These findings suggest that learning enhances behavioral sensitivity to visual forms and fMRI sensitivity in higher visual areas by enhancing neural responses to behaviorally relevant features, whereas decreasing responses to nonpreferred features.

Received April 29, 2010; revised Aug. 17, 2010; accepted Aug. 19, 2010.

The research leading to these results has received funding from Biotechnology and Biological Sciences Research Council Grants BB/D52199X/1 and BB/E027436/1 and European Community Seventh Framework Programme Grant FP7/2007-2013 under Grant Agreement 214728. Thanks to M. Dexter for help with the development of analysis tools.

Correspondence should be addressed to Zoe Kourtzi, University of Birmingham, School of Psychology, Edgbaston, Birmingham B15 2TT, UK. E-mail: z.kourtzi@bham.ac.uk.

DOI:10.1523/JNEUROSCI.2204-10.2010

Materials and Methods

Participants

Ten observers participated in the study (two males, eight females; age range, 19–37 years). All observers had normal or corrected-to-normal vision, gave written informed consent, and were paid for their participation. The data from one observer were excluded as a result of low behavioral performance after training and from a second observer as a result of poor fMRI signals even for high-signal stimuli. The study was approved by the local ethics committee.

Stimuli

Stimuli were Glass patterns (Glass, 1969) generated using previously described methods (Li et al., 2009) ensuring that coherent form patterns are reliably perceived for the stimulus generation parameters we used. In particular, stimuli were defined by white dot pairs (dipoles) displayed within a square aperture (7.7° × 7.7°) on a black background (100% contrast). The dot density was 3%, and the Glass shift (i.e., the distance between two dots in a dipole) was 16.2 arc min. The size of each dot was 2.3° × 2.3° arc min^2. For each dot dipole, the spiral angle was defined as the angle between the dot dipole orientation and the radius from the center of the dipole to the center of the stimulus aperture. Each stimulus comprised dot dipoles that were aligned according to the specified spiral angle (signal dipoles) for a given stimulus and noise dipoles for which the spiral angle was randomly selected. The proportion of signal dipoles defined the stimulus signal level. We generated concentric and radial Glass patterns by placing dipoles tangentially (concentric stimuli) or orthogonally (radial stimuli) to the circumference of a circle centered on the fixation dot. Further, we generated intermediate patterns between these two Glass pattern types by parametrically varying the spiral angle of the pattern from 0° (radial pattern) to 90° (concentric pattern) (Fig. 1*A*). Half of the observers were presented with clockwise patterns (0° to 90° spiral angle) and half with counterclockwise patterns (0° to −90° spiral angle). A new pattern was generated for each stimulus presented in a trial, resulting in stimuli that were locally jittered in their position.

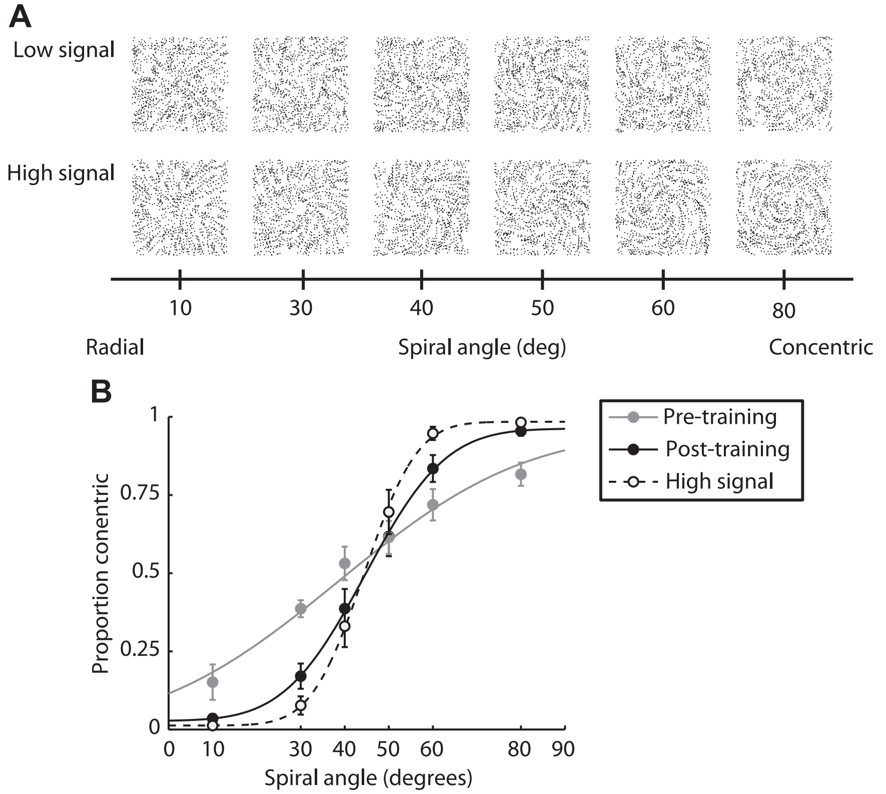


Figure 1. Stimuli and behavioral performance. ***A***, Low- and high-signal Glass pattern stimuli. ***B***, Behavioral performance (average data across observers) across spiral angle during scanning. The curves indicate the best fit of the cumulative Gaussian function. Error bars indicate the SEM.

Design

All observers participated in three psychophysical training sessions and three fMRI sessions. In the psychophysical sessions, observers were presented with stimuli at 45% signal level. For the pretraining and posttraining fMRI sessions, observers were presented with stimuli at 45% signal level (low-signal stimulus sessions), whereas for the third fMRI session, observers were presented with stimuli at 80% signal level (high-signal stimulus session). The high-signal stimulus session followed the low-signal stimulus sessions to avoid priming the observers with highly visible versions of the stimuli before training.

Psychophysical training

Observers were first familiarized with the task during a short practice session (20 trials). Then observers performed one pretest session (one run without feedback), followed by three training sessions that were conducted on different days. Each session comprised five training runs with audio feedback on error trials and was followed by one test run without feedback. Each psychophysical run comprised 160 trials (16 trials per stimulus condition). Stimuli were presented at 10 possible spiral angles (5°, 15°, 25°, 35°, 42°, 48°, 55°, 65°, 75°, and 85°). Each trial lasted 1.5 s, and the stimulus was presented for 200 ms. Observers were instructed to indicate whether each presented stimulus was similar to a radial Glass pattern (0° spiral angle) or a concentric Glass pattern (90° spiral angle), by pressing one of two (left or right) buttons on a mouse. The buttons for different stimulus categories were counterbalanced across observers.

fMRI sessions

Observers participated in three scanning sessions: one high-signal stimulus session (i.e., observers were presented with stimuli at 80% signal level), one low-signal stimulus session (i.e., observers were presented with stimuli at 45% signal level) before training (after the psychophysical pretest), and one low-signal stimulus session after training (after the last psychophysical session). During scanning, observers performed the same categorization task as during the psychophysical sessions.

Each scanning session comprised eight experimental runs, each of which lasted 364 s. Each run comprised eighteen 18-s-long stimulus blocks. A 10 s fixation block (i.e., only a fixation square was presented on the screen) was presented after every six stimulus blocks, as well as in the beginning and the end of each run. Each stimulus block was repeated three times in each run. The order of the blocks was randomized within each run, and each block was presented only once between two fixation blocks. Each stimulus block comprised 12 trials, including target and distractor stimuli. That is, 10 trials contained target stimuli presented at one of six conditions (i.e., spiral angles), whereas two trials contained distractor stimuli from another condition. Possible combinations of spiral angles (target/distractor stimuli) presented in a block were 10°/50°, 30°/60°, 40°/80°, 50°/10°, 60°/30°, and 80°/40°. The presentation order of target and distractor stimuli within each block was randomized; one of the distractors was presented in the first six trials and the other in the last six trials. Each trial lasted 1.5 s. Stimuli were presented for 200 ms each and separated by a 1300 ms interstimulus interval, during which observers made their response to the stimulus by pressing one of two keys. The color of the fixation square, which was presented during fixation blocks and throughout each trial, served as a cue for the motor response. If the cue was red, observers used the same key–category matching as during the psychophysical training sessions (e.g., left key for concentric pat-

