Advancing the Understanding of Brain Function with Multivariate Pattern Analysis

Der Fakultät für Naturwissenschaften
der Otto-von-Guericke-Universität Magdeburg
zur Erlangung des akademischen Grades

doctor rerum naturalium
(Dr. rer. nat.)

am 24. März 2009

eingereichte Dissertation,

vorgelegt von Dipl.-Psych. Michael Hanke

Abstract

Decoding patterns of neural activity onto cognitive states is one of the central goals of functional brain imaging. Standard univariate fMRI analysis methods, which correlate cognitive and perceptual function with the blood oxygenation-level dependent (BOLD) signal, have proven successful in identifying anatomical regions based on signal increases during cognitive and perceptual tasks. Recently, researchers have begun to explore new multivariate techniques that have proven to be more flexible, more reliable, and more sensitive than standard univariate analysis. Drawing on the field of statistical learning theory, these new multivariate pattern analysis (MVPA) techniques possess explanatory power that could provide new insights into the functional properties of the brain.

However, unlike the wealth of software packages for univariate analyses, there are few packages that facilitate multivariate pattern classification analyses of fMRI data. This in turn prevents the adoption of these methods by a large number of research groups to fully assess their potential with respect to cognitive neuroscience research. Here, a novel, Python-based, cross-platform, and open-source software framework, called PyMVPA, for the application of multivariate pattern analysis techniques to fMRI datasets is introduced. PyMVPA makes use of Python's ability to access libraries written in a large variety of programming languages and computing environments to interface with the wealth of existing machine-learning packages. The framework is presented in this thesis, and illustrative examples on its usage, features, and programmability are provided.

In addition, this thesis provides an overview of promising strategies for the application of MVPA to neuroimaging datasets. While various possibilities are reviewed based on previously published studies, the primary focus lies on the *sensitivity analysis* technique that is shown to provide interesting additional information which are readily available as part of any typical MVPA-based study. Moreover, this technique is eminently suited for modality-independent data analysis, a feature that is demonstrated by an example of a uniform analysis of datasets from four different brain imaging domains.

The thesis concludes with a discussion about the challenges that have to be faced to establish MVPA as a standard analysis procedure, including the statistical evaluation of results, as well as potential pitfalls in their interpretation.

Acknowledgements

A dissertation is a big step in a scientific career, and the path towards it is peppered with challenges. Some are purely intellectual, some are just laborious tasks. Either one can be exhaustive when faced alone. Fortunately, I had many people that helped and guided me on this journey.

First of all, I would like to thank Stefan Pollmann for his generous and unconditional support from the very beginning (actually even before that). Without his attitude to risk something, and reach out for new ground this project would never have gotten in the shape in which it is now. He never seemed to have lost the faith in progress, despite some traces of what could be interpreted as an occasional procrastination of mine (only at first sight of course).

Ten years ago, Yaroslav Halchenko and I could have served as the main characters in a light-hearted movie about two guys that met "on the internet". Having very similar beliefs about "how it should be done *rright*", and sharing a slight tendency towards unsolicited preaching, I can only say that it was all fun – from honeymoon in Paris to Russian hiking, but also the countless hours of fruitful discussions about research, free software, and the amusing asides of life. After proving that earth is indeed a small planet, and six hours time difference actually help you to work 24 hours a day, I am really looking forward to have his office right next to mine for the coming years.

Nevertheless, a poor guy who only has virtual friends. Without my real ones life would be sad, and without my wonderful colleagues work would be a burden. I want to thank Marianne Maertens for her friendship and encouragement over almost ten years, Angela Manginelli for casual discussions about Italian lifestyle, and tough ones about the scientific topics. Ingo Fründ was the one who orginally introduced me to the Python world: Thanks! Moreover, besides German, English, and a bit of Turkish he is also fluent in linear algebra, which was a tremendous help in straightening things out more than a few times. More recently, I very much enjoyed the company of Reka Daniel and Florian Baumgartner and hope they don't mind my frequent appearances.

I would also like to thank my fellow PyMVPA developers Per Sederberg and Emanuele Olivetti for having these occasional discussions that extend my horizon within minutes, and for the joy of writing papers with them. Along that line, I also want to thank Jim Haxby and Steve Hanson for their sustained support, and even more for the memorable post-barbecue discussions.

Most importantly, however, I want to thank my beloved girl friend and prospective wife Karoline Makosch for being my best companion, unselfish supporter, and the anchor in my life. Furthermore, I'm glad to have our kids Charlotte and Moritz, who enrich our life with a completely non-scientific, but essential facet. And finally, I am really grateful for the support of their grandparents, that always provided us with a bit of

extra-freedom, whenever it was necessary.

Contents

1. Introduction **1**

2. Advancing the Comprehension of Brain Function with Multivariate Pattern Analysis **2**
2.1. Exploring the brain voxel by voxel 3
2.2. Considering brain-response *patterns* 6
2.3. Linking brain-responses to cognitive states with MVPA 8
2.3.1. Technical advantages 8
2.3.2. Theoretical advances 11
2.3.3. Practical advantages 11
2.4. The need for transparency 12
2.5. Practical difficulties: Where is the software? 14

3. The PyMVPA Framework **16**
3.1. Python: *lingua franca* of computational (neuro)science 16
3.2. Bridging fMRI data and machine learning software 17
3.3. Dataset handling 19
3.4. Machine learning algorithms 23
3.4.1. Classifier abstraction 23
3.4.2. Feature measures 24
3.5. Workflow abstraction 25
3.5.1. Dataset resampling 25
3.5.2. Feature selection procedures 26
3.6. Demonstrating the high-level interface 27
3.6.1. Loading a dataset 27
3.6.2. Simple full-brain analysis 28
3.6.3. Feature selection 28
3.7. Community-driven development 30

4. Analysis Strategies **33**
4.1. Labeling functional systems by prediction performance 34
4.1.1. Classify and dissect 34
4.1.2. Leaving the voxel-space: Brain response components 35
4.1.3. Multivariate searchlight 36
4.2. More information is available: Sensitivity analysis 40
4.2.1. Event-related example dataset 42

4.2.2. Spatio-temporal MVPA 43
4.3. Sensitivity analysis as a modality-independent technique 49
4.3.1. EEG 50
4.3.2. MEG 51
4.3.3. Extracellular recordings 56
4.3.4. fMRI 57
4.4. Using an unsupervised method to investigate information representations 62
4.4.1. Self-organizing maps 62
4.4.2. Example: Looking for categorical information in the ventral object-processing stream 63
4.5. Statistical safeguarding of results 65
4.5.1. Feature selection stability 67
4.5.2. Statistical measures beyond accuracies and *t*-tests 67

5. General Conclusions **71**
5.1. Fathom the full potential 72
5.2. Powerful methods for cognitive neuroscience 74
5.3. Offering the blueprint of a *mind-reader* 75

6. References **77**

A. Curriculum vitae **86**

1. Introduction

The thesis put forth in this dissertation is that multivariate pattern analysis (MVPA) techniques can be used to complement established analysis procedures in neuroimaging research and that they provide a rich, model-free, and domain-neutral approach to gain insights into the functioning of the brain. To this end, the main focus of the thesis lies on demonstrating the application of numerous linear MVPA methods on a variety neural datasets and to inspect the structure of the resulting multivariate models to shed some light on the contained information. No attempt will be made to promote a specific algorithm, instead multiple techniques will be used to show their flexibility and behavior.

After a short introduction into the most widely used methods for functional magnetic resonance imaging (fMRI) data analysis (sections 2.1 and 2.2), the general potential of MVPA techniques for this purpose is evaluated (section 2.3). In addition to the theoretical properties the following sections will also discuss some implications and practical problems of the application of MVPA.

In chapter 3 a novel software framework for MVPA-based data analysis, called PyMVPA, is introduced, that is specifically tailored towards neuroimaging research and aims to address the majority of the previously discussed practical difficulties.

The illustration of its basic concepts and components is followed by the demonstration of several analysis strategies to outline the flexibility of the MVPA approach and show the richness of the information available from inspection of the MVPA models. (chapter 4).

While the bulk of this thesis is mostly concern with the analysis of fMRI data, the underlying concepts apply to many neuroimaging data modalities. Therefore, illustrative example of analyses of other modalities will be given to show the potential of MVPA techniques for multi-modal or modality-agnostic data analysis (section 4.3).

The thesis concludes with an outlook on future developments concerning a subset of the many open questions in this field, and finally, reflects on the explanatory power of the method as a general purpose tools to uncover information encoded in the brain activity patterns (chapter 5).

A substantial proportion of this thesis has been previously published in two articles that appeared in peer-reviewed journals. The introduction into PyMVPA in chapter 3 in conjunction with section 4.1.3 overlaps to a large degree with **?**. The discussion of aspects related to general software development procedures in section 3.7, major parts concerned with the features of the Python programming language and its associated software packages (sections 3.1 and 3.2), as well as the examples for modality independent data analyses with MVPA in section 4.3 were originally published in Hanke et al. (2009).

2. Advancing the Comprehension of Brain Function with Multivariate Pattern Analysis

Recently, neuroscientists have reported surprising results when they applied machine learning techniques based on statistical learning theory in their analysis of fMRI data[1]. For example, two studies employing multivariate pattern analyses by Haynes & Rees (2005) and Kamitani & Tong (2005) were able to predict the orientation of visual stimuli from fMRI data recorded in human primary visual cortex. These studies aggregated information contained in variable, subtle response biases in large numbers of voxels which would not be detected by univariate analysis. These small signal biases were sufficient to disambiguate stimulus orientations despite the fact that their fMRI data were recorded at 3 mm spatial resolution. This is especially notable because the organization of the primary visual cortex in monkeys indicates that the orientation-selective columns are only approximately 0.5 mm in diameter (Vanduffel, Tootell, Schoups, & Orban, 2002), consequently any individual voxel carries only a small amount discriminating information on its own.

Other MVPA-based studies have further highlighted the strength of a multivariate analysis approach. For example, MVPA was first used to investigate neural representations of faces and objects in ventral temporal cortex and showed that the representations of different object categories are spatially distributed and overlapping and revealed that they have a similarity structure that is related to stimulus properties and semantic relationships (Haxby et al., 2001; Hanson, Matsuka, & Haxby, 2004; O'Toole, Jiang, Abdi, & Haxby, 2005).

Although the first use of MVPA on neuroimaging data dates back into the early 1990s (Moeller & Strother, 1991; Kippenhahn, Barker, Pascal, Nagel, & Duara, 1992, with both studies using data from positron emission tomography, PET), these striking developments have just recently attracted considerable interest throughout the neuroscience community (see Norman et al., 2006; Haynes & Rees, 2006, for reviews). While classical, multivariate statistical techniques, such multivariate analysis of variance (MANOVA), are not well suited for the specifics of fMRI data (*e.g.* due to its high-dimensionality with low number of observations; O'Toole et al., 2007), machine learning (ML) re-

[1] In the literature, authors have referred to the application of machine learning techniques to neural data as *decoding* (Kamitani & Tong, 2005; Haynes et al., 2007), *information-based analysis* (*e.g.* Kriegeskorte, Goebel, & Bandettini, 2006) or *multi-voxel pattern analysis* (*e.g.* Norman, Polyn, Detre, & Haxby, 2006). Throughout this thesis the more general term *multivariate pattern analysis* (MVPA) will be used to refer to all these methods.

search has spawned a more powerful set of multivariate analysis techniques. They are typically generic, flexible (*e.g.* classification, regression, clustering), powerful (*e.g.* multivariate, linear and non-linear) and often applicable to various data modalities with minor modality-specific preprocessing. These techniques, which were developed outside the neuroscience community, could provide another valuable approach to the analysis of neural data. This chapter aims to assess their potential by contrasting their most important properties with currently prevailing analysis procedures to pinpoint general differences, limitations, and advantages. This introduction will necessarily be brief and limited to the topics which are closely related to the scope of this thesis. For a general overview of inferential and non-inferential fMRI analysis techniques the reader is referred to Petersson, Nichols, Poline, & Holmes (1999a,b).

2.1. Exploring the brain voxel by voxel

The most prevalent approach used in the analysis of fMRI data is the *statistical parametric mapping* (SPM; Friston, Holmes, et al., 1994). SPM employs a voxel-wise analysis procedure that aims to localize units in the functional neuroanatomy by comparing every voxel's behavior to modeled signal timecourses derived from prior knowledge about the hemodynamic system of the brain.

Although in principle all fMRI analysis techniques rely on the hemodynamic signal measured by fMRI, SPM requires specific assumptions about the nature of this signal to be made. Hence, a few important details need to be mentioned to be able to outline potential problems that arise from these assumptions.

A subset of the fMRI signal is related to the ratio of oxyhemoglobin and deoxyhemoglobin in a particular voxel, where an increase in the proportion of deoxyhemoglobin is causing local decreases in the strength of the magnetic field, and vice versa (Ogawa, Lee, Kay, & Tank, 1990). The effect is called the *blood oxygenation level-dependent* (BOLD) response, and this physiological signal is thought to reflect the intensity of the neural processing (*i.e.* firing of neurons) in the surrounding brain tissue – an idea originally postulated by Roy & Sherington (1890). Evidence in favor of this hypothesis comes from two studies by Logothetis (Logothetis, Pauls, Augath, Trinath, & Oeltermann, 2001; Logothetis, 2002) which found strong correlations between the BOLD signal and local field potentials (LFPs) simultaneously measured by intracortical electrodes. However, the association of the BOLD signal and neural processing has not gone unquestioned and the corresponding debate does not seem to be resolved yet (see *e.g.* Heeger & Ross, 2002, for a review).

The basic analysis routine of SPM involves the generation of (potentially many) BOLD response models from the actual experimental design, covering the full duration of a scanning session. The hemodynamic signal in response to a short stimulation is significantly smeared over time (Bandettini, 1999), so that when building the BOLD-response model, this temporal forward contamination of the signal is typically accounted for by convolving a stimulation protocol with an assumed hemodynamic response function (HRF). Moreover, the extension of the stimulus-related signal over time causes a significant over-

lap of responses to rapidly occuring stimulations. A central assumption in SPM is that overlapping responses combine in an additive fashion (Boynton, Engel, Glover, & Heeger, 1996). The panels (A) and (B) in figure 2.1 show an exemplary "translation" of a stimulation protocol with events of decreasing temporal distance into the BOLD-response model.

For any further analysis the models have to be downsampled to match the temporal resolution of the recorded fMRI data (Fig. 2.1C). Afterwards each response timeseries model can be used as a regressor in a general linear model (GLM), which is then fitted to the actual timeseries of each voxel in the dataset individually, *i.e.* univariately (Fig 2.1D and E). The regressor parameter estimates (often referred to as *beta-weights*, emphasizing their multiple linear regression origin) are the intermediate analysis results, indicating the magnitude of a voxel's response to a particular experimental condition. Research questions are essentially framed as contrasts between experimental conditions, and hence as contrasts of parameter estimates. Such a contrast is equivalent to asking the question whether a certain voxel differs in its response behavior between experimental conditions.

This analysis approach implies two problems. The first problem is that being model-based, SPM relies on appropriate HRF models to prevent impairing the sensitivity to actually detect brain responses to a respective stimulation. However, the BOLD response is subject to substantial variability across subjects (Aguirre, Zarahn, & D'esposito, 1998). But also within subjects there is variation in the response timing. Schacter, Buckner, Koutstaal, Dale, & Rosen (1997) have shown a timing difference of several seconds between anterior and dorsal prefrontal cortex in a memory study. Buckner et al. (1998) report a difference of one second between extrastriate and prefrontal cortex during a word generation task, and even within visual cortex Bandettini (1999) has shown a variance of response timing of 1-2 s. Moreover, the shape and timing of the BOLD response are also influenced by the actual experimental paradigm employed in a study. Miezin, Maccotta, Ollinger, Petersen, & Buckner (2000) report a significant decrease in the response amplitudes for temporally dense events, as opposed to spaced stimulation. This variation is typically accounted for by adding additional regressors to the GLM design matrix, such as the temporal derivatives of the original HRFs, or more generally the use of HRF basis functions (see *e.g.* Woolrich, Behrens, & Smith, 2004).

The second, and more vexing problem is that typically several ten thousand univariate statistical tests are performed to assess the significance of parameter estimates, or their contrasts, which inevitably leads to type I error rate inflation, *i.e.* rejecting the null hypothesis (there is no difference) when it is actually true. That this is indeed a problem becomes obvious if one considers that a standard fMRI dataset contains about 30000 voxels. Testing each of them individually using a common criterion of $p < 0.05$ would lead to approximately 1500 voxels being labeled as significantly "activated" – regardless of the actual signal in the data, merely by chance.

This problem is usually addressed by performing *alpha*-level corrections for *multiple comparisons*. However, techniques such as Bonferroni correction typically lead to overly conservative tests and hence increase the type II error rate, *i.e.* not rejecting the null hypothesis whenever it is false. Therefore several more sophisticated techniques have been developed that aim to improve the situation, *e.g.* false discovery rate (FDR; Ben-

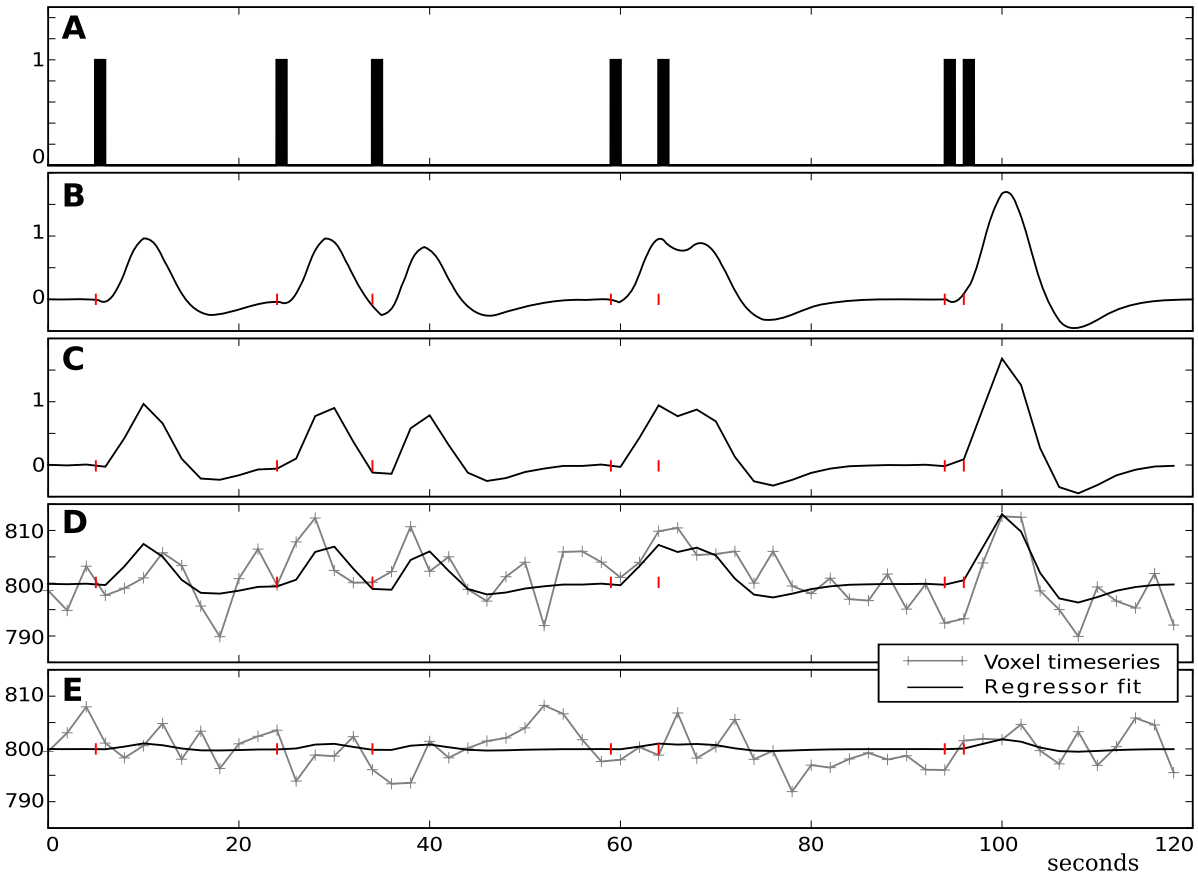

Figure 2.1.: Basic SPM analysis procedure. Based on the stimulation protocol (A; stimulus intensity over time) a model regressor (B) is generated by convolving the protocol with a model hemodynamic response function (HRF) representing the temporal properties of the blood oxygenation level-dependent (BOLD) response. (B) shows model shapes for stimulation events with decreasing temporal distance leading to a substantial overlap of the corresponding BOLD-responses. Standard analysis procedures typically assume additivity of individual responses. The model regressor is assumed to represent the signal timecourse of a voxel that responds to the stimulation. For the actual analysis the model has to be downsampled to the temporal resolution of the fMRI data (C). Finally, the GLM is fitted to each voxel's timeseries individually, yielding per regressor β weights indicating the magnitude of the response. (D) shows a model fit for artificially generated data based on the regressor shape itself and (E) shows a fit for a timeseries generated from Gaussian noise only (both timeseries with an arbitrary signal baseline). To take the goodness of fit of each regressor into account typically z-statistics are reported instead of raw β weights (in this example (D) $z = 6.8$ and (E) $z = 0.95$). The red ticks mark the corresponding stimulus onsets.

jamini & Hochberg, 1995) or Gaussian random field (GRF; Sigmund & Worsley, 1995) theory that tries to account for the spatial distribution of "activated" areas.

To summarize, SPM represents a mass-univariate analysis technique that analyzes each voxel's timeseries independently from all others, and is therefore explicitly excluding potentially available information in the global covariance structure of a dataset. Although this property seems to be unfavorable for the analysis of a system that is performing a massively parallel signal processing task, it was nevertheless motivated by two valid reasons (Monti, 2006). First, fMRI data is typically very high-dimensional (*i.e.* usually several 10000 voxels), but at the same time most of the datasets consist of only a couple of hundred volumes, and even less experimental trials, which leads to difficulties with some multivariate, statistical algorithms, such as linear discriminant analysis[2]. The second reason is incident with the primary intention of SPM to identify and localize functional subsystems of the brain. Multivariate techniques consider a functional volume as a whole and are able to integrate complex signals, which in turn might be difficult to interpret in terms of the contribution of a specific region of interest (ROI). Using a univariate technique, such as SPM always provides inherent localization information since each voxel, and hence a specific location in the brain, is tested individually with the aim to find clusters of voxels with a similar temporal behavior.

2.2. Considering brain-response *patterns*

Data-driven techniques, such as independent component analysis (ICA), or principle component analysis (PCA) are able to address some problems of model-based approaches that have been outlined earlier. First of all they are multivariate techniques that are able to take interactions between voxels into account, and hence have access to a huge amount of information that is invisible to SPM. However, in addition to that, these model-free methods are also applicable to cognitive paradigms where there is no appropriate a priori model of the anticipated brain response (Calhoun, Pekar, McGinty, Adali, & Watson, 2002).

The basic analysis pipeline contains similar pre-processing steps as for SPM. However, instead of a univariate GLM-model fit, the data is factored into a set of components that capture a certain amount of the variance in the dataset, with PCA and ICA implementing different assumptions about the relation of these components (in terms of orthogonality or independence). In the case of fMRI data typically a spatial ICA is performed (Calhoun, Adali, Hansen, Larsen, & Pekar, 2003) that yields a number of spatial patterns (*i.e.* voxel maps) that share a common temporal profile (Fig. 2.2).

In contrast to SPM, this flexible analysis approach is both an advantage and a disadvantage. Data-driven techniques are able to pickup *any* signal in a dataset, regardless of its association with the experimental design. This property makes them a useful tool to identify artifacts, such as slice-dropout, head-motion related signals, eye-movement artifacts, physiological, or scanner-related noise components (see Beckmann, 2009, for

[2]More voxels (variables) than volumes or trials (observations) leads to the inversion of a rank-deficient matrix, and hence is impossible to process.

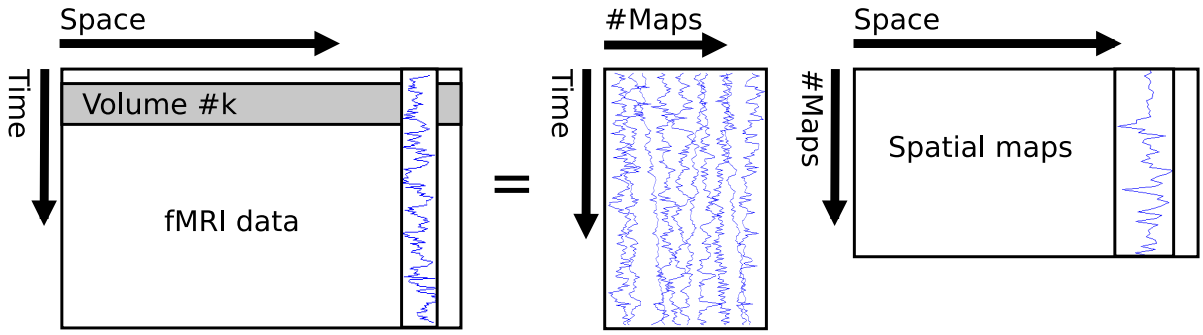

Figure 2.2.: Spatial independent component analysis (ICA) of fMRI data, factors the dataset into a set of spatial maps and their associated temporal profiles. The figure has been adopted from Beckmann (2009).

examples). Because the analysis offers the temporal profile of each component, they can be used to *denoise* the data, by *e.g.* using the profiles of artifact components as confound regressors in a GLM.

However, noise-components will be intermixed with the ones that are interesting in terms of the original research question, and neither ICA nor PCA provide intrinsic measures to associate components with the experimental design. Instead, components are usually ordered by the amount of explained variance. If one recalls that the expected BOLD-response magnitude at 3 Tesla is just about 1-2 % signal-change (Aguirre et al., 1998), it becomes obvious that components explaining lots of variance do not necessarily explain "interesting" variance, but rather things like motion artifacts, that cause substantial changes in signal intensity (O'Toole et al., 2007).

The "hypothesis-free" analysis approach forces researchers to interpret and label each estimated component by eye, hence reversing the analysis logic as opposed to SPM, where anticipated signal timecourses are determined first, and the data is analyzed with respect to them afterwards. The situation is made even worse by the fact that the size of the analysis results is as large as the input data, *i.e.* a dataset with 600 volumes is factored into 600 components with their associated temporal profile. Inspecting, interpreting and finally labeling all those components by eye will identify some as artifacts, and some as "interesting", but the nature of most components remains ambiguous, and they are typically ignored (O'Toole et al., 2007).

At least two approaches exists that aim to improve the interpretation of ICA results. The first is an attempt to reduce the number of estimated ICA components by assessing the actual dimensionality of the dataset, via a PCA-analysis step prior to the ICA procedure (Beckmann & Smith, 2005). However, that still leaves the researcher with many components to interpret, including those artifact-related ones that explain lots of variance. The second approach is a post-hoc labeling of each component, by correlating the respective temporal profile with a model-timeseries, basically identical to a GLM-regressor in SPM analyses. This will automatically identify components which are related to the experimental design. However, reverting to a response-model effectively demolishes the ability of ICA/PCA techniques to emphasize components for which no

appropriate a priori model exists.

2.3. Linking brain-responses to cognitive states with MVPA

The application of MVPA to neuroimaging data promises to resolve many of the shortcomings of established model-based or model-free techniques that have been outlined so far. Their basic flaw is that they either provide a quantitative link between experimental design and fMRI data, or are pattern-based (*i.e.* multivariate), but not both.

O'Toole et al. (2007) provide an excellent classification of MVPA with respect to neuroimaging research and other fields of psychological research. The authors claim that "pattern-based classification analyses have the potential [...] to become *the* standard approach in functional neuroimaging analysis", and list a number of *technical, theoretical,* and *practical* advances. The following sections will review these and some additional advantages in comparison to established and prevailing techniques, but at the same time also pinpoint problems that must be resolved before MVPA has the chance to become a *standard* method.

2.3.1. Technical advantages

Looking at the basic elements of the MVPA pipeline will quickly reveal two major advantages that make these methods superior to both SPM and ICA/PCA or similar algorithms. In general, MVPA makes use of a classifier algorithm that is trained on a dataset to associate brain response patterns with the corresponding experimental conditions. During training a multivariate model is generated that tries to capture important information to allow for such an association to be established. As a second step the trained model is usually tested against a second independent dataset. The aim is to assess whether the supposedly learned association can be used to correctly predict the experimental condition of yet unseen samples of brain response pattern, which in turn validates the model (Pereira, Mitchell, & Botvinick, 2009).

MVPA shares its multivariate nature with other data-driven or unsupervised methods, such as ICA. Supervised procedures, such as a classifier, however, resolve their labeling problem by providing a direct quantifiable link between neuroimaging data and experimental design (O'Toole et al., 2007). The prediction performance of classifiers on new, previously unseen data can be interpreted as a test of the goodness of fit of the extracted multivariate model to the structure of the underlying signal. Hence, a classifier not only links information contained in a neuroimaging dataset to experimental conditions, it also provides a measure how well that information can describe them.

Such a link is available despite the fact there is no explicit a priori model specification of the expected signal, which is, for example, necessary to perform a posthoc labeling of ICA components by correlating their temporal profile with standard GLM regressors. However, classifiers can obviously also operate on model-fit parameters instead of "raw" data, and hence make use of a priori assumptions about an anticipated signal.

Moreover, in contrast to ICA results, MVPA dramatically reduces the size of its model, since typically one model parameter per feature/voxel is estimated, instead of one parameter per voxel and timepoint for ICA, which therefore merely performs a redescription of the data. This reduced multivariate model can nevertheless be subject of an exploratory analysis, which will be the main theme of chapter 4.

