ARTICLE **Communicated by Francisco Pereira**

Comparing Classification Methods for Longitudinal fMRI Studies

Tanya Schmah
schmah@cs.toronto.edu
Department of Computer Science, University of Toronto, Toronto, Ontario, M5S 3G4, Canada

Grigori Yourganov
gyourganov@rotman-baycrest.on.ca
Rotman Research Institute of Baycrest Centre and Institute of Medical Science, University of Toronto, Toronto, Ontario, M6A 2E1, Canada

Richard S. Zemel
zemel@cs.toronto.edu
Geoffrey E. Hinton
hinton@cs.toronto.edu
Department of Computer Science, University of Toronto, Toronto, Ontario, M5S 3G4, Canada

Steven L. Small
small@uchicago.edu
Department of Neurology, University of Chicago, Chicago, IL 60637, U.S.A.

Stephen C. Strother
sstrother@rotman-baycrest.on.ca
Rotman Research Institute of Baycrest Centre, Department of Medical Biophysics, and Institute of Medical Science, University of Toronto, Toronto, Ontario, M6A 2E1, Canada

We compare 10 methods of classifying fMRI volumes by applying them to data from a longitudinal study of stroke recovery: adaptive Fisher's linear and quadratic discriminant; gaussian naive Bayes; support vector machines with linear, quadratic, and radial basis function (RBF) kernels; logistic regression; two novel methods based on pairs of restricted Boltzmann machines (RBM); and K-nearest neighbors. All methods were tested on three binary classification tasks, and their out-of-sample classification accuracies are compared. The relative performance of the methods varies considerably across subjects and classification tasks. The best overall performers were adaptive quadratic discriminant, support

vector machines with RBF kernels, and generatively trained pairs of RBMs.

1 Introduction

Pattern classification approaches to analyzing functional neuroimaging data have become increasingly popular (see, e.g., Morch et al., 1997; Haxby et al., 2001; Hanson, Matsuka, & Haxby, 2004; Mitchell et al., 2004; Strother et al., 2004, Haynes & Rees, 2005; Pessoa, Japee, Sturman, & Ungerleider, 2006; Hansen, 2007; Hanson & Halchenko, 2008, and the reviews: Norman, Polyn, Detre, & Haxby, 2006; O'Toole et al., 2007; Pereira, Mitchell, & Botvinick, 2009). The aim of these methods is to discriminate, on the basis of imaging data, among different conditions in the brain. From a biomedical point of view, there are two broad goals. The first is to find general classification methods that can eventually be applied in clinical settings, for example, to classify patients as likely responders or nonresponders to certain treatments. The second goal is to interpret the models built by the classification methods so as to give insight into underlying neural representations, for example, by highlighting the brain regions that are most informative with respect to particular experimental variables. This article focuses on classification accuracy as opposed to issues of neural representation.

Given a wide selection of available multivariate classification methods of varying complexity, which are the most appropriate for the specific application of longitudinal fMRI studies? Compared to other fMRI studies, longitudinal ones have larger numbers of volumes and a potentially higher degree of heterogeneity due to functional changes in the subjects' brains over the course of the study. These two properties suggest that the classifiers that are most accurate for longitudinal studies may not be those that are most accurate for single-session data. We cannot expect to definitively prescribe the best classifiers for longitudinal fMRI because of the variety of data sets and classification tasks. However, this article provides a starting point by comparing 10 classification methods on three binary classification tasks based on a longitudinal stroke recovery data set. The classification methods include some that have been widely applied to fMRI: linear and quadratic discriminants, gaussian naive Bayes, and support vector machines with three different kernels. We also include two methods based on restricted Boltzmann machines, a form of artificial neural network that has not before been applied to fMRI but has achieved excellent results in other domains (Hinton, 2007a). The remaining two methods tested are logistic regression and K-nearest neighbors. We do not use an initial feature selection step except for the linear and quadratic discriminants, which are applied to a subspace defined by principal components.

The classification tasks used for this comparison are deliberately chosen to be complex in the sense that for each task, each of the two classes is known to be heterogeneous (see section 2.2). Accuracy results could presumably be increased by simplifying the classification tasks. However, our aim is not to obtain the highest possible accuracy values, but to compare methods on tasks that more closely resemble the highly complex classification tasks one might expect in the setting of a multisubject clinical application.

The rest of the article is organized as follows. In section 2 we describe our data and methods, section 3 contains our results and analysis, and section 4 concludes with a discussion.

2 Methods

2.1 Data and Preprocessing. We use fMRI data from a unique longitudinal study of recovery from stroke described in Small, Hlustik, Noll, Genovese, and Solodkin (2002). That study analyzed mean volumes of activation over four regions of interest in each hemisphere. Twelve subjects were studied at 1, 2, 3, and 6 months after stroke. Due to data irregularities, we study only nine of the subjects in this article. Each of the four imaging sessions consisted of four continuous recording runs. During each run, the subject alternated two kinds of hand movement: tapping finger and thumb together or wrist flexion and extension, with rest breaks in between. The movement was paced auditorily at 1 Hz. Within each run, the experimental design was 12 seconds rest, 24 seconds finger tap, 12 seconds rest, and 24 seconds wrist flexion, repeated eight times. During each run, the subject moved one hand only, with the pattern of hand movement within each session being healthy hand, impaired hand, healthy, impaired. Whole-brain fMRI volumes, each consisting of 24 axial slices, were collected at the University of Maryland in a 1.5T scanner (slice thickness: 6 mm; pixel size: 1.88 mm $\times$ 1.88 mm; field of view (FOV): 240 $\times$ 240 $\times$ 144; flip angle (FA) = 60; echo time (TE) = 35 ms; repetition time (TR) = 4000 ms). In terms of the TR, the design of each run may be summarized as 3 TR rest, 6 TR finger tap, 3 TR rest, 6 TR wrist flexion), repeated eight times (see Figure 1).