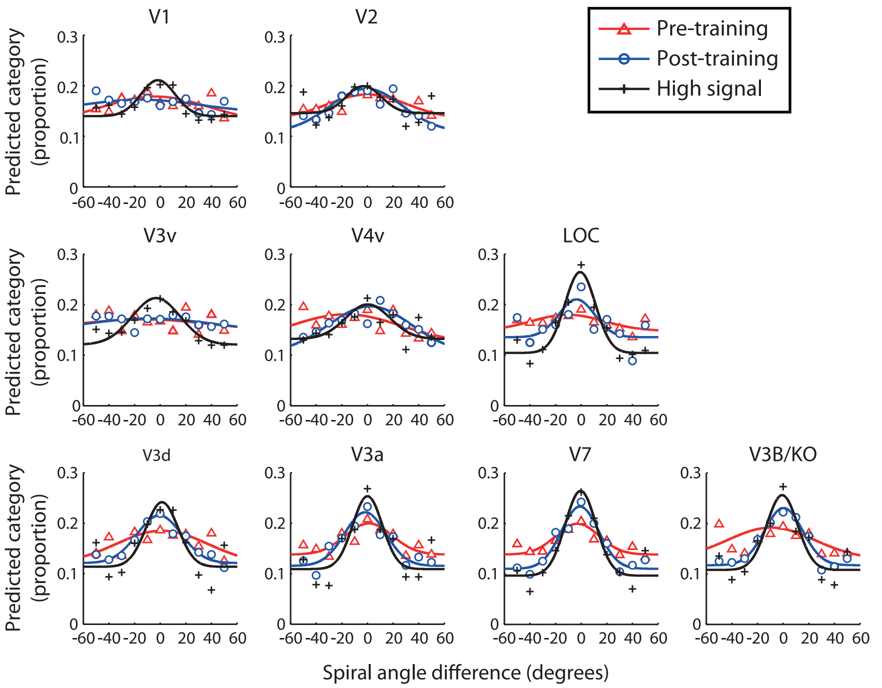


Figure 2. fMRI pattern-based tuning functions. The proportion of predictions made to each stimulus condition in terms of the difference in spiral angle between the viewed stimulus and the prediction. Symbols indicate average data across observers; solid lines indicate the best fit of a Gaussian to the data from 1000 bootstrap samples.

terns), whereas if the cue was green, observers switched finger–key matching (e.g., left key for radial patterns). The color of the fixation square changed after every six stimulus blocks (i.e., before each fixation block) and was counterbalanced across runs.

fMRI data acquisition

The experiments were conducted at the Birmingham University Imaging Centre using a 3 T Philips Achieva MRI scanner. T2*-weighted functional and T1-weighted anatomical (1 × 1 × 1 mm resolution) data were collected with an eight-channel head coil. Echo planar imaging data (gradient echo-pulse sequences) were acquired from 28 slices (repetition time, 2000 ms; echo time, 34 ms; 1.5 × 1.5 × 2 mm resolution). Slices were oriented near coronal covering the entire occipital and posterior temporal cortex.

fMRI data analysis

Data preprocessing. MRI data were processed using Brain Voyager QX (Brain Innovation B.V.). T1-weighted anatomical data were used for coregistration, three-dimensional cortex reconstruction, inflation, and flattening. Preprocessing of the functional data involved slice-scan time correction, three-dimensional head movement correction, temporal high-pass filtering (three cycles), and removal of linear trends. No spatial smoothing was performed on the functional data used for the multivariate analysis. The functional images were aligned to anatomical data under careful visual inspection, and the complete data were transformed into Talairach space (voxel size of 1 × 1 × 1 mm, nearest-neighbor interpolation). Transforming the data into Talairach space ensured that the coordinates of the selected regions of interest (ROIs) for each individual subject were comparable with previous studies. When aligning the functional data to the anatomical scans, we used a nearest-neighbor interpolation method for resampling the data at high resolution (1 × 1 × 1 mm) and included only the unique voxels for the pattern classification analysis. For each participant, the functional imaging data between sessions were coaligned, registering all volumes of each observer to the first functional volume. This procedure ensured a cautious registration across sessions.

Mapping regions of interest. For each individual observer, we identified retinotopic visual areas (V1, V2, V3d, V3a, V7, V3v, and V4v) based on standard mapping procedures (Engel et al., 1994; Sereno et al., 1995; DeYoe et al., 1996). We also identified V3B/KO (kinetic occipital area) and the lateral occipital complex (LOC) in two independent scans. Area V3B/KO was defined as the set of contiguous voxels anterior to V3a that showed significantly stronger activation ($p < 0.005$) for kinetic boundaries than transparent motion (Dupont et al., 1997). LOC was defined as set of contiguous voxels in the ventral occipitotemporal cortex that showed significantly stronger activation ($p < 0.005$) for intact than scrambled images of objects (Kourtzi and Kanwisher, 2000).

Multivoxel pattern analysis

Voxel selection. For each observer and session, we selected voxels in each ROI (retinotopic areas, V3B/KO, and LOC) that showed stronger response to stimulus conditions than fixation ($p < 0.05$). To enable comparisons across ROIs and observers, we selected the average number of voxels across ROIs and observers with the highest difference between stimulus conditions ($p < 0.05$). This procedure resulted in the selection of 250 voxels per ROI, by which point classification accuracies had saturated in all ROIs. If an ROI had fewer than 250 voxels (4.94% of cases across subjects and ROIs), we selected the classification accuracy at the maximum number of voxels in the region. The time course of each voxel was *z*-score normalized for each run and shifted by 4 s to account for the hemodynamic delay. For each pattern, we averaged the fMRI responses across all trials per block, resulting in 24 patterns per session for each condition and ROI.

Pattern classification. We trained linear support vector machine (SVM) classifiers using these patterns per ROI and calculated mean classification accuracies following a leave-one-run-out cross-validation procedure (supplemental Fig. S1*A*, available at www.jneurosci.org as supplemental material). That is, we trained binary classifiers on 21 training patterns and tested their accuracy on three test patterns per condition and ROI using an eightfold cross-validation procedure. For each cross-validation, we selected voxels using only the training dataset, thus ensuring that the classifier was not confounded by using the same data for pattern classification and voxel selection. Also, to ensure that the classifier output did not simply result from univariate differences across conditions, we subtracted the grand mean response across voxels from each voxel.

To determine whether we could predict the viewed stimulus from the six possible stimulus conditions (i.e., spiral angles), we used multiple pairwise (one-against-one) binary classifiers (supplemental Fig. S1*B*, available at www.jneurosci.org as supplemental material) (Kamitani and Tong, 2005; Preston et al., 2008; Serences et al., 2009). In particular, we trained and tested all possible pairwise classifiers (15 comparisons) and collated their results for each test pattern. The predicted stimulus category corresponded to the category that received the fewest "votes against" when collating the results across all pairwise classifications. In the event of a tie, the prediction was randomly assigned to one of the categories. We expressed the accuracy of the six-way classifier as the proportion of test patterns for which it correctly predicted the viewed stimulus.

Pattern-based tuning functions. We examined the pattern of predictions made by the classifier when trained on a particular stimulus condition (i.e., spiral angle). We calculated the proportion of patterns for which the classifier predicted each stimulus condition from fMRI activity associated with each of the six different stimulus conditions. This gave us six sets of predictions for each of the six spiral angles: one prediction indicated the classification accuracy, whereas the rest indicated the classification errors. We plotted these 36 predictions as a function of the difference in spiral angle between the stimulus that evoked the fMRI response and the stimulus predicted by the classifier (supplemental Fig. S1*B*, available at www.jneurosci.org as supplemental material). This allowed us to generate pattern-based tuning functions for spiral angle in each ROI. In particular, each proportion predicted $P(i)$ with stimulus distance (i.e., spiral angle difference) i was calculated as follows:

$$P(i) = \frac{n(i)}{N(i)},$$

where $n(i)$ is the number of patterns predicted to have distance i (from the stimulus condition), and $N(i)$ is the total number of patterns that is possible to be predicted to have distance i. We then fitted the averaged pattern-based tuning functions across observers using a Gaussian function:

$$y = \frac{\alpha}{\sqrt{2\pi s^2}}\exp\left(-\frac{(x-\mu)^2}{2s^2}\right) + \beta,$$

where α is the scaling parameter, μ is the mean, s is the standard deviation, and β is the baseline. Data for ±70° difference in spiral angle were excluded because they were derived from a single prediction (±10° vs ±80°) resulting in outlier values.

Note that, for this approach, the area under each tuning function is not constrained to unity (or a constant number) because the proportion of patterns predicted is a function of the total number of patterns that is possible to be predicted for each condition (i.e., spiral angle difference) rather than the total number of patterns across conditions. To quantify the pattern-based tuning functions, we measured the amplitude (value at $x = 0$) and the width (standard deviation s) of these functions based on 1000 bootstrap samples.