As the MVPA approach is free of a priori response models, it can also account for the aforementioned variability of the BOLD-response across subjects, brain regions, physiological states, and experimental designs. This is an important advantage, since SPM relies on an appropriate specification of the expected signals, as all other information in the dataset is considered to be noise and hence affects, via the residual variance, the likelihood to identify a particular area as significantly activated.

Another important advantage of MVPA as opposed to SPM, which also offers a quantifiable link between experimental condition and brain responses, is that it represents a multivariate analysis technique that, by design, does not suffer from the methodological side-effects of the mass-univariate approach. Voxels, or more general, variables are not treated as independent entities that are tested for model compliance individually. While considering fMRI volumes as multivariate samples much better reflects the massively parallel nature of the brain, it also obviates to a certain degree the need for sophisticated algorithms that are applied to assess the statistical relevance of SPM "activation maps" (*e.g.* Bonferroni correction for multiple comparisons, false discovery rate, or Gaussian random field theory). MVPA does not make any assumptions about the inherent smoothness of the data, so that aside from the intention to boost the signal to noise ratio, there is no need for spatial filtering to, for example, control the true number of degrees of freedom to prevent an overly conservative statistical testing.

At the latest with the development of the support vector machine (SVM; Vapnik, 1995) classifier there are methods available that have been derived from statistical learning theory, and do not suffer from the limitations of classical statistical techniques, such as MANOVA or LDA. SVMs for examples can easily deal with feature-spaces (*i.e.* number of variables/voxels) of well over one million features.

The use of multivariate information integration instead of having to rely on significant individual features in turn opens the door for a further increase in the dimensionality of the recorded datasets to *e.g.* make use of the power of modern high-field MRI-scanners to record functional volumes at spatial resolutions of as low as 1 mm (Kriegeskorte & Bandettini, 2007). Measuring functional data at such high resolutions moves fMRI closer to exploit information about signal encoding at the level of the columnar organization of the human brain (Tanaka, 1996) – promising exciting new insights into details of information processing *within* individual functional subsystems. Multivariate procedures also offer the possibilities to investigate the interactions between functional units of the brain, which is a critical feature to advance the understanding of brain function beyond the level of localization efforts.

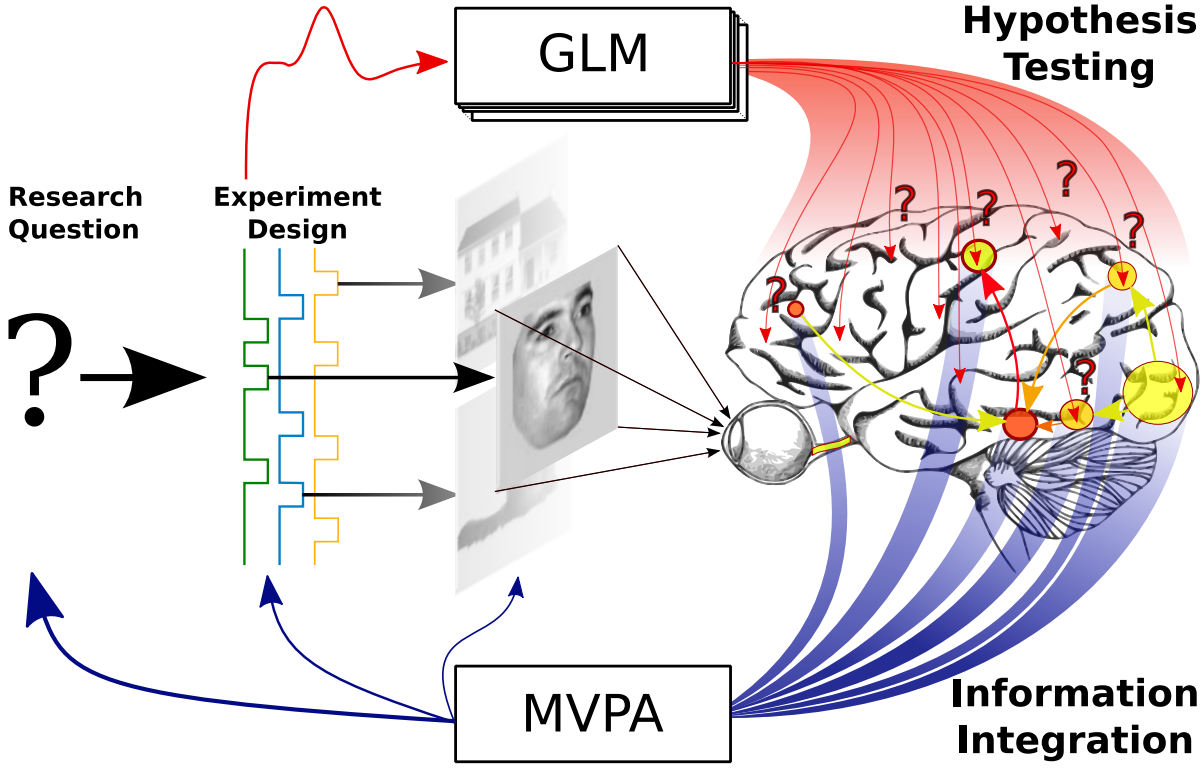

Figure 2.3.: Conceptual differences between MVPA and SPM analyses. SPM aims to identify brain areas that are activated by a specific task or experimental condition by correlating a response model with the recorded signal of each individual voxel. SPM represents a mass-univariate procedure that cannot account for interactions between multiple voxels. MVPA reverses this analysis logic and provides a direct quantifiable link between brain response patterns and experimental design. A multivariate model is fitted that identifies information that can be used to predict certain experimental conditions, or even individual stimuli from fMRI data. The use of feature selection methods can restrict the complexity of the multivariate model by identifying information that is "important" with respect to the intended classification task. Being free of a priori response models, the exploratory analysis of the MVPA model parameters can even yield new research questions, by identifying unexpected, yet informative signals. This figure is a modified and merged variant of two figures that have been adopted from Halchenko (2009).

2.3.2. Theoretical advances

The theoretical advances could be considered as mere by-products of the technical advantages of MVPA. However, while being caused by properties of the techniques that have been developed outside the neurosciences, they offer an enormous potential for new insights into the functional properties of the brain and could lead to a change of experimental paradigms in cognitive neuroscience research.

The most important improvement is once again the switch from univariate to multivariate analysis. The explanatory power of SPM is effectively limited to the localization of "activated" regions, which additionally have to be spatially extended to be detectable whenever the typical preprocessing routine (*i.e.* spatial filtering) is applied. Although it is possible to perform complex analyses by using sophisticated experimental paradigms and studying interaction contrasts of the fitted GLM regressors, any detectable signal nevertheless has to be available within a single voxel.

Employing multivariate methods allows to refocus the research on *how* information is encoded, instead of exclusively looking at *where* it is in the brain (O'Toole et al., 2007). Pioneering work on this topic has been done by *e.g.* Hanson et al. (2004) who revealed a combinatorial encoding of object category information in human ventral temporal cortex and Kriegeskorte, Mur, Ruff, et al. (2008) who were able to show striking correlations between the similarity structure of categorical object representations in the inferior temporal cortex of humans and monkeys.

Moreover, being able to access the spatio-temporal structure of fMRI data allows to perform causal modeling of activation patterns to determine functional networks in the brain that are associated with a specific task (Halchenko, 2009). Although a similar analysis logic is possible using the dynamic causal modeling (DCM; Friston, Harrison, & Penny, 2003) technique, MVPA remains a model-free method that allows to achieve a similar goal with much less a priori assumptions (which can of course have both advantages and disadvantages, depending on the appropriateness of those assumptions).

A last notable improvement of MVPA over established procedures is the inherent model-testing and the reversed direction of concluding. The principle question underlying SPM can be paraphrased as "Does a voxel behave like a responsive voxel according to my model for a certain experimental condition?" whereas MVPA is asking "Does a set of voxels contain enough information to reliably predict the experimental condition that caused a particular brain state?" (see figure 2.3 for a conceptual comparison of the analysis logic of SPM and MVPA).

2.3.3. Practical advantages

A third and final set of improvements is associated with the practical aspects of the application of MVPA. In general they can be subsumed as "becoming standard" by discarding the unique analysis procedures that are commonly applied in brain imaging research. Although high-dimensional datasets are very common in neuroimaging research they are not unique to this field of science (*e.g.* much larger datasets are usual in human genome research). However, rather unique is the mass-univariate approach to analyze

these datasets.

Its isolated position and the substantial restrictiveness with respect to alternative analysis techniques effectively limits fMRI research by preventing benefits from developments in other areas, such as the machine learning community. Adopting a more widespread analysis paradigm could help to facilitate cross-talk between adjacent research communities.

In that respect MVPA has two primary advantages. First, as being based on statistical learning theory, it is a *standard* tool in many fields of science, *e.g.* computer science, signal processing, image analysis, or artificial intelligence. Its usefulness is constantly being evaluated and new algorithms are developed by skilled researchers outside the neuroscience, hence significantly enlarging the fraction of scientists working to improve analysis techniques that are potentially applicable to neuroimaging data. The second advantage is the aspect of familiarity (O'Toole et al., 2007). Many neuroscientists are familiar with the concept of artificial neural networks (due to connectionism in psychological research in the 80s and early 90s), which are simply another facet of MVPA.

Moreover, from a conceptual point of view, studying classifier performance when predicting category labels of brain response patterns is very similar to the analysis of behavioral data of humans performing a categorization task (*e.g.* in a typical two-alternative-forced-choice (2AFC) paradigm). Procedures such as those originating in the signal detection theory (Green & Swets, 1966) are well understood and provide familiar measures (*e.g.* d' and receiver operating characteristics curves, ROC) to assess the quality of classifier model performances.

This long list of advantages is a strong indicator that neuroimaging research could enormously benefit from a wide adoption of MVPA. However, two aspects are critical with respect to a successful adoption: immediate availability of the associated technology to the whole neuroscience community, and the awareness about its potentials as well as its limitations, most importantly with respect to the interpretation of its results. Both aspects will be the subject of the remainder of this thesis.

2.4. The need for transparency

The analysis of neuroimaging data is a complex process that comprises of many different steps ranging from simple data conversion to sophisticated modeling procedures, and the interpretation as well as statistical evaluation of these models. However, as with all methods one has to be careful to obey limitations or requirements of a particular method since conclusions drawn from inappropriate analyses might be distorted or even plain wrong.

For a proper assessment of the value of a scientific study it is critical to decide whether the employed methods were used appropriately, and hence to judge whether the conclusions drawn have at least the potential to be valid. Obviously, this judgement becomes more difficult with increasing complexity of the analysis procedure. The situation gets even worse if the particular methods used are not part of the standard toolbox of a scientist or reviewer in a certain field, since in this case there is no common ground to

base an evaluation on.

Unfortunately, this is pretty much the current situation of MVPA of fMRI data. For the conventional SPM-based approach there is a huge amount of literature that allows to derive at least a reasonable guess of the many parameters that have to be considered in each analysis. Based on this literature a scientist evaluating a particular study is able to decide whether during preprocessing a reasonable filtering kernel was used, or whether the data was modeled with an appropriate HRF-function. In the worst case, it raises at least a question if some article does not justify the use of extraordinary parameter settings. For the MVPA of fMRI there is very little literature concerned with the evaluation of the many different methods. This is of course not very surprising since the total number of studies using this approach is negligible in comparison to the, at least, 15 years of SPM-based fMRI data analysis.

Despite the absence of a tested set of *good practices* there is nevertheless an increasing number of studies being published that employ MVPA to answer actual scientific questions. A lot of these studies are published in high-ranked journals, possibly partly due to the current excitement about the method (*e.g.* Mitchell et al., 2008; Haynes et al., 2007; Kamitani & Tong, 2005). But what weight can be put on the conclusions made in these pioneering studies?

Typically the research process answers this question by replication. If an effect is found using similar or even different methods, by a different research group, using different data acquisition equipment, its existance will generally be accepted by the science community.

However, in the context of the application of MVPA to fMRI data the intention to replicate a study is hindered by at least two main factors. First, in contrast to the ML community, datasets are typically not available to the general public. This is probably due to the fact that in brain imaging research for each study new datasets are acquired, while ML research typically focuses on few datasets with a well-known structure. The second problem is that published studies typically only contain a verbal description of the applied analysis.

This second factor is much less important for studies employing conventional SPM-based fMRI data analysis, since they are typically performed by one of the widely-used fMRI toolkits, like *AFNI*[3] (R. W. Cox, 1996), *BrainVoyager*[4] (Goebel, Esposito, & Formisano, 2006), *FSL*[5] (Smith et al., 2004), *Lipsia*[6] (Lohmann et al., 2001), or *SPM*[7] (Friston, Jezzard, & Turner, 1994). The behavior of these toolkits is known, they are available to the whole research community, and the algorithms they implement are published in peer-reviewed publications. All these aspects allow to summarize an analysis performed by these toolkits by listing relatively few parameters.

For an ML analysis the situation is totally different. In the absence of an established toolkit for neuroscientific research, an analysis generally involves the combination of many different tools, combined with custom developed, usually unpublished code. While

[3] http://afni.nimh.nih.gov/afni
[4] http://www.brainvoyager.com
[5] http://www.fmrib.ox.ac.uk/fsl
[6] http://www.cbs.mpg.de/institute/software/lipsia
[7] http://www.fil.ion.ucl.ac.uk/spm

it would be nevertheless possible to provide a comprehensive verbal description of an algorithm regardless of its complexity, such description is often not included in the respective publications. For a researcher intending to replicate a study translating a verbal – potentially incomplete, or too superficial – description into running analysis code is a lengthy and error-prone task that turns a replication attempt into a costly project.

Nevertheless, the solution to this problem is quite simple. Instead of exclusively providing a verbal analysis description, a publication should be accompanied by the actual source code that was used to run an analysis. The source code, by definition, provides the highest possible level of detail of a description. Access to the source code can immediately lead to a facilitation of replication efforts, enabling the potential for timely feedback with respect to newly developed analysis strategies, while simultaneously fostering the use of the ones that turned out to be generally successful.

To make such effort worthwhile for the majority of MVPA-based studies, analysis descriptions should be provided in a language that is comprehensible by scientists in the field. The least common denominator for such an endeavor would be some form of pseudo-code. However, chapter 3 introduces a novel software framework that offers an alternative that is significantly more powerful, by actually providing scientists with executable analysis code.

2.5. Practical difficulties: Where is the software?

Various factors have delayed the adoption of MVPA methods for the analysis of neural information. First and foremost, existing conventional techniques are well-tested and often perfectly suitable for the standard analysis of data from the modality for which they were designed. Most importantly, however, a set of sophisticated software packages has evolved over time that allow researchers to apply these conventional and modality-specific methods without requiring in-depth knowledge about low-level programming languages or underlying numerical methods. In fact, most of these packages come with convenient graphical and command line interfaces that abstract the peculiarities of the methods and allow researchers to focus on designing experiments and to address actual research questions without having to develop specialized analyses for each study.

On the other hand, only a few software packages exist that are specifically tailored towards straightforward and interactive exploration of neuroscientific data using a broad range of ML techniques. At present only independent component analysis (ICA), an unsupervised method, seems to be supported by numerous software packages (see Beckmann & Smith, 2005, for fMRI, and Makeig, Debener, Onton, & Delorme, 2004, for EEG data analysis). Therefore, the application of MVPA usually involves the development of a significant amount of custom code. Hence, users are typically required to have in-depth knowledge about both data modality peculiarities and software implementation details.

At the time of this writing there seem to be only two publicly available software packages designed for MVPA of fMRI data. One is the *3dsvm* plugin for AFNI (LaConte, Strother, Cherkassky, Anderson, & Hu, 2005) and the other is the Matlab-based

MVPA toolbox (Detre et al., 2006). However, both packages only cover a fraction of the available algorithms that have been developed in machine learning research (see NIPS[8] community) over the past decades.

For example, the recently founded *Machine learning open source software*[9] project shows an impressive, nonetheless still incomplete, sample of available software packages. At the very least starting with these already available high-quality software libraries has the potential to accelerate scientific progress in the emerging field of MVPA of brain-imaging data. Although these libraries are freely available, their usage typically assumes a high-level of programming expertise and statistical or mathematical knowledge. Therefore, it would be of great value to have a unifying framework that helps to bridge well-established neuroimaging tools and machine learning software packages and provides ease of programmability, cross-library integration and transparent fMRI data handling. Such a framework should at least have the five following features:

User-centered programmability with an intuitive user interface Since most neuroimaging researchers are not also trained as computer scientists, it should require only a minimal amount of programming ability. Workflows for typical analyses should be supported by a high-level interface that is focused on the experimental design and language of the neuroimaging scientist. That being said, of course, all interfaces should allow access to detailed information about the internal processing for comprehensive extensibility. Finally, reasonable documentation is a primary requirement.

Extensibility It should be easy to add support for additional external machine learning toolboxes to prevent duplicating the effort that is necessary when a single algorithm has to be implemented multiple times.

Transparent reading and writing of datasets Because the toolbox is focused on neuroimaging data, the default access to data, should require little or no specification for the user. The toolbox framework should also take care of proper conversions into any target data format required for the external machine learning algorithms.

Portability It should not impose restrictions about hardware platforms and should be able to run on all major operating systems.

Open source software It should be open source software, as it allows one to access and to investigate every detail of an implementation, which improves the reproducibility of experimental results, leading to more efficient debugging and gives rise to accelerated scientific progress (Sonnenburg et al., 2007).

As an attempt to provide such a framework *PyMVPA*[10] (MultiVariate Pattern Analysis in Python) was implemented. The next chapters will introduce the framework and highlight the features that ease the access to the advantages of ML methods and help to overcome the associated difficulties that have been identified so far.

[8] Neural Information Processing Systems http://nips.cc/
[9] http://www.mloss.org
[10] http://www.pymvpa.org

3. The PyMVPA Framework

PyMVPA is a free and open-source framework to facilitate uniform analysis of information obtained from different neural modalities through the use of MVPA. PyMVPA heavily utilizes libraries written in a large variety of programing languages and computing environments to interface with the wealth of existing machine learning packages developed outside the neuroscience community. Although the framework is eminently suited for neuroscientific datasets, it is by no means limited to this field. However, the neuroscience tuning is a unique aspect of PyMVPA in comparison to other ML or computing toolboxes, such as *MDP*[1], *scipy-cluster*[2], or the *Spider toolbox*[3] which are developed as domain-neutral packages.

This chapter offers an overview of PyMVPA's key features and design principles. A justification for the choice of programing language will be presented, and common analysis steps will be illustrated by actual code snippets. The chapter concludes with an outline of the efforts that have been undertaken to promote a community-driven development process to foster the PyMVPA project.

3.1. Python: *lingua franca* of computational (neuro)science

The choice of the programming language is an important decision for a project, since it determines the number of potential developers, as well as the flexibility, and maintainability of the resulting code base. To fulfill the significant computational demands PyMVPA had to be implemented in a programing language that allows for high-performance computing solutions. With increasing size of neuroscientific datasets and also increasing complexity of analysis procedures, there is a strong trend away from the use of individual workstations towards cluster or cloud computing. However, at the same time the choice of language had to be compatible with the primary requirements that have been put forth in section 2.5, mainly being open-source, portable, feature-rich, and easy to learn.

To this end, the *Python*[4] language was selected for PyMVPA. Python is a free and open-source scripting language and is available for all major platforms and operating systems. It has become the open-source scripting language of choice in the research

[1] http://mdp-toolkit.sourceforge.net
[2] http://code.google.com/p/scipy-cluster/
[3] http://www.kyb.mpg.de/bs/people/spider
[4] http://www.python.org

community to prototype and carry out scientific data analyses or to develop complete software solutions quickly. It has attracted attention due to its openness, flexibility, and the availability of a constantly evolving set of tools for the analysis of many types of data. Python's automatic memory management, in conjunction with its powerful libraries for efficient computation (*NumPy*[5] and *SciPy*[6]) abstracts users from low-level "software engineering" tasks and allows them to fully concentrate their attention on the development of computational methods.

Furthermore, Python extensions make it easy to wrap high-performance libraries written in low-level programing languages like C, C++, or Fortran while preserving their speed (*e.g.* via ctypes, SWIG, SIP, Cython), and use them in addition to the available NumPy and SciPy packages which already provide a fast n-dimensional array library with comprehensive signal processing toolboxes. Two other Python packages provide PyMVPA with the possibility to access an even larger code base. The *RPy*[7] module allows PyMVPA scripts to make use of the full functionality of the statistical programing language *R*[8] and all its extensions and support packages. Also, *pymat*[9] and *mlabwrap*[10] provide a similar interface for easy access to Matlab.

Finally, the IPython project provides a powerful Matlab-like command-line interface for interactive data exploration (Perez & Granger, 2007). Recent releases added support for interactive cluster computing, enabling PyMVPA users to make use of high-performance computing hardware.

In addition to its technical advantages, Python is a well documented, easy to learn, interpreted high-level scripting language, which is instrumental in making PyMVPA an easy and powerful multivariate analysis framework. Table 3.1 summarizes the features of a Python-based environment by listing available Python modules which might be of interest in the neuroscientific context.

3.2. Bridging fMRI data and machine learning software

Despite the huge number of specialized Python modules and the resulting fact that it is possible to perform complex data analyses solely within Python, it *once again* often requires in-depth knowledge of numerous Python modules, as well as the development of a large amount of code to lay the foundation for one's work. Therefore, it would be of great value to have a framework that helps to abstract from both data modality specifics and the implementation details of a particular analysis method. The task of PyMVPA, which is aiming to be such a framework, is to help to expose any form of data in an optimal format applicable to a broad range of machine learning methods, and on

[5] http://numpy.scipy.org
[6] http://www.scipy.org
[7] http://rpy.sourceforge.net
[8] http://www.r-project.org
[9] http://claymore.engineer.gvsu.edu/~steriana/Python/pymat.html
[10] http://mlabwrap.sourceforge.net

Table 3.1.: Various free and open-source projects, either written in Python or providing Python bindings, which are germane to acquiring or processing neuroimaging datasets using MVPA. The last column indicates whether PyMVPA internally uses a particular project or provides public interfaces to it.

Name	Description	URL	PyMVPA
Machine Learning			
Elephant	Multi-purpose library for ML	http://elefant.developer.nicta.com.au	
Shogun	Comprehensive ML toolbox	http://www.shogun-toolbox.org/	√
Orange	General-purpose data mining	http://www.ailab.si/orange	
PyML	ML in Python	http://pyml.sourceforge.net	
MDP	Modular data processing	http://mdp-toolkit.sourceforge.net	√
hcluster	Agglomerative clustering	http://code.google.com/p/scipy-cluster/	√
–	Other Python modules	http://www.mloss.org/software/language/python	
Neuroscience Related			
NiPy	Neuroimaging data analysis	http://neuroimaging.scipy.org	
PyMGH	Access FreeSurfer's *.mgh* files	http://code.google.com/p/pyfsio	
PyNIfTI	Access NIfTI/Analyze files	http://niftilib.sourceforge.net/pynifti	√
OpenMEEG	EEG/MEG inverse problems	http://www-sop.inria.fr/odyssee/software/OpenMEEG	
Stimuli and Experiment Design			
PyEPL	Create complete experiments	http://pyepl.sourceforge.net/	
VisionEgg	Visual Stimuli Generation	http://www.visionegg.org	
PsychoPy	Create psychophysical stimuli	http://www.psychopy.org/	
PIL	Python Imaging Library	http://www.pythonware.com/products/pil/	
Interfaces to Other Computing Environments			
RPy	Interface to R	http://rpy.sourceforge.net/	√
mlabwrap	Interface to Matlab	http://mlabwrap.sourceforge.net/	
Generic			
Matplotlib	2D Plotting	http://matplotlib.sourceforge.net	√
Mayavi2	Interactive 3D visualization	http://code.enthought.com/projects/mayavi	
PyExcelerator	Access MS Excel files	http://sourceforge.net/projects/pyexcelerator	
pywavelets	Discrete wavelet transforms	http://www.pybytes.com/pywavelets/	√

the other hand provide a versatile, yet simple, interface to plug in additional algorithms operating on the data.

In the neuroscience context it would also be useful to bridge between well-established neuroimaging tools and ML software packages by providing cross library integration and transparent data handling for typical containers of neuroimaging data, *e.g.* data formats for anatomical and functional volumes in fMRI research. Although there are many fMRI data formats, over the last decade the neuroimaging community has converged on *NIfTI* as a standard data format that most fMRI analysis packages support – either directly or by conversion into their respective native format. Thus it was an obvious choice for the primary data storage format supported by PyMVPA.

While *PyNIfTI*[11] makes it easy to read and write NIfTI files from within the PyMVPA framework, merely being able to access data is not sufficient for a full analysis pipeline. FMRI data typically has to undergo multiple preprocessing steps before an actual anal-

[11] http://niftilib.sourceforge.net/pynifti

ysis can be applied. This involves several procedures to address the specifics of the modality, *e.g.* motion and distortion correction, temporal detrending, and spatial filtering.

Although, with the project *Neuroimaging in Python* (NIPY; Millman & Brett, 2007) there is already an ongoing effort to provide a comprehensive software library for traditional fMRI data analysis, including data preprocessing, there are many other toolkits offering different sophisticated routines. Here, PyMVPA tries to follow the same path as with connecting to various ML packages, by offering simple, yet sufficient, interfaces to access relevant information. Due to NIfTI as the common data format, it is already very easy to import data that has been preprocessed elsewhere. Sometimes, however, information becomes interesting that was generated during preprocessing, but is not available in the dataset itself, *e.g.* if one wants to consider the motion correction parameters for a data detrending procedure in PyMVPA. For such situations PyMVPA tries to provide support to access information in additional data formats, such as the parameter output of the McFLIRT motion correction tool (Jenkinson, Bannister, Brady, & Smith, 2002).

As Table 3.1 highlighted, PyMVPA is not the only ML framework available for scripting and interactive data exploration in Python. In contrast to some of the primarily GUI-based ML toolboxes (*e.g.* Orange, Elephant), PyMVPA is designed to provide not just a toolbox, but a framework for concise, yet intuitive, scripting of possibly complex analysis pipelines. To achieve this goal, PyMVPA provides a number of building blocks that can be combined in a very flexible way. They can be categorized into three distinct components: dataset handling, machine learning algorithms and high-level workflow abstractions. Each component provides interfaces that connect the framework with a large variety of existing software packages. Figure 3.1 shows a schematic representation of the framework design and its building blocks. In the following sections the interfaces to neuroimaging and machine learning software, and how the three components combine to create complete analyses, are discussed.

3.3. Dataset handling

Input, output, and conversion of datasets are a key task for PyMVPA. A dataset representation has to be simple enough to allow for maximum interoperability with other toolkits, but simultaneously also has to be comprehensive in order to make available as much information as possible to *e.g.* domain-specific analysis algorithms. In PyMVPA a dataset consists of three parts: the *data samples*, *sample attributes* and *dataset attributes*. While the data samples are the actual patterns that shall be used for training or validation, sample attributes hold additional information on a per sample basis (see Fig. 3.3). First and foremost of these are the labels that index each data sample with a certain experimental condition and, therefore, define the mapping that will be learned by the classifier.

Additionally, it is often necessary to define groups of data samples. For example, when performing a cross-validation it is necessary to have independent training and validation sets. In the case of fMRI data, with its significant forward temporal contamination

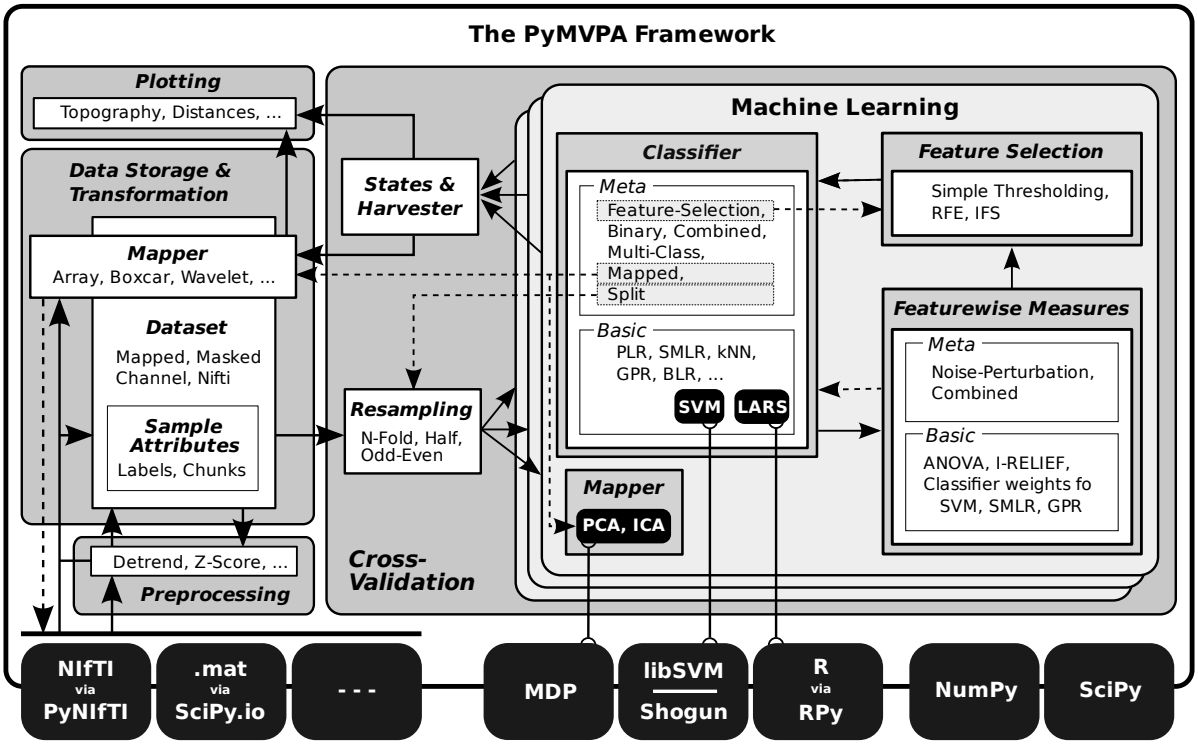

Figure 3.1.: framework. It consists of several components (gray boxes) such as ML algorithms or dataset storage facilities. Each component contains one or more modules (white boxes) providing a certain functionality, *e.g. classifiers*, but also feature-wise measures (*e.g.* I-RELIEF; Sun, 2007), and feature selection methods (recursive feature elimination, RFE; Guyon et al., 2002; Guyon & Elisseeff, 2003). Typically, all implementations within a module are accessible through a uniform interface and can therefore be used interchangeably, *i.e.* any algorithm using a classifier can be used with any available classifier implementation, such as, *support vector machine* (SVM), or *sparse multinomial logistic regression* (SMLR; Krishnapuram et al., 2005). Some ML modules provide generic *meta* algorithms that can be combined with the *basic* implementations of ML algorithms. For example, a *Multi-Class* meta classifier provides support for multi-class problems, even if an underlying classifier is only capable of dealing with binary problems. Additionally, most of the components in PyMVPA make use of some functionality provided by external software packages (black boxes). In the case of *SVM*, classifiers are interfaced to the implementations in *Shogun* or *LIBSVM*. PyMVPA only provides a convenience wrapper to expose them through a uniform interface. By providing simple, yet flexible interfaces, PyMVPA is specifically designed to connect to and use externally developed software. Any analysis built from those basic elements can be cross-validated by running them on multiple dataset splits that can be generated with a variety of data resampling procedures (*e.g.* bootstrapping, Efron & Tibshirani, 1993). Detailed information about analysis results can be queried from any building block and can be visualized with various plotting functions that are part of PyMVPA, or can be mapped back into the original data space and format to be further processed by specialized tools (*i.e.* to create an overlay volume analogous to a statistical parametric mapping).