The data for all nine subjects were coregistered and motion-corrected using 12-parameter affine transformations found using the Automated Image Registration (AIR) package (Woods, Grafton, Holmes, Cherry, & Mazziotta, 1998). A multiresolution, iterative approach was used, with a least squares cost function. After this, for computational ease, we retained only seven axial fMRI slices: slices 2, 3, 4, 5, 21, 22, and 23, where 24 is the top of the head. These slices were chosen to ensure coverage of the cerebellum and sensorimotor cortex. This results in 10,499 voxels. Next, for every active volume—those recorded during finger or wrist movement—we divided the intensity of each voxel by the mean of the corresponding voxel intensities in the previous two rest volumes. The purpose of this procedure is to normalize relative to a baseline, but in such a way that the baseline

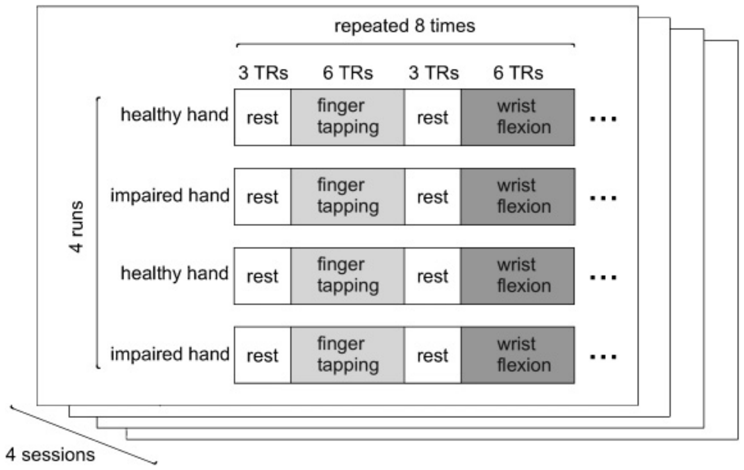

Figure 1: The data were collected during four sessions, each consisting of four continuous recording runs. During the first and third runs, the subject moved only the healthy hand, while during the second and fourth runs, the subject moved only the impaired hand. Within a run, experimental design was 12 seconds rest, 24 seconds finger tap, 12 seconds rest, 24 seconds wrist flexion, repeated eight times. The figure shows these times expressed as multiples of the TR, which was 4 seconds. Ignoring the rest volumes, there are 16 condition groups per run: 8 groups of finger tapping, interleaved with 8 groups of wrist flexion. Overall (across the runs and sessions) there are 256 condition groups. During preprocessing, the first finger or wrist volume in each group is discarded, leaving 5 volumes per condition group.

changes with temporal drift. Following this normalization, the rest volumes were discarded. This procedure has been widely used in neuroimaging as part of partial least squares (PLS) analysis (McIntosh & Lobaugh, 2004). The remaining normalized data are in groups of six consecutively recorded volumes, each corresponding to either finger or wrist movement. So as to reduce the hemodynamic response transients intrinsic to fMRI imaging, we discard the first volume in each group.

The data now consist of 256 condition groups of five consecutively recorded normalized volumes, each corresponding to either finger or wrist movement, resulting in 1280 volumes overall. Within each condition group, the five volumes were recorded consecutively, in the same run of the same session, under the same experimental conditions (i.e., the subject was making the same movement). Since the TR is 4 seconds, which is less than typical estimates of the latency of the BOLD signal, a high degree of correlation between volumes in each condition group is to be expected. The minimum separation in recording times between volumes in different condition

Table 1: Coding of the Classes in the Three Classification Tasks.

0	1
Early (E)	Late (L)
Healthy (H)	Impaired (I)
Finger (F)	Wrist (W)

groups is 16 seconds, which is longer than typical estimates for the latency of the BOLD signal. Thus, we expect at most weak correlations between volumes in different condition groups.

The final preprocessing step is to scale the data as follows, separately for each subject. First, for each voxel, we subtract the mean intensity of that voxel across all volumes for the given subject. Then we compute the standard deviation of each voxel's intensity, across all volumes for the given subject. We divide each voxel's intensity by the mean of the standard deviations (thus using the same scaling factor for all voxels). This scaling can be considered a robust version of scaling by pooled variance: more robust because it is less affected by voxels with exceptionally high or low variance.

2.2 Classification Tasks. We have chosen three classification tasks for comparison of the classification methods. For each task, the input is one fMRI volume, and the classifier assigns the volume into one of two classes. The first task is to predict whether the volume was recorded early in the study, defined as the first or second recording session (1 or 2 months post-stroke) or late in the study, defined as the third or fourth recording session (3 or 6 months post-stroke). The second task is to predict whether the volume was recorded during movement of the healthy or impaired hand. The third task is to predict whether the volume was recorded during finger or wrist movement (irrespective of which hand was used). For each of the three tasks, the two classes are coded as 0 and 1, as shown in Table 1.

As in any other fMRI study, there are many sources of variability between volumes that are extraneous to the classification task, including cardiac and respiratory physiological noise, fatigue, and attention to the task (Strother, 2006). In other respects, the classification tasks may seem simple, being based on simple hand movements. However, they are not necessarily simple, for two reasons. The first is that all of our subjects have lesions involving the motor system, so they are unlikely to be using the normal pathways to perform the motor tasks with the impaired hand. The second reason is that for each classification task, each of the two classes is known to be heterogeneous. For example, in the first classification task (early versus late), the early class is known to contain volumes in four subclasses: healthy finger movement, healthy wrist movement, impaired finger movement, and impaired wrist movement; and similarly for the late class.

The early-versus-late task is of particular interest from the point of view of elucidating neural changes that occur during recovery from stroke. Classification methods that are successful at this task must have identified changes that occur during the recovery process. In principle, this information sheds light on the changing neural representations during recovery, though extracting this information from the classifiers (i.e., opening the box), can be challenging. We touch on this issue with the activation maps in Figure 5 and associated discussion.