Voxel-based tuning functions. We generated tuning functions based on the fMRI response of individual voxels, as described previously (Serences et al., 2009). For each ROI and scanning session, we selected the same 250 voxels as for the pattern-based tuning functions following the same procedure for voxel selection and using only the training data. The time course of each voxel was *z*-score normalized for each run and shifted by 4 s to account for the hemodynamic delay. We then determined the preference of each voxel by the stimulus condition that evoked the largest mean response when considering data from all experimental runs except one (test run). Then using the data from the test run, we determined the response of each voxel in each condition by the difference (in spiral angle) between the stimulus condition and the preference of the voxel. Akin to the MVPA procedure, we averaged the results from this leave-one-run-out eightfold cross-validation procedure to obtain voxel-based tuning functions for each observer and ROI. We then fitted the average tuning functions across observers with the Gaussian function and estimated the amplitude and the width of these functions from 1000 bootstrap samples.

Results

Behavioral results

We tested the observers' ability to categorize global form patterns as radial or concentric when stimuli were presented at high versus low signal before and after training (Fig. 1*A*). Our results showed that training improved the observers' sensitivity in discriminating visual forms, that is, the 78% threshold performance for low-signal stimuli was reduced after training (Fig. 1*B*). A repeated-measures ANOVA showed higher standard deviation (estimated from cumulative Gaussian fits on individual subject data) for low- than high-signal stimuli before training ($F_{(1,7)} = 14.35$, $p < 0.01$), which decreased significantly after training ($F_{(1,7)} = 10.24$, $p < 0.05$). Similar results were observed during testing in the laboratory (supplemental Fig. S2, available at www.jneurosci.org as supplemental material), indicating that training enhanced the observers' sensitivity to stimulus category.

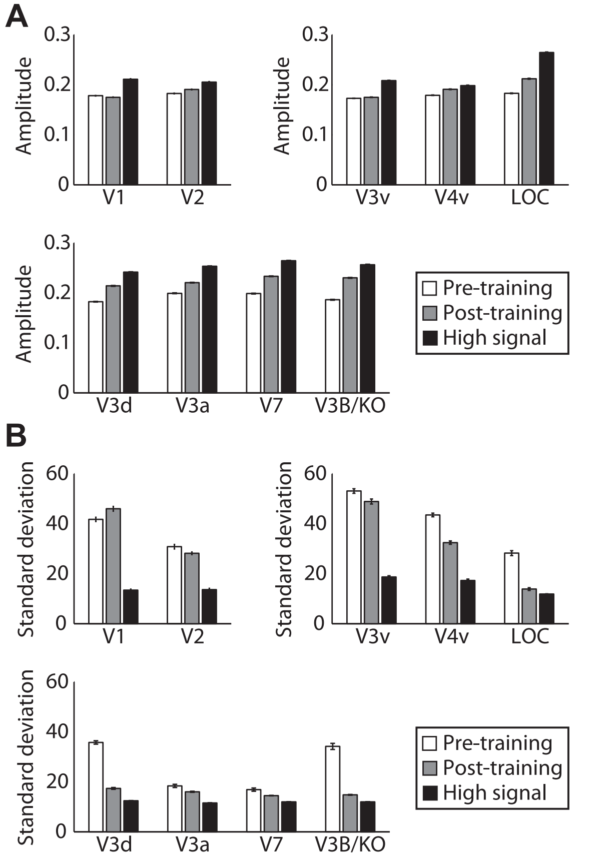


Figure 3. Measures of fMRI pattern tuning. ***A***, ***B***, Amplitude (***A***) and width (***B***) (standard deviation of the Gaussian fit) of the pattern-based tuning functions. Error bars indicate 95% confidence intervals calculated from 1000 bootstrap samples.

fMRI results: pattern-based tuning functions

We used multivoxel pattern classification (Haynes and Rees, 2005; Kamitani and Tong, 2005) to investigate which visual areas encode selective information about shape category (concentric, radial Glass patterns) as determined by the orientation of the local stimulus dipoles (i.e., spiral angle). In particular, we used a six-way linear SVM classifier to discriminate fMRI responses evoked by each stimulus condition from fMRI responses for the other five conditions. We calculated the predictions of the classifier for each stimulus condition using the fMRI activity associated with each of the six different stimulus conditions. From these predictions, we generated pattern-based tuning functions across spiral angle in each ROI (Fig. 2). For high-signal stimuli, we observed a tuned response across visual areas. That is, the classifiers mispredicted stimuli at similar spiral angles more frequently than stimuli at dissimilar spiral angles, suggesting fMRI selective responses for visual forms differing in local orientation signals. To assess the effect of learning on fMRI selectivity for visual forms, we compared the amplitude and the width (i.e., standard deviation of the Gaussian fits) of the pattern-based tuning functions for high-signal stimuli with those for low-signal stimuli before and after training. The amplitude at $x = 0$ indicates accuracy for each predicted stimulus category relating to the classification accuracy of the six-way MVPA (supplemental Fig. S3, available at www.jneurosci.org as supplemental material).

Consistent with the behavioral results, we observed higher amplitude for high- than low-signal stimuli before training in higher visual areas. However, after training, the amplitude for low-signal stimuli increased significantly in higher dorsal and

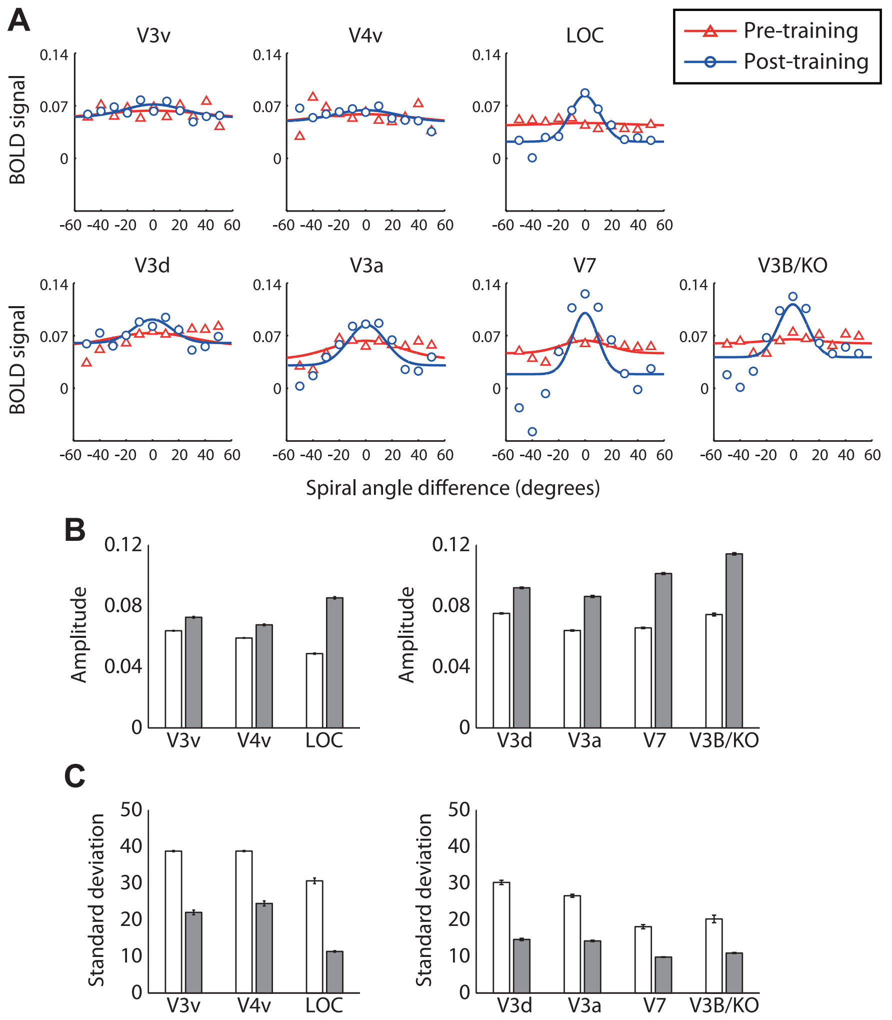


Figure 4. Voxel-based tuning functions. ***A***, BOLD signal (average across voxels) plotted against the difference in spiral angle between each stimulus condition and the preferred condition of each voxel. ***B***, ***C***, Amplitude (***B***) and width (***C***) (standard deviation of the Gaussian fit) of the voxel-based tuning functions. Error bars indicate 95% confidence intervals calculated from 1000 bootstrap samples.

ventral areas (Fig. 3*A*). A repeated-measures ANOVA showed significant differences across sessions in dorsal ($F_{(2,14)} = 15.33$, $p < 0.001$) and ventral ($F_{(2,14)} = 27.26$, $p < 0.001$) but not early (V1, V2) visual areas ($F_{(2,14)} = 2.02$, $p = 0.17$). For dorsal visual areas, amplitude was higher for high- than low-signal stimuli before training ($F_{(1,7)} = 64.49$, $p < 0.001$), but it increased significantly for low-signal stimuli after training ($F_{(1,7)} = 7.84$, $p < 0.05$). Similar results were observed in the LOC rather than earlier ventral areas (V3v, V4), as indicated by a significant interaction between session and ventral regions ($F_{(4,28)} = 3.13$, $p < 0.05$).