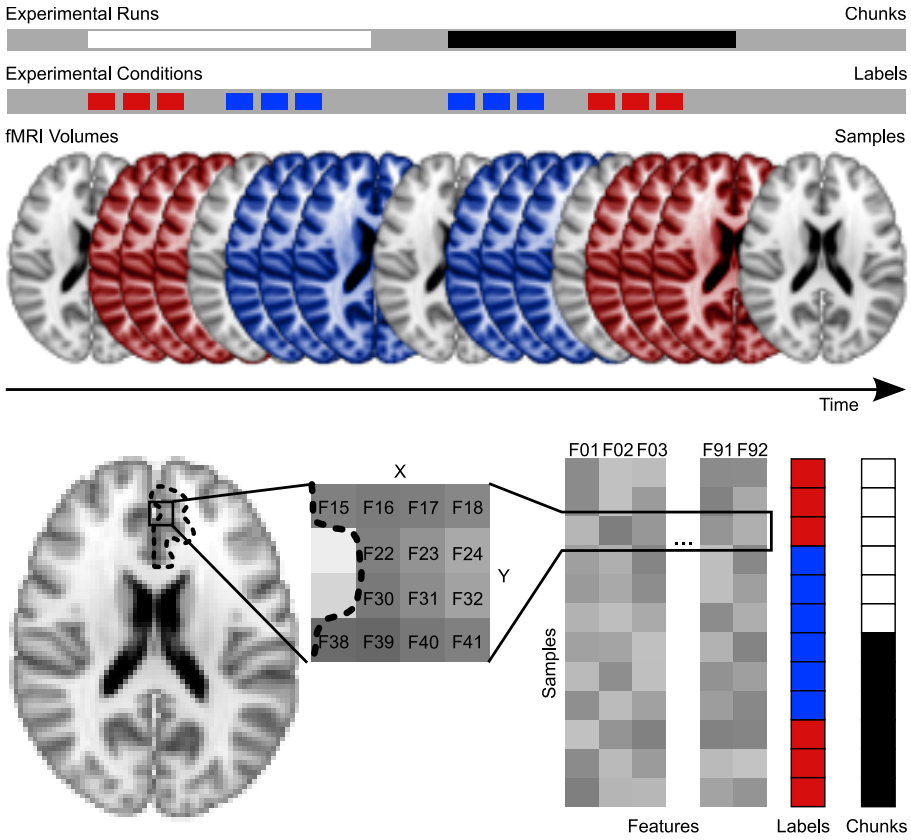

Figure 3.2.: Terminology for MVPA as implemented in PyMVPA. The upper part shows a simple block-design experiment with two experimental conditions (*red* and *blue*) and two experimental runs (*black* and *white*). Experimental runs are referred to as independent *chunks* of data and fMRI volumes recorded in certain experimental conditions are data *samples* with the corresponding condition *labels* attached to them (for the purpose of visualization the axial slices are taken from the MNI152 template downsampled to 3 mm isotopic resolution). The lower part shows an example ROI analysis of that paradigm. All voxels in the defined ROI are considered as *features*. The three-dimensional data samples are transformed into a two-dimensional *samples*×*feature* representation, where each row (sample) of the data matrix is associated with a certain label and data chunk.

across the samples, it is mandatory to take actions to ensure this independence by *e.g.* sufficiently separating training and validation datasets in time. This is typically achieved by splitting an experiment into several runs that are recorded separately. In PyMVPA this type of information can be specified by a special *chunks* sample attribute, where each sample is associated with the numerical identifier of its respective data chunk or run (see Fig. 3.3). However, an arbitrary number of auxiliary sample attributes can be defined in addition to *labels* and *chunks*.

One of the key features of PyMVPA is its ability to read fMRI datasets and transform them into a generic format that makes it easy for other data processing toolboxes to inherit them. Most machine learning software requires data to be represented in a simple two-dimensional $samples \times features$ matrix (see Fig. 3.3, bottom), however, fMRI datasets are typically four-dimensional. Although it is possible to view each volume as a simple vector of voxels, doing so discards information about the spatial properties of the volume samples. This is a potentially serious disadvantage because in the context of brain imaging, spatial metrics, and especially distance information, are of interest. In addition, some analysis algorithms such as the *multivariate searchlight* (Kriegeskorte et al., 2006) make use of this information when calculating spheres of voxels.

PyMVPA follows a different approach. Each dataset is accompanied by a transformation or mapping algorithm that preserves all required information and stores it as a dataset attribute. These mappers allow for bidirectional transformations from the original data space into the generic 2-D matrix representation and vice versa. In the case of fMRI volumes the mapper indexes each feature with its original coordinate in the volume. It can optionally compute customizable distances (*e.g.* Cartesian) between features by taking the voxel size along all three dimensions into account. Using the mapper in the reverse direction, from generic feature space into original data space makes it easy to visualize analysis results. For example, feature sensitivity maps can be easily projected back into a 3-D volume and visualized similar to a statistical parametric map.

PyMVPA comes with a specialized dataset type for handling import from and export to images in the NIfTI format[12]. It automatically configures an appropriate mapper by reading all necessary information from the NIfTI file header. Upon export, all header information is preserved (including embedded transformation matrices). This makes it very easy to do further processing or use the visualization capabilities of any other NIfTI-aware software package, like any of the major fMRI toolkits that have been listed previously.

Since many algorithms are applied only to a subset of voxels, PyMVPA provides convenient methods to select voxels based on ROI masks. Successively applied feature selections will be taken into account by the mapping algorithm of NIfTI datasets and reverse mappings from the new subspace of features into the original dataspace, *e.g.* for visualization, is automatically performed upon request.

However, the mapper construct in PyMVPA, which is applied to each data sample, is more flexible than a simple 3-D data volume to 1-D feature vector transformation.

[12] ANALYZE format is supported as well but it is inferior to NIfTI thus is not explicitly advertised here.

The original dataspace is not limited to three dimensions. For example, when analyzing an experiment using an event-related paradigm it might be difficult to select a single volume that is representative for some event. A possible solution is to select all volumes covering an event in time (as suggested by *e.g.* Mitchell et al., 2004), which results in a four-dimensional dataspace. A mapper can also be easily used for EEG/MEG data, *e.g.* mapping spectral decompositions of the time series from multiple electrodes into a single feature vector. PyMVPA provides convenient methods for these use-cases and also supports reverse mapping of results into the original dataspace, which can be of any dimensionality.

3.4. Machine learning algorithms

3.4.1. Classifier abstraction

PyMVPA itself does not at present implement all possible classifiers, even if that were desirable. Currently included are implementations of a k-nearest-neighbor classifier as well as ridge, penalized logistic, Bayesian linear, Gaussian process (GPR), and sparse multinomial logistic regressions (SMLR; Krishnapuram et al., 2005). However, instead of distributing yet another implementation of popular classification algorithms the toolbox defines a generic classifier interface that makes it possible to easily create software wrappers for existing machine learning libraries and enable their classifiers to be used within the PyMVPA framework. At the time of this writing, wrappers for SVMs of the widely used *LIBSVM* package (Chang & Lin, 2001) and *Shogun* machine learning toolbox (Sonnenburg, Raetsch, Schaefer, & Schoelkopf, 2006) are included. Additional classifiers implemented in the statistical programing language R are provided within PyMVPA, *e.g.* least angle regression (LARS; Efron, Trevor, Johnstone, & Tibshirani, 2004) and elastic net (ENET; Zou & Hastie, 2005). The software wrappers expose as much functionality of the underlying implementation as necessary to allow for a seamless integration of the classification algorithm into PyMVPA. Wrapped classifiers can be treated and behave exactly as any of the native implementations.

Some classifiers have specific requirements about the datasets they can be trained on. For example, SVMs do not provide native support for multi-class problems, *i.e.* discrimination of more than two classes. To deal with this fact, PyMVPA provides a framework to create meta-classifiers (see Fig. 3.1). These are classifiers that utilize several basic classifiers, both those implemented in PyMVPA and those from external resources, that are each trained separately and their respective predictions are used to form a joint meta-prediction, sometimes referred to as *boosting* (see Schapire, 2003). Besides generic multi-class support, PyMVPA provides a number of additional meta-classifiers *e.g.* a classifier that automatically applies a customizable feature selection procedure prior to training and prediction. Another example is a meta-classifier that applies an arbitrary mapping algorithm to the data to implement a data reduction step, such as principal component analysis (PCA), independent component analysis (ICA),

both using implementations from *MDP*[13] or wavelet decomposition via *pywavelets*[14].

Despite its high-level interface PyMVPA offers detailed information to the user. To achieve a useful level of transparency, all classifiers can easily store any amount of additional information. For example, a logistic regression might optionally store the output values of the regression that are used to make a prediction. PyMVPA provides a framework to store and pass this information to the user if it is requested. The type and size of such information is in no way limited. However, if the use of additional computational or storage resources is not required, then it can be switched off at any time, to allow for an optimal tradeoff between transparency and performance.

3.4.2. Feature measures

A primary goal for brain-mapping research is to determine where in the brain certain types of information are processed or which regions are engaged in a specific task. In univariate analysis procedures the localization of information is automatically achieved because each feature is tested independently of all others. In contrast, MVPA-based techniques, however, incorporate information from the entire feature set to *e.g.* determine whether or not a classifier can extract sufficient information in order to predict the experimental condition from the recorded brain activity. Although classifiers can use the joint signal of the whole feature set to perform their predictions, it is nevertheless important to know which features contribute to the classifiers' correct predictions. Some classifiers readily provide information about *sensitivities*, *i.e.* feature-wise scores measuring the impact of each feature on the decision made by the classifier. For example, a simple artificial neural network or a logistic regression, such as SMLR, bases its decisions on a weighted sum of the inputs. Similar weights can also be extracted from any linear classifier including SVMs.

However, there are also *classifier-independent algorithms* to compute featurewise measures. While neural network and SVM weights are inherently multivariate, a feature-wise ANOVA, *i.e.* the fraction of within-class and across class variances, is a univariate measure, as is simple variance or entropy measures of each voxel over all classes. In addition to a simple ANOVA measure PyMVPA provides linear SVM, GPR, LARS, ENET, and SMLR weights as basic feature sensitivities. As with the classifiers discussed in the previous section, a simple and intuitive interface makes it easy to extend PyMVPA with custom measures (*e.g.* information entropy). Among others, the SciPy package provides a large variety of measures that can be easily used within the PyMVPA framework.

PyMVPA provides some algorithms that can be used on top of the basic featurewise measures to potentially increase their reliability. Multiple feature measures can be easily computed for sub-splits of the training data and combined into a single featurewise measure by averaging, t-scoring or rank-averaging across all splits. This might help to stabilize measure estimates if a dataset contains spatially distributed artifacts. While a SPM is rather insensitive to such artifacts as it looks at each voxel individually (Chen,

[13] http://mdp-toolkit.sourceforge.net
[14] http://www.pybytes.com/pywavelets/

Pereira, Lee, Strother, & Mitchell, 2006), classifiers usually pick up such signal if it is related to the classification decision. But, if the artifacts are not equally distributed across the entire experiment, computing measures for separate sub-splits of the dataset can help to identify and reduce their impact on the final measure.

In addition, PyMVPA enables researchers to easily conduct *noise perturbation* analyses, where one measure of interest, such as cross-validated classifier performance, is computed many times with a certain amount of noise added to each feature in turn. Feature sensitivity is then expressed in terms of the difference between computed measures with and without noise added to a feature (see Rakotomamonjy, 2003; Hanson et al., 2004, for equivalence analyses between noise perturbation and simple sensitivities for SVM).

3.5. Workflow abstraction

MVPA-based analyses typically consist of some basic procedures that are independent of the classification algorithm or decision process that was actually used, *e.g.* error calculation, cross-validation of prediction performance, and feature selection. PyMVPA provides support for all of these procedures and, to maximize flexibility, it allows for arbitrary combinations of procedures with any classifiers, featurewise measures, and feature selectors. The two most important procedures are dataset resampling and feature selection.

3.5.1. Dataset resampling

During a typical MVPA a particular dataset has to be resampled several times to obtain an unbiased generalization estimate of a specific classifier. In the simplest case, resampling is done via splitting the dataset, so that some part serves as a validation dataset while the remaining dataset is used to train a classifier. This is done multiple times until a stable estimate is achieved or the particular sampling procedure exhausts all possible choices to split the data. Proper splitting of a dataset is very important and might not be obvious due to the aforementioned forward contamination through the hemodynamic response function. If the strict separation of training and validation datasets was violated, all subsequent analyses would be biased because the classifier might have had access to the data against which it will be validated.

PyMVPA provides a number of resampling algorithms. The most generic one is an N-M splitter where M out of N dataset chunks are chosen as the validation dataset while all others serve as training data until all possible combinations of M chunks are drawn. This implementation can be used for leave-one-out cross-validation, but additionally provides functionality that is useful for bootstrapping procedures (Efron & Tibshirani, 1993). Additional splitters produce first-half-second-half or odd-even splits. Each splitter may base its splitting on any sample attribute. Therefore it is possible to split not just into different data chunks but also *e.g.* into pairs of stimulus conditions.

Most algorithms implemented in PyMVPA can be parameterized with a splitter, making them easy to apply within different kinds of splitting or cross-validation procedures. Like with other parts of PyMVPA, it is trivial to add other custom splitters, due to a common interface definition.

The dataset resampling functionality in PyMVPA also eases non-parametric testing of classification and generalization performances via a data randomization approach, *e.g.* Monte Carlo permutation testing (Nichols & Holmes, 2001). By running the same analysis multiple times with permuted dataset labels (independently within each data chunk) it is possible to obtain an estimate of the baseline or chance performance of a classifier or some sensitivity measure. This allows one to estimate statistical significance (in terms of p-values) of the results achieved on the original (non-permuted) dataset. The reader is kindly referred to section 4.5 for a discussion of the statistical evaluation of classifier accuracies.

3.5.2. Feature selection procedures

As mentioned above, featurewise measure maps can easily be computed with a variety of algorithms. However, such maps alone cannot answer the question of which features are necessary or sufficient to perform some classification. Feature selection algorithms address this question by trying to determine the relevant features based on a featurewise measure. As such, feature selection can be performed in a data-driven or classifier-driven fashion. In a data-driven selection, features could be chosen according to some criterion such as a statistically significant ANOVA score for the feature given a particular dataset, or a statistically significant t-score of one particular weight across splits. Classifier-driven approaches usually involve a sequence of training and validation actions to determine the feature set which is optimal with respect to some classification error (*e.g.* transfer, inherent leave-one-out, theoretical upper-bound) value. It is important to mention that to perform unbiased feature selection using the classifier-driven approach, the selection has to be carried out without observing the main validation dataset for the classifier.

Among the existing algorithms *incremental feature search* (IFS) and *recursive feature elimination* (RFE, Guyon et al., 2002; Guyon & Elisseeff, 2003) are widely used (*e.g.* Rakotomamonjy, 2003) and both are available within PyMVPA. The main differences between these procedures are starting point and direction of feature selection. RFE starts with the full feature set and attempts to remove the least-important features until a stopping criterion is reached. IFS on the other hand starts with an empty feature set and sequentially adds important features until a stopping criterion is reached. The implementations of both algorithms in PyMVPA are very flexible as they can be parameterized with all available basic and meta featurewise measures, and expose any desired amount of the progress and internal state of the computation. In addition, the specifics of the iteration process and the stopping criteria are both fully customizable.

3.6. Demonstrating the high-level interface

An important feature of PyMVPA is that it allows, by design, researchers to compress complex analyses into a small amount of code. This makes it possible to complement publications with the source code actually used to perform the analysis as supplementary material (a feature that has been demonstrated in Hanke et al., 2009). Making this critical piece of information publicly available allows for in-depth reviews of the applied methods on a level well beyond what is possible with verbal descriptions. Although such an approach is generally desirable in the scientific context, it is of particular importance in an emerging field, such as the application of MVPA to neural data, as has been argued in section 2.4.

To demonstrate PyMVPA's high-level interface the following sections provide a few examples of common analysis steps and their actual implementation in PyMVPA.

3.6.1. Loading a dataset

Dataset representation in PyMVPA builds on NumPy arrays. Anything that can be converted into such an array can also be used as a dataset source for PyMVPA. Possible formats range from various plain text formats to binary files. However, the most important input format from the functional imaging perspective is NIfTI[15], which PyMVPA supports with a specialized module.

The following short source code snippet demonstrates how a dataset can be loaded from a NIfTI image. Among various other approaches PyMVPA supports reading the sample attributes from a simple two-column text file that contains a line with a label and a chunk id for each volume in the NIfTI image (line 0). To load the data samples from a NIfTI file it is sufficient to create a `NiftiDataset` object with the filename as an argument (line 1). The previously-loaded sample attributes are passed to their respective arguments as well (lines 2-3). Optionally, a mask image can be specified (line 4) to easily select a subset of voxels from each volume based on the non-zero elements of the mask volume. This would typically be a mask image indicating brain and non-brain voxels.

```
0 attr = SampleAttributes('sample_attr_filename.txt')
1 dataset = NiftiDataset(samples='subj1_bold.nii.gz',
2                        labels=attr.labels,
3                        chunks=attr.chunks,
4                        mask='subj1_roi_mask.nii.gz')
```

Once the dataset is loaded, successive analysis steps, such as feature selection and classification, only involve passing the `dataset` object to different processing objects. All following examples assume that a dataset was already loaded.

[15] To a certain degree PyMVPA also supports importing ANALYZE files.

3.6.2. Simple full-brain analysis

The first analysis example shows the few steps necessary to run a simple cross-validated classification analysis. After a dataset is loaded, it is sufficient to decide which classifier and type of splitting shall be used for the cross-validation procedure. Everything else is automatically handled by `CrossValidatedTransferError`. The following code snippet performs the desired classification analysis via leave-one-out cross-validation. Error calculation during cross-validation is conveniently performed by `TransferError`, which is configured to use a linear C-SVM classifier[16] on line 6. The leave-one-out cross-validation type is specified on line 7.

```
5 cv = CrossValidatedTransferError(
6           transfer_error=TransferError(LinearCSVMC()),
7           splitter=NFoldSplitter(cvtype=1))
8 mean_error = cv(dataset)
```

Simply passing the dataset to `cv` (line 8) yields the mean error. The computed error defaults to the fraction of incorrect classifications, but an alternative error function can be passed as an argument to the `TransferError` call. If desired, more detailed information is available, such as a confusion matrix based on all the classifier predictions during cross-validation.

3.6.3. Feature selection

Feature selection is a common preprocessing step that is also routinely performed as part of a conventional fMRI data analysis, *i.e.* the initial removal of non-brain voxels. This basic functionality is provided by `NiftiDataset` as it was shown on line 4 to provide an initial operable feature set. Nevertheless, PyMVPA provides additional means to perform feature selection, which are not specific to the fMRI domain, in a transparent and unified way.

Machine learning algorithms often benefit from the removal of noisy and irrelevant features (see Guyon et al., 2002, Section V.1. "The features selected matter more than the classifier used"). Retaining only features relevant for classification improves learning and generalization of the classifier by reducing the possibility of overfitting the data. Therefore, providing a simple interface to feature selection is critical to gain superior generalization performance and get better insights about the relevance of a subset of features with respect to a given contrast.

The `FeatureSelectionClassifier` in PyMVPA offers an easy way to perform feature selections. It is a meta-classifier that enhances any other classifier with an arbitrary initial feature selection step. This approach is very flexible as the resulting classifier can be used as any other classifier, *e.g.* for unbiased generalization testing using `CrossValidatedTransferError`. For instance, the following example shows a classifier that operates only on 5% of the voxels that have the highest ANOVA score across the

[16] LIBSVM C-SVC (Chang & Lin, 2001) with trade-off parameter C being a reciprocal of the squared mean of Frobenius norms of the data samples.

data categories in a particular dataset. It is noteworthy that the source code looks almost identical to the example given on lines 5-8, with just the feature selection method added to it. No changes are necessary for the actual cross-validation procedure.

```
9  clf = FeatureSelectionClassifier(
10   clf=LinearCSVMC(),
11   feature_selection=SensitivityBasedFeatureSelection(
12     sensitivity_analyzer=OneWayAnova(),
13     feature_selector=
14       FractionTailSelector(0.05, mode='select', tail='upper'))
15 cv = CrossValidatedTransferError(
16   transfer_error=TransferError(clf),
17   splitter=NFoldSplitter(cvtype=1))
18 mean_error = cv(dataset)
```

It is important to emphasize that feature selection (lines 11-13) in this case is *not* performed first on the full dataset, which could bias generalization estimation. Instead, feature selection is being performed as a part of classifier training, thus, only the actual training dataset is visible to the feature selection. Due to the unified interface, it is possible to create a more sophisticated example, where feature selection is performed via *recursive feature elimination* (Guyon et al., 2002; Guyon & Elisseeff, 2003):

```
19 rfesvm = LinearCSVMC()
20 clf = SplitClassifier(
21   FeatureSelectionClassifier(
22     clf=rfesvm,
23     feature_selection=RFE(
24       sensitivity_analyzer=
25         LinearSVMWeights(clf=rfesvm,
26                          transformer=Absolute),
27       transfer_error=TransferError(rfesvm),
28       stopping_criterion=FixedErrorThresholdStopCrit(0.05),
29       feature_selector=
30         FractionTailSelector(0.2, mode='discard', tail='lower'),
31       update_sensitivity=True)),
32   splitter=NFoldSplitter())
```

On line 19 the main classifier that is reused in many aspects of the processing is defined: line 22 specifies that classifier to be used to make the final prediction operating only on the selected features, line 25 instructs the sensitivity analyzer to use it to provide sensitivity estimates of the features at each step of recursive feature elimination, and on line 27 it is specified that the error used to select the best feature set is a generalization error of that same classifier. Utilization of the same classifier for both the sensitivity analysis and for the transfer error computation prevents us from re-training a classifier twice for the same dataset.

The fact that the RFE approach is classifier-driven requires us to provide the classifier with two datasets: one to train a classifier and assess its features sensitivities and the other one to determine stopping point of feature elimination based on the trans-

fer error. Therefore, the `FeatureSelectionClassifier` (line 21) is wrapped within a `SplitClassifier` (line 20), which in turn uses `NFoldSplitter` (line 32) to generate a set of data splits on which to train and test each independent classifier. Within each data split, the classifier selects its features independently using RFE by computing a generalization error estimate (line 27) on the internal validation dataset generated by the splitter. Finally, the `SplitClassifier` uses a customizable voting strategy (by default *MaximalVote*) to derive the joint classification decision.

As before, the resultant classifier can now simply be used to obtain an unbiased generalization estimate of the trained classifier within `CrossValidatedTransferError` . The step of validation onto independent validation dataset is often overlooked by the researchers performing RFE (Guyon et al., 2002). That leads to biased generalization estimates, since otherwise internal feature selection of the classifier is driven by the full dataset.

Fortunately, some machine learning algorithms provide an internal theoretical upper bound on the generalization performance, thus they could be used as a `transfer_error` criterion (line 27) with RFE, which eliminates the necessity of additional splitting of the dataset. Some other classifiers perform feature selection internally (*e.g.* SMLR, also see figure 4.16), which removes the burden of external explicit feature selection and additional data splitting.

The next chapter offers additional information about the feature selection facilities of PyMVPA and analysis examples on actual neural datasets.

3.7. Paving the way for solid, community-driven methods development

The PyMVPA projects aims for a community-driven development to ensure high-quality software and the timely inclusion of newly developed analysis techniques. A sufficiently large developer base is a critical requirement to extend the lifetime of such a project beyond the scope of a dissertation project. While attracting developers is to a certain degree subject to "social engineering" a project nevertheless has to provide the technical means to scale well with an increasing number of developers.

But first of all it has to provide functionality that outweighs the negative side-effects of participating in a multi-developer project, as opposed to individual development, such as the hassles of dealing with divergent opinions. PyMVPA achieves that by its modularity, that allows to independently add novel algorithms without having to worry about data import and export or even storage of analysis results. In PyMVPA, each building block (*e.g.* all classifiers) follows a simple, standardized, interface. This allows one to use various types of classifiers interchangeably, without additional changes in the source code, and makes it easy to test the performance of newly developed algorithms on one of the many didactical neuroscience-related examples and datasets that are included in PyMVPA. In addition, any implementation of an analysis method/algorithm benefits from the basic *house-keeping* functionality done by the base classes, reducing

the necessary amount of code needed to contribute a new fully-functional algorithm. PyMVPA takes care of hiding implementation-specific details, such as a classifier algorithm provided by an external C++ library. At the same time it tries to expose all available information (*e.g.* classifier training performance) through a consistent interface (for reference, this interface is called *states* in PyMVPA).

As it has already been mentioned, PyMVPA makes use of a number of external software packages, including other Python modules and low-level libraries (*e.g. LIBSVM*[17]) and computing environments (*e.g. R*[18]). Using externally developed software instead of reimplementing algorithms has the advantage of a larger developer and user base and makes it more likely to find and fix bugs in a software package to ensure a high level of quality. However, using external software also carries the risk of breaking functionality when any of the external dependencies break. To address this problem PyMVPA utilizes an automatic testing framework performing various types of tests ranging from unittests (currently covering 84% of all lines of code) to sample code snippet tests in the manual and the source code documentation itself to more evolved "real-life" examples. This facility allows one to test the framework within a variety of specific settings, such as the unique combination of program and library versions found on a particular user machine.

At the same time, the testing framework also significantly eases the inclusion of code by a novel contributor by catching errors that would potentially break the project's functionality. Being open-source does not always mean *easy to contribute* due to various factors such as a complicated application programing interface (API) coupled with undocumented source code and unpredictable outcomes from any code modifications (bug fixes, optimizations, improvements). PyMVPA welcomes contributions, and thus, addresses all the previously mentioned points:

Accessibility of source code and documentation All the source code (including website and examples) together with the full development history is publicly available via a distributed version control system[19] which makes it very easy to track the development of the project, as well as to develop independently and to submit back into the project.

Inplace code documentation Large parts of the source code are well documented using *reStructuredText*[20], a lightweight markup language that is highly readable in source format as well as being suitable for automatic conversion into HTML or PDF reference documentation. In fact, *Ohloh.net*[21] source code analysis judges PyMVPA as having "extremely well-commented source code".

Developer guidelines A brief summary defines a set of coding conventions to facilitate uniform code and documentation look and feel. Automatic checking of compli-

[17] http://www.csie.ntu.edu.tw/~cjlin/libsvm/
[18] http://www.r-project.org
[19] http://en.wikipedia.org/wiki/Version_control_system
[20] http://en.wikipedia.org/wiki/ReStructuredText
[21] http://www.ohloh.net/projects/pymvpa/factoids

ance to a subset of the coding standards is provided through a custom *PyLint*[22] configuration, allowing early stage minor bug catching.

Moreover, PyMVPA does not raise barriers by being limited to specific platforms. It could fully or partially be used on any platform supported by Python (depending on the availability of external dependencies). However, to improve the accessibility, binary installers for Windows, and MacOS X, as well as binary packages for Debian GNU/Linux (included in the official repository), Ubuntu, and a large number of RPM-based GNU/Linux distributions, such as OpenSUSE, RedHat, CentOS, Mandriva, and Fedora are provided. Additionally, the available documentation provides detailed instructions on how to build the packages from source on many platforms.

[22] http://www.logilab.org/projects/pylint

4. Analysis Strategies

Previous chapters have reviewed the general, rather theoretical advantages of the application of MVPA to neuroimaging data, and also have introduced a new software framework that aims to provide scientists with the means to quickly adopt and evaluate these powerful new methods in their own research.