2.3 Validation and Testing Procedures. Data for each subject are treated separately. For each subject, the preprocessed data from all sessions are pooled, resulting in 1280 volumes. Note that pooling the data makes sense because the data have been normalized relative to rest volumes in a way that minimizes temporal drift, as explained in section 2.1. For each classification method and each classification task, we compare the methods in 20 trials, with each trial run on a different random splitting of the pooled data into three subsets: 75% training (960 volumes), 12.5% validation (160 volumes), and 12.5% test (160 volumes). The splittings of the data were created using a resampling method, with the sampling units being condition groups, which are groups of five normalized volumes recorded consecutively under the same experimental conditions (see section 2.1). By assigning all volumes in the same condition group to the same subset (training, validation, or test), we reduce the dependence between volumes in different subsets. The sampling method was chosen so as to create training, validation, and test sets that are balanced with respect to all experimental conditions, that is, for each of the three sets in a splitting, the set contains the same number of volumes corresponding to finger movement as to wrist movement and equal numbers of volumes from each run. The latter condition implies that each set contains equal numbers of volumes from each hand and equal numbers from early and late in the experiment.

The sampling method is illustrated in Figure 2. Recall that each run contains 16 condition groups: 8 corresponding to finger movement and 8 to wrist movement. We call a set of 8 condition groups in the same run corresponding to the same condition a half-run. From each half-run, one of the 8 condition groups is randomly chosen and assigned to the validation set, a second condition group is randomly chosen and assigned to the test set, and the remaining 6 are assigned to the training set. The training sets from all half-runs are merged into one large training set of 960 volumes, the validation sets are merged into a set of 160 volumes, and the test sets are merged into a set of 160 volumes. Finally, within each of the training, validation, and test sets, the volumes are randomly reordered, this time without regard to the condition groups. This splitting process is repeated 20 times with different random seeds, leading to 20 uncorrelated splittings.

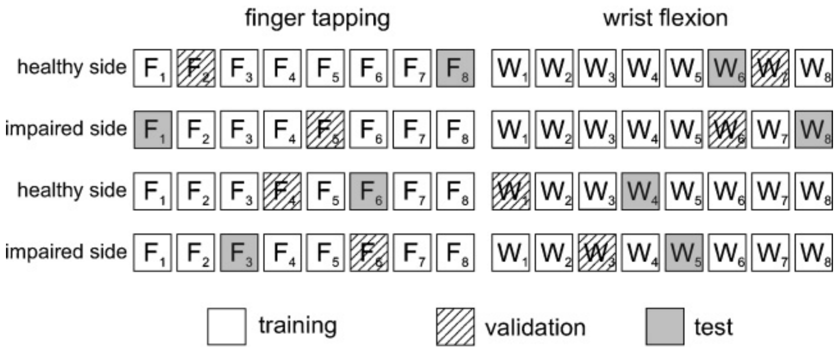

Figure 2: This diagram illustrates the method used to randomly split each subject's data into three groups: training, validation, and test. The data from one session are shown, with each row corresponding to a run. Each square represents a condition group, which is a group of five normalized volumes recorded consecutively under the same experimental conditions; the square is labeled F or W according to whether the group was recorded during finger tapping (F) or wrist flexion (W). During the experiment, these two conditions alternated, but in the diagram, all of the finger condition groups in one run are grouped into the half-run shown on the left, and all of the wrist condition groups are grouped into the half-run shown on the right. From each half-run, one condition group is randomly chosen for inclusion in the test set, and one condition group is randomly chosen for inclusion in the validation set, while the remaining six condition groups are assigned to the training set.

The purpose of the validation set is to tune hyperparameters before final testing of a method. Ideally, for each method and each random splitting of the data, the training set is used to build a family of classifiers that are identical except for certain hyperparameter values, and these classifiers are tested on the validation set in order to determine the optimal hyperparameter values. The classifier with the optimal hyperparameter values is then tested on the test set. This is the procedure that was followed for all of the methods that used an optimal-dimension principal component subspace: linear and quadratic discriminants and the "adaptive" version of naive Bayes. For logistic regression, K-nearest neighbors, and support vector machines with RBF kernels, a variation on this method was used. After the validation set was used to tune a hyperparameter, the validation set was merged with the training set, and this enlarged training set was used to build the final classifier with the optimal hyperparameter value. This was done because we determined that for these methods (but not for linear discriminant), this enlargement of the training set led to slightly higher accuracy. For the remaining methods—naive Bayes applied directly to voxels, support vector machines with linear and quadratic kernels, and both of the restricted Boltzmann machine (RBM) methods—we did not use the validation set to optimize hyperparameter values, so we merged the validation set into the training set. For naive Bayes (applied directly to voxels) and support vector machines with linear and quadratic kernels, no hyperparameter optimization was done. For the RBM methods (only), some informal hyperparameter optimization was done using all data for two of the subjects (see section 2.4.5 for details and an explanation of this approach).

All accuracy values reported are out-of-sample accuracy—the accuracy of the classifier when evaluated on a test set, after training (and optionally tuning) the classifier on the corresponding training and validation sets. Note that since each of the training, validation, and test sets contains volumes from every run of every session, the classification tasks involve only the weakest form of generalization.

2.4 Classification Methods. We compare several binary classification methods: adaptive Fisher's linear discriminant; adaptive quadratic discriminant; gaussian naive Bayes; support vector machines with linear, quadratic, and radial basis function kernels; logistic regression; two methods based on restricted Boltzmann machines; and K-nearest neighbors.

All of these methods produce a decision function, D, such that if $D(\mathbf{x}) < 0$ for a certain volume $\mathbf{x}$, then that volume is predicted to be in class 0, while if $D(\mathbf{x}) > 0$, it is predicted to be in class 1. (See Table 1 for the meaning of classes 0 and 1 for the three classification tasks.) Zero values of the decision function correspond to ambiguous cases of volumes that cannot be classified by the method, or volumes for which the method predicts equal probability of membership in each of the two classes.