Next, we observed narrower tuning of the pattern-based functions for high- than low-signal stimuli before training in higher visual areas (Fig. 3*B*). However, training enhanced the tuning width in both dorsal and ventral areas. A repeated-measures ANOVA showed significant differences in the tuning width across sessions for dorsal ($F_{(2,14)} = 20.09$, $p < 0.001$) and ventral ($F_{(2,14)} = 16.90$, $p < 0.001$) areas but not early (V1, V2) visual areas ($F_{(2,14)} = 1.15$, $p = 0.27$). For both dorsal ($F_{(1,7)} = 64.05$, $p < 0.0.001$) and ventral ($F_{(1,7)} = 31.99$, $p < 0.01$) areas, the tuning width was narrower for high- than low-signal stimuli before training. However, after training, the tuning width for low-signal stimuli decreased significantly in dorsal ($F_{(1,7)} = 6.02$, $p < 0.05$) and ventral ($F_{(1,7)} = 12.27$, $p < 0.01$) areas.

In summary, our results show that training results in behavioral improvement and changes in the tuning of multivoxel patterns in higher dorsal and ventral visual areas. It is interesting to note that, although behavioral and fMRI sensitivity for low-signal stimuli improved with training, it did not reach the levels of sensitivity for high-signal stimuli. In particular, the 78% threshold was lower for high- than low-signal stimuli after training ($F_{(1,7)} = 19.44$, $p < 0.01$), suggesting that observers' sensitivity for discriminating global forms was higher for high- than low-signal stimuli. Similarly, the amplitude of the pattern-based tuning functions was higher ($F_{(1,7)} = 10.87$, $p < 0.05$) whereas the width significantly lower ($F_{(1,7)} = 6.78$, $p < 0.05$) for high- than low-signal stimuli after training. It is possible that more extensive training would result in equivalent performance and fMRI pattern-based tuning for high- and low-signal stimuli.

Control analyses

To discern learning-dependent changes to fMRI signals related to preferred compared with nonpreferred shape categories, we compared the amplitude and width of pattern-based tuning functions. However, these parameters can be coupled; that is, changes in tuning width can relate to changes in amplitude. Therefore, we performed two additional analyses. First, we examined the pattern of mispredictions by excluding prediction data from the preferred shape category (i.e., zero difference in spiral angle). Regression analysis (supplemental Fig. S3, available at www.jneurosci.org as supplemental material) showed steeper slopes (i.e., reduced mispredictions far from the preferred category) across higher dorsal and ventral areas after training, indicating narrower tuning independent of changes in the peak of the pattern-based tuning functions. Second, we tested learning-dependent changes in the preferences of individual voxels for shape categories (Fig. 4). This method (Serences et al., 2009) does not rely on probability distributions and produces voxel-based tuning functions for which amplitude and standard deviation are independent. In agreement with the MVPA results, we observed changes in both the amplitude and standard deviation of tuning in higher dorsal and ventral areas. These findings provide additional evidence for learning-dependent changes not only in the overall responsiveness of large neural populations (at the level of single voxels or patterns) to trained stimuli but also in the neural sensitivity to shape categories.

To avoid confounds and ensure that our data treatment was appropriate, we conducted the following additional analyses. First, we tested for differences in the overall fMRI responsiveness that could result from differences in task difficulty across scanning sessions. Analysis of the percentage signal change from fixation baseline across cortical regions did not show any significant differences across scanning sessions ($F_{(2,14)} = 1.08$, $p = 0.37$), suggesting that differences in the pattern-based tuning functions could not be attributed to differences in the overall fMRI signal. Second, to ensure that our classification approach was not overpowered and did not suffer from any bias, we ran the classification with the data labels shuffled. The results for the classification of 1000 permutation of the six-way classifier (supplemental Table S1, available at www.jneurosci.org as supplemental material) were at chance. Finally, analysis of eye movement data collected during scanning did not show any significant differences between sessions in the eye position or number of saccades (supplemental Fig. S5, available at www.jneurosci.org as supplemental material), suggesting that differences in the pattern-based tuning functions across sessions could not be significantly attributed to eye movement differences.

Discussion

Our findings demonstrate that training alters the observers' sensitivity to visual forms and fMRI selectivity in higher dorsal and ventral visual areas. In particular, we show that training on low-signal stimuli increases the amplitude but reduces the width of pattern-based tuning in higher dorsal and ventral visual areas. Increased amplitude after training indicates higher stimulus discriminability that may relate to enhanced neural responses for the preferred stimulus category at the level of large neural populations across voxels. Reduced tuning width after training indicates fewer classification mispredictions, suggesting that learning decreases neural responses to nonpreferred stimuli. Thus, our findings suggest that learning of visual patterns is implemented in the human visual cortex by enhancing the response to the preferred stimulus category, whereas reducing the response to nonpreferred stimuli.

Our findings advance our understanding of learning brain mechanisms in two main respects. First, we provide evidence for learning-dependent changes related to neural sensitivity rather than simply overall responsiveness (i.e., increased or decreased fMRI responses) to trained stimuli as reported in previous imaging studies (Kourtzi et al., 2005; Sigman et al., 2005; Op de Beeck et al., 2006; Mukai et al., 2007; Yotsumoto et al., 2008). This previous work does not allow us to discern whether learning-dependent changes in fMRI signals relate to changes in the overall magnitude of neural responses or changes in neuronal selectivity of neural populations. Previous work using fMRI adaptation has suggested selectivity changes related to learning (Jiang et al., 2007; Gillebert et al., 2009). Here, we take advantage of the sensitivity of high-resolution fMRI recordings and MVPA methods to discern the mechanisms that mediate learning-dependent changes in visual selectivity. This combination of methods allows us to discern whether learning changes the magnitude of responses to preferred or nonpreferred stimuli by comparing fMRI tuning functions before and after training. Although the low spatial resolution of fMRI compared with neurophysiology does not allow us to investigate learning-dependent changes at the level of single neurons, our methodology provides sensitive tools for testing how learning shapes the fine-tuned representation of visual forms across large neural populations.

Second, our findings provide novel evidence for the role of dorsal areas in learning visual forms. Although the evidence for experience-dependent plasticity in V1 remains controversial (Crist et al., 2001; Schoups et al., 2001; Ghose et al., 2002; Furmanski et al., 2004; Li et al., 2004), recent work indicates that learning shapes visual processing in ventral stream areas. In particular, training is shown to result in greater changes in orientation tuning in V4 than in V1 (Yang and Maunsell, 2004; Raiguel et al., 2006). However, our results demonstrate learning-dependent changes in fMRI selectivity in dorsal visual areas, consistent with our previous work showing that these areas are involved in the integration of local orientation signals into global forms (Ostwald et al., 2008). Furthermore, our results show learning-dependent changes in fMRI selectivity in the LOC, consistent with the role of learning in shaping inferotemporal processing of complex visual features (Sigala and Logothetis, 2002; Freedman et al., 2006), multiple-part configurations (Baker et al., 2002), and objects (Logothetis et al., 1995; Rolls, 1995; Kobatake et al., 1998).

Thus, our findings suggest that learning alters the tuning of activation patterns in regions selective for task-relevant visual features. We have shown previously that the categorization of visual forms is achieved by integrating local visual features and configurations in dorsal visual areas, whereas global form structure in the LOC (Ostwald et al., 2008). Here we propose that learning shapes these processes by decreasing responses to distractor stimuli during the integration of visual forms, whereas enhancing responses for the selective representation of behaviorally relevant stimuli. Recent neurophysiology work suggests that these learning-dependent changes may be mediated by changes in the readout of signals rather than stimulus encoding in visual areas (Law and Gold, 2008). Although, the high-resolution fMRI used in our study limited our recordings to the visual cortex, it is possible that higher frontoparietal circuits may modulate neural plasticity and optimize visual processing through attention-gated learning mechanisms (Hochstein and Ahissar, 2002; Roelfsema and van Ooyen, 2005). Additional work using multimodal imaging (e.g., EEG–fMRI measurements) is necessary for understanding these interactions across cortical circuits and the spatiotemporal dynamics that mediate learning-dependent plasticity in the human brain.

References

Baker CI, Behrmann M, Olson CR (2002) Impact of learning on representation of parts and wholes in monkey inferotemporal cortex. Nat Neurosci 5:1210–1216.

Crist RE, Li W, Gilbert CD (2001) Learning to see: experience and attention in primary visual cortex. Nat Neurosci 4:519–525.

DeYoe EA, Carman GJ, Bandettini P, Glickman S, Wieser J, Cox R, Miller D, Neitz J (1996) Mapping striate and extrastriate visual areas in human cerebral cortex. Proc Natl Acad Sci U S A 93:2382–2386.