The main focus of the current chapter is now to illustrate this power with some concrete examples that show the flexibility of the MVPA approach. To this end a series of analyses will be presented that cover a wide variety of experimental paradigms, ranging from object perception, over auditory information processing, to memory-related research. Several different datasets will be the subject of these analyses to point out potentially interesting processing schemes and layout a set of valid and informative procedures. The selection of datasets includes block-design as well as event-related fMRI data. Moreover, due to the aforementioned absence of required a priori models, MVPA analyses of other data modalities are equally possible and fruitful, therefore similar methods will be applied to data from extracellular recordings, EEG, and MEG as well. All analysis steps including pre-processing of the data have been performed with PyMVPA, unless otherwise noted[1].

To begin with a set of analysis strategies will be discussed that primarily focus on interpreting the *direct quantifiable link* between brain response patterns and experimental conditions (O'Toole et al., 2007) and so far make up the majority of applied methods in the studies that have been published in this field.

However, the second part of the chapter will introduce a *sensitivity analysis* scheme that offers alternative information sources that are readily available as part of any MVPA study, but haven't received much attention in the literature so far.

What is not going to be *discussed*, however, are many possible decisions to be made when applying MVPA, such as which type of classifier to use, or which preprocessing to apply. Of course the actual analysis parameters will be specified, but no attempt is made to promote a particular choice as superior to an alternative. For a comprehensive overview about important aspects that should be considered when such decisions have to be made (*i.e.* whether or not a non-linear classifier should be employed), the reader is kindly refered to Pereira et al. (2009) for a comprehensive tutorial overview about these topics.

[1] Note that PyMVPA internally makes use of a number of other aforementioned Python modules, such as NumPy and SciPy.

4.1. Labeling functional systems by prediction performance

It has been mentioned previously that one of the primary features of MVPA is that it provides a direct quantifiable link between brain response patterns and the experimental design. This link is the accuracy measure of a trained classifier predicting the experimental condition of new data samples. A high accuracy indicates that a classifier extracted a multivariate model that appropriately captures the underlying signal, and does not suffer from severe overfitting of the specific noise pattern in the training data. The implication with respect to the original research question is that somewhere in the dataset there has to be enough information of some kind to allow for this prediction to be made. However, quite often, if not always, the actual research question deals with either specifically localizing this information, or characterizing its structure, or both. Hence, researchers typically need to go beyond the level of reliable predictions and need to characterize the cause for this situation. The following sections introduce strategies that use the generalization accuracy as the primary dependent variable.

4.1.1. Classify and dissect

The first analysis strategy that shall be introduced has been labeled as "classify and dissect" by O'Toole et al. (2007). For this approach a classifier is trained on data from a, possibly hypothesis-led, selection of ROIs first. The critical requirement for this analysis is that the classifier is able to perform reliable predictions for generalization test datasets. To investigate the sources of the signals that contribute to the classification each ROI is excluded from the training data. Afterwards a new classifier is trained on the reduced dataset, and its prediction accuracy is compared to the initial classifier, trained on the full dataset. A reduction in the accuracy of the reduced classifier might be interpreted as that with the exclusion of the particular ROI a relatively important chunk of information is no longer available, hence the reduction of the accuracy is used as a measure of importance.

Pessoa & Padmala (2007) offer an application example of this strategy, who aimed to determine the emotional content of brief visual presentations of face (*i.e.* fearful vs. neutral faces) from single-trial fMRI data employing a slow event-related paradigm. They identified a set of ROIs based on results available from previous studies, performed a "virtual lesioning" analysis using the procedure outlined above, and were able to identify a set of brain areas (including the amygdala) that contribute to the perception of the emotional content of the stimulus faces (Fig. 4.1).

However, this approach to deduce the function or even just the contribution of brain areas with respect to a specific cognitive task, implies at least one problem. Excluding one ROI from the training dataset at a time might not necessarily lead to a reduction in the prediction accuracy even if a particular ROI contributes meaningful signal with respect to the classification task. That might happen, for example, if multiple ROIs contain redundant information, or even when that information is not redundant, but

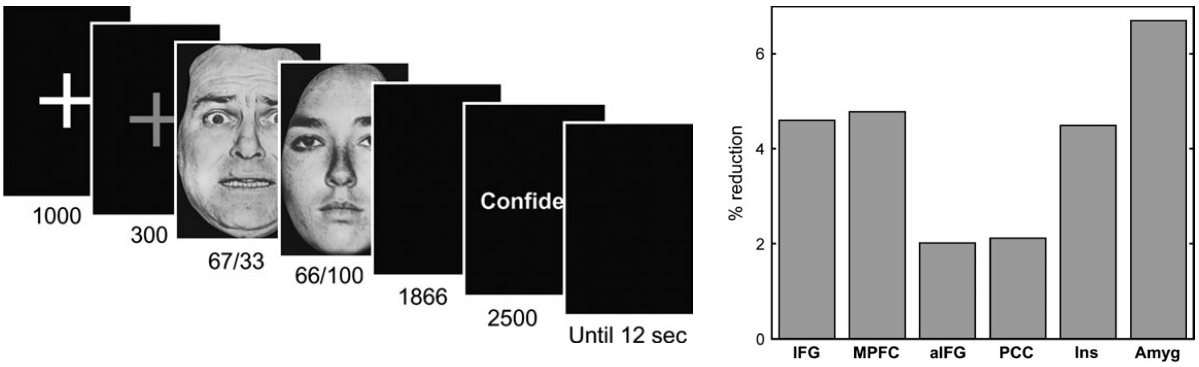

Figure 4.1.: Assessing the involvement of brain areas in the perception of near-threshold fear. Pessoa & Padmala (2007) trained a classifier on data from multiple ROIs to distinguish between trials with masked fear-related and neutral stimuli. The multivariate model was then inspected with respect to the contribution of individual ROIs by successively excluding each of them from the training dataset. ROIs were then scored by the reduction of the achieved generalization accuracy of the reduced model with respect to the full model including all ROIs, where an increasing reduction was considered as stronger involvement in the processing. Reprinted from Pessoa & Padmala (2007)

complementary yet each subset alone is sufficient to allow for a high prediction accuracy. Therefore, using this analysis strategy requires additional in-depth analysis of the extracted multivariate models to be able to interpret the type of contribution of a specific brain area. Several possibilities how such an in-depth analysis can be done are presented in the remainder of this chapter.

4.1.2. Leaving the voxel-space: Brain response components

One of the possibilities to analyze brain response patterns mentioned in chapter 2 is to investigate them in terms of components as estimated by PCA/ICA algorithms instead of voxels, or ROIs of those. However, the labeling issue, *i.e.* to decide whether a particular components represents a meaningful signal, or noise, was listed as the primary factor limiting the usefulness of this approach.

Classifiers can help to address this issue by assigning a measure to each component that reflects how informative it is with respect to a particular classification. It is basically irrelevant for a classifier whether its input features originate from voxel signal timecourses, or from the temporal profiles of a spatial ICA components. Therefore it is equally possible to apply the previously outlined "classify and dissect" strategy to datasets that have been transformed into component space. However, this time not for certain spatial ROIs, but for individual components, *i.e.* features.

It is worth mentioning that transforming an fMRI dataset into a component representation causes a dramatic reduction of the dimensionality, hence often being labeled as *data-compression*. For example, a spatial ICA of a dataset with 30k voxels and 500 volumes would yield 500 components (*i.e.* features), which only 1.7% of the original number

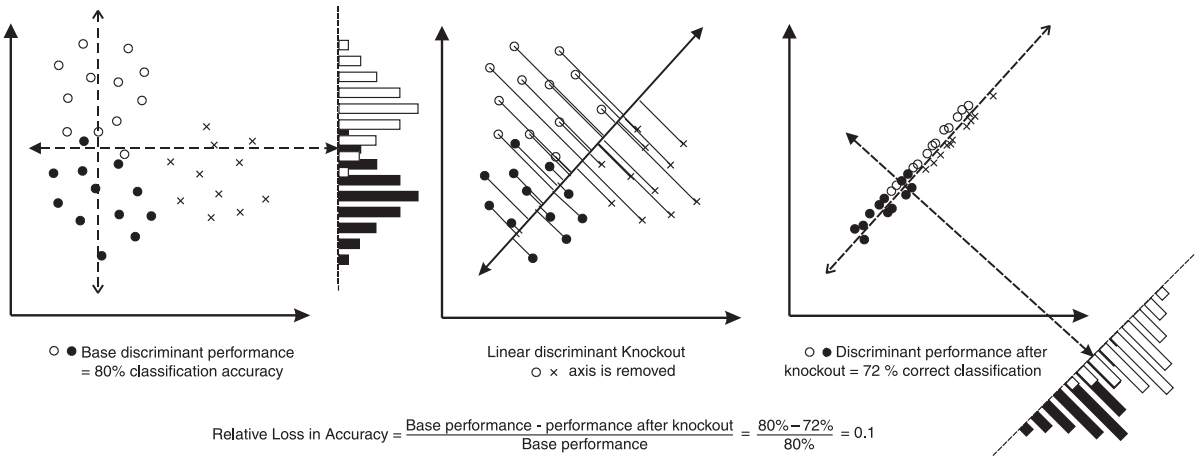

Figure 4.2.: Carlson et al. (2003) proposed a hierarchical knockout procedure to investigate the importance of brain response components. This strategy successively removes the axes of the component space, and projects the original data points into the reduced space. Removing an informative component yields an impaired prediction accuracy, where the relative loss of information can be used to quantify the contribution. Reprinted from Carlson et al. (2003)

of feature. Such a reduction also typically also allows to employ classical mutlivariate algorithms, such as LDA.

Carlson, Schrater, & Sheng (2003) proposed a slightly different strategy that accounts for the ordering of the estimated PCA/ICA components. They suggest a "knockout" procedure that starts with the full set of components, and successively removes them from the dataset, starting with those explaining most of the variance. The importance of each component is now expressed as the relative loss of generalization accuracy when excluding that component from the dataset (Fig. 4.2). In contrast to the previous strategy, that analysis approach is implemented as a hierarchical procedure where previously excluded components are not put back into the dataset.

However, it has been mentioned before that the ordering of the components might not be very useful in terms of the experimental design (with *i.e.* motion artifacts explaining substantial proportions of the signal variance, in contrast to the tiny changes that are due to the BOLD-response). Therefore, both the hierarchical exclusion strategy, as well as the "classify and dissect" approach might be suboptimal. It would be very useful to have an analysis procedure that is able to assign importance-scores to all features simultaneously. Fortunately, such a strategy will be introduced in section 4.2.

4.1.3. Multivariate searchlight

An alternative algorithm for information localization, again operating in voxel-space, not on components, was proposed by Kriegeskorte et al. (2006). The basic idea is to scan the whole brain by automatically running multiple ROI analyses. The so-called *multivariate searchlight* does a MVPA on spherical ROIs of a given radius centered around any voxel

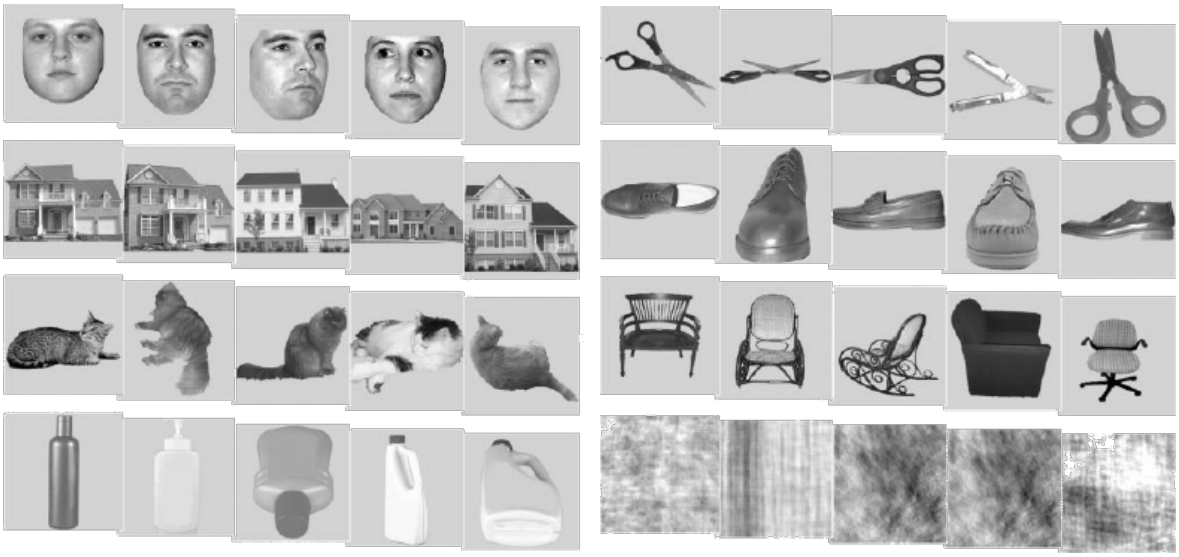

Figure 4.3.: Example set of the stimuli used in Haxby et al. (2001).

covering brain matter. A searchlight analysis computing *e.g.* generalization accuracies yields a map showing where in the brain a relevant signal can be identified while still harnessing the power of multivariate techniques (for application examples see Haynes et al., 2007; Kriegeskorte, Formisano, Sorger, & Goebel, 2007).

A searchlight performs well if the target signal is available within a relatively small area. By increasing the size of the searchlight the information localization becomes less specific because, due to the anatomical structure of the brain, each spherical ROI will contain a growing mixture of grey-matter, white-matter, and non-brain voxels. Additionally, a searchlight operating on volumetric data will integrate information across brain-areas that are not directly connected to each other *i.e.* located on opposite borders of a sulcus. This problem can be addressed by running a searchlight on data that has been transformed into a surface representation. PyMVPA supports analyses with spatial searchlights (not extending in time), operating on both volumetric and surface data (given an appropriate mapping algorithm and using circular patches instead of spheres). The searchlight implementation can compute arbitrary measures within the spheres.

For the following exemplary application of a searchlight analysis data of a single participant (participant 1) from the now classical study by Haxby et al. (2001) are used. This dataset was chosen because, since its first publication, it has been repeatedly reanalyzed (Hanson et al., 2004; O'Toole et al., 2007; Hanson & Halchenko, 2008) and parts of it also serve as an example dataset of the Matlab-based MVPA toolbox (Detre et al., 2006). This dataset will also be subject of further analyses in the following sections.

The dataset itself is a block-design fMRI experiment consisting of 12 runs. In each run, the participant passively viewed greyscale images of eight object categories (Fig. 4.3), grouped in 24 s blocks, separated by rest periods. Each image was shown for 500 ms and followed by a 1500 ms inter-stimulus interval. Full-brain fMRI data were recorded with a volume repetition time of 2500 ms, thus, a stimulus block was covered by roughly 9

volumes. For a complete description of the experimental design and fMRI acquisition parameters see Haxby et al. (2001).

Prior to any analysis, the raw fMRI data were motion corrected using FLIRT from FSL (Jenkinson et al., 2002). After motion correction, linear detrending was performed for each run individually by fitting a straight line to each voxels timeseries and subtracting it from the data. No additional spatial or temporal filtering was applied.

For the sake of simplicity, the binary CATS vs. SCISSORS classification problem was selected for this example. All volumes recorded during either CATS or SCISSORS blocks were extracted and voxel-wise z-scored with respect to the mean and standard deviation of volumes recorded during rest periods. Z-scoring was performed individually for each run to prevent any kind of information transfer across runs.

Because the actual source code to perform a searchlight analysis in PyMVPA is so short it is printed in its full length below.

```
1 cv = CrossValidatedTransferError(
2           transfer_error=TransferError(LinearCSVMC()),
3           splitter=OddEvenSplitter())
4 sl = Searchlight(cv, radius=5)
5 sl_map = sl(dataset)
6 dataset.map2Nifti(sl_map).save('searchlight_5mm.nii.gz')
```

In the source code the measure to be computed by the searchlight is configured first. Similar to the code examples in the previous chapter it is a cross-validated transfer or generalization error, but this time it will be computed on a run-wise odd-even split of the dataset and with a linear C-SVM classifier (lines 1-3). On line 4 the searchlight is set up to compute this measure in all possible 5 mm-radius spheres when called with a dataset (line 5) The final call on line 6 transforms the computed error map back into the original data space and stores it as a compressed NIfTI file. Such a file can then be viewed and further processed with any NIfTI-aware toolkit.

Figure 4.4 shows the searchlight error maps for the CATS vs. SCISSORS classification on single volumes from the example dataset for radii of 1, 5, 10 and 20 mm respectively. Utilizing only a single voxel in each sphere (1 mm radius), yields a generalization error as low as 17% in the best performing sphere, which is located in the left occipito-temporal fusiform cortex. With increases in the radius there is a tendency for further error reduction, indicating that the classifier performance benefits from integrating signal from multiple voxels. However, better classification accuracy is achieved at the cost of reduced spatial precision of signal localization. The distance between the centers of the best-performing spheres for 5 and 20 mm searchlights totals almost 18 mm. The lowest overall error in the right occipito-temporal cortex with 8% is achieved by a searchlight with a radius of 10 mm. The best performing sphere with 20 mm radius (12% generalization error) is centered between right inferior temporal and fusiform gyrus. It comprises approximately 700 voxels and extends from right lingual gyrus to the right inferior temporal gyrus, also including parts of the cerebellum and left lateral ventricle. It, therefore, includes a significant proportion of voxels sampling cerebrospinal fluid or white matter, indicating that a sphere of this size is not optimal given the structural organization of

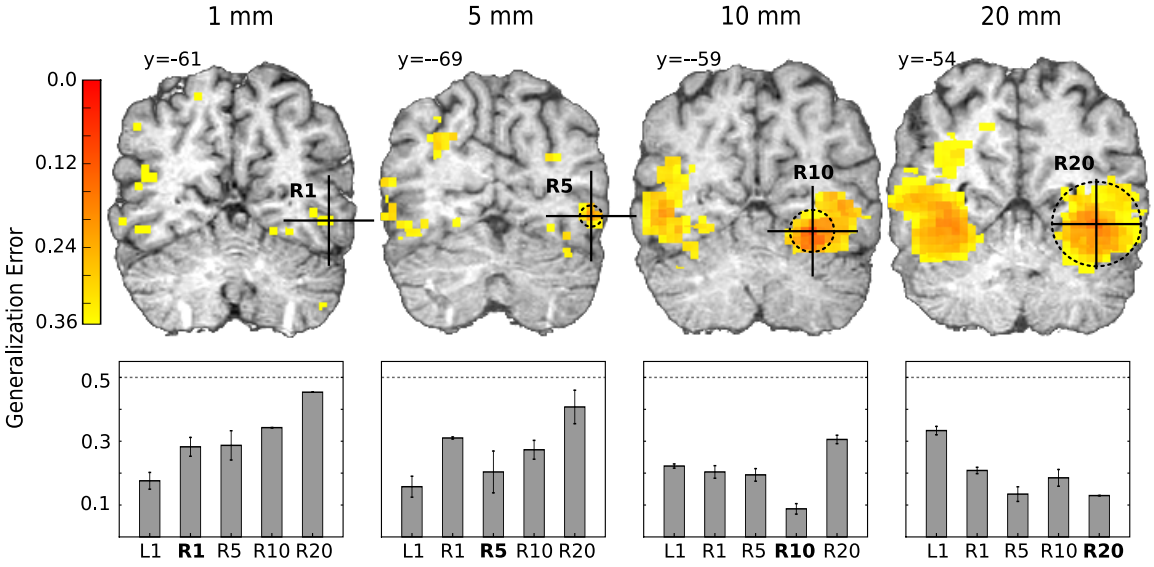

Figure 4.4.: Searchlight analysis results for CATS vs. SCISSORS classification for sphere radii of 1, 5, 10 and 20 mm (corresponding to approximately 1, 15, 80 and 675 voxels per sphere respectively). The upper part shows generalization error maps for each radius. All error maps are thresholded arbitrarily at 0.35 (chance level: 0.5) and are not smoothed to reflect the true functional resolution. The center of the best performing sphere (*i.e.* lowest generalization error) in right temporal fusiform cortex or right lateral occipital cortex is marked by the cross-hair on each coronal slice. The dashed circle around the center shows the size of the respective sphere (for radius 1 mm the sphere only contains a single voxel). MNI-space coordinates (x, y, z) in mm for the four sphere centers are: 1 mm (*R1*): (48, -61, -6), 5 mm (*R5*): (48, -69, -4), 10 mm (*R10*): (28, -59, -12) and 20 mm (*R20*): (40, -54, -8). The lower part shows the generalization errors for spheres centered around these four coordinates, plus the location of the univariately best performing voxel (*L1*: -35, -43, -23; left occipito-temporal fusiform cortex) for all radii. The error bars show the standard error of the mean across cross-validation folds.

the brain surface. Kriegeskorte et al. (2006) suggest that a sphere radius of 4 mm yields near-optimal performance. However, while this assumption might be valid for representations of object properties or low-level visual features, a searchlight of this size could miss signals related to high-level cognitive processes that involve several spatially distinct functional subsystems of the brain. However, there are nevertheless studies trying to employ a searchlight to identify complex cognitive signals, such as hidden intentions (Haynes et al., 2007).

To summarize, a searchlight represents a relatively simple way to localize regions whos brain response patterns contain relevant information. However, due to its assumptions about the spatial layout and size of the targeted regions, it is limited in its ability to detect signals that are encoded in the interaction of spatially distinct areas. Reducing the dataset to just a few features from a local neighborhood might nevertheless be necessary, if one wants to apply classical statistical algorithms, such as LDA to datasets with a low number of observations. However, classifier algorithms such as SVMs are easily able to deal with full-brain datasets. Especially in conjunction with more flexible feature selection strategies (*i.e.* based on classifier weights), they are able to achieve impressive prediction accuracies, hence estimating multivariate models that appropriately capture the underlying signal in the dataset.

Table 4.1 shows the prediction error of a variety of classifiers trained on the full-brain dataset with and without any prior feature selection. Most of the classifiers perform near chance performance without prior feature selection[2], but even simple feature selection (*e.g.* some percentage of the population with highest scores on some measure) boosts generalization performance significantly of all classifiers, including the non-linear algorithms radial basis function SVM, and kNN.

Inspecting which features are preserved during the feature selection stage, again provides inherent localization information. Moreover, the stability of the selection patterns also provide information about the variability of the underlying signal – a topic that will be discussed in section 4.5. However, more versatile measures than binary selection maps are available. The next section will provide some hints on how the MVPA model parameters itself can be interpreted in meaningful ways.

4.2. More information is available: Sensitivity analysis

Hanson et al. (2004) trained a multi-layer artificial neural network on the data from Haxby et al. (2001) (the same dataset that has been the subject of the analysis example in the previous section). By inspecting the weights of the trained neural network they were able to provide convincing evidence in favor of a distributed combinatorial coding of object category related information in the fMRI data. Such a *sensitivity analysis*, *i.e.* the analysis of the multivariate model parameters is easily possible with any linear classifier

[2] Chance performance without feature selection was not the norm for all category pairs in the dataset. For example, the SVM classifier generalized well for other pairs of categories (*e.g.* FACE vs HOUSE) without prior feature selection. Consequently, SCISSORS vs CATS was chosen to provide a more difficult analysis case.

Table 4.1.: Performance of various classifiers with and without feature selection. For classifiers with feature selection the classifier algorithm is followed by the sensitivity measure the feature selection was based on (*e.g.* LinSVM on 50(SVM) reads: linear SVM classifier using 50 features selected by their magnitude of weight from a trained linear SVM).

Classifier	Training			Transfer	
	Features utilized	Error	Time (sec)	Error (±stderr)	Time (sec)
Without feature selection					
LinSVM(C=def)	29125	0.00	22.7	0.40±0.07	2.0
LinSVM(C=10*def)	29125	0.00	22.7	0.36±0.07	2.0
LinSVM(C=1)	29125	0.00	22.6	0.36±0.07	2.0
RbfSVM()	29125	0.00	23.8	0.50±0.07	2.1
kNN()	29125	0.02	0.0	0.44±0.03	2.9
With feature selection					
SMLR(lm=0.1)	314	0.00	19.0	0.09±0.03	0.1
SMLR(lm=1.0)	92	0.00	5.0	0.11±0.03	0.1
SMLR(lm=10.0)	42	0.00	2.4	0.09±0.03	0.1
RbfSVM on SMLR(lm=10) non-0	42	0.00	2.5	0.11±0.02	0.0
kNN on 5%(ANOVA)	1456	0.00	0.8	0.28±0.05	0.2
kNN on 50(ANOVA)	50	0.01	0.8	0.07±0.02	0.0
kNN on SMLR(lm=10) non-0	42	0.00	2.5	0.12±0.02	0.0
LinSVM on 5%(SVM)	1456	0.00	23.1	0.18±0.04	0.1
LinSVM on 50(SVM)	50	0.00	22.7	**0.03**±0.02	0.0
LinSVM on 5%(ANOVA)	1456	0.00	1.6	0.13±0.04	0.1
LinSVM on 50(ANOVA)	50	0.00	0.8	0.09±0.03	0.0
LinSVM on SMLR(lm=1) non-0	92	0.00	5.3	0.08±0.02	0.0
LinSVM on SMLR(lm=10) non-0	42	0.00	2.5	0.12±0.03	0.0
LinSVM+RFE(N-Fold)	4587	0.00	2010.0	0.12±0.03	3.4
LinSVM+RFE(OddEven)	42	0.09	260.9	0.24±0.04	0.0

(in contrast non-linear sensitivities, such as those from radial basis function SVMs are much harder to interpret (see *e.g.* Kienzle, Schölkopf, Wichmann, & Franz, 2007, for an analysis approach). An interesting example is the investigation of the spatio-temporal structure of a multivariate model, which is going to be introduced in this section.

It has been already mentioned in chapter 3 that when analyzing event-related fMRI datasets it might be hard to decide which volume during an event should be selected as the "representative" brain response sample for that particular trial. There are several possibilities to make this decision. First, considering the temporal properties of the BOLD-response, one could argue that the peak amplitude is probably reached 5-6 s after stimulus onset. Simply choosing the volume that is closest in time would yield a reasonable candidate. However, it has been noted earlier that there is substantial variation in the timing of the BOLD-response across subjects, brain areas, and experimental paradigms. To account for a certain variability some studies average a number of successively recorded volumes, aiming to capture the peak response, and at the same time seek to boost the signal-to-noise ratio (see *e.g.* Pessoa & Padmala, 2007; Haynes et al., 2007, for applications). But volume averaging inevitably destroys the temporal structure of the fMRI signal over the course of a trial, and might significantly demolish the available signal, especially for relatively fast event-related designs.

The analysis approach that is proposed here is centered around the idea to explicitly make use of the full spatio-temporal structure of the fMRI signal, and use the powerful classifier algorithms to determine the relevant information, instead of selectively excluding information based on a priori assumptions. The dataset that will be used to demonstrate the method has not been published previously, therefore a short summary of the employed paradigm and recording parameters will be given first.

4.2.1. Event-related example dataset

fMRI datasets from eleven participants have been recorded while they were performing an object categorization task. Colored images of computer-generated faces and photographs of shoes were presented to the participants. All object stimuli were displayed centered on top of a rectangular area that subtended $7.2° \times 4.8°$ degrees of visual angle and consisted of a phase scrambled version of the actual stimulus itself (Fig. 4.5). This was done to equalize the stimuli with respect to their visual appearance (*e.g.* shape outline).

The participants had to categorize each stimulus with respect to one of two possible aspects. Either they had to make a two-alternative forced-choice response concerning the *width* (*i.e.* slim vs. wide) or the *gender* (*i.e.* male vs. female) of the presented image. Each trial began with the presentation of a stimulus for two seconds. Afterwards the stimulus was replaced by a new phase-scrambled image without an object on top of it. This "filler" image was replaced by a new one after another two seconds, and a total of three filler images were presented in each trial, hence yielding a trial length of eight seconds. Participants were asked to respond within six seconds after trial onset by a button-press with their right hand and no feedback was given to them. Trials were grouped into mini blocks consisting of five trials of each stimulus condition (5×4

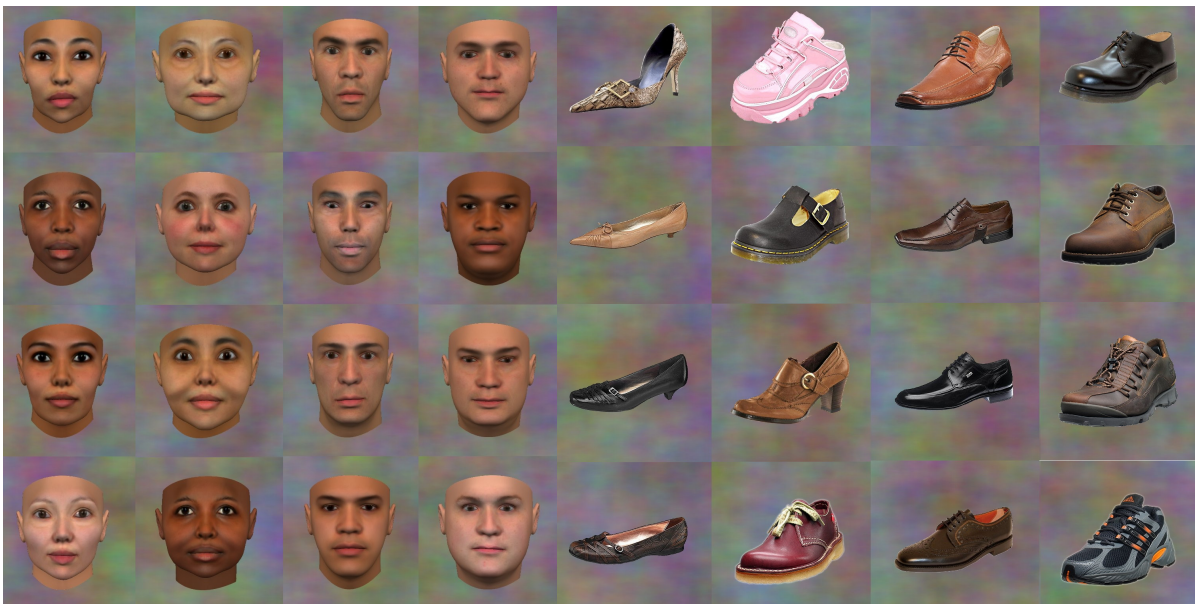

Figure 4.5.: Computer-generated stimulus examples. The columns show examples of slim female, wide female, slim male, and wide male faces and shoes used in the object categorization experiment analyzed in section 4.2.

trials) and another five null-event trials in which the initial stimulus image was replaced by a forth purely scrambled image. The trial order in each mini block was pseudo-randomized to ensure the each condition and the null-events were equally often followed by any other condition or null-event. The participants task and the stimulus object category (*i.e.* being a face or a shoe) was kept constant during a mini block. Over the course of a scanning run four mini blocks were presented to the participants covering all combinations of stimulus object category and task. The mini block order was randomized in each run and a total of four runs were performed.