For all of the methods except for support vector machines and K-nearest neighbors, the decision function can be interpreted as the log odds of the volume being in class 1:

$$D(\mathbf{x}) = \log\left(\frac{P(\text{class} = 1 \,|\, \mathbf{x})}{P(\text{class} = 0 \,|\, \mathbf{x})}\right). \tag{2.1}$$

This is equivalent to

$$P(\text{class} = 1 \,|\, \mathbf{x}) = \frac{1}{1 + \exp(-D(\mathbf{x}))}. \tag{2.2}$$

The right-hand side of the previous equation can be expressed in terms of the sigmoid function σ defined by $\sigma(z) = 1/(1 + \exp(-z))$, so that $P(\text{class} = 1 \,|\, \mathbf{x}) = \sigma(D(\mathbf{x}))$.

By Bayes's theorem,

$$P(\text{class} = 1 \,|\, \mathbf{x}) = \frac{P(\mathbf{x} \,|\, \text{class} = 1)\, P(\text{class} = 1)}{P(\mathbf{x})}. \tag{2.3}$$

We assume that, a priori, membership in each of the two classes is equally probable, that is, $P(\text{class} = 1) = P(\text{class} = 0) = 0.5$. (If this is not the case, the formulas presented here need minor modifications.) Under this assumption, substituting equation 2.3 and the analogous formula for class 0 into equation 2.1 gives an equivalent formula for $D(\mathbf{x})$:

$$D(\mathbf{x}) = \log\left(\frac{P(\mathbf{x}\,|\,\text{class} = 1)}{P(\mathbf{x}\,|\,\text{class} = 0)}\right). \tag{2.4}$$

We note that all of the probabilistic methods used in this study (i.e., all of the methods except for support vector machines and K-nearest neighbors) assume that the data points in each class are independent and identically distributed. In our application, the fMRI volumes are not independent. This is a potentially serious mismatch of method to data. However, as noted in section 2.1, it is more reasonable to assume independence between volumes in different condition groups. With this in mind, one approach would be to work only with the means of the condition groups. This would make the independence assumption more reasonable, but at the expense of losing some of the information in the data. It is unclear whether on balance this would improve our results, and we have chosen instead to always work with the individual volumes.

In the following descriptions of the classification methods, M is the number of voxels in each volume, N is the number of volumes, and each volume $\mathbf{x}$ will be treated as a single vector in $\mathbb{R}^M$.

2.4.1 Linear and Quadratic Discriminants. Both linear and quadratic discriminant analysis assume that, for each class, the volumes are sampled from multivariate gaussian distributions. For the quadratic discriminant (QD), no further assumptions are made. Substituting gaussian distributions into equation 2.4 leads to the following decision function (the discriminant), which is a quadratic function of $\mathbf{x}$:

$$\begin{aligned} D_{QD}(\mathbf{x}) = {} & \frac{1}{2}\log\frac{|S_0|}{|S_1|} - \frac{1}{2}(\mathbf{x}-\mathbf{m}_1)^T S_1^{-1}(\mathbf{x}-\mathbf{m}_1) \\ & + \frac{1}{2}(\mathbf{x}-\mathbf{m}_0)^T S_0^{-1}(\mathbf{x}-\mathbf{m}_0). \end{aligned}$$

Here, $\mathbf{m}_c$ and S_c are means and covariance matrices for class c, which are taken to be the sample means and covariances of the training data for class c. (See below for the case where one or both of the sample covariance matrices are not invertible.) Note that the factors of 1/2 could be cancelled to give an alternative decision function that still has the property that $D(\mathbf{x}) > 0$ for predicted class 1 and $D(\mathbf{x}) < 0$ for predicted class 0. However, in this case, D would no longer have the probabilistic interpretation in equation 2.2.

Table 2: Median Number of Principal Components Selected Using Validation Sets for the Classification Tasks: Early Versus Late (E/L), Healthy Versus Impaired (H/I), and Finger Versus Wrist (F/W).

	E/L	H/I	F/W	Overall Median
Linear discriminant	251	301	341	311
Quadratic discriminant	61	251	281	201
Naive Bayes (pooled-variance version)	241	306	351	311
Naive Bayes (distinct-variance version)	61	101	341	131

Note: Later figures and tables show results for the naive Bayes method applied directly to voxels, not using principal components.

Linear discriminant analysis (LD) is similar to quadratic discriminant analysis, but makes the extra assumption that the covariance matrices of the two classes are equal. In place of S_0 and S_1, we use S, the pooled sample covariance matrix (the mean of S_0 and S_1). The decision function simplifies to

$$D_{LD}(\mathbf{x}) = -\left(\mathbf{x} - \frac{1}{2}(\mathbf{m}_0 + \mathbf{m}_1)\right)^T S^{-1}(\mathbf{m}_0 - \mathbf{m}_1).$$

This function is linear in $\mathbf{x}$, and the equation $D_{LD}(\mathbf{x}) = 0$ defines a hyperplane that separates the two classes of vectors.

In our application, the number of voxels, M, is greater than the number of observations, N (the number of volumes), so the sample covariance matrices are always rank deficient and cannot be inverted. To avert this problem, we mean-center all data (using the grand mean of all training, validation, and test volumes), and then compute the first K principal components of the training data using the singular value decomposition (see, e.g., Mardia, Kent, & Bibby, 1979). We then apply LD and QD to the orthogonal projection of the centered data onto the first K principal components. The value of K is chosen to optimize the classification performance on a validation set. In our implementation, the K values tried were $1, 11, 21, \ldots, 471$, with the maximum value determined by the number of volumes per class in the training set, which is 480. We found that the optimal values of K ranged from 1 to 471. The median values for the three tasks are shown in Table 2. We call the combined method we have described, principal component analysis (PCA), followed by the application of LD (or QD) to an optimal-dimension PC subspace, adaptive LD (or QD).

2.4.2 Gaussian Naive Bayes. The gaussian naive Bayes (NB) classifier assumes that for each class, each component of the data vector has an independent gaussian distribution, so that

$$P(\mathbf{x} \mid \text{class} = c) = \prod_{i=1}^{M} \frac{1}{\sigma_{ci}\sqrt{2\pi}} \exp\left(-\frac{(x_i - \mu_{ci})^2}{2\sigma_{ci}^2}\right), \tag{2.5}$$

where μ_{ci} and σ_{ci}^2 are the mean and variance of the ith component, for class c. When we apply this method directly to the fMRI volumes, each component is a voxel.