Dupont P, De Bruyn B, Vandenberghe R, Rosier AM, Michiels J, Marchal G, Mortelmans L, Orban GA (1997) The kinetic occipital region in human visual cortex. Cereb Cortex 7:283–292.

Engel SA, Rumelhart DE, Wandell BA, Lee AT, Glover GH, Chichilnisky EJ, Shadlen MN (1994) fMRI of human visual cortex. Nature 369:525.

Freedman DJ, Riesenhuber M, Poggio T, Miller EK (2006) Experience-dependent sharpening of visual shape selectivity in inferior temporal cortex. Cereb Cortex 16:1631–1644.

Furmanski CS, Schluppeck D, Engel SA (2004) Learning strengthens the response of primary visual cortex to simple patterns. Curr Biol 14:573–578.

Ghose GM, Yang T, Maunsell JH (2002) Physiological correlates of perceptual learning in monkey V1 and V2. J Neurophysiol 87:1867–1888.

Gilbert CD, Sigman M, Crist RE (2001) The neural basis of perceptual learning. Neuron 31:681–697.

Gillebert CR, Op de Beeck HP, Panis S, Wagemans J (2009) Subordinate categorization enhances the neural selectivity in human object-selective cortex for fine shape differences. J Cogn Neurosci 21:1054–1064.

Glass L (1969) Moire effect from random dots. Nature 223:578–580.

Haynes JD, Rees G (2005) Predicting the orientation of invisible stimuli from activity in human primary visual cortex. Nat Neurosci 8:686–691.

Hochstein S, Ahissar M (2002) View from the top: hierarchies and reverse hierarchies in the visual system. Neuron 36:791–804.

Jiang X, Bradley E, Rini RA, Zeffiro T, Vanmeter J, Riesenhuber M (2007)

Categorization training results in shape- and category-selective human neural plasticity. Neuron 53:891–903.

Kamitani Y, Tong F (2005) Decoding the visual and subjective contents of the human brain. Nat Neurosci 8:679–685.

Kobatake E, Wang G, Tanaka K (1998) Effects of shape-discrimination training on the selectivity of inferotemporal cells in adult monkeys. J Neurophysiol 80:324–330.

Kourtzi Z, DiCarlo JJ (2006) Learning and neural plasticity in visual object recognition. Curr Opin Neurobiol 16:152–158.

Kourtzi Z, Kanwisher N (2000) Cortical regions involved in perceiving object shape. J Neurosci 20:3310–3318.

Kourtzi Z, Betts LR, Sarkheil P, Welchman AE (2005) Distributed neural plasticity for shape learning in the human visual cortex. PLoS Biol 3:e204.

Law CT, Gold JI (2008) Neural correlates of perceptual learning in a sensory-motor, but not a sensory, cortical area. Nat Neurosci 11:505–513.

Li S, Mayhew SD, Kourtzi Z (2009) Learning shapes the representation of behavioral choice in the human brain. Neuron 62:441–452.

Li W, Piëch V, Gilbert CD (2004) Perceptual learning and top-down influences in primary visual cortex. Nat Neurosci 7:651–657.

Logothetis NK, Pauls J, Poggio T (1995) Shape representation in the inferior temporal cortex of monkeys. Curr Biol 5:552–563.

Mukai I, Kim D, Fukunaga M, Japee S, Marrett S, Ungerleider LG (2007) Activations in visual and attention-related areas predict and correlate with the degree of perceptual learning. J Neurosci 27:11401–11411.

Op de Beeck HP, Baker CI, DiCarlo JJ, Kanwisher NG (2006) Discrimination training alters object representations in human extrastriate cortex. J Neurosci 26:13025–13036.

Ostwald D, Lam JM, Li S, Kourtzi Z (2008) Neural coding of global form in the human visual cortex. J Neurophysiol 99:2456–2469.

Preston TJ, Li S, Kourtzi Z, Welchman AE (2008) Multivoxel pattern selectivity for perceptually relevant binocular disparities in the human brain. J Neurosci 28:11315–11327.

Raiguel S, Vogels R, Mysore SG, Orban GA (2006) Learning to see the difference specifically alters the most informative V4 neurons. J Neurosci 26:6589–6602.

Roelfsema PR, van Ooyen A (2005) Attention-gated reinforcement learning of internal representations for classification. Neural Comput 17:2176–2214.

Rolls ET (1995) Learning mechanisms in the temporal lobe visual cortex. Behav Brain Res 66:177–185.

Schoups A, Vogels R, Qian N, Orban G (2001) Practising orientation identification improves orientation coding in V1 neurons. Nature 412: 549–553.

Serences JT, Saproo S, Scolari M, Ho T, Muftuler LT (2009) Estimating the influence of attention on population codes in human visual cortex using voxel-based tuning functions. Neuroimage 44:223–231.

Sereno MI, Dale AM, Reppas JB, Kwong KK, Belliveau JW, Brady TJ, Rosen BR, Tootell RB (1995) Borders of multiple visual areas in humans revealed by functional magnetic resonance imaging. Science 268:889–893.

Sigala N, Logothetis NK (2002) Visual categorization shapes feature selectivity in the primate temporal cortex. Nature 415:318–320.

Sigman M, Pan H, Yang Y, Stern E, Silbersweig D, Gilbert CD (2005) Top-down reorganization of activity in the visual pathway after learning a shape identification task. Neuron 46:823–835.

Yang T, Maunsell JH (2004) The effect of perceptual learning on neuronal responses in monkey visual area V4. J Neurosci 24:1617–1626.

Yotsumoto Y, Watanabe T, Sasaki Y (2008) Different dynamics of performance and brain activation in the time course of perceptual learning. Neuron 57:827–833.

Supplemental Material

Learning alters the tuning of fMRI multi-voxel patterns for visual forms

Jiaxiang Zhang, Alan Meeson, Andrew E Welchman & Zoe Kourtzi

Table S1: MVPA with shuffled data labels

We randomly assigned conditions to the multi-voxel patterns and calculated the prediction accuracy from the 6-way classification for each region of interest and observer. Mean prediction accuracies from 1000 shuffling iterations were not significantly different from chance (0.167) across observers.

	Pre-training		Post-training		High signal	
	t(7)	p	t(7)	p	t(7)	p
V1	-2.06	0.08	-0.16	0.88	0.11	0.92
V2	-1.42	0.20	-1.54	0.17	0.72	0.49
V3d	-0.69	0.51	-1.14	0.29	-0.08	0.94
V3a	0.20	0.85	-0.43	0.68	-0.99	0.36
V3v	-0.85	0.43	-2.27	0.06	-1.02	0.34
V4v	2.00	0.09	-1.45	0.19	0.50	0.63
V7	-2.11	0.07	-0.66	0.53	-0.26	0.80
V3B/KO	-2.25	0.06	-2.04	0.08	-1.42	0.20
LOC	-0.68	0.52	-0.48	0.64	-1.35	0.22

Supplementary Figures

Figure S1: Method for MVPA-based tuning functions

A. Schematic diagram of the MVPA procedure. fMRI data from a given ROI is split into a training and a test set. Voxel selection is then performed based only on the training data set. Multi-voxel patterns are generated by averaging the time course of the selected voxels within each block. Classification results are averaged across leave-one-run-out cross-validations.

B. Illustration of generating pattern-based tuning functions using MVPA. The top panel shows patterns associated with 3 experimental conditions (shown by different shapes and colors) and the prediction for each pattern given by a multiclass classifier. We then calculate the confusion matrix for a given example. Each row of the matrix represents the number of patterns in a given condition, while each column represents the number of predicted patterns for each condition. We then calculate the proportion of patterns that can be predicted for each condition offset (i.e. spiral angle difference) and plot the tuning profile (data and Gaussian fit) as a function of the condition difference.

Figure S2: Behavioral performance in the lab.

Average behavioural data (proportion concentric) across observers from lab testing sessions across stimulus conditions (i.e. spiral angles). Error bars indicate the standard error of the mean. The curves indicate the best fit of the cumulative Gaussian function. Training improved the observers' sensitivity as shown by performance at the 78% threshold. In particular, the 78% threshold after training (55.32° ±2.44°) was lower than before training (85.17°±3.36°). A repeated measures ANOVA showed significant differences in the standard deviation of the cumulative Gaussian fits for individual subjects data across sessions (F(1, 7)= 32.43, p<0.001) indicating that training enhanced the observers' sensitivity to stimulus category.

Figure S3: MVPA accuracy

Prediction accuracy (proportion correct) of the six-way classification (0.167 chance level) per ROI. Error bars denote standard error of the mean across observers. Similar patterns of results were observed for MVPA accuracy and amplitude of pattern-based tuning functions (Figure 3A).