A standard echo-planar imaging (EPI) sequence was used to acquire the fMRI data using a Siemens Trio 3 Tesla MRI scanner (TR=2 s, TE=30 ms, and flip-angle $\alpha = 90°$). The spatial resolution was 3×3 mm (64×64 matrix in-plane matrix) and 32 slices (3 mm thickness and 10% interslice-gap). A total of 1564 volumes (4×391) were recorded.

4.2.2. Spatio-temporal MVPA

For the purpose of this example analysis the stimulus variation with respect to width and gender was ignored and only the object category classification problem was considered – which has been studied repeatedly, and hence, good estimates for the expected classifier performance are available (Haxby et al., 2001; Hanson et al., 2004; Hanson & Halchenko, 2008; O'Toole et al., 2007).

The initial pre-processing of the fMRI data was very similar to the procedure applied in the previous section. The functional timeseries were first motion-corrected using McFLIRT from FSL. Afterwards, run-wise detrending was performed. This time, how-

ever, not only by linear detrending, but additionally by regressing out quadratic trends in the data to account for the long run length of approximately 13 minutes (which had a significant positive impact on the classifier performance as opposed to linear detrending alone). Except for a feature-wise z-scoring of the data for each run individually, no further pre-processing was done, especially no slice-timing correction or any spatial or temporal filtering of the data.

The critical step in this analysis was the selection of brain response patterns for each trial in the experiment. To be able to look at spatio-temporal patterns, not only a single volume was selected, but a series of five volumes from a 10 s-window starting at stimulus onset, and therefore extending 2 s past the beginning of the next trial. For a full brain analysis this would yield about 200k voxel-timepoints, which PyMVPA is easily able to deal with (when running on reasonably modern desktop hardware). However, for the purpose of this example, the analysis was limited to a bilateral occipito-temporal ROI (as defined by the probabilistic Harvard-Oxford cortical atlas as *occipito-temporal fusiform cortex*, Flitney et al., 2007). Aligning each subject's functional space to the MNI-template and backprojecting the corresponding ROI mask yielded ROIs of approximately 300 voxels per volume, and hence brain response samples with about 1700 features each.

Linear SVM classifiers were trained on each participant's dataset individually, using a 4-fold cross-validation procedure, that routinely held out all trials from a full scanning run as a test dataset. All classifiers were able to achieve resonable accuracies in the generalization tests, with a mean of 79% correct single-trial predictions across all subjects (one-sample t-test against the chance-performance of 50%, $t = 13.75$, $df = 10$, $p < 0.001$). With the trained classifiers being able to reliably predict the object category of stimuli from single-trial data it was now posssible to inspect the classifier model parameters. Please note, that it is necessary to achieve reasonably high generalization accuracies (not just a little better-than-chance), as otherwise interpreting the model parameters is not meaningful, since the multivariate model most likely suffers from substantial overfitting or did not learn at all.

The SVM classifier provides one weight per feature – basically a score that attributes how informative a feature is. The higher the classifier weight, the larger is the impact of a feature on the classifier's decision. This association is largely identical to the weights of a logistic regression. Since the features in this example analysis are actually voxel-timepoints, it is possible to look at the temporal "importance" profile of each voxel.

Figure 4.6 shows such profiles (extracted from the classifier of participant 1) for the two voxels with the highest absolute classifier weight on any voxel-timestep in comparison to the event-related signal change. Voxel A located on the lateral side of the fusiform gyrus responds relatively similar to both object categories. However, there are small but reliable differences in the signal amplitude, that are picked-up by the classifier. Voxel B on the opposite, medial part of the fusiform gyrus shows a response to shoe stimuli that is about twice a large as the response to faces. Although the magnitudes of the assigned weights are about as large as the ones of voxel A, the impact of the signal of voxel B on the classifier decision is nevertheless stronger due to the larger response differences of the timepoints four and six seconds after stimulus onset. Both voxels clearly show

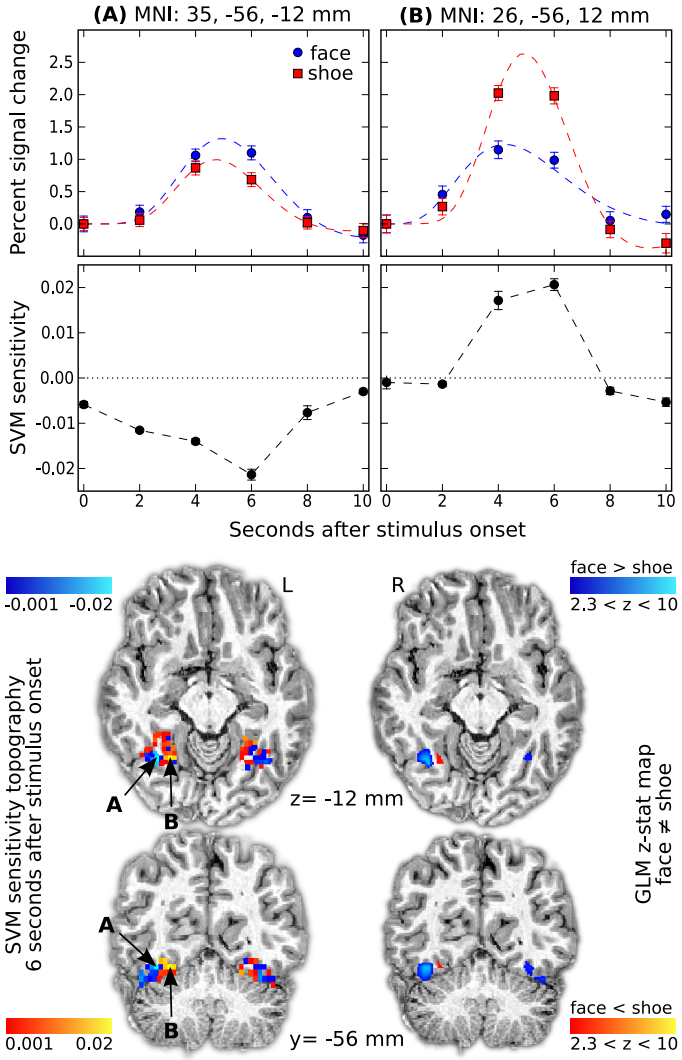

Figure 4.6.: Event-related SVM classifier sensitivities for object category discrimination of face and shoe stimuli (subject 1; occipital temporal fusiform ROI). The upper half shows the mean trial timecourse (errorbars show the standard error of the mean across all trials) for each stimulus condition and the associated SVM sensitivties (81% SVM prediction accuracy; errorbars show the standard error of the mean across cross-validation folds). The classifier automatically picks-up reliable signal differences, without having a notion of BOLD-response timing or shape. Positive and negative sensitivities are equally informative and represent a weighting towards one of the two alternatives in this binaries discrimination. The lower half contains a comparison of the sensitivity topography six seconds after stimulus onset and the corresponding SPM z-statistic map (arbitrarily thresholded at $z = 2.3$) of the *face vs. shoe* contrast. For the SPM analysis the data has been spatially smoothed using a 5 mm FWHM gaussian kernel, while the sensitivity analysis was run on unsmoothed data. The typical SPM-based analysis (including preprocessing) substantially obscures the response pattern pattern revealed by the classifier sensitivity topography.

a univariate effect of object category, and consequently the corresponding locations are also identified by an SPM analysis (also see Fig. 4.6).

Figure 4.7 offers a more detailed view on spatio-temporal structure of the multivariate model for a set of eight exemplary voxels on the surface of the right fusiform gyrus of participant 4 (that was chosen as another example of a classifier model with high generalization accuracy). When looking at the event-related signal change of these voxels it becomes obvious that there is a substantial variation in the response patterns. While most voxels seem to be responsive to both stimulus conditions there are nevertheless differences in amplitude (*e.g.* voxel B), but also in timing within a voxel (voxel H), and across voxels (*e.g.* voxel D vs. G).

The corresponding SPM z-score computed on the spatially smoothed data seems to be mainly driven by the small difference in the amplitudes, with faces causing a slightly stronger BOLD-response. However, this type of analysis completely cancels out local variations in the signal (as can be seen in the z-scores of smoothed and unsmoothed data of voxel E). Moreover, SPM is largely incapable to pick-up reliable signal differences originating in shape or timing difference of the BOLD-response (voxel H). Although it has to be mentioned that SPM is capable of framing hypotheses about timing-differences through the use of additional regressors, such as HRF basis functions, most studies nevertheless focus on contrasts about pure amplitude differences. MVPA on the other hand readily provides this type of information without the requirement of explicit hypotheses about them.

Finally, figure 4.8 shows an assessment of the stability of the structure of the multivariate model. Across all participants MVPA identifies continuous clusters of informative regions on the lateral and medial parts of the fusiform gyrus. The clustering of informative voxels can be considered as converging evidence that the model parameters capture a meaningful signal in the dataset, since the BOLD-response itself is known to be spatially extended ("watering the garden for the sake of the single flower"; Malonek & Grinvald, 1997). However, please note the variation in the spatial constellations across participants. One part shows the voxel clusters where a larger response indicates a face stimulus (blue color) on the lateral part of the fusiform gyrus, while others have this pattern reversed. This might be due to the known variability of the chunk of cortex that is supposed to perform the face processing in the human ventral stream (Kanwisher, McDermott, & Chun, 1997).

This pattern of results clearly shows that the employed SVM classifiers automatically picked-up meaningful information in the dataset that is based on amplitude and timing differences in the BOLD-response across subjects. This confirms the aforementioned property of MVPA to generate meaningful multivariate models of high-dimensional datasets without the need to specify a priori models of the anticipated signal.

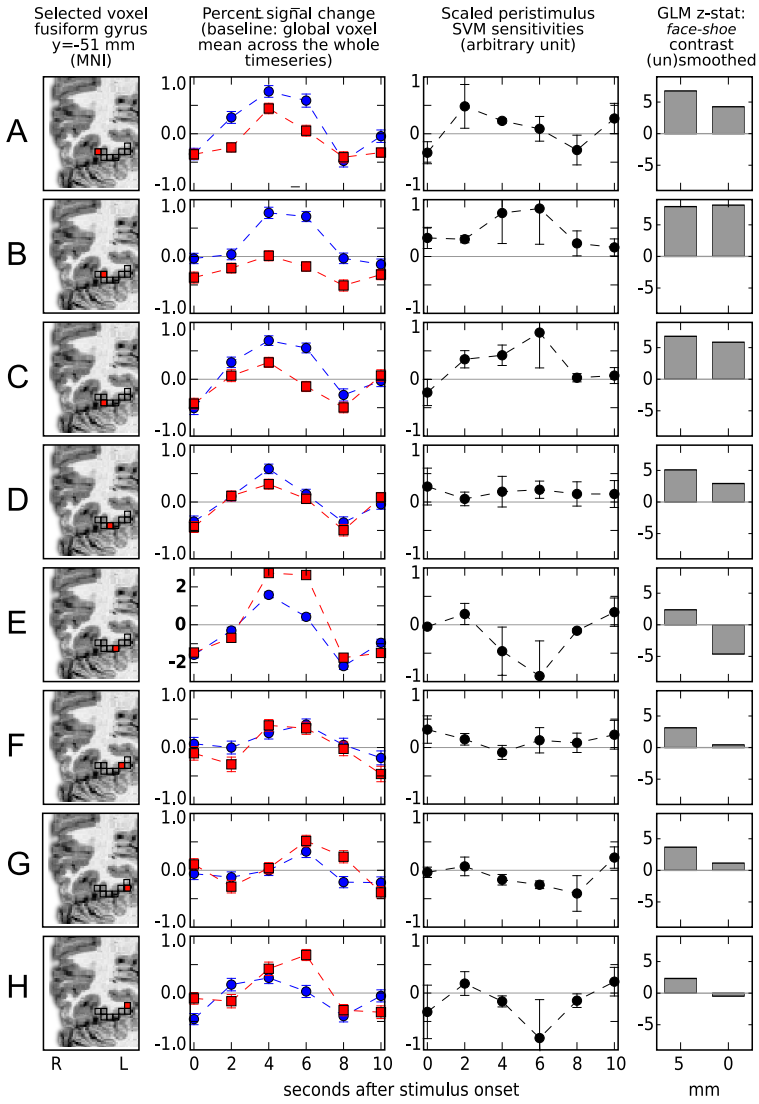

Figure 4.7.: Event-related SVM classifier sensitivities for eight adjacent target voxels on the right fusiform gyrus of subject 4. The classifier performed at a prediction accuracy of 92% using data from the occipito temporal fusiform ROI. The shown plots are analogous to figure 4.6, but in addition the SPM z-statistic of the *face - shoe* contrast of each voxel is plotted for data that has been smoothed using a 5 mm FWHM spatial Gaussian filter, and for unsmoothed data. The BOLD signal change is computed relative the global mean of a voxel's signal across the whole timeseries, to prevent obscuring a potential baseline difference between both stimulation conditions, that might be picked-up by the classifier. Voxel (E) shows a substantially larger BOLD-response than all other voxel in both conditions (note the divergent y-axis scaling). This might be due to a potential intersection of this voxel with a draining vein. This is reflected in both classifier sensitivities, and SPM z-statistic of unsmoothed data, but is not visible anymore after spatial filtering with a 5 mm kernel.

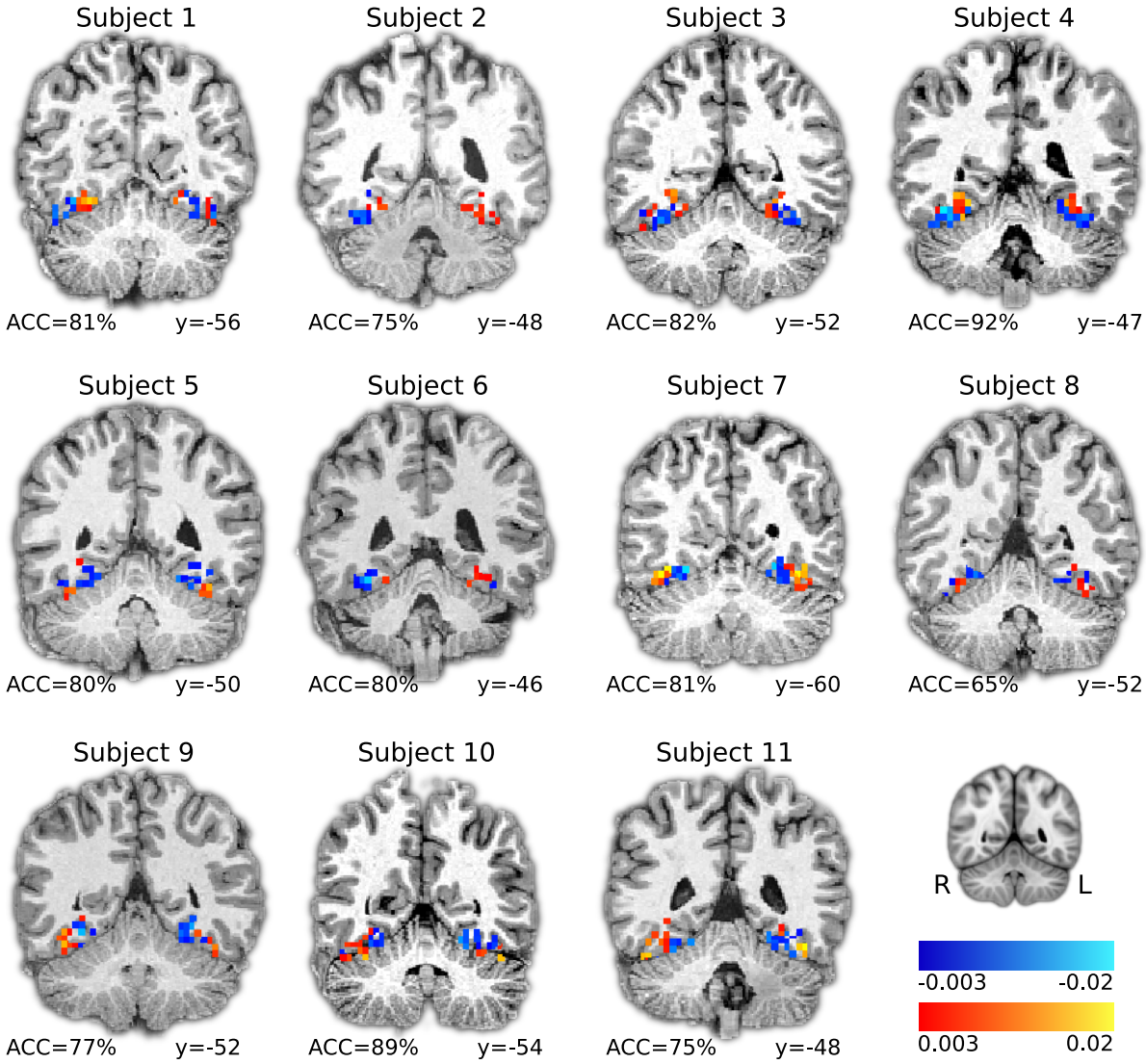

Figure 4.8.: SVM accuracies (ACC) and sensitivity distributions for all eleven subjects. The coronal slices show the sensitivities of voxels recorded 4-6 s after stimulus onset (no slicetime correction was performed). The sensitivity analysis can reliably identify informative voxel clusters on the lateral and medial surface of the right fusiform gyrus in all subjects. Please note, that the sign of the sensitivity is no uniform indicator of selectivity for any of the stimulus categories. The position of all slices is specified using the MNI coordinate system. Reported accuracies refer to the mean generalization performance, and analogously sensitivities are means across cross-validation folds. All SVMs were trained on voxels from a occipito-temporal ROI and multiple timepoints from a 10-seconds window after stimulus onset. The sensitivity plots are unsmoothed to reflect the true functional resolution.

4.3. Sensitivity analysis as a modality-independent technique

Understanding how the brain is able to give rise to complex behavior has stimulated a plethora of brain measures such as non-invasive EEG[3], MEG[4], MRI[5], PET[6], optical imaging, and invasive extracellular and intracellular recordings, often in conjunction with new methods, models, and techniques. Each data acquisition method has offered a unique set of properties in terms of spatio-temporal resolution, signal to noise, data acquisition cost, applicability to humans, and the corresponding neural correlates that result from the measurement process.

Neuroscientists often focus on only one or a smaller subset of these neural modalities partly due to the kinds of questions investigated and partly due to the cost of learning to analyze data from these different modalities. The diverse measurement approaches to brain function can heavily influence the selection of a research question and, in turn, the development of specific software packages to answer them. Consequently, the peculiarities of each data acquisition modality and the lack of strong interaction between the neuroscience communities employing them have produced distinct software packages specialized for the conventional analyses within a particular modality. Some analysis techniques have become, due to normative concerns, *de facto* standards despite their limitations and inappropriate assumptions for the given data type (*e.g.* SPM for fMRI).

While specialized procedures, and the software packages implementing them, are useful when dealing with the specific properties of a single data modality, they limit the flexibility to transfer newly developed analysis techniques to other fields of neuroscience. This issue is compounded by the closed-source, or restrictive licensing of many software packages, which further limits software flexibility and extensibility.

MVPA offers the chance to apply its powerful methods to data from various modalities, and hence loosen the otherwise tight coupling between analysis method and data modality. A central idea of PyMVPA is to expose any data in a optimal format for compatibility with a wide range of external software packages. However, at the same time it preserves modality specific information, such as the metric of the original dataspace. This is a critical feature to allow for meaningful interpretation of the multivariate model parameters as it has been outlined in the previous section. The concept of a sensitivity analysis is equally well applicable to other data modalities than fMRI, and PyMVPA helps to access the embedded information in modality-specific ways.

To illustrate this ability this section contains example analyses of four datasets, each from a different modality (EEG, MEG, fMRI, and extracellular recordings). All examples follow the same basic analysis pipeline: initial modality-specific preprocessing, application of MVPA methods, and visualization of the results. For the modality-independent machine-learning stage, all four examples employ the same analysis with *exactly* the same

[3]Electroencephalography
[4]Magnetoencephalography
[5]Magnetic resonance imaging
[6]Positron emission tomography

source code. Specifically, a cross-validation with one or more classifiers on each dataset is performed first and afterwards feature-wise sensitivity measures are computed. These measures can then be examined to reveal their implications in terms of the underlying research question.

4.3.1. EEG

The dataset used for the EEG example consists of a single participant from a previously published study on object recognition (Fründ et al., 2008). In the experiment, participants indicated, for a sequence of images, whether they considered each particular image a meaningful object or just object-like with a meaningless configuration. This task was performed for two sets of stimuli with different statistical properties and under two different speed constraints. EEG was recorded from 31 electrodes at a sampling rate of 500 Hz using standard recording techniques. Details of the recording procedure can be found in Fründ et al. (2008). A detailed description of the stimuli can be found in Busch, Herrmann, Müller, Lenz, & Gruber (2006, colored images) and in Herrmann, Lenz, Junge, Busch, & Maess (2004, line-art pictures).

Fründ et al. (2008) performed a wavelet-based time-frequency analysis of channels from a posterior region of interest (*i.e.* no multivariate methods were employed). Here, multivariate methods are used to differentiate between two conditions: trials with colored stimuli (broad spectrum of spatial frequencies and a high level of detail) and trials with black and white line-art stimuli (Fig. 4.9A), collapsing the data across all other conditions. This discrimination is orthogonal to the participants task of indicating object vs. non-object stimuli.

The data for this analysis were 700 ms EEG segments starting 200 ms prior to the stimulus onset of each trial, to which the following preprocessing procedure was applied. Only trials that passed the semi-automatic artifact rejection procedure performed in the original study were included, yielding 852 trials (422 color and 430 line-art). Each trial timeseries was downsampled to 200 Hz, leaving 140 sample points per trial and electrode. Each trial, including the EEG signal of all sample points from all channels, was defined as a sample to be classified (4340 features total). Finally, all features for each sample were normalized to zero mean and unit variance (z-scored).

As the main analysis a standard 6-fold cross-validation[7] procedure with *linear support vector machine* (linCSVM; Vapnik, 1995), *sparse multinomial logistic regression* (SMLR; Krishnapuram et al., 2005) and *Gaussian process regression* with linear kernel (linGPR; Rasmussen & Williams, 2006) classifiers was applied. Additionally, the multivariate I-RELIEF (Sun, 2007) feature sensitivity measures, and, for comparison, a univariate analysis of variance (ANOVA) F-score were computed on the same cross-validation dataset splits.

All three classifiers performed with high accuracy on the independent test datasets, achieving 86.2% (linCSVM), 91.8% (SMLR), and 89.6% (linGPR) correct single trial predictions, respectively. However, more interesting than the plain accuracy are the

[7] http://en.wikipedia.org/wiki/Cross-validation#K-fold_cross-validation

features each classifier relied upon to perform its predictions. Figure 4.9B shows the computed sensitivities from all classifiers and measures. There is a striking similarity between the shape of the classifier sensitivities plotted over time and the corresponding event-related potential (ERP) difference wave between the two experimental conditions (Fig. 4.9A; example shown for electrode *Pz*).

While all measures are able to identify the large signal differences at around 150 ms after stimulus onset, one particularly interesting result is the difference between the multivariate sensitivities and the univariate ANOVA F-scores from 300 ms to 400 ms. Only the multivariate methods (especially SMLR, linCSVM and linGPR) detected a relevant contribution to the classification task of the signal in this time window.

The head topography plots of the sensitivities (Fig. 4.10) nicely illustrate this situation for a selection of representative time windows (around 150, 200, 250 and 370 ms). They reveal a high variability with respect to the specificity among the multivariate measures. SVM, GPR and SMLR weights congruently identify three posterior electrodes as being most informative (SMLR weights provide the highest contrast of all measures). The I-RELIEF topography is much less specific and more similar to the ANOVA topography in its global spatio-temporal structure than to the other multivariate measures.

The late signal at around 370 ms that is completely missed by the univariate ANOVA may be related to the intracranial EEG gamma-band responses that Lachaux et al. (2005) observed at around the same time range when participants viewed complex stimuli. Given that the present data also seem to show a similar evoked gamma-band response (Fründ et al., 2008), it is possible that the multivariate methods are sensitive to the gamma-band activity in the data. Still, further work would be required to prove this correlation.

4.3.2. MEG

The example MEG dataset was collected with the aim to test whether it is possible to predict the recognition of briefly presented natural scenes from single trial MEG-recordings of brain activity (Rieger et al., 2008) and to use machine learning methods to investigate the properties of the brain activity that is predictive of later recognition. On each trial participants saw a briefly presented photograph (37 ms) of a natural scene that was immediately followed by a pattern mask (1000 ms – 1400 ms). The short masked presentation effectively limits the processing interval of the scene in the brain (Rieger, Braun, Bülthoff, & Gegenfurtner, 2005) and, therefore, participants will later recognize only some of the scenes. After the mask was turned off, participants indicated via button presses whether they would subsequently recognize the photograph or if they would fail. Immediately after this judgement, four natural scene photographs were presented and participants had to indicate which of the four scenes had been previously presented (*i.e.* a four-alternative forced-choice delayed match to sample task).

The MEG was recorded with a 151 channel CTF Omega MEG system from the whole head (sampling rate 625 Hz and a 120 Hz analogue low pass filter) while participants performed this task. The 600 ms interval of the MEG time series data that was used for the analysis started at the onset of the briefly presented scene and ended before the

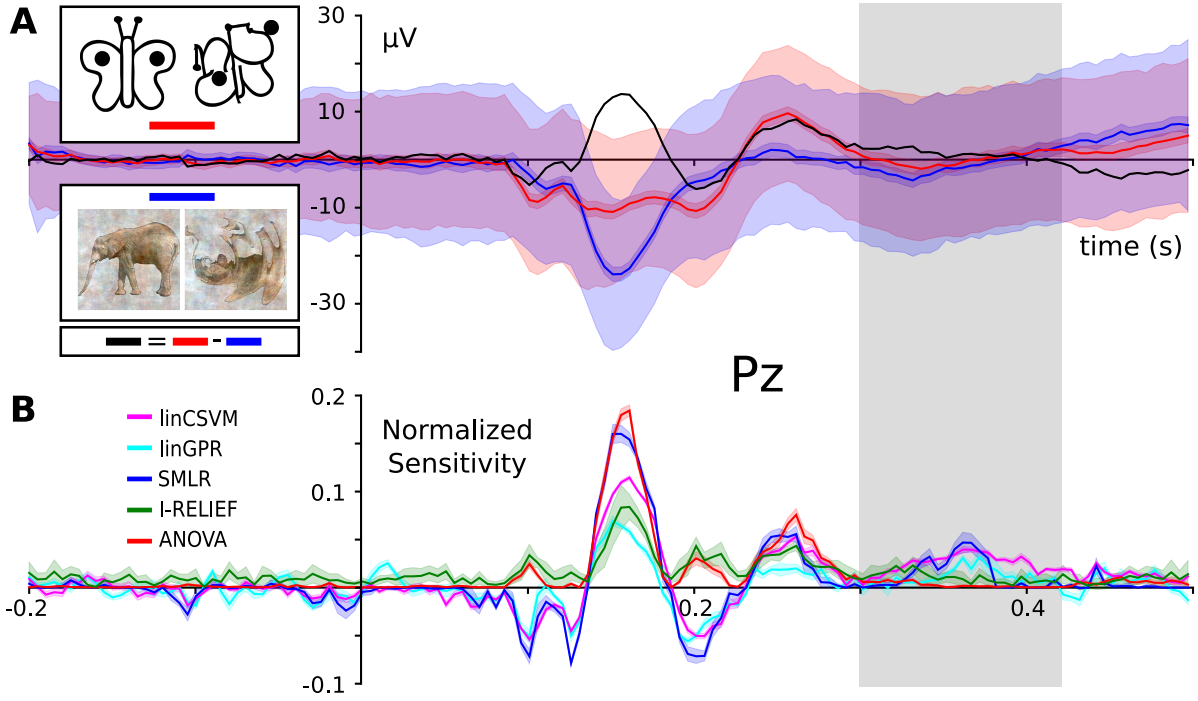

Figure 4.9.: Sensitivities for the classification of color and line-art conditions. Panel (A) shows event-related potentials (ERP) of each condition for electrode *Pz*. The light shaded area shows the standard deviation, the darker shade the 95% confidence interval around the mean ERP of each condition. The black curve is the difference wave of both ERPs. The stimulus example images are from Fründ et al. (2008). Panel (B) shows feature sensitivity measures for the different methods plotted over time for the *Pz* electrode. This electrode was chosen as Fründ et al. (2008) made it the subject of most visualizations. Sensitivities were normalized by scaling the vector norm of each sensitivity vector (covering all timepoints from all electrodes) to unit length. This allows for comparison of the relative weight each classifier puts on each feature. The shape of the sensitivity curves nicely resemble the ERP difference wave. Interestingly, for a time window around 370 ms after stimulus onset (indicated by the grey bar), all multivariate sensitivity measures assign a considerable amount of weight on the respective timepoints, whereas the univariate ANOVA is completely flat at zero.