Substitution into equation 2.4 leads to the decision function,

$$D_{NB}(\mathbf{x}) = \sum_{i=1}^{M} \log \frac{\sigma_{0i}}{\sigma_{1i}} - \sum_{i=1}^{M} \frac{(x_i - \mu_{1i})^2}{2\sigma_{1i}^2} + \sum_{i=1}^{M} \frac{(x_i - \mu_{0i})^2}{2\sigma_{0i}^2}.$$

There are two variants of this classifier. In the first, which we call the distinct-variance version, the variances σ_{0i}^2 and σ_{1i}^2 are allowed to be unequal, so they are estimated by the sample variances of component i in each of the two classes of training data. In the second version, which we call the pooled-variance version, the variance of component i is assumed to be the same for both classes, $\sigma_{0i}^2 = \sigma_{1i}^2$, and this is estimated by the pooled sample variance—the mean of the sample variances of the two classes. The pooled-variance model is also known as the general linear model (GLM), which is widely used in fMRI analysis (Friston, Ashburner, Kiebel, Nichols, & Penny, 2007). Note that the conventional use of the GLM in fMRI analysis is in inferential testing under the null hypothesis of no mean effect, whereas this article focuses on classification accuracy defined by posterior probability in a cross-validation framework.

An equivalent description of the naive gaussian class model is that each class has a multivariate gaussian distribution with a diagonal covariance matrix, that is, an axis-aligned gaussian distribution. Thus, the NB classifier with distinct variances is a special case of the quadratic discriminant, while the NB classifier with pooled variances is a special case of the linear discriminant.

The assumption that each voxel has an independent distribution is clearly false for fMRI volumes. Thus, the NB classifier is mismatched to our application, and we might anticipate that it will perform relatively poorly, in particular when compared to linear or quadratic discriminants, which are similar to NB but without this independence assumption. However, the smaller number of parameters of the NB model could be an advantage when the number of training examples is relatively small, as is our case.

We also evaluated an alternative version of the method, which we call adaptive naive Bayes: the NB classifier is applied to the orthogonal projection of the data onto the first K principal components. The number of principal components, K, was chosen so as to optimize performance of the

NB classifier; otherwise, the details of the optimization are as explained above for LD and QD. The median optimal K value for each of the tasks is shown in Table 2.

2.4.3 Support Vector Machines. Support vector machines (SVMs) have recently become popular as classifiers of fMRI volumes (LaConte, Strother, Cherkassky, Anderson, & Hu, 2005). Linear SVMs, like linear discriminants, separate the classes with a hyperplane:

$$D_{LSVM}(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b.$$

The vector $\mathbf{w}$ and the number b are found by minimizing the expression

$$\frac{1}{2}\,\|\mathbf{w}\|^2 + C\sum_{n=1}^{N} \xi_n^2, \tag{2.6}$$

subject to the constraints:

$$y_n\,(\mathbf{w} \cdot \mathbf{x}_n + b) \geq 1 - \xi_n, \quad \text{for } n = 1 \ldots N,$$

where $\mathbf{x}_1, \ldots, \mathbf{x}_N$ are the training examples and y_n is -1 for training vectors in class 0 and 1 for vectors in class 1. (Though b does not appear in the expression to be minimized, it is adjusted along with $\mathbf{w}$ and $\xi_1, \ldots, \xi_N$ during the optimization.) This optimization problem has a unique solution, which can be found by quadratic programming. The solution has the property that $\mathbf{w}$ is a linear combination of the training vectors, and in fact,

$$\mathbf{w} = \sum_{n=1}^{N} \alpha_n y_n \mathbf{x}_n,$$

for some $\alpha_n \geq 0$. The vectors $\mathbf{x}_n$ for which $\alpha_n > 0$ are called support vectors.

The quantities ξ_n in equation 2.6 are called slack variables. If the two classes are linearly separable, then it is possible to require that all slack variables be zero, a case referred to as hard margin SVM. The hyperparameter C is a trade-off parameter sometimes called the penalty factor. High C values force slack variables to be smaller, approximating the behavior of hard margin SVM. The quantity $2/\|\mathbf{w}\|$ is called the margin. When all slack variables are zero, the margin is the width of the gap between the classes. Nonzero slack variables allow a degree of violation of the margin, in the sense that the corresponding training examples are allowed to be inside the gap defined by the hyperplane and the margin. Slack variables greater than 1 allow some training examples to be misclassified.

When the number of training points is less than or equal to the dimension of the space containing them, the case for the data set in this article, it is always possible to find a hyperplane that separates the two classes. However, even in this case, the slack variables may play a useful regularizing role by allowing a different choice of hyperplane that gives a wider margin that is violated only by a small number of points.

In nonlinear SVM methods, vectors are mapped to a high-dimensional feature space: $\mathbf{z} = g(\mathbf{x})$. The decision function is based on a hyperplane in the feature space,

$$D_{Feat}(\mathbf{z}) = \mathbf{w} \cdot \mathbf{z} + b = \sum_{n=1}^{N} \alpha_n y_n \mathbf{z}_n \cdot \mathbf{z} + b.$$

For any function $g(\mathbf{x})$, we can write the decision function in terms of the kernel function $K(\mathbf{x}, \mathbf{y}) = g(\mathbf{x}) \cdot g(\mathbf{y})$, as follows:

$$D_{Kernel}(\mathbf{x}) = \sum_{n=1}^{N} \alpha_n y_n g(\mathbf{x}_n) \cdot g(\mathbf{x}) + b = \sum_{n=1}^{N} \alpha_n y_n K\left(\mathbf{x}_n, \mathbf{x}\right) + b.$$

We use the quadratic kernel $K(\mathbf{x}, \mathbf{y}) = \mathbf{x} \cdot \mathbf{y} + 1$, as well as the radial basis function (RBF) kernel $K(\mathbf{x}, \mathbf{y}) = \exp\left(-\gamma \|\mathbf{x} - \mathbf{y}\|^2\right)$, in which γ is a hyperparameter. The corresponding decision functions are

$$D_{QSVM}(\mathbf{x}) = \sum_{n=1}^{N} \alpha_n y_n \left(\mathbf{x}_n \cdot \mathbf{x} + 1\right)^2 + b,$$

$$D_{RSVM}(\mathbf{x}) = \sum_{n=1}^{N} \alpha_n y_n \exp\left(-\gamma \|\mathbf{x}_n - \mathbf{x}\|^2\right) + b.$$

The parameters α_n and b are found by minimizing the same expression as in linear SVMs (subject to the same constraints), only now it is done in feature space, using the transformed points $\mathbf{z}_n = g(\mathbf{x}_n)$ in place of the original points $\mathbf{x}_n$.