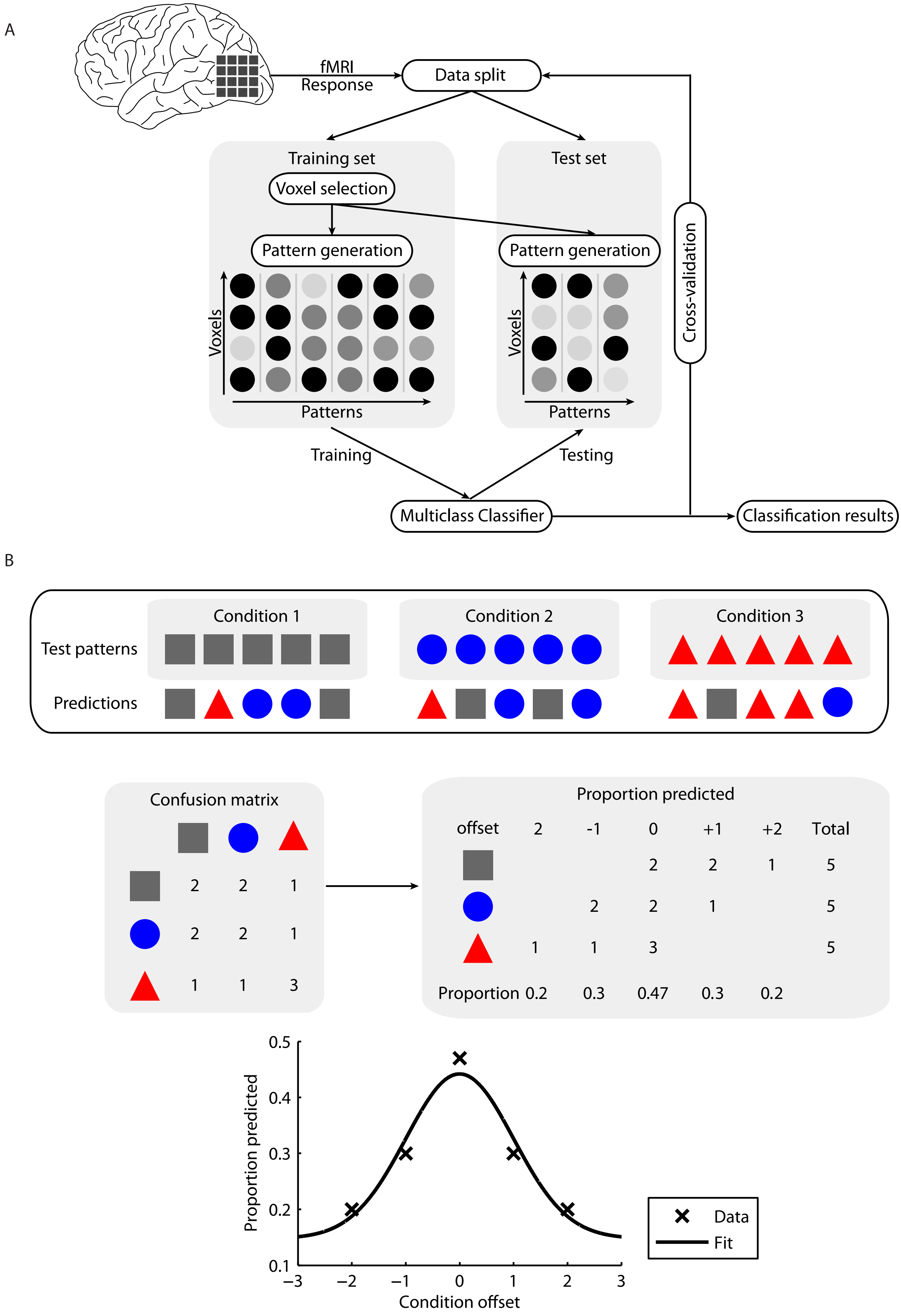
Figure S4: Learning-dependent changes in MVPA mis-predictions

A. Linear regression of the proportion of predictions across absolute difference in spiral angle between the viewed stimulus and the prediction, excluding predictions at zero difference in spiral angle. Predictions are obtained from the six-way classification and averaged across observers and across positive and negative spiral angle distance. Descending slopes indicate greater mis-predictions with an increasing difference in spiral angle. Solid lines indicate the best linear regression fit. B. Slope of the linear regressions. Higher absolute slope values indicate narrower tuning for the mis-prediction distributions. Error bars indicate 95% confidence intervals calculated from 1000 bootstrap samples.

Figure S5: Eye movement recordings

We recorded eye-movements for observers during scanning. Eye movements were recorded using the ASL 6000 Eye-tracker (Applied Science Laboratories, Bedford, MA). Eye tracking data were pre-processed using the Eyenal software (Applied Science Laboratories, Bedford, MA) and analyzed using custom Matlab (Mathworks, MA) software. For each scan session we computed horizontal (X) eye position, vertical (Y) eye position, saccades amplitude and number of saccades per trial per condition. In all sessions the horizontal and vertical eye positions for each stimulus type peaked and were cantered on the fixation at zero degrees. A repeated measurement ANOVA indicated that there was no significant difference between stimulus conditions on mean horizontal eye position (before training: F(5, 10)=0.14, p=0.98; after training: F(5, 10)=0.29, p=0.91; high signal: F(5, 5)=1.00, p=0.50), mean vertical eye position (before training: F(5, 10)=0.81, p=0.57; after training: F(5, 10)=0.65, p=0.67; high signal: F(5, 5)=1.05, p=0.48), mean saccade amplitude (before training: F(5, 10=0.89, p=0.52; after training: F(5, 10)=1.14, p=0.40; high signal: F(5, 5)=0.39, p=0.84) or the number of saccades per trial per condition (before training: F(1.35, 2.69)=1.92, p=0.18; after training: F(5, 10)=1.46, p=0.29; high signal: F(5, 5)=0.46, p=0.79). In addition, no significant differences were observed between sessions for horizontal eye position (F(2, 2)=0.88, p=0.53), vertical eye position (F(2, 2)=1.18, p=0.46), mean saccade amplitude (F(2, 2)=3.24, p=0.24) or number of saccades (F(2, 2)=4.84, p=0.17). These analyses suggest that it is unlikely that our findings were significantly confounded by eye movements.

Figure S1 - Zhang et al.



	Condition 1	Condition 2	Condition 3
Test patterns	■ ■ ■ ■ ■	● ● ● ● ●	▲ ▲ ▲ ▲ ▲
Predictions	■ ▲ ● ● ■	▲ ■ ● ■ ●	▲ ■ ▲ ▲ ●

Confusion matrix

	■	●	▲
■	2	2	1
●	2	2	1
▲	1	1	3

Proportion predicted

offset	2	-1	0	+1	+2	Total
■			2	2	1	5
●		2	2	1		5
▲	1	1	3			5
Proportion	0.2	0.3	0.47	0.3	0.2	

Figure S1 - Zhang et al.