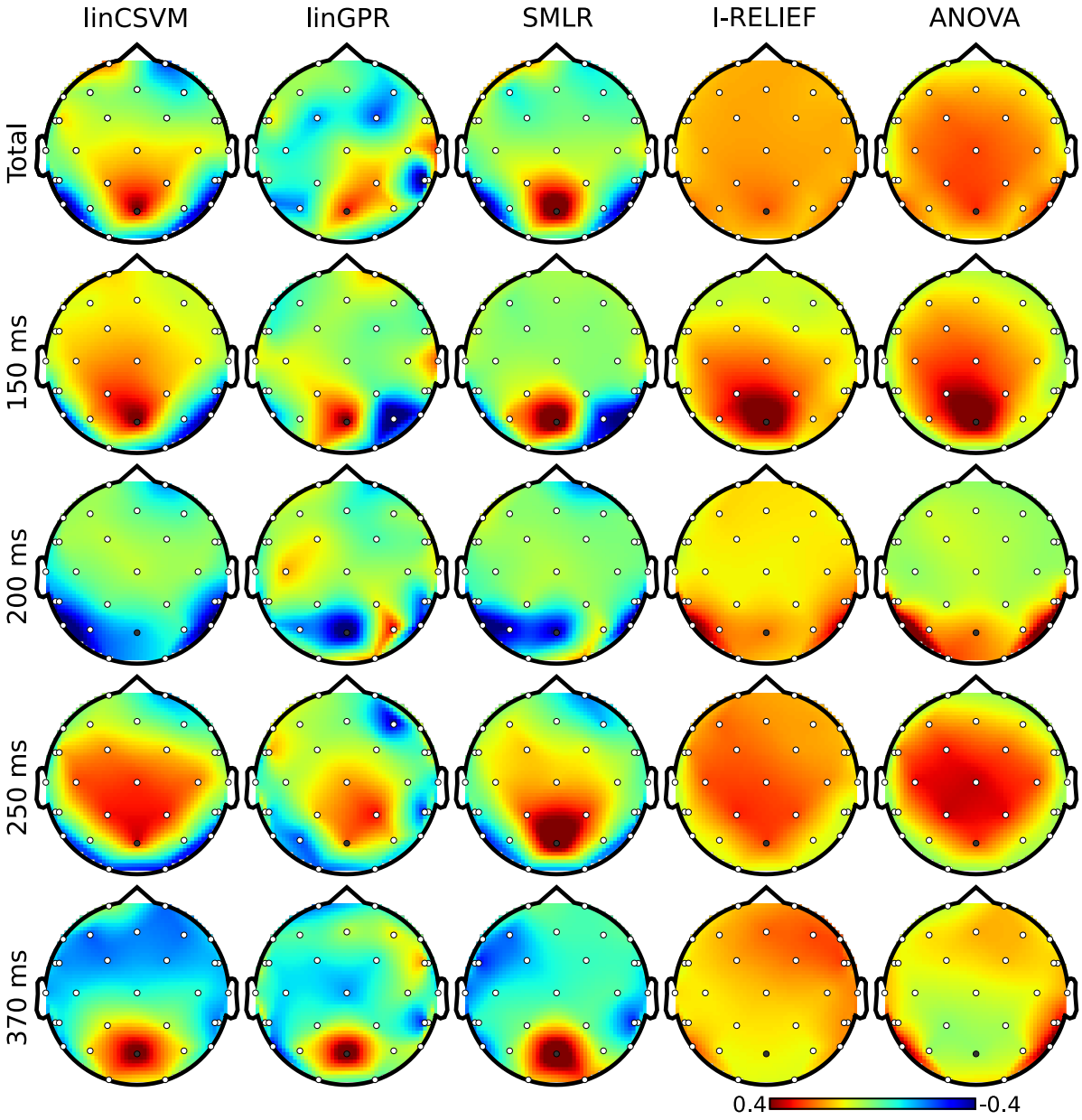

Figure 4.10.: Timecourse of the sensitivity topographies of various multivariate classifiers and measures, as well as the univariate ANOVA for the classification of color and line-art conditions. While the multivariate I-RELIEF shows a spatio-temporal profile that is very similar to ANOVA all other classifiers determine additional information in time-windows other than the one around 150 ms that contains a large signal difference in the ERP of electrode *Pz* between the stimulus conditions. The topographies show the channel-wise average of the sensitivities, either across the whole trial-timecourse (top-row) or from a 20 ms-window around 150, 200, 150, or 370 ms after stimulus onset.

mask was turned off. As in the original study, only those trials were analyzed in which participants both judged they would be correct and also correctly recognized the scene (*RECOG*) and the trials in which participants both predicted they would fail and gave an incorrect response (*NRECOG*). For details about the rationale of this selection, the stimulus presentation information, and the recording procedure see Rieger et al. (2008). In this example analysis data from a single participant was used (labeled *P1* in the original publication).

The MEG timeseries were first downsampled to 80 Hz and then all trial segments were channel-wise normalized by subtracting their mean baseline signal (determined from a 200 ms window prior to scene onset). Only timepoints within the first 600 ms after stimulus onset were considered for further analysis. The resulting dataset consisted of 151 channels with 48 timepoints each (7248 features), and a total of 294 samples (233 *RECOG* trials and 61 *NRECOG* trials).

The original study contained analyses based upon SVM classifiers, which revealed, by means of the spatio-temporal distribution of the sensitivities, that the theta band alone provides the most discriminative signal. The authors also addressed the topic of how to interpret heavily unbalanced datasets[8]. Given this comprehensive analysis, the aim here was to replicate their basic analysis strategy with PyMVPA and it was possible to achieve almost identical results.

As with the EEG data, a standard cross-validation procedure was applied, this time 8-fold, using linear SVM and SMLR classifiers. Additionally, a univariate ANOVA F-scores was again computed on the same cross-validation dataset splits. The SVM classifier was configured to use different per-class C-values[9], scaled with respect to the number of samples in each class to address the unbalanced number of samples. Similar to Rieger et al. (2008), a second cross-validation was ran on balanced datasets (by performing multiple selections of a random subset of samples from the larger *RECOG* category).

Both, classifiers performed almost identically on the full, unbalanced dataset, achieving 84.69% (SMLR) and 82.31% (linCSVM) correct single trial predictions (83.0% in the original study). Figure 4.11 shows sample timeseries of the classifier sensitivities and the ANOVA F-score of two posterior channels. Due to the significant difference in the number of samples of each category, it is important to additionally report mean true positive rate (TPR)[10], that amounted to 72% (SMLR), and 76% (linCSVM) respectively. The second SVM classifier trained on the balanced dataset achieved a comparable accuracy of 76.07% correct predictions (mean across 100 subsampled datasets), which is a slightly larger drop in accuracy when compared to the 80.8% achieved in the original study (refer to table 3 in Rieger et al., 2008).

Importantly, these results show that PyMVPA produces reproducible results that

[8] Unbalanced datasets have a dominant category (category which has considerably more samples than any other category). That potentially leads to the problem when a classifier prefers to assign the label of that category to all samples to minimize total prediction error.

[9] Parameter C in soft-margin SVM controls a trade-off between width of the SVM margin and number of support vectors (see Veropoulos, Campbell, & Cristianini, 1999, for an evaluation of this approach)

[10] Mean TPR is equivalent to accuracy in balanced sets, and is 50% at chance performance even with unbalanced set sizes (see Rieger et al., 2008, for a discussion of this point).

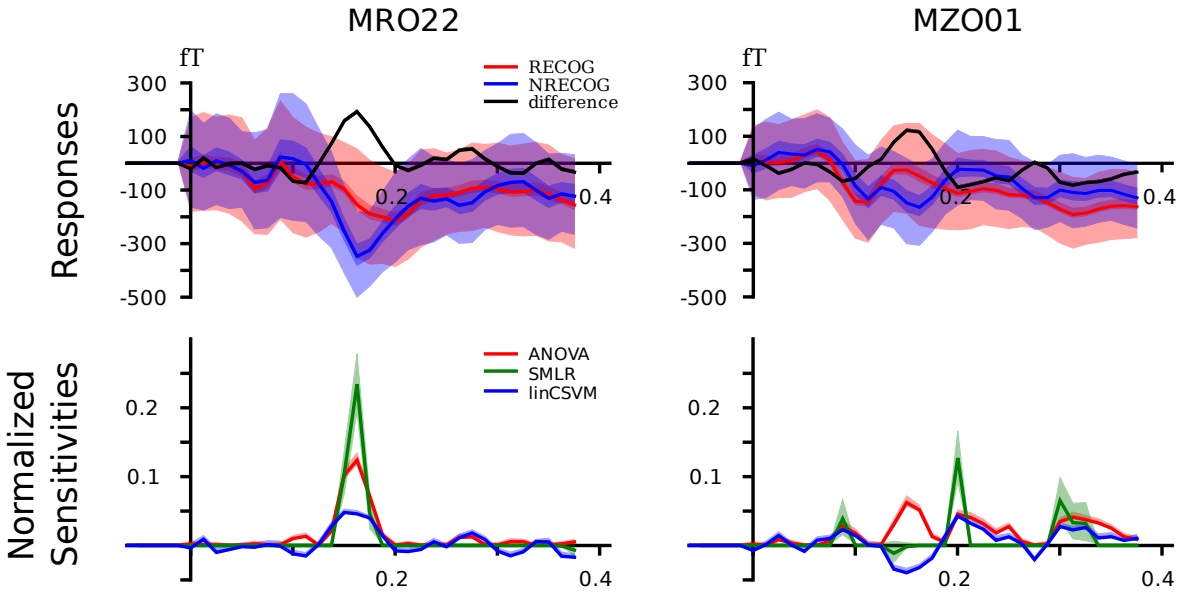

Figure 4.11.: Event-related magnetic fields (EMF) and classifier sensitivities. The upper part shows EMFs for two exemplary MEG channels. On the left sensor MRO22 (right occipital), and on the right sensor MZO01 (central occipital). The lower part shows classifier sensitivities and ANOVA F-scores plotted over time for both sensors. Both classifiers showed equivalent generalization performance of approximately 82% correct single trial predictions.

depend on the ML methods employed, but not on a particular implementation. However, according to the original authors, the integrated framework of PyMVPA allows to achieve these results with much less effort than what was necessary in the original study.

4.3.3. Extracellular recordings

The extracellular dataset analyzed in this section is previously unpublished[11], thus, the experimental and acquisition setup is briefly described first. Animal experiments were carried out in accordance with the National Institute of Health Guide for the Care and Use of Laboratory Animals and approved by Rutgers University. Sprague-Dawley rats (300-500 g) were anaesthetized with urethane (1.5 g/kg) and held with a custom naso-orbital restraint. After preparing a 3 mm square window in the skull over auditory cortex, the dura was removed and a silicon microelectrode consisting of eight four-site recording shanks (NeuroNexus Technologies, Ann Arbor, MI) was inserted. The recording sites were in the primary auditory cortex, estimated by stereotaxic coordinates, vascular structure (Sally & Kelly, 1988) and tonotopic variation of frequency tuning across recording shanks, and located within layer V, determined by electrode depth and firing patterns.

Five pure tones (3, 7, 12, 20, 30 kHz at 60 dB) and five different natural sounds (extracted from the CD "Voices of the Swamp", Naturesound Studio, Ithaca, NY) were used as stimuli. Each stimulus had a duration of 500 ms followed by 1500 ms of silence. All stimuli were tapered at beginning and end with a 5 ms cosine window. The data acquisition took place in a single-walled sound isolation chamber (IAC, Bronx, NY) with sounds presented free field (RP2/ES1, Tucker-Davis, Alachua, FL).

Individual units[12] were isolated by a semiautomatic algorithm (*KlustaKwik*[13]) followed by manual clustering (*Klusters*[14]). Post-stimulus time histograms (PSTH) of spike counts per each unit for all 1734 stimulation onsets were estimated using a bin size of 3.2 ms. To ensure an accurate estimation of PSTHs only units with mean firing rates higher than 2 Hz were selected for further analysis, leaving us with a total of 105 units.

Since the segregation of individual units out of the extracellular recordings is carried out without taking the respective stimulus condition into account, *i.e.* in unsupervised fashion (in ML terminology), it does not guarantee that the activity of any particular unit can be easily attributed to some set of stimulus conditions. From the stimulus-wise descriptive statistics of the units presented in the top plots of (Fig. 4.12) it is difficult to state that the activity of any particular unit at some moment in time is specific for a given stimulus. Furthermore, due to the inter-trial variance in the spike counts, it is even more difficult to reliably assess what stimulus condition any particular trial belongs

[11] The dataset was acquired and provided by Dr. Artur Luczak and Dr. Kenneth D. Harris (CMBN, Rutgers University, Newark, NJ, USA), and the primary data analysis, as published in Hanke et al. (2009), was performed by Yaroslav Halchenko.

[12] The term "unit" in the text refers to a single entity, which was segregated from the recorded data, and is expected to represent a single neuron.

[13] http://klustakwik.sourceforge.net

[14] http://klusters.sourceforge.net

to. Hence, the purpose of the PyMVPA analysis was to complement the results of the unsupervised clustering with a characterization of all extracted units in terms of their specificity to any given stimulus at any given time.

The analysis pipeline was similar to the one used for EEG, and MEG data. A standard 8-fold cross-validation procedure for an SMLR (Krishnapuram et al., 2005) classifier was ran, which achieved a mean of 77.57% accuracy estimate across all 10 types of stimuli. This generalization accuracy is well above chance (10%) for all stimulus categories and allows one to conclude that the neuronal population activity pattern at the recording site carries a differential signal across all 10 stimuli. Misclassifications mostly occurred for low-frequency stimuli. Pure tones with 3 kHz and 7 kHz were more often confused with each other than tones with a larger frequency difference (see Fig. 4.13), which suggests a high similarity in the spiking patterns for these stimuli. It could be further speculated that this neuronal population is more tuned towards the processing of higher frequency tones.

Besides being able to label yet unseen trials with high accuracy, the trained classifier can readily provide its sensitivity estimates for each unit, time bin, and stimulus condition (see bottom plots of (Fig. 4.12)). Temporal sensitivity profiles of any particular unit (see unit #42 profiles in lower left plot of (Fig. 4.12)) can reveal that the stimulus specific information is contained in spike times relative to stimulus onset or can be represented as slowly modulated pattern of spike counts (see 3 kHz stimuli). An aggregate sensitivity (in this case the sum of absolute sensitivities) across all time-bins provides a summary statistic of any unit's sensitivity to a given stimulus condition (see lower right plot of (Fig. 4.12)). In contrast to a simple variance measure, it provides an easier way to associate any given unit to a set of stimulus conditions. Additionally, it can identify units which might lack a substantial amount of variance, but nevertheless carrying a stimuli-specific signal (e.g. unit #28 and 30 kHz stimulus).

4.3.4. fMRI

Although example fMRI data analyses have been presented before an alternative analysis approach is introduced here, that combines different types of MVPA techniques, but is otherwise identical to the procedure used for the other data modalities. The example fMRI analysis again uses the dataset from Haxby et al. (2001) that already has been targeted in section 4.1.3, and employed an exactly identical pre-processing procedure. However, again for the sake of simplicity, the dataset was reduced – this time to a four-class problem (*faces*, *houses*, *cats* and *shoes*). All volumes recorded during any of these blocks were extracted and voxel-wise z-scored. This normalization was performed individually for each run to prevent any kind of information transfer across runs.

After preprocessing, the same sensitivity analysis was performed as for all other data modalities. Here, only a SMLR classifier was used (6-fold cross-validation, with two of the twelve experimental runs grouped into one chunk, and trained on single fMRI volumes that covered the full brain). For comparison, a univariate ANOVA was again computed for the same cross-validation dataset splits.

The SMLR classifier performed very well on the independent test datasets, correctly

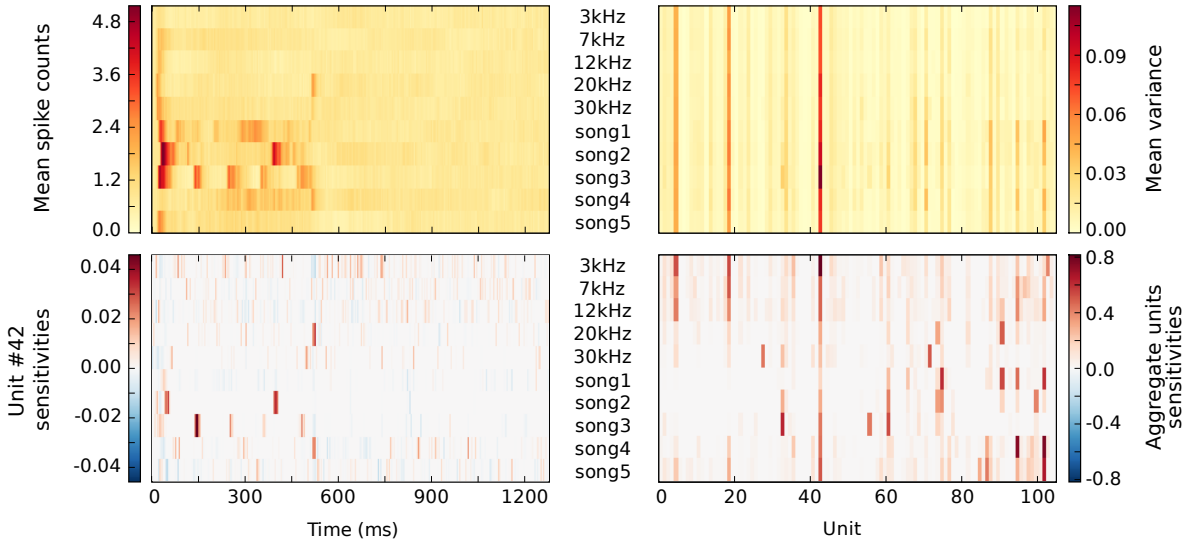

Figure 4.12.: Statistics of multiple single units extracellular simultaneous recordings and corresponding classifier sensitivities. All plots sweep through different stimuli along vertical axis, with stimuli labels presented in the middle of the plots. The upper part shows basic descriptive statistic of spike counts for each stimulus per each time bin (on the left) and per each unit (on the right). Such statistics seems to lack stimulus specificity for any given category at a given time point or unit. The lower part on the left shows the temporal sensitivity profile of a representative unit for each stimulus. It shows that stimulus specific information in the response can be coded primarily temporally (few specific offsets with maximal sensitivity like for *song2* stimulus) or in a slowly modulated pattern of spikes counts (see 3kHz stimulus). Associated aggregate sensitivities of all units for all stimuli in the lower right figure indicate each unit's specificity to any given stimulus. It provides better specificity than simple statistics like variance, *e.g.* unit 19 active in all stimulation conditions according to its high variance, but according to classifier sensitivity it carries little, if any, stimuli specific information for natural songs 1-3.

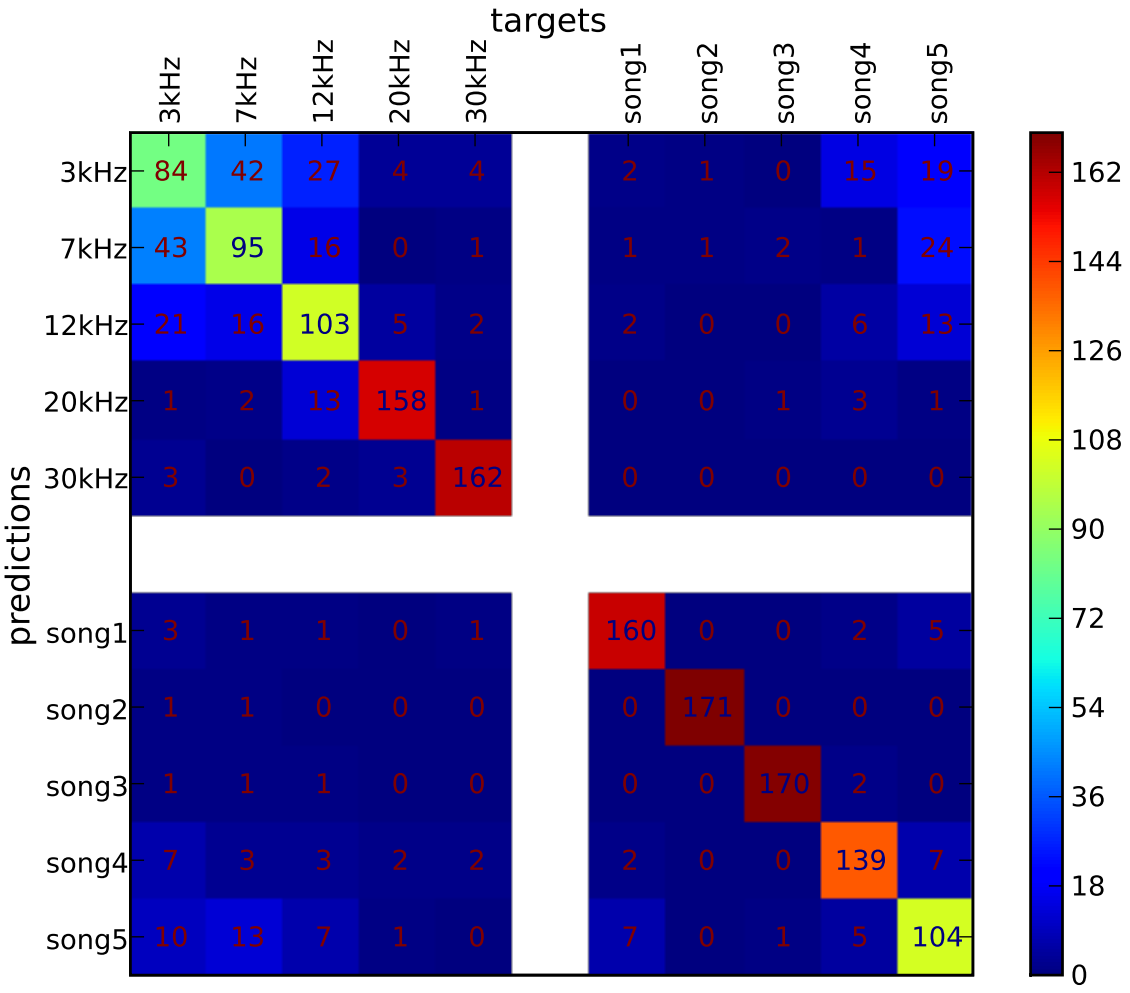

Figure 4.13.: Confusion matrix of SMLR classifier predictions of stimulus from multiple single units recorded simultaneously. The classifier was trained to discriminate between stimuli of five pure tones and five natural sounds. Elements of the matrix (numeric values and color-mapped visualization) show the number of trials which were correctly (diagonal) or incorrectly (off-diagonal) classified by a SMLR classifier during an 8-fold cross-validation procedure. The results suggest a high similarity in the spiking patterns for stimuli of low-frequency pure tones, which lead the classifier to confuse them more often, whenever responses to natural sound stimuli and high-frequency tones were hardly ever confused with each other.

predicting the category for 94.7% of all single volume samples in the test datasets. To examine what information was used by the classifier to reach this performance level, region of interest (ROI) based sensitivity scores for 48 non-overlapping structures defined by the probabilistic Harvard-Oxford cortical atlas (Flitney et al., 2007) were computed. To create the ROIs, the probability maps of all structures were thresholded at 25%, and assigned ambiguous voxels to the structure with the higher probability. The resulting map was projected into the space of the functional dataset using an affine transformation and nearest neighbor interpolation.

In order to determine the contribution of each ROI, the sensitivity vector was first normalized (across all ROIs), so that all absolute sensitivities summed up to 1 (L_1-normed). Afterwards ROI-wise scores were computed by taking the sum of all sensitivities in a particular ROI. The upper part of figure 4.14 shows these scores for the 20 highest-scoring and the three lowest-scoring ROIs.

The lower part of the figure shows dendrograms from a hierarchical cluster analysis[15] on relevant voxels from a block-averaged variant of the dataset (but otherwise identical to the classifier training data). For SMLR, only voxels with a non-zero sensitivity were considered in each particular ROI. For ANOVA, only the voxels with the highest F-scores (limited to the same number as for the SMLR case) were considered. For visualization purposes the dendrograms show the distances and clusters computed from the average samples of each condition in each dataset chunk (i.e, two experimental blocks), yielding 6 samples per condition.

The four chosen ROIs clearly show four different cluster patterns. The 92 selected voxels in temporal occipital fusiform cortex (TOFC) show a clear clustering of the experimental categories, with relatively large sample distances between categories. The pattern of the 36 voxels in angular gyrus reveals an animate/inanimate clustering, although with much smaller distances. The largest group of 148 voxels in the frontal pole ROI seems to have no obvious structure in their samples. Despite that, both sensitivity measures assign substantial importance to this region. This might be due to the large inter-sample distances, visualized in the corresponding dendrogram in figure 4.14. Each leaf node (in this case an average volume of two stimulation blocks) is approximately as distinct from any other leaf node, in terms of the employed distance measure, as the semantic clusters identified in the TOFC ROI. Finally, the ROI covering the anterior division of the superior temporal gyrus shows no clustering at all, and, consequently, is among the lowest scoring ROIs of both measures. On the whole, the cluster patterns from voxels selected by SMLR weights and F-scores are very similar in terms of inter-cluster distances.

Given that these results only include the data of a single participant, no far-reaching implications can be drawn from them. However, the distinct cluster patterns might provide indications for different levels of information encoding that could be addressed in future studies. Although voxels selected in both angular gyrus and the frontal pole ROIs do not provide a discriminative signal for all four stimulus categories, they nevertheless provide some disambiguating information and, thus, are picked up by the classifier. In

[15] PyMVPA provides hierarchical clustering facilities through *hcluster* (Eads, 2008).

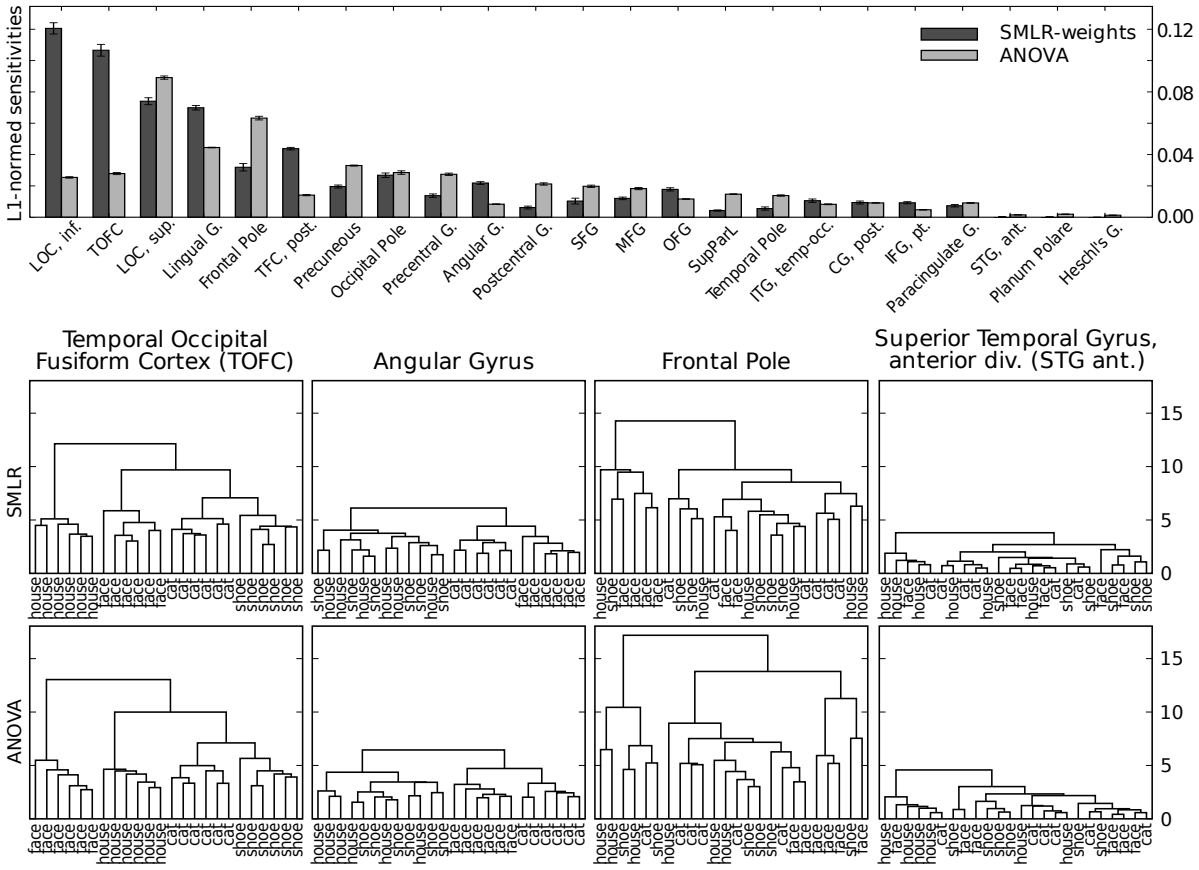

Figure 4.14.: Sensitivity analysis of the four-category fMRI dataset. The upper part shows the ROI-wise scores computed from SMLR classifier weights and ANOVA F-scores (limited to the 20 highest and the three lowest scoring ROIs). The lower part shows dendrograms with clusters of average category samples (computed using squared Euclidean distances) for voxels with non-zero SMLR-weights and a matching number of voxels with the highest F-scores in each ROI.

angular gyrus, this seems to be an animate/inanimate pattern that additionally also differentiates between the two categories of animate stimuli. Finally, in the frontal pole ROI the pattern remains unclear, but the relatively large inter-sample distances indicate a differential code of some form that is not closely related to the semantic stimulus category.