We used SVM-light, an efficient implementation of the SVM methods (Joachims, 1999), with Matlab wrappers written by Tom Briggs. We initially tuned the hyperparameter C using validation sets but noticed that for all kernels, subjects, tasks, and kernel hyperparameter settings (in the case of the RBF kernel), out-of-sample classification accuracy is either a sigmoid, or near-sigmoid, function of C, with optimal or near-optimal accuracy usually obtained for any C value above a certain threshold. For the quadratic kernel, all C values above 10^{-5} gave optimal median accuracy (median across all 20 random splittings of the data) in all but two subject-task combinations.

For the RBF kernel, when C and the kernel hyperparameter γ are jointly optimized (we tested only C values 0.1, 1, 10, 100), both $C = 10$ and $C = 100$ were found to give identical and optimal median accuracies for 19 out of 27 subject-task combinations. For the linear kernel, optimal median accuracies were obtained at C values between 10^{-5} and 10^{-2}. However, for $C > 10^{-2}$, median accuracy was constant and usually no worse than 0.01 below the optimal, with the exception of 4 subject-task combinations.

In the results reported here, C was always set to 10^{30}. For the linear kernel, we checked that the policy of setting C to 10^{30} led to higher or equal median test accuracies, for most subjects and tasks (21 subject-task combinations out of 27), than optimizing C on the validation set. For the remaining 6 subject-task combinations, the difference in median test accuracies was at most 0.025, and it was less than 0.004 in four cases. For the linear and quadratic kernels, since we fixed the C value, no hyperparameter optimization was used. For the RBF kernel, the hyperparameter γ was optimized using validation sets, resulting in median optimal values between 10^{-5} and 10^{-4} for all but 2 subject-task combinations. The resulting numbers of support vectors are high for all kernels and tasks, ranging from 383 to 851, out of 1120 data points (training plus validation sets merged).

2.4.4 Logistic Regression. The object of logistic regression is to find a weight vector $\mathbf{w} \in \mathbb{R}^M$ and a scalar bias b, such that the following is a good decision function:

$$D_{LOG}(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b. \tag{2.7}$$

This decision function implies the following prediction of the class labels,

$$P(\text{class } = 1 \mid \mathbf{x}) = \sigma(D_{LOG}(\mathbf{x})) = \sigma(\mathbf{w} \cdot \mathbf{x} + b),$$

where $\sigma(z) = 1/(1 + \exp(-z))$ (see equation 2.2). For every volume $\mathbf{x}_n$ in the training set, let

$$p_n = P(\text{class } = 1 \mid \mathbf{x}_n) = \sigma(\mathbf{w} \cdot \mathbf{x}_n + b),$$

and let c_n be the actual class of $\mathbf{x}_n$. The likelihood of $\mathbf{w}$ and b given the class labels is

$$\begin{aligned}\prod_{n=1}^{N} P(c_n \mid \mathbf{x}_n) &= \prod_{n=1}^{N} \begin{cases} p_n, & \text{if } c_n = 1 \\ 1 - p_n, & \text{if } c_n = 0 \end{cases} \\ &= \prod_{n=1}^{N} p_n^{c_n}(1 - p_n)^{1-c_n}.\end{aligned}$$

To find the maximum likelihood estimate of $\mathbf{w}$ and b, we minimize a loss function that is the negative logarithm of the likelihood,

$$\begin{aligned} L &:= -\sum_{n=1}^{N} \log P(c_n \mid \mathbf{x}_n; \mathbf{w}, b) \\ &= \sum_{n=1}^{N} [-c_n \log p_n - (1 - c_n) \log(1 - p_n)]. \end{aligned} \tag{2.8}$$

To guard against overfitting, we add an L1 regularization term $\alpha \sum_{i=1}^{M} |w_i|$ to this loss function, with the parameter α tuned using the validation data. We found that this gave slightly better classification accuracy than L2 regularization. Note that with L2 regularization, logistic regression is a form of ridge regression. One interpretation of regularization terms is that they correspond to prior distributions over the weights, so that minimizing the regularized loss function produces a maximium a posteriori (MAP) estimate.

With or without regularization, the loss function is convex and may easily be numerically minimized, for example, by gradient descent. We have instead used Polack-Ribiere conjugate gradient descent with quadratic-cubic line search and Wolfe-Powell stopping criteria (Press, Teukolsky, Vetterling, & Flannery, 1992), as implemented by Carl Rasmussen's minimize.m (available online at http://www.kyb.tuebingen.mpg.de/bs/people/carl/code/minimize).

We found that the median (across splittings of the data) of the optimal L1 regularization hyperparameter α varied across tasks and subjects, but was always between 10^{-5} and 0.2.

2.4.5 Restricted Boltzmann Machines. A restricted Boltzmann machine (RBM) is a two-layer network of random variables in which a layer of visible units is connected to a layer of hidden units using symmetrically weighted connections, with no inter-unit connections within layers (Hinton, 2002; Hinton, 2007a). In our application, the visible units of the RBM correspond to voxels, while the hidden units can be thought of as feature detectors. In the typical RBM, both visible and hidden units are binary variables, but we use a version in which the visible units are continuous and have gaussian distributions, conditional on the hidden units (Welling, Rosen-Zvi, & Hinton, 2005; Hinton & Salakhutdinov, 2006; Bengio, Lamblin, Popovici, & Larochelle, 2007).