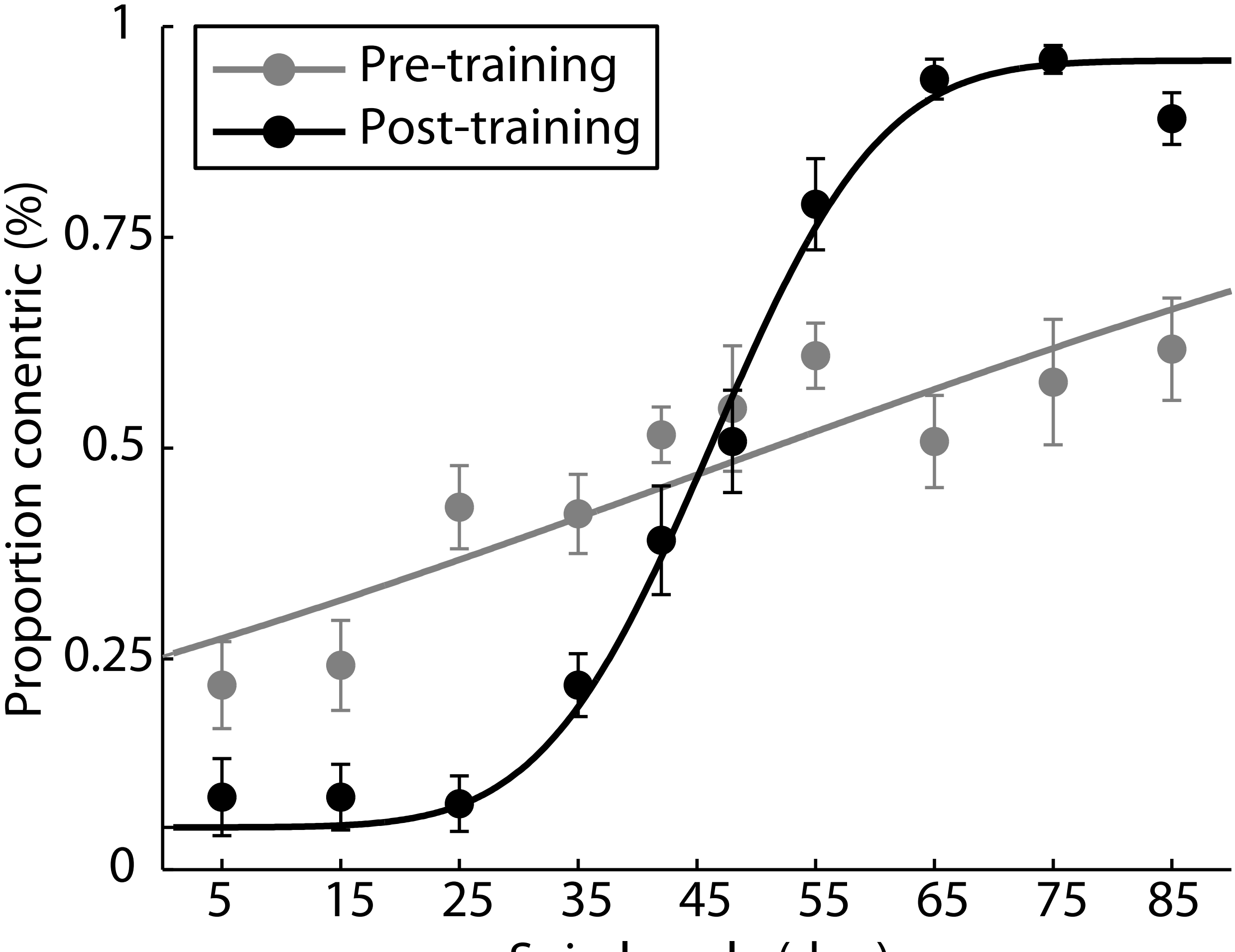


Figure S3 - Zhang et al.

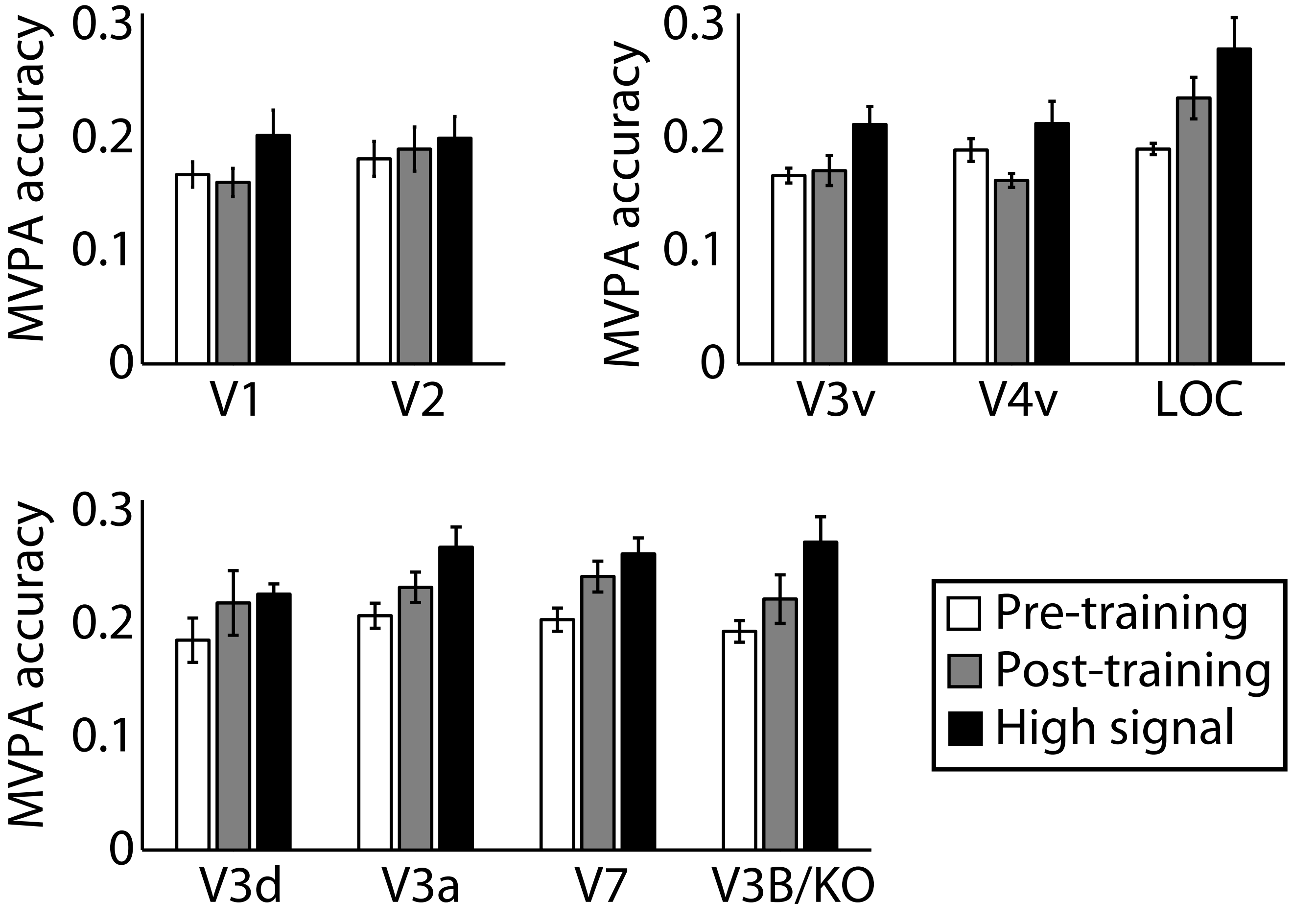


Figure S4 - Zhang et al.

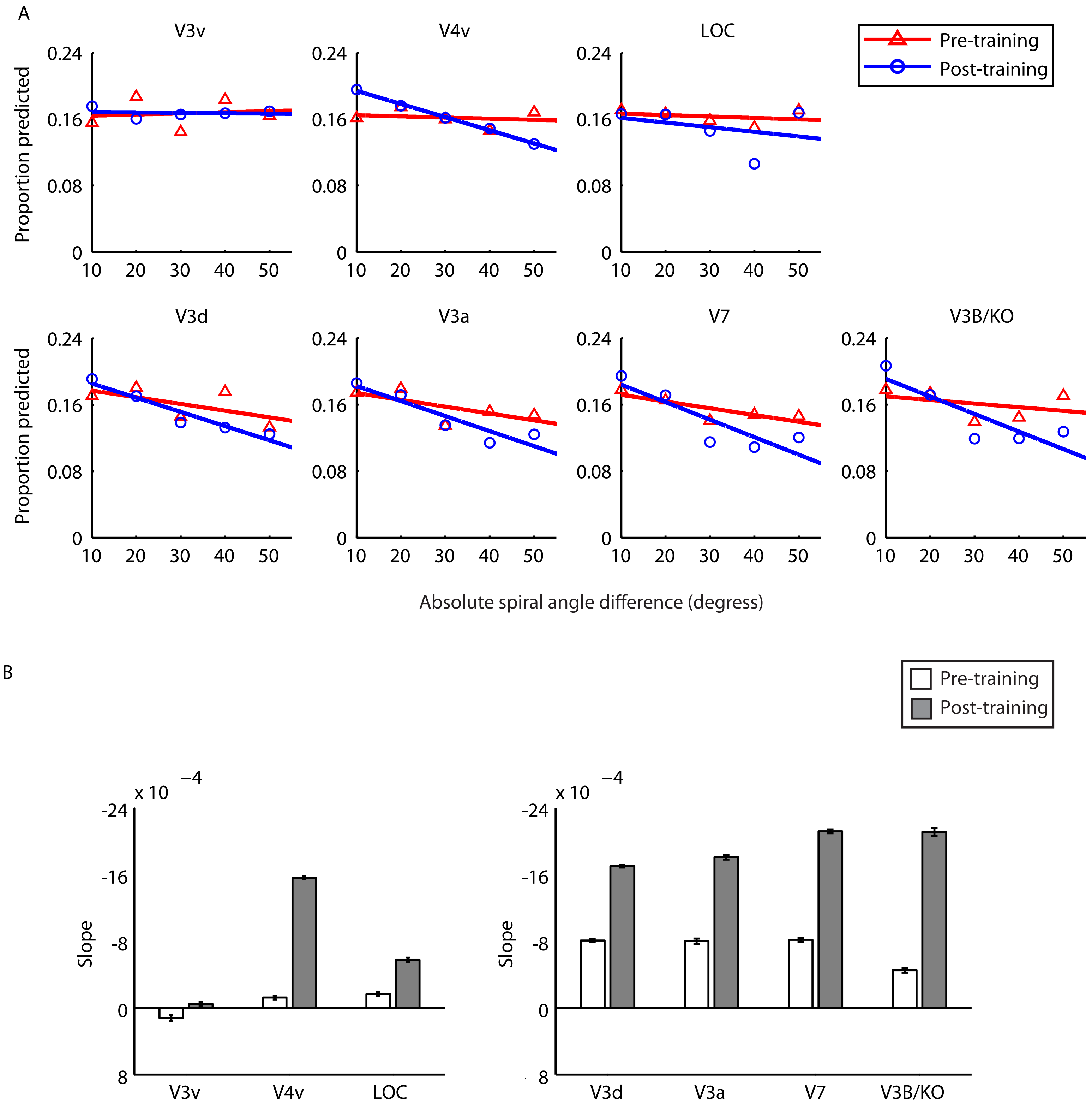


Figure S5 - Zhang et al.