4.4. Using an unsupervised method to investigate information representations

The last section introduced the use of *unsupervised* MVPA techniques (in that case a clustering analysis) to complement *supervised* methods, such as classifiers. In this context supervised means that the training process is guided by explicitly specifying a model that should be learned by a classifier (note that this does not refer to a priori models of the signal, but a rather general classification of the available variance). This is typically done by labeling all samples in a dataset with their corresponding experimental condition or associating a certain regressor value with each sample. However, even completely unsupervised approaches can provide insight into the structure of brain response patterns. These methods do not require any model-specification. They are completely data-driven, and only operate on the brain response patterns themselves without ever observing condition labels, or model regressors.

4.4.1. Self-organizing maps

An example of such a method is a self-organizing map (SOM). This versatile machine learning technique was originally invented by Kohonen (1981), and is also often refered to as a *Kohonen-network*. SOMs basically generate a low-dimensional topological map that preserves structural properties of high-dimensional input data, and thus makes it an interesting tool to investigate the similarity structure of neural data. Its abilities to visualize high-dimensional data structures in typically two dimensions are similar to *multidimensional scaling* (Kruskal & Wish, 1978), although the underlying algorithm is different. Kohonen (2001) lists numerous variations of the SOM algorithm and application examples in statistics, signal processing and financial analysis.

In its simplest form a SOM is a single-layer neural network. The nodes in that layer are usually aranged in a two-dimensional hexagonal or rectangular grid. Each node receives input patterns over weighted connections, *i.e.* each node is associated with its own weight vector $\vec{w}$ that is initialized with random values. More sophisticated initialization methods are also commonly used. For example this could be randomly sampled values from a plane spanned by the first components estimated with a principal component analysis (PCA).

The training of the network is performed in a two-step process. For each input pattern a single node is determined whose weight vector matches the input pattern best. The best matching node is typically considered as the one where the euclidean distance between

weight vector and input pattern is minimal throughout the network. Afterwards, the weights of a node i are adjusted according to the following formula:

$$\vec{w_i}(t+1) = \vec{w_i}(t) + \phi(i,j,t)\,\alpha(t)(\vec{d_k} - \vec{w_i}(t))$$

where ϕ is a temporally decreasing neighborhood kernel that depends on the distance between the to be adjusted node i and the best matching node j in the Kohonen-layer, an also temporally decreasing learning coefficient α, and the difference between the current weight and input pattern $\vec{d_k}$. Depending on the actual implementation, weight adjustments can be applied *online*, *i.e.* per each input pattern and iteration, or in *batch-mode*, where the weight deltas for all input patterns are accumulated first and finally applied all at once at the end of each iteration. The training phase is finished after a predetermined number of iterations or the weight-adjustment drops below a certain threshold. Zell (2004, chapter 15) provides an overview of many possible parameters of SOM implementations and their practical implications.

The mapping of input patterns by a trained SOM, which is in a sense the classification step, is performed by determining the best match node and returning its coordinates in the Kohonen-layer, hence mapping from a high-dimensional input space into a typically two-dimensional discretized output space.

The properties of these maps are heavily influenced by two features of the training algorithm. First, the neighborhood kernel causes weights of spatially adjacent nodes to be changed more strongly than those of distant nodes. Since the kernel decreases in width over time, the network topology, after an initial global structuring, converges to a certain state during training, as later iterations only influence a relatively local area in the SOM. Second, weight adjustments become larger with increasing difference between current weight vector and input pattern, *i.e.* the weights for the best matching node for a particular input pattern are never changed. The conjunction of both properties cause similiar input patterns to be represented in adjacent locations. Fine-grained data structures sampled by many patterns in a dataset are automatically represented in more detail (*i.e.* using more nodes) then sparse or even empty areas in the high-dimensional input space. In this respect, SOMs are very similar to the retinotopic representation of information in primary visual cortex, where the amount of cortical surface corresponding to a certain fraction of the visual field is related to the receptor density on the retina (Kohonen, 2001).

4.4.2. Example: Looking for categorical information in the ventral object-processing stream

It has been long known that there are two distinct visual pathways (Ungerleider & Mishkin, 1982; Millner & Goodale, 1992, 1995). The dorsal pathway is considered to be involved in the processing of action-related information, while the ventral path is associated with the identification and processing of features of objects. However, the details of information encoding along the different stages in the ventral stream are still under debate (*e.g.* Kanwisher et al., 1997; Gauthier, Skudlarski, Gore, & Anderson,

2000; Haxby et al., 2001; Hanson & Halchenko, 2008). One of the major questions is a characterization of the subsystem in the ventral stream in terms of the respective information representation. One possibility to shed light on this topic is to look at the similarity of brain response pattern in those areas for different stimuli. The basic idea is that areas responding differently to some stimuli must somehow encode the differences between them, while similar responses indicate an insensitivity towards the discriminating features. Recently, researchers have started looking at the similarity structure of brain-response to visual objects using *representational dissimilarity matrices* (RMD; Kriegeskorte, Mur, & Bandettini, 2008) for different data-modalities and even across species (Kriegeskorte, Mur, Ruff, et al., 2008).

The purpose of this section is to provide a simple example to show that the non-linear vector quantization performed by a SOM, can be used in a similar fashion to look at the development of categorical information along the ventral stream (see *e.g.* Liao, Chen, Yang, & Lei, 2008, for another interesting example of SOM to fMRI data).

With `SimpleSOMMapper` PyMVPA provides a very simple SOM implementation using a rectangular Kohonen-grid, a gaussian neighborhood kernel and a euclidean distance measure. SOM is implemented as a mapper, and therefore it cannot only be used to visualize high-dimensional data, but also for data transformation similar to a PCA or wavelet decomposition. Since the number of features is effectively reduced to two, a SOM might also be a useful preprocessing step for simple non-linear classifiers as k-Nearest-Neighbor (kNN), which behaves suboptimal for high-dimensional datasets.

The fMRI dataset used here was again the same as in sections 4.3.4, and 4.1.3 – the single participant from Haxby et al. (2001), for the purpose of visualization again limited to the stimulus categories *face, house, cat,* and *shoe*, and preprocessed using the same procedure. Analogous to the previous analysis the originally full-brain dataset was processed in a ROI-wise fashion, with the anatomical ROIs defined again by the Harvard-Oxford probabilistic cortical atlas (Flitney et al., 2007). However, only a subset of ROIs along the ventral stream was considered, namely *occipital pole, inferior lateral occipital cortex, occipital fusiform cortex, temporal occipital fusiform cortex*, and *temporal fusiform cortex, posterior division* – all bilaterally.

The main differences between the current and the previous analysis approach is that here exclusively unsupervised techniques were applied, hence no predefined model was fitted to the data. Moreover, although already figure 4.14 provided some insight with respect to the similarity structure of the data using cluster analysis, in this case no block-averaging of fMRI scans is performed, but single volume samples were analyzed. Also, there was no classifier-based feature selection, but all data from all voxels in each ROI were fed into the SOM.

Figure 4.15 shows that SOMs clearly reflect the evolving object category related information along the ventral processing stream that has been hypothesized and shown in the literature. While there is no visible structure in the data from the occipital pole, a semantic clustering is successively built-up, with an almost perfect cluster pattern derived from the data from temporal occipital fusiform cortex. It is worth mentioning again that the structure of all SOMs evolved without information of the category membership of the brain response samples it has been trained on. This information was only added

afterwards to visualize the topological structure of the resulting maps.

Even more insightful analyses would be possible by conducting experiments using stimuli systematically varying along more than one dimension, and to track individual stimuli (each dot in figure 4.15 actually corresponds to a single fMRI volume) along a processing stream in the brain.

4.5. Statistical safeguarding of results

Drawing conclusions from classifier-based analyses often involves the comparison of classifier performances on data from different experimental conditions, or from distinct ROIs. For example, evidence in favor of a hypothesis postulating the presence of a signal of some sort in a particular ROI could be expressed by a *better-than-chance* performance of a classifier operating on data from that ROI. However, this raises the question how can be determined what exactly is *better than chance*?

Fortunately, classifier accuracies for predicting the labels of data samples are hardly any different from accuracies (or equivalent error rates) of human observers performing a classification task. Such tasks are used in the majority of psychophysical experiments where subjects judge the presence or absent of a signal, or associate a stimulus with some category in a multi-alternative forced choice design. This similarity allows to use the full bandwidth of statistical methods developed for psychophysical research to be used for inferential analysis of classifier accuracies.

A simple case is the evaluation of accuracies across several subjects. The analysis is performed individually for each subject and afterwards the achieved accuracies are tested against the expected chance performance using a one-sample t-test (see *e.g.* section 4.1.3 or Sterzer, Haynes, & Rees, 2008, for an example application). For balanced datasets the chance-performance is $\frac{1}{k}$, where k is the number of categories in the classification task. For unbalanced datasets the chance-performance can be substantially different from that (see Rieger et al., 2008, for a comprehensive discussion).

However, combining accuracies across subjects carries the risk of comparing apples to oranges. It is in no way guaranteed that a discrimination performance on two datasets from two different subjects is based on a similar or even the same signal. This applies to both the spatial or temporal sources of the utilized signal as well as its nature. The classifier might pick-up different signals in each subject's dataset, which, despite being different, are nevertheless related to the experimental paradigm. Such situation would have a significant impact on the validity of the conclusions drawn, which typically aim at the underlying representation of information in the brain.

Although this potential problem is equally present in single-subject cross-validation analyses, it can be addressed more easily. The major problem in multi-subject analyses is that the features in individual datasets cannot be easily aligned to each other and hence it is very hard to identify common patterns in the brain responses. Such common patterns would be a good source of confidence that the results of a group analysis actually summarize similarly structured multivariate models. However, the established approach of aligning subjects anatomies and co-registering datasets by affine transfor-

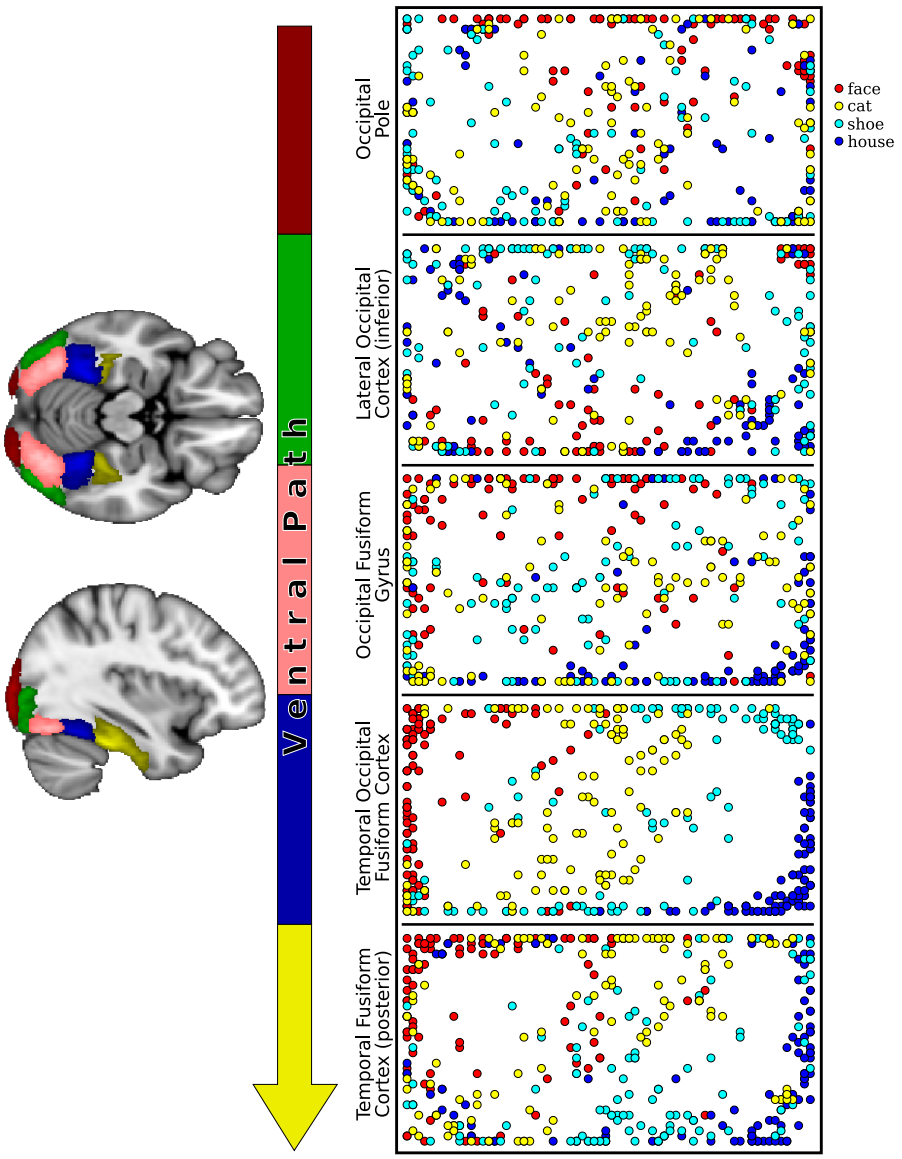

Figure 4.15.: Self-organizing map (SOM) structures derived from brain responses along the ventral pathway to four visual stimulus categories: *faces, cats, shoes*, and *houses*. For each of the five areas a separate SOM was training with a total of 432 single fMRI volume samples recorded during block-wise stimulus presentation. The non-linear vector quantization performed by the SOMs reveals an evolving category-related signal. Starting at the occipital pole, with no or very little clustering, the signal becomes more refined while moving in anterior direction, with a beginning separation of *face* and *house* conditions in occipital fusiform gyrus. This is followed by clear category-related clustering of the brain responses to all four conditions measured in voxels in temporal occipital fusiform cortex. Even further anterior in the posterior temporal fusiform cortex the specificity of the categorical clustering is substantially impaired.

mations is unlikely to achieve a voxel to voxel correspondence between individuals. It is anyway questionable if such alignment is desired, since the variability in (functional) brain anatomy makes it almost impossible that two voxels measured in two different humans actually sample the same set of neurons.

4.5.1. Feature selection stability

It is nevertheless important to assess the structure of the multivariate models generated within a cross-validation analysis. This is especially the case for algorithms involving feature selection procedures. Feature selection, although in general beneficial for the analysis of high-dimensional datasets, might lead to the weird situation that a classifier predicts the test data category labels with high accuracy in each cross-validation fold, but across all folds exclusively non-overlapping feature sets are used to achieve this performance. Such behavior could be interpreted in the way that a number of different features carry redundant information and they are only selected or rejected due to random variation in the noise pattern of the data. However, it might also be the case that the utilized signal really changes between cross-validation folds, which are typically associated with different parts of the experiment (*i.e.* experimental runs) and hence might be influenced by effects of *e.g.* fading vigilance. It is therefore very important to report the variability of the feature sets and other measures as has been previously suggested (O'Toole et al., 2007; Chen et al., 2006).

Figure 4.16 shows examples of ratios of cross-validation folds in which any given feature was chosen by the corresponding feature selection method used by some classifiers listed in Table 4.1 (trained on the example dataset used in section 4.1.3). PyMVPA already provides convenient methods to assess the stability of feature selections within cross-validation procedures. However, more research is required to address information localization problems in different contexts. For example, when implementing a brain-computer interface it is beneficial to identify a set of features that provides both an optimal generalization performance as well as a high stability of spatial configuration and accuracy across different datasets, *i.e.* to reduce the number of false-positive feature selections. On the other hand, in a clinical setting one may want to identify all voxels that could possibly contribute some information in a pre-surgery diagnostic tool and, thus, would focus on minimizing false-negatives instead.

4.5.2. Statistical measures beyond accuracies and t-tests

In general, it might be beneficial to report statistical scores, such as t-scores or significance levels instead of plain accuracies. These statistics incorporate additional, and important information about the variance of the underlying results. However, the gold standard of reporting such analysis results is yet to be found, and statistical scores might also cause problems for some of the recently proposed analysis strategies. For example, a p-value map of probabilities of better-than-chance classification accuracy generated during a searchlight run must be considered as a massive multiple-comparison problem,

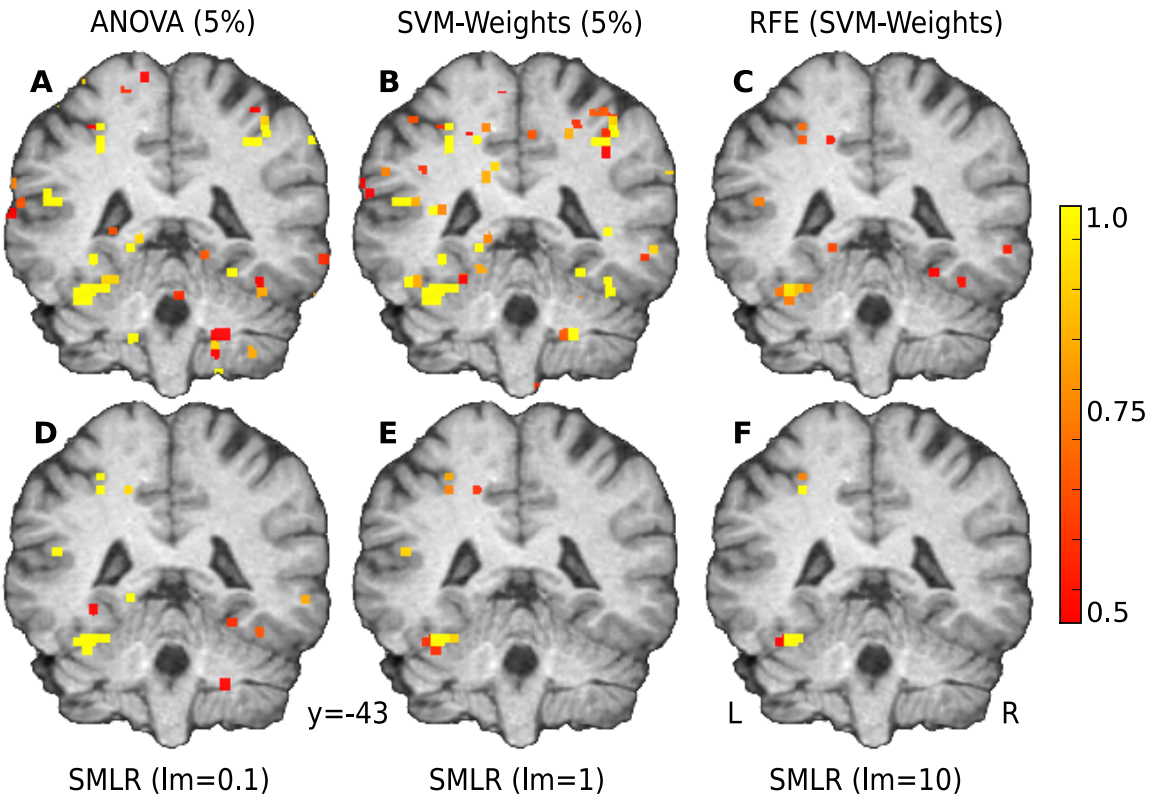

Figure 4.16.: Feature selection stability maps for the CATS vs. SCISSORS classification (using the dataset of Haxby et al., 2001). The color maps show the fraction of cross-validation folds in which each particular voxel is selected by various feature selection methods used by the classifiers listed in Table 4.1: 5% highest ANOVA F-scores (A), 5% highest weights from trained SVM (B), RFE using SVM weights as selection criterion (C), internal feature selection performed by the SMLR classifier with penalty term 0.1 (D), 1 (E) and 10 (F). All methods reliably identify a cluster of voxels in the left fusiform cortex, centered around MNI: -34, -43, -20 mm. All stability maps are thresholded arbitrarily at 0.5 (6 out of 12 cross-validation folds).

very much like an SPM result. Therefore it also has to be subject of appropriate α-level adjustments.

However, there is more information available from classifier cross-validations than simple accuracies. Even for binary classification problems the plain fraction of correct classifications hides interesting facts. As it was already mentioned, MVPA results are very similar to those of psychophysical experiments. A look at the prevalent analysis techniques in that field suggest that the methods derived from the *signal detection theory* (Green & Swets, 1966) provide a more comprehensive description of classifier behavior. The well-known d'-score and ROC curves (Swets, 1996) can be computed from a classifier's confusion matrix. A confusion matrix is generated by cross-tabulating actual classifier predictions and prediction targets (see *e.g.* figure 4.13). Every classifier that is implemented in PyMVPA provides confusion matrices (single and accumulated across cross-validation folds) and hence can be easily analyzed in this respect.

Moreover, a χ^2-test on the confusion matrix provides a statistical evaluation of the classifier accuracy that is also applicable to multi-class problems. Pereira et al. (2009) suggests that in case of a low number of observations in the confusion matrix cells individual binomial tests might be more appropriate, given independent samples. However, especially for fMRI data, independence is typically not the case, due to the aforementioned temporal forward-contamination of the signal by the sluggish BOLD-response. Currently there is no generally accepted way to address this problem. Perhaps a data-driven deconvolution approach can be used to achieve sample-independence while preserving the advantages of MVPA that originate in the non-necessity of a priori models, but this is a hypothesis that needs careful evaluation.

All statistical methods listed so far are mostly concerned with classifier prediction error or accuracies, but the demand for statistical evaluation applies equally well for the parameters of a multivariate model. Previous sections have shown that the interpretation of classifier sensitivities can provide additional insight into the functioning of a particular information processing system. However, to derive valid interpretation two requirements have to be fulfilled. First, the multivariate model generated by the classifier has been proven to generalize to yet unseen data with reasonable performance. Second, the assigned sensitivities are stable across cross-validation folds.

The first condition is critical since a model that does not generalize is not an appropriate description of the signal of interest and hence its model parameters have to be considered meaningless. The second condition is very similar to the aspects of feature selection stability discussed earlier and primarily influence the conclusions that can be drawn from a particular sensitivity distribution.

Validation of sensitivities with parametric tests is difficult since it is hard to justify (or even to make assumptions) about their distribution under the null hypothesis. Fortunately, *Monte-Carlo permutation tests* provide a way to estimate the corresponding distributions without having to make prior assumptions (Golland & Fischl, 2003) – at the cost of a significant increase in the demand for computational resources.

To estimate the distribution of a classifier sensitivity (or any other measure) under the null hypothesis (*i.e.* no relevant signal in the dataset) the classifier is trained multiple times (usually several thousand times) on a dataset with permuted category labels. If

there is no relevant information in the dataset, the association between category labels and data samples can be randomized without altering the classifier performance.

For a one-sided test the probability under the null hypothesis of a given sensitivity derived from a classifier trained on the dataset with the original, unpermuted category labels is simply the fraction of Monte-Carlo samples that is greater or equal than the respective value. This technique provides a flexible way to assess the probability of any result, ranging from classifier accuracies to sensitivity measures (*e.g.* SVM or regression weights) to other feature-wise scores, such as ANOVA F-scores.

However, estimating the null distribution is a computationally demanding task that can easily take days depending on the size of the dataset and the complexity of the classification task. Therefore parametric tests remain attractive whenever reasonable assumptions about the distribution of the corresponding variable can be made, and hence PyMVPA provides support for both types of statistical evaluation of analysis results.

5. General Conclusions

By now, numerous studies have illustrated the power of MVPA, harnessing machine learning techniques based on statistical learning theory to extract information about the functional properties of the brain previously thought to be below the signal-to-noise ratio of fMRI data (for reviews see Haynes & Rees, 2006; Norman et al., 2006).

This thesis reviewed many advantages of MVPA over established analysis procedures. It primarily emphasized two aspects. First, being a multivariate technique it allows to access information in the covariance structure of fMRI data that has been completely ignored in the vast majority of studies that have been published over the last two decades. The second aspect focused on the flexibility of MVPA with respect to its independence of a priori assumptions about the targeted signals. While prior knowledge can be built into any MVPA-based analysis, the non-necessity to do so enables researchers to discover meaningful brain response patterns that an SPM-based analysis would rather accumulate into the residual variance. Moreover, this thesis could show that the model-free approach is equally well applicable to other neuroimaging data modalities. This represents a major advantage that should not be underestimated, since it moves distinct research communities closer together that have been specialized on a particular modality, and could potentially help to intensify the exchange between them.

Although MVPA-based studies have already proven their potential to advance the understanding of brain function in a number of fields, ranging from human memory to visual perception, it is important to note that they were performed by relatively few research groups. This may be due to the fact that very few software packages that specifically address MVPA of fMRI data are available to a broad audience. Such packages require a significant amount of software development, starting from basic problems, such as how to import and process fMRI datasets, to more complex problems, such as the implementation of classifier algorithms. This results in an initial overhead requiring significant resources before actual neuroimaging datasets can be analyzed.

The main advance put forth in this dissertation project is undoubtedly the PyMVPA analysis framework. It aims to be a solid base for conducting MVPA. In contrast to other software packages, such as the *3dsvm* plugin for AFNI (LaConte et al., 2005), it follows a more general approach by providing a collection of common algorithms and processing steps that can be combined with great flexibility. Consequently, the initial overhead to start an analysis once the fMRI dataset is acquired is significantly reduced because the toolbox also provides all necessary import, export and preprocessing functionality.

It has been demonstrated that PyMVPA is specifically tuned towards fMRI data analysis. But much like the great flexibility of the MVPA techniques themselves, the generic design of the framework allows to work with other data modalities equally well. The flexible dataset handling encourages extensions to support other data formats, while

at the same time extending the mapping algorithms to represent other data spaces and metrics, such as the sparse surface sampling of EEG channels or MEG datasets (see Thulasidas, Guan, & Wu, 2006; Guimaraes, Wong, Uy, Grosenick, & Suppes, 2007, for examples of MVPA of these data modalities).

However, the key feature of PyMVPA is that it provides a uniform interface to bridge from standard neuroimaging tools to machine learning software packages. This interface makes it easy to extend the toolbox to work with a broad range of *existing* software packages, which should significantly reduce the need to recode available algorithms for the context of brain-imaging research. Moreover, all external and internal classifiers can be freely combined with the classifier-independent algorithms for *e.g.* feature selection, making this toolbox an ideal environment to compare different classification algorithms.

5.1. Attracting researchers to fathom the full potential of MVPA

While the promising list of MVPA properties alone should be a convincing argument to adopt these methods in the research process, this is not guaranteed to happen automatically. O'Toole et al. (2007) emphasize that basically the same technical advantages were already offered by the *partial least squares* (PLS; McIntosh, Bookstein, Haxby, & Grady, 1996) method, 15 years ago. While it is not clear what prevented PLS to become a standard method, it nevertheless shows that it is important to undertake great endeavors to advertise the advantages, demonstrate the power of MVPA, and make it available to a broad scientific community.

The last aspect is most likely the one that still requires most of the work. While the prospect of being able to do "brain reading" already carries a certain level of attraction, it does not imply that everybody can afford to adopt the technology. The statistical learning theory origin requires researcher who are already proficient with SPM to familiarize themselves with a substantial amount of MVPA background theory. To this end it requires appropriate literature that explains valid analysis approaches of neuroimaging data. Fortunately, very recently an initial set of articles covering this topic for different target audiences appeared (*e.g.* Pereira et al., 2009; Mur, Bandettini, & Kriegeskorte, 2009).

However, knowledge about the theory is a critical requirement to employ MVPA techniques, but it is again not sufficient. In this thesis it has been argued that the necessary software development is both costly, and moreover of questionable usefulness if the resulting product is not publicly available and subject to intensive peer-review. Here, PyMVPA is trying to provide a framework to incorporate the necessary technology, while the framework itself is aiming for a maximum of transparency (fully available source code, tests, documentation, and examples) and accessibility.

Its intended audience is threefold. First, there are *neuroscience researchers* interested in testing ML algorithms on neural data, *e.g.* people working on brain-computer interfaces (BCI, see Lebedev & Nicolelis, 2006; Birbaumer & Cohen, 2007). PyMVPA pro-

vides researchers with the ability to execute complex analysis tasks in very concise code. Second, it is also designed for *ML researchers* interested in testing new ML algorithms on neural data. PyMVPA offers a highly-modularized architecture designed to minimize the effort of adding new algorithms. Moreover, the availability of neuroscience-related code-examples (like the ones presented in this thesis) and datasets greatly reduces the time to get actual results. Finally, PyMVPA is welcoming *code contributors* from both neuroscience and ML communities interested in improving or adding modality-specific functions or new algorithms. PyMVPA offers a community-based development model together with a distributed version control system and extensive reference documentation.

PyMVPA code is tested to be portable across multiple platforms, and its limiting set of essential external dependencies in turn has proven to be portable. In fact, PyMVPA only depends on a moderately recent version of Python and the NumPy package. Although PyMVPA can make use of other external software, the functionality provided by them is completely optional. For an up-to-date list of possible extensions the reader is referred to the PyMVPA project website[1]. To allow for convenient installation and upgrade procedures, the authors are providing binary packages for ten different operating systems, including various GNU/Linux distributions (in their native package format), as well as installers for MacOS X and Windows. This comprises PyMVPA itself and a number of additional packages (*e.g.* NumPy), if they are not available from other sources for a particular target platform.

Although PyMVPA aims to be especially user-friendly it does not provide a graphical user interface (GUI). The reason not to include such an interface is that the toolbox explicitly aims to encourage novel combinations of algorithms and the development of new analysis strategies that are not easily foreseeable by a GUI designer[2]. The toolbox is nevertheless user-friendly, enabling researchers to conduct highly complex analyses with just a few lines of easily readable code. It achieves this by taking away the burden of dealing with low-level libraries and providing a great variety of algorithms in a concise framework. The required skills of a potential PyMVPA user are not much different from neuroscientists using the basic building blocks needed for one of the established fMRI analysis toolkits (e.g., shell scripting for AFNI, Lipsia, and FSL command line tools, or Matlab-scripting of SPM functions).