The energy of the joint configuration $(\mathbf{v}, \mathbf{h})$ of the visible and hidden units is

$$E(\mathbf{v}, \mathbf{h}) := -\sum_{i,j} \frac{v_i}{\sigma_i} w_{ij} h_j - \sum_{j} c_j h_j + \sum_{i} \frac{(v_i - b_i)^2}{2\sigma_i^2}, \tag{2.9}$$

where $w_{ij}, b_i, c_j, \sigma_i$ are fixed parameters. The joint distribution over visible and hidden variables is

$$P(\mathbf{v}, \mathbf{h}) := \frac{1}{Z} \exp\left(-E(\mathbf{v}, \mathbf{h})\right), \tag{2.10}$$

with partition function $Z := \int d\mathbf{u} \sum_{\mathbf{g}} \exp\left(-E(\mathbf{u}, \mathbf{g})\right)$. This implies the following conditional distributions,

$$P(v_i|\mathbf{h}) = \mathcal{N}\left(\mu_i\,, \sigma_i^2\right), \qquad \text{where } \mu_i = b_i + \sigma_i \sum_j w_{ij} h_j, \tag{2.11}$$

and $\mathcal{N}(\mu_i, \sigma_i^2)$ is the gaussian distribution with mean μ_i and variance σ_i^2, and

$$P(h_j = 1|\mathbf{v}) = \sigma\left(\sum_i \frac{v_i}{\sigma_i} w_{ij} + c_j\right), \tag{2.12}$$

where σ is the logistic function, $\sigma(z) := 1/1 + \exp(-z)$. Note that the conditional probabilities of the hidden units are the same as for binary-only RBMs.

The marginal distribution over the visible units can be expressed as

$$P(\mathbf{v}) = \sum_{\mathbf{h}} P(\mathbf{v}, \mathbf{h}) = \frac{1}{Z} \exp\left(-F(\mathbf{v})\right), \tag{2.13}$$

where F is the free energy:

$$\begin{aligned} F(\mathbf{v}) &= -\log\left(\sum_{\mathbf{h}} \exp\left(-E(\mathbf{v}, \mathbf{h})\right)\right) \\ &= -\sum_j \log\left(1 + \exp\left(\sum_i \frac{v_i}{\sigma_i} w_{ij} + c_j\right)\right) + \sum_i \frac{(v_i - b_i)^2}{2\sigma_i^2}. \end{aligned} \tag{2.14}$$

The marginal distribution over the visible units is typically intractable because of the partition function Z. However, Gibbs sampling can be used to sample from an approximation to the marginal distribution, since the conditional probability distributions $P(\mathbf{v}|\mathbf{h})$ and $P(\mathbf{h}|\mathbf{v})$, given by equations 2.11 and 2.12, are tractable.

The aim of generative training of an RBM is to model the marginal distribution of the visible units $P(\mathbf{v})$ by adjusting the parameters w_{ij}, b_i, c_j. In this study, we fix the variances σ_i^2, but these could also be adjusted, as in

Bengio et al. (2007). In maximum likelihood learning, the aim is to minimize the negative log probability of the training data,

$$L_{\mathrm{gen}} = -\sum_{\mathbf{v}\in\mathcal{S}} \log P(\mathbf{v}|\theta), \tag{2.15}$$

where $\mathcal{S}$ is the training set and θ is the vector of all parameters w_{ij}, b_i, c_j. The gradient of this function is intractable; however, there is an approximation to maximum likelihood learning, called contrastive divergence, which works well in practice (Hinton, 2002). We use an n-step version of contrastive divergence in which, at each iteration, the parameter increments are

$$\begin{aligned}
\Delta w_{ij} &= \frac{\varepsilon}{\sigma_i}\left(\langle v_i h_j\rangle_0 - \langle v_i h_j\rangle_n\right),\\
\Delta b_i &= \frac{\varepsilon}{\sigma_i^2}\left(\langle v_i\rangle_0 - \langle v_i\rangle_n\right),\\
\Delta c_j &= \varepsilon\left(\langle h_j\rangle_0 - \langle h_j\rangle_n\right).
\end{aligned}$$

In this definition, angle brackets denote an expected value with respect to a certain joint distribution of $\mathbf{v}$ and $\mathbf{h}$. The subscript 0 indicates that the data distribution is used for the visible units, that is, they take values corresponding to observed fMRI volumes, with the hidden units distributed according to the conditional distribution $P(\mathbf{h}|\mathbf{v})$. A subscript n indicates that n steps of Gibbs sampling have been done, beginning at data points, to give an approximation to the joint distribution $P(\mathbf{v}, \mathbf{h})$. The constant ε is called the learning rate.

Classification Using a Pair of Generatively Trained RBMs. We begin by generatively training two RBMs—one for each data class. We use the same hyperparameters for both RBMs, though this is not required by the method. For maximum likelihood learning, the loss function is the negative log probability of the training data:

$$\begin{aligned}
L_{\mathrm{gen}} &= L^{A}_{\mathrm{gen}}(\mathcal{S}_A) + L^{B}_{\mathrm{gen}}(\mathcal{S}_B)\\
&:= -\sum_{\mathbf{v}\in\mathcal{S}_A} \log P(\mathbf{v}|\theta_A) - \sum_{\mathbf{v}\in\mathcal{S}_B} \log P(\mathbf{v}|\theta_B),
\end{aligned} \tag{2.16}$$

where $\mathcal{S}_A$ and $\mathcal{S}_B$ are the training data from classes A and B, and θ_A and θ_B are the parameter vectors for the two RBMs. In practice, we approximate gradient descent of this loss function by contrastive divergence, as explained above. We also regularize by adding a term to L_{gen} that corresponds to putting a prior distribution on the weights w_{ij}.