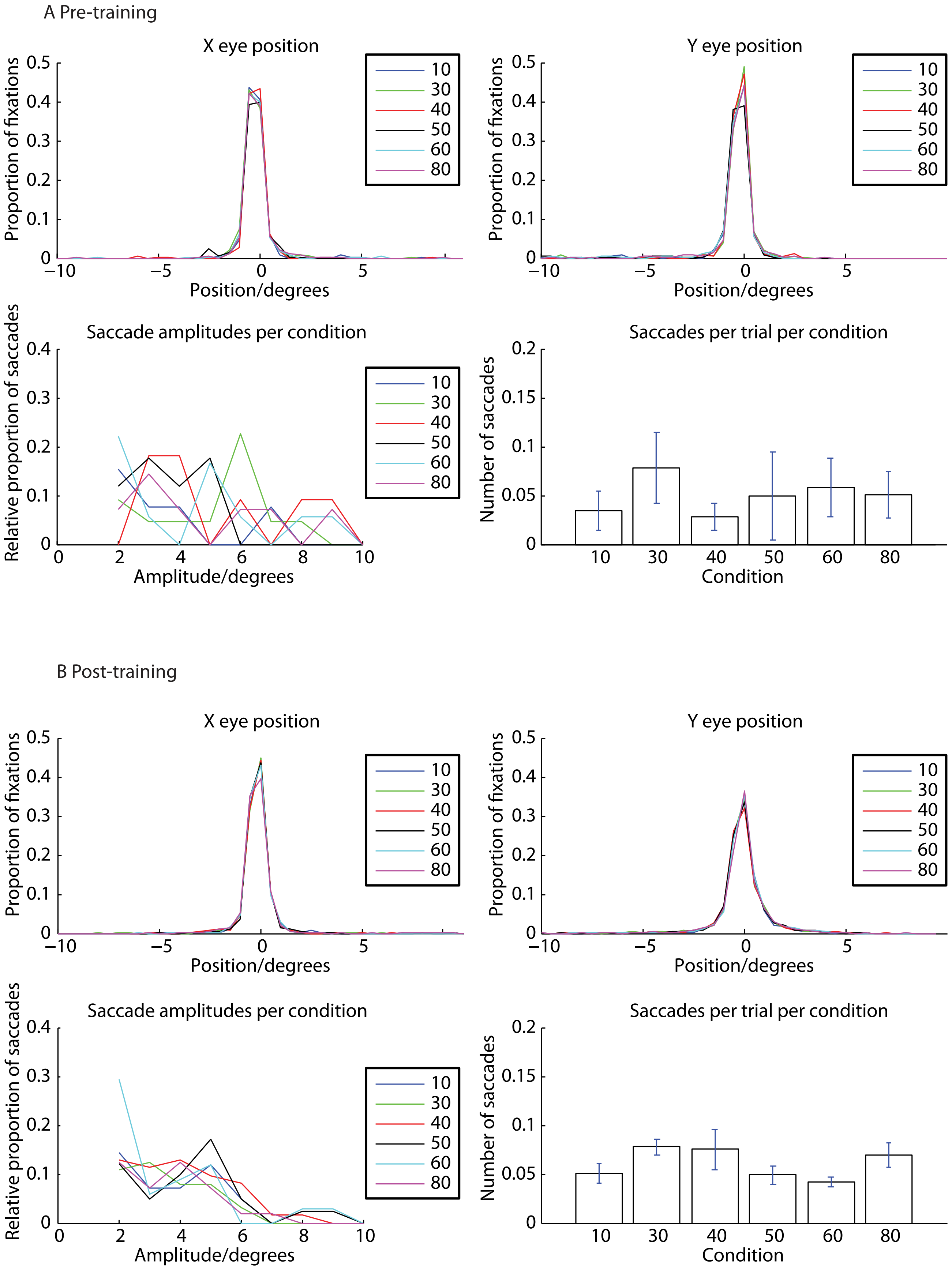
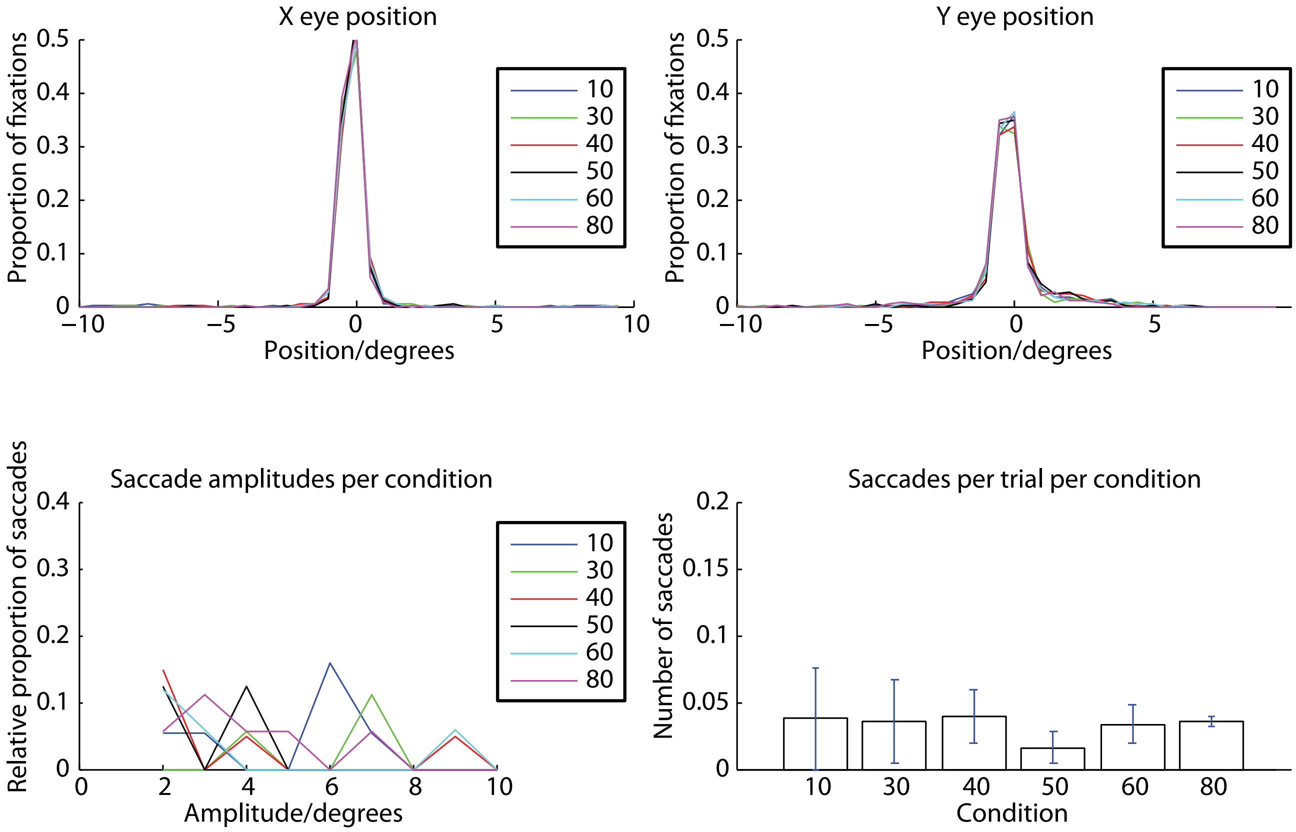


Figure S5 - Zhang et al.

C High signal


X eye position
Proportion of fixations
Position/degrees
Y eye position
Proportion of fixations
Position/degrees
Saccade amplitudes per condition
Relative proportion of saccades
Amplitude/degrees
Saccades per trial per condition
Number of saccades
Condition