Recent releases of PyMVPA added support for visualization of analysis results, such as, classifier confusions, distance matrices, topography plots and plotting of time-locked signals. However, while PyMVPA does not provide extensive plotting support, it nevertheless makes it easy to use existing tools for MRI-specific data visualization. Similar to the data import PyMVPA's mappers make it also trivial to export data into the original data space and format, *e.g.* using a reverse-mapped sensitivity volume as a statistical overlay, in exactly the same way as a statistical parametric map derived from a conventional analysis. This way PyMVPA can fully benefit from the functionality provided by

[1] http://www.pymvpa.org/installation.html#dependencies

[2] Nothing prevents a software developer from adding a GUI to the toolbox using one of the many GUI toolkits that interface with Python code, such as PyQT (http://www.riverbankcomputing.co.uk/software/pyqt/) or wxPython (http://www.wxpython.org/).

the numerous available MRI toolkits.

The features of PyMVPA outlined here cover only a fraction of the currently implemented functionality. More information is, however, available on the PyMVPA project website, which contains a user manual with an introduction into the main concepts and the design of the framework, a wide range of examples, a comprehensive module reference as a user-oriented summary of the available functionality, and finally a more technical reference for extending the framework.

5.2. Powerful methods for cognitive neuroscience

The emerging field of MVPA of fMRI data is beginning to complement the established analysis techniques and has great potential for novel insights into the functional architecture of the brain. However, there are a lot of open questions how the wealth of algorithms developed by those motivated by statistical learning theory can be optimally applied to brain-imaging data.

An important aspect of this topic are the underlying intentions of cognitive neuroscience research. The ultimate goal is to understand how the brain works, and this goal has important implications on almost every single step of an analysis procedure. Consider for example a feature selection algorithm. In ML research, feature selection is usually performed to remove unimportant information from a dataset that does not improve, or even has a negative impact on the generalization accuracy of a particular classifier. In the neuroscience context, however, the primary focus is not on the accuracy level, but on the structure and origin of the information that allows for correct predictions to be made (the accuracy simply has to be reasonably high to allow for an interpretation of the model at all). It therefore has significant side-effects on the conclusions that can be drawn from an analysis, when carelessly removing features providing redundant information, and hence, it would be of great value to have a set of techniques that acknowledges the specifics of cognitive neuroscience research, to provide researchers with the tools to access the full structure of brain-response patterns.

The current lack of a *gold standard* for MVPA demands software that allows one to apply a broad range of available techniques and test an even broader range of hypotheses. PyMVPA tries to reach this goal by providing a unifying framework that allows to easily combine a large number of basic building blocks in a flexible manner to help neuroscientists to do rapid initial data exploration, and consecutive custom data analysis. PyMVPA even facilitates the integration of additional algorithms in its framework that are not yet discovered by neuro-imaging researchers.

An important direction for future developments is moving from the localization and description of brain-response patterns to the modeling of the system dynamics of the brain or its subsystems. This thesis already offered some examples of the analysis of spatio-temporal patterns to predict or explain their association with certain experimental conditions. However, MVPA is not limited to simple classification tasks. Multivariate regression-variants are able to build models that link the brain response patterns themselves. Investigating the temporal relationship of signals from distinct brain areas would

allow for an in-depth analysis of the functional connectivity structure of the brain.

A related topic is cross-modal data analysis. The flexibility of MVPA with respect to the analysis of several data modalities that has been shown in this thesis, allows researchers to target the joint analysis of multiple neural datasets, to combine the advantages of each brain activity measure (see Halchenko, 2009, for a first example of a cross-modal data analysis with PyMVPA). However, candidate data modalities are by no means limited to the neuroimaging domain. It is long known, that *e.g.* eye-movement patterns reflect conscious and unconscious cognitive processes (see *e.g.* Hayhoe & Ballard, 2005; Ferreira, Apel, & Henderson, 2008, for reviews), and hence might also provide valuable information.

An important topic that has yet to be confronted by PyMVPA is the problem of model selection. Most MVPA-based studies published so far successfully used linear classifiers and found no practical advantages of non-linear algorithms (see *e.g.* D. D. Cox & Savoy, 2003, for an evaluation of linear vs. non-linear classifiers). However, when moving forward to the description of system dynamics non-linear techniques will probably become necessary. One particular problem of non-linear classifiers is that they require a sensible setting of their hyper-parameters to achieve optimal results. In PyMVPA only Gaussian process regression currently has the ability to guide the selection of hyper-parameters. Uniform model selection for ML methods within PyMVPA is planned for the next major release of the project. It will provide the facility to automatically search for the best set of parameters for each classifier without sacrificing unbiased estimates of the generalization performance.

5.3. Offering the blueprint of a *mind-reader*

With the recent increase in attention of the neuroscience community, discussions about the application of MVPA to neural datasets are often accompanied by the concepts of "decoding" (Kamitani & Tong, 2005) or "mind-reading" (Norman et al., 2006). There is even a begining discussion about the ethical consequences of the ability to read people's minds (Haynes & Rees, 2006). While these concepts are first of all quite fancy labels, and undoubtedly contributed to the perceived attractiveness of the method, it is nevertheless important to reflect on their implications.

What does it mean, if a classifier is able to reliably predict whether a person was looking at the image of a house or a cat, solely from fMRI data that has been recorded at that time? Is it the case that this classifier can now predict any visual stimulation by reading it out from this person's brain, given that an fMRI dataset is available? Does it mean that the classifier identified the true encoding algorithm of object category information in the brain? Or does it just mean that the classifier identified the location of object category processing in the brain? The answer to all those question is either *no*, or *we don't know*.

The reason for this becomes obvious if one looks at the technical implementation of this problem. When training a classifier on this binary *house vs. cat* problem it eventually will extract a model of the multivariate fMRI signal that allows it to perform

this prediction on an independent dataset. However, this fact does not even justify to conclude that the classifier is able to discriminate between houses and cats. The classifier is simply distinguishing between two different patterns in the fMRI data, but we don't know what stimulus property is reflected in the data. With respect to this example it could equally likely be the difference of stimuli with and without fur, or even the desire cuddle the stimulus object – with a dramatic impact on the *correct* interpretation.

In the context of psychological experiments this problem is all but new. Confounds always limit the interpretability of experimental results, and there are ways to conduct more appropriate variants of this example paradigm that allow to draw more specific conclusion (*i.e.* by adding control conditions).

However, with respect to deciphering the encoding of information in the brain it is important to consider the origin of the data, even given an appropriate experimental paradigm and stimuli. In the case of fMRI data it is still not completely understood what exactly is represented by the BOLD-response – whether it is the firing of neurons that are sampled by a particular voxel, or if it is rather the magnitude of inhibitory input from spatially distinct areas (Heeger & Ross, 2002). Therefore conclusions drawn from the identified structure of patterns in the fMRI signal may not be equally applicable to the pattern of neuron firing in the corresponding brain-tissue.

In that sense MVPA-based studies have to obey the same principles as those employing SPM analysis procedures. Moreover, the same guidelines for conclusion-drawing apply. Specifically, studies observing an involvement of a certain brain area in a specific classification cannot easily infer the nature of this involvement from other studies having shown a contribution of that area in some other cognitive task (*i.e. inverse conclusions*, Poldrack, 2006).

Logothetis (2008) summarizes the situation in very few words: "In fact, fMRI is not and will never be a mind reander...". Even if one does not completely follow this strong oppinion, giving all mentioned limitations, MVPA is far from becoming, or even already being a general purpose mind-reader. This fact is of significant importance for the application of MVPA in the context of lie-detection, and the reader is refered to an interesting discussion about the impact of deception on the "decodability" of lies (Sip, Roeppstorff, McGregor, & Frith, 2008a; Haynes, 2008; Sip, Roeppstorff, McGregor, & Frith, 2008b).

Nevertheless, MVPA represents a major step forward in the analysis of the brain, and its current applications prepare the ground for an even larger step. Many possibilities have not been mentioned in this thesis, such as the confirmatory analysis of models describing the structure and behavior of subsystems of the brain (*e.g.* Kay, Naselaris, Prenger, & Gallant, 2008, for an example of reconstructing hundreds of individual images from brain response patterns by employing a relatively simple model of the signal processing in the visual system). Many additional studies offering high-level descriptions of brain function will be necessary, because otherwise even if we know all the nitty-gritty details to readout every bit of information from the *brain*, we might still not know how the *mind* really works (Mausfeld, 2007).

6. References

Aguirre, G. K., Zarahn, E., & D'esposito, M. (1998). The variability of human, bold hemodynamic responses. *NeuroImage*, *8*, 360–369.

Bandettini, P. A.(1999). The temporal resolution of MRI. In C. T. W. Moonen & P. A. Bandettini (Eds.), *Functional MRI* (pp. 205–220). Berlin: Springer.

Beckmann, C. F. (2009). *Model-free functional data analysis.* Available online from FMRIB, Oxford: http://www.fmrib.ox.ac.uk/fslcourse/lectures/melodic.pdf.

Beckmann, C. F., & Smith, S. M.(2005). Tensorial extensions of independent component analysis for multisubject fMRI analysis. *NeuroImage*, *25*, 294–311.

Benjamini, Y., & Hochberg, Y.(1995). Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society. Series B (Methodological)*, *57*, 289–300.

Birbaumer, N., & Cohen, L. G.(2007). Brain-computer interfaces: communication and restoration of movement in paralysis. *Journal of Physiology*, *579*, 621–636.

Boynton, G. M., Engel, S. A., Glover, G. H., & Heeger, D. J.(1996). Linear systems analysis of functional magnetic resonance imaging in human v1. *Journal of Neuroscience*, *16*, 4207–4221.

Buckner, R. L., Koutstaal, W., Schacter, D. L., Dale, A. M., Rotte, M., & Rosen, B. R.(1998). Functional-anatomic study of episodic retrieval. ii. selective averaging of event-related fMRI trials to test the retrieval success hypothesis. *NeuroImage*, *7*, 163–175.

Busch, N. A., Herrmann, C. S., Müller, M. M., Lenz, D., & Gruber, T.(2006). A cross-laboratory study of event-related gamma activity in a standard object recognition paradigm. *NeuroImage*, *33*, 1169–1177.

Calhoun, V., Adali, T., Hansen, L., Larsen, J., & Pekar, J.(2003). ICA of functional MRI data: an overview. In *Fourth international symposium on independent component analysis and blind source separation* (pp. 281–288).

Calhoun, V., Pekar, J., McGinty, T., Adali, T. D., & Watson, G. D.(2002). Different activation dynamics in multiple neural systems during simulated driving. *Human Brain Mapping*, *16*, 158–167.

Carlson, T. A., Schrater, P., & Sheng, H. (2003). Patterns of activity in the categorical representations of objects. *Journal of Cognitive Neuroscience*, *15*, 704–717.

Chang, C.-C., & Lin, C.-J. (2001). *LIBSVM: a library for support vector machines.* (Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm)

Chen, X., Pereira, F., Lee, W., Strother, S., & Mitchell, T. (2006). Exploring predictive and reproducible modeling with the single-subject FIAC dataset. *Human Brain Mapping*, *27*, 452–461.

Cox, D. D., & Savoy, R. L. (2003). Functional magnetic resonance imaging (fMRI) "brain reading": detecting and classifying distributed patterns of fMRI activity in human visual cortex. *NeuroImage*, *19*, 261–270.

Cox, R. W. (1996). AFNI: software for analysis and visualization of functional magnetic resonance neuroimages. *Computers and Biomedical Research*, *29*, 162–173.

Detre, G., Polyn, S. M., Moore, C., Natu, V., Singer, B., Cohen, J., et al. (2006). *The Multi-Voxel Pattern Analysis (MVPA) toolbox.* Poster presented at the Annual Meeting of the Organization for Human Brain Mapping (Florence, Italy).

Eads, D. (2008). *hcluster: Hierarchical clustering for SciPy.* (Software available at: http://scipy-cluster.googlecode.com/)

Efron, B., & Tibshirani, R. (1993). *An Introduction to the Bootstrap.* Chapman & Hall/CRC.

Efron, B., Trevor, H., Johnstone, I., & Tibshirani, R. (2004). Least Angle Regression. *Annals of Statistics*, *32*, 407–499.

Ferreira, F., Apel, J., & Henderson, J. M. (2008). Taking a new look at looking at nothing. *Trends in Cognitive Sciences*, *12*, 405–410.

Flitney, D., Webster, M., Patenaude, B., Seidman, L., Goldstein, J., Tordesillas Gutierrez, D., et al. (2007). Anatomical brain atlases and their application in the FSLView visualisation tool. In *Thirteenth Annual Meeting of the Organization for Human Brain Mapping.*

Friston, K., Harrison, L., & Penny, W. (2003). Dynamic causal modelling. *NeuroImage*, *19*, 1273–1302.

Friston, K., Holmes, A., Worsley, K., Poline, J., Frith, C. D., & Frackowiak, S. (1994). Statistical parametric maps in functional imaging: A general linear approach. *Human Brain Mapping*, *2*, 189–210.

Friston, K., Jezzard, P., & Turner, R. (1994). Analysis of functional mri time-series. *Human Brain Mapping*, *1*, 153–171.

Fründ, I., Busch, N. A., Schadow, J., Gruber, T., Körner, U., & Herrmann, C. S.(2008). Time pressure modulates electrophysiological correlates of early visual processing. *PLoS ONE*, *3*, e1675.

Gauthier, I., Skudlarski, P., Gore, J. C., & Anderson, A. W.(2000). Expertise for cars and birds recruits brain areas involved in face recognition. *Nature Neuroscience*, *3*, 191–197.

Goebel, R., Esposito, F., & Formisano, E.(2006). Analysis of functional image analysis contest (FIAC) data with Brainvoyager QX: From single-subject to cortically aligned group general linear model analysis and self-organizing group independent component analysis. *Human Brain Mapping*, *27*, 392–401.

Golland, P., & Fischl, B.(2003). Permutation tests for classification: towards statistical significance in image-based studies. *Information Processing and Medical Imaging*, *18*, 330–341.

Green, D. M., & Swets, J. A.(1966). *Signal detection and recognition by human observers*. New York: Wiley.

Guimaraes, M. P., Wong, D. K., Uy, E. T., Grosenick, L., & Suppes, P.(2007). Single-trial classification of MEG recordings. *IEEE Transactions on Biomedical Engineering*, *54*, 436–443.

Guyon, I., & Elisseeff, A.(2003). An Introduction to Variable and Feature Selection. *Journal of Machine Learning*, *3*, 1157–1182.

Guyon, I., Weston, J., Barnhill, S., & Vapnik, V.(2002). Gene Selection for Cancer Classification using Support Vector Machines. *Machine Learning*, *46*, 389–422.

Halchenko, Y. O.(2009). *Predictive decoding of neural data*. Unpublished doctoral dissertation, New Jersey Institute of Technology.

Hanke, M., Halchenko, Y. O., Sederberg, P. B., Olivetti, E., Fründ, I., Rieger, J. W., et al.(2009). PyMVPA: A Unifying Approach to the Analysis of Neuroscientific Data. *Frontiers in Neuroinformatics*, *3:3*.

Hanson, S. J., & Halchenko, Y. O.(2008). Brain reading using full brain support vector machines for object recognition: there is no "face" identification area. *Neural Computation*, *20*, 486–503.

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

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

Hayhoe, M., & Ballard, D. (2005). Eye movements in natural behavior. *Trends in Cognitive Sciences*, *9*, 188–194.

Haynes, J.-D. (2008). Detecting deception from neuroimaging signals – a data-driven perspective. *Trends in Cognitive Sciences*, *12*, 126–127.

Haynes, J.-D., & Rees, G. (2005). Predicting the orientation of invisible stimuli from activity in human primary cortex. *Nature Neuroscience*, *8*, 686–691.

Haynes, J.-D., & Rees, G.(2006). Decoding mental states from brain activity in humans. *Nature Reviews Neuroscience*, *7*, 523–534.

Haynes, J.-D., Sakai, K., Rees, G., Gilbert, S., Frith, C., & Passingham, R. E. (2007). Reading hidden intentions in the human brain. *Current Biology*, *17*, 323–328.

Heeger, D. J., & Ross, D. (2002). What does fMRI tell us about neuronal activity? *Nature Reviews Neuroscience*, *3*, 142–151.

Herrmann, C. S., Lenz, D., Junge, S., Busch, N. A., & Maess, B.(2004). Memory-matches evoke human gamma-responses. *BMC Neuroscience*, *5*(13).

Jenkinson, M., Bannister, P., Brady, J., & Smith, S.(2002). Improved optimisation for the robust and accurate linear registration and motion correction of brain images. *NeuroImage*, *17*, 825–841.

Kamitani, Y., & Tong, F. (2005). Decoding the visual and subjective contents of the human brain. *Nature Neuroscience*, *8*, 679–685.

Kanwisher, N., McDermott, J., & Chun, M. M.(1997). The fusiform face area: a module in human extrastriate cortex specialized for face perception. *Journal of Neuroscience*, *17*, 4302–4311.

Kay, K. N., Naselaris, T., Prenger, R. J., & Gallant, J. L. (2008). Identifying natural images from human brain activity. *Nature*, *452*, 352–355.

Kienzle, W., Schölkopf, B., Wichmann, F., & Franz, M. O.(2007). How to find interesting locations in video: a spatiotemporal interest point detector learned from human eye movements. In *Lecture notes in computer science: Pattern recognition (DAGM).* (pp. 405–414).

Kippenhahn, J. S., Barker, W. W., Pascal, S., Nagel, J., & Duara, R.(1992). Evaluation of a neural-network classifier for pet scans of normal and alzheimer's disease subjects. *Journal of Nuclear Medicine*, *33*, 1459–1467.

Kohonen, T. (1981). Automatic formation of topological maps of patterns in a self-organizing system. In E. Oja & O. Simula (Eds.), *Proceedings of the Scandinavian Conference on Image Analysis* (pp. 214–220). Finland: Suomen Hahmontunnistus-tutkimuksen Seura r. y.

Kohonen, T.(2001). *Self-organizing Maps* (3 ed.). New York: Springer.

Kriegeskorte, N., & Bandettini, P.(2007). Analyzing for information, not activation, to exploit high-resolution fMRI. *NeuroImage*, *38*, 649–662.

Kriegeskorte, N., Formisano, E., Sorger, B., & Goebel, R.(2007). Individual faces elicit distinct response patterns in human anterior temporal cortex. *Proceedings of the National Academy of Sciences of the USA*, *104*, 20600–20605.

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

Kriegeskorte, N., Mur, M., & Bandettini, P.(2008). Representational similarity analysis - connecting the branches of systems neuroscience. *Frontiers in System Neuroscience*, *2*, 4.

Kriegeskorte, N., Mur, M., Ruff, D. A., Kiani, R., Bodurka, J., Esteky, H., et al.(2008). Matching categorical object representations in inferior temporal cortex of man and monkey. *Neuron*, *60*, 1126–1141.

Krishnapuram, B., Carin, L., Figueiredo, M. A., & Hartemink, A. J. (2005). Sparse multinomial logistic regression: fast algorithms and generalization bounds. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *27*, 957–968.

Kruskal, J. B., & Wish, M.(1978). Multidimensional Scaling. In *Sage University Paper series on Quantitative Application in the Social Sciences* (pp. 7–11). Sage Publications.

Lachaux, J.-P., George, N., Tallon-Baudry, C., Martinerie, J., Hugueville, L., Minotti, L., et al. (2005). The many faces of the gamma band response to complex visual stimuli. *NeuroImage*, *25*, 491–501.

LaConte, S., Strother, S., Cherkassky, V., Anderson, J., & Hu, X. (2005). Support vector machines for temporal classification of block design fMRI data. *NeuroImage*, *26*, 317–329.

Lebedev, M. A., & Nicolelis, M. A. L.(2006). Brain-machine interfaces: past, present and future. *Trends in Neuroscience*, *29*, 536–546.

Liao, W., Chen, H., Yang, Q., & Lei, X.(2008). Analysis of fMRI data using improved self-organizing mapping and spatio-temporal metric hierarchical clustering. *Medical Imaging, IEEE Transactions on*, *27*, 1472–1483.

Logothetis, N. K.(2002). The neural basis of the blood-oxygen-level-dependent functional magnetic resonance imaging signal. *Philosophical Transactions of the Royal Society B: Biological Sciences*, *357*, 1003–1037.

Logothetis, N. K.(2008). What we can do and what we cannot do with fMRI. *Nature*, *453*, 869–878.

Logothetis, N. K., Pauls, J., Augath, M., Trinath, T., & Oeltermann, A.(2001). Neurophysiological investigation of the basis of the fMRI signal. *Nature*, *412*, 150–157.

Lohmann, G., Mueller, K., Bosch, V., Mentzel, H., Hessler, S., Chen, L., et al.(2001). LIPSIA–a new software system for the evaluation of functional magnetic resonance images of the human brain. *Computerized Medical Imaging and graphics: the official journal of the Computerized Medical Imaging Society*, *25*, 449.

Makeig, S., Debener, S., Onton, J., & Delorme, A.(2004). Mining event-related brain dynamics. *Trends in Cognitive Sciences*, *8*, 204–210.

Malonek, D., & Grinvald, A.(1997). Interactions between electrical activity and cortical microcirculation revealed by imaging spectroscopy: implications for functional brain mapping. *Science*, *272*, 551–554.

Mausfeld, R.(2007). Über Ziele und Grenzen einer naturwissenschaftlichen Zugangsweise zur Erforschung des Geistes. In A. Holderegger, B. Sitter-Liver, & C. Hess (Eds.), *Hirnforschung und Menschenbild.* Schwabe.

McIntosh, A. R., Bookstein, F. L., Haxby, J. V., & Grady, C. L.(1996). Spatial pattern analysis of functional brain images using partial least squares. *NeuroImage*, *3*, 143–157.

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*, 735–759.

Millman, K., & Brett, M.(2007). Analysis of Functional Magnetic Resonance Imaging in Python. *Computing in Science & Engineering*, *9*, 52–55.

Millner, A. D., & Goodale, M. A.(1992). Separate visual pathways for perception and action. *Trends in Neuroscience*, *15*, 20–25.

Millner, A. D., & Goodale, M. A.(1995). *The visual brain in action.* Oxford: Oxford University Press.

Mitchell, T., Hutchinson, R., Niculescu, R. S., Pereira, F., Wang, X., Just, M., et al.(2004). Learning to Decode Cognitive States from Brain Images. *Machine Learning*, *57*, 145–175.

Mitchell, T., Shinkareva, S. V., Carlson, A., Chang, K.-M., Malave, V. L., Mason, R. A., et al.(2008). Predicting human brain activity associated with the meanings of nouns. *Science*, *320*, 1191–1195.

Moeller, J. R., & Strother, S.(1991). A regional covariance approach to the analysis of functional patterns in positron emission tomographic data. *Journal of Cerebral Blood Flow and Metabolism*, *11*, 121–135.

Monti, M. M.(2006). *Statistical analysis of fMRI time-series: A critical evaluation of the GLM approach.* Online available at: http://www.mrc-cbu.cam.ac.uk/people/martin.monti/pdf/Monti_GLM_Critical_Review.pdf.

Mur, M., Bandettini, P. A., & Kriegeskorte, N.(2009). Revealing representational content with pattern-information fMRI–an introductory guide. *Social Cognitive and Affective Neuroscience*, *4*, 101–109.

Nichols, T. E., & Holmes, A. P.(2001). Nonparametric Permutation Tests For Functional Neuroimaging: A Primer with Examples. *Human Brain Mapping*, *15*, 1–25.

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*, 424–430.

Ogawa, S., Lee, T., Kay, A., & Tank, D.(1990). Brain magnetic resonance imaging with contrast dependent on blood oxygenation. *Proceedings of the National Academy of Sciences*, *87*, 9868–9872.

O'Toole, A. J., Jiang, F., Abdi, H., & Haxby, J. V.(2005). Partially Distributed Representations of Objects and Faces in Ventral Temporal Cortex . *Journal of Cognitive Neuroscience*, *17*, 580–590.

O'Toole, A. J., Jiang, F., Abdi, H., Penard, 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*, 1735–1752.

Pereira, F., Mitchell, T., & Botvinick, M.(2009). Machine learning classifiers and fMRI: A tutorial overview. *NeuroImage*, *45*, 199–209.

Perez, F., & Granger, B.(2007). IPython: A System for Interactive Scientific Computing. *Computing in Science & Engineering*, *9*, 21–29.

Pessoa, L., & Padmala, S. (2007). Decoding near-threshold perception of fear from distributed single-trial brain activation. *Cerebral Cortex*, *17*, 691–701.

Petersson, K., Nichols, T. E., Poline, J. B., & Holmes, A. P.(1999a). Statistical limitations in functional neuroimaging: II. Signal detection and statistical inference. *Philosophical Transactions of the Royal Society B: Biological Sciences*, *354*, 1261–1281.

Petersson, K., Nichols, T. E., Poline, J. B., & Holmes, A. P.(1999b). Statistical limitations in functional neuroimaging: I. Non-inferential methods and statistical models. *Philosophical Transactions of the Royal Society B: Biological Sciences*, *354*, 1239–1260.

Poldrack, R. A. (2006). Can cognitive processes be inferred from neuroimaging data? *Trends in Cognitive Sciences*, *10*, 59–63.

Rakotomamonjy, A. (2003). Variable Selection Using SVM-based Criteria. *Journal of Machine Learning Research*, *3*, 1357–1370.

Rasmussen, C. E., & Williams, C. K.(2006). *Gaussian Processes for Machine Learning*. MIT Press.

Rieger, J. W., Braun, C., Bülthoff, H. H., & Gegenfurtner, K. R.(2005). The dynamics of visual pattern masking in natural scene processing: A magnetoencephalography study. *Journal of Vision*, *5*, 275–286.

Rieger, J. W., Reichert, C., Gegenfurtner, K. R., Noesselt, T., Braun, C., Heinze, H.-J., et al. (2008). Predicting the recognition of natural scenes from single trial MEG recordings of brain activity. *NeuroImage*, *42*, 1056–1068.

Roy, C. S., & Sherington, C. S.(1890). The regulation of the blood supply of the brain. *Journal of Physiology*, *11*, 85–108.

Sally, S. L., & Kelly, J. B. (1988). Organization of auditory cortex in the albino rat: sound frequency. *Journal of Neurophysiology*, *59*, 1627–1638.

Schacter, D. L., Buckner, R. L., Koutstaal, W., Dale, A. M., & Rosen, B. R.(1997). Late onset of anterior prefrontal activity during true and false recognition: an event-related fMRI study. *NeuroImage*, *6*, 259–269.

Schapire, R. E. (2003). The boosting approach to machine learning: An overview. In D. D. Denison, M. H. Hansen, C. Holmes, B. Mallick, & B. Yu (Eds.), *Nonlinear Estimation and Classification*. New York: Springer.

Sigmund, D. O., & Worsley, K.(1995). Testing for a signal with unknown location and scale in a stationary Gaussian random field. *Annals of Statistics*, *23*, 608–639.

Sip, K. E., Roeppstorff, A., McGregor, W., & Frith, C. D.(2008a). Detecting deception: the scope and limits. *Trends in Cognitive Sciences*, *12*, 48–53.

Sip, K. E., Roeppstorff, A., McGregor, W., & Frith, C. D.(2008b). Response to haynes: There's more to deception than brain activity. *Trends in Cognitive Sciences*, *12*, 127–128.

Smith, S. M., Jenkinson, M., Woolrich, M. W., Beckmann, C. F., Behrens, T. E. J., Johansen-Berg, H., et al. (2004). Advances in functional and structural MR image analysis and implementation as FSL. *NeuroImage*, *23*, 208–219.

Sonnenburg, S., Braun, M., Ong, C. S., Bengio, S., Bottou, L., Holmes, G., et al. (2007). The Need for Open Source Software in Machine Learning. *Journal of Machine Learning Research*, *8*, 2443–2466.

Sonnenburg, S., Raetsch, G., Schaefer, C., & Schoelkopf, B.(2006). Large Scale Multiple Kernel Learning. *Journal of Machine Learning Research*, *7*, 1531–1565.

Sterzer, P., Haynes, J.-D., & Rees, G.(2008). Fine-scale activity patterns in high-level visual areas encode the category of invisible objects. *Journal of Vision*, *8*, 10.1–12.

Sun, Y.(2007). Iterative RELIEF for Feature Weighting: Algorithms, Theories and Applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *29*, 1035–1051.

Swets, J. A.(1996). *Signal detection theory and roc analysis in psychology and diagnostics: Collected papers.* Mahwah, NJ: Lawrence Erlbaum Associates.

Tanaka, K.(1996). Inferotemporal cortex and object vision. *Annual Review of Neuroscience*, *19*, 109–139.

Thulasidas, M., Guan, C., & Wu, J.(2006). Robust classification of EEG signal for brain-computer interface. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, *14*, 24–29.

Ungerleider, L. G., & Mishkin, M.(1982). Two cortical visual systems. In *Analysis of visual behaviour.* MIT Press.

Vanduffel, W., Tootell, R. B. H., Schoups, A. A., & Orban, G. A.(2002). The organization of orientation selectivity throughout macaque visual cortex. *Cerebral Cortex*, *12*, 647–662.

Vapnik, V.(1995). *The Nature of Statistical Learning Theory.* New York: Springer.

Veropoulos, K., Campbell, C., & Cristianini, C.(1999). Controlling the sensitivity of support vector machines. In *Proceedings of the International Joint Conference on AI.*

Woolrich, M. W., Behrens, T. E. J., & Smith, S. M.(2004). Constrained linear basis sets for hrf modelling using variational bayes. *NeuroImage*, *21*, 1748–1761.

Zell, A.(2004). *Simulation neuronaler Netze* (2 ed.). München: Oldenbourg.

Zou, H., & Hastie, T.(2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society Series B*, *67*, 301–320.