In general, given probabilistic generative models for each of two classes, A and B, data can be classified by Bayes's theorem. For brevity, we write A for "$\mathbf{v}$ is of class A," and similarly for B. We assume that $\mathbf{v}$ is a priori equally likely to belong to both classes, so Bayes's theorem implies that

$$P(A|\mathbf{v}) = \frac{P(\mathbf{v}|A)}{P(\mathbf{v}|A) + P(\mathbf{v}|B)}. \tag{2.17}$$

If the distributions $P(\mathbf{v}|A)$ and $P(\mathbf{v}|B)$ are defined by RBMs, they can be expressed in terms of free energies F_A and F_B and partition functions Z_A and Z_B, as in equation 2.13. Substituting this into equation 2.17 gives

$$P(A|\mathbf{v}) = \sigma\left(F_B(\mathbf{v}) - F_A(\mathbf{v}) - T\right), \tag{2.18}$$

where $T := \log(Z_A/Z_B)$. The free energies in this formula can be calculated easily using equation 2.14. However, the partition functions are intractable for RBMs with large numbers of hidden and visible units. For this reason, we replace the unknown "threshold" T with an independent parameter Δ and fit it discriminatively. (Thus, this method is not pure generative training.) The aim of discriminative training is to model the conditional probability of the class labels given the visible units. In maximum likelihood learning, the loss function to be minimized is the negative log-conditional probability of the class labels of the training data, which is

$$\begin{aligned} L_{\text{disc}} &= -\sum_{\mathbf{v}\in\mathcal{S}} \log P(\text{class of } \mathbf{v}|\mathbf{v}, \theta_A, \theta_B, \Delta) \\ &= -\sum_{\mathbf{v}\in\mathcal{S}_A} \log\sigma\left(F_B(\mathbf{v}) - F_A(\mathbf{v}) - \Delta\right) \\ &\quad - \sum_{\mathbf{v}\in\mathcal{S}_B} \log\sigma\left(\Delta + F_A(\mathbf{v}) - F_B(\mathbf{v})\right). \end{aligned} \tag{2.19}$$

Classification via Purely Discriminative Training. As an alternative to generative training, the function L_{disc} (defined in the previous equation) can be minimized directly, with respect to all parameters simultaneously: the w_{ij}, b_i, and c_j of both RBMs and the threshold parameter Δ. We emphasize that this method uses exactly the same model of $P(\text{class of } \mathbf{v}|\mathbf{v})$ as before, summarized in equation 2.18; only the training method changes.

By substituting equation 2.14 into equation 2.19, the gradient of L_{disc} with respect to all parameters can be calculated exactly. Indeed, since $\frac{d}{dz}\log\sigma(z) = \sigma(-z)$, the partial derivative of L_{disc} with respect to the threshold parameter is

$$\frac{\partial L_{\text{disc}}}{\partial\Delta} = \sum_{\mathbf{v}\in\mathcal{S}_A} \sigma\left(\Delta + F_A(\mathbf{v}) - F_B(\mathbf{v})\right) - \sum_{\mathbf{v}\in\mathcal{S}_B} \sigma\left(F_B(\mathbf{v}) - F_A(\mathbf{v}) - \Delta\right).$$

The free energies depend on the weights of the RBMs (suppressed in the above equation for ease of notation). If the parameters for the two RBMs are not linked in any way, then any given parameter θ affects either model A or model B but not both, so either $\partial F_B/\partial\theta = 0$ or $\partial F_A/\partial\theta = 0$. From equation 2.14 we have,

$$\frac{\partial}{\partial w_{ij}} F(\mathbf{v}) = -\frac{v_i}{\sigma_i} p_j, \quad \frac{\partial}{\partial c_j} F(\mathbf{v}) = -p_j, \quad \frac{\partial}{\partial b_i} F(\mathbf{v}) = \frac{1}{\sigma_i^2}(b_i - v_i),$$

where $p_j := \sigma(z_j) = P(h_j = 1|\mathbf{v})$. It follows that setting $M(\mathbf{v}) := \sigma(F_B(\mathbf{v}) - F_A(\mathbf{v}) - \Delta)$, the derivatives for the parameters of model A are

$$\frac{\partial L_{\text{disc}}}{\partial w_{ij}} = \sum_{\mathbf{v}\in\mathcal{S}_A} (1 - M(\mathbf{v}))\left(-\frac{v_i}{\sigma_i} p_j\right) + \sum_{\mathbf{v}\in\mathcal{S}_B} M(\mathbf{v})\left(\frac{v_i}{\sigma_i} p_j\right),$$

$$\frac{\partial L_{\text{disc}}}{\partial c_j} = \sum_{\mathbf{v}\in\mathcal{S}_A} (1 - M(\mathbf{v}))\left(-p_j\right) + \sum_{\mathbf{v}\in\mathcal{S}_B} M(\mathbf{v})\left(p_j\right),$$

$$\frac{\partial L_{\text{disc}}}{\partial b_i} = \frac{1}{\sigma_i^2}\sum_{\mathbf{v}\in\mathcal{S}_A} (1 - M(\mathbf{v}))(b_i - v_i) + \frac{1}{\sigma_i^2}\sum_{\mathbf{v}\in\mathcal{S}_B} M(\mathbf{v})(v_i - b_i).$$

The formulas for model B are the same with opposite signs.

The factors $M(\mathbf{v})$ and $(1 - M(\mathbf{v}))$ can be thought of as magnification factors that are largest for vectors that are most likely to be misclassified by the current pair of models. This focusing of training effort on cases that are modeled most poorly can be considered a form of boosting. We note that discriminative training of a single RBM was suggested in Hinton (2007b) and Larochelle and Bengio (2008). Generative and discriminative training of a pair of RBMs, as well as blends of the two kinds of training, are studied in (Schmah, Hinton, Zemel, Small, & Strother, 2009) using the same data set as in this article.

Implementation Details. For generative training, we used 3500 groups of 30 hidden units. Within each group, the same weights were used for all units in the group. The large number of hidden units was necessary for numerical stability, and the grouping was done to ease the large memory and time requirements of the computations. For discriminative training, we used the same strategy with 3500 groups of 10 hidden units. For both kinds of training, we used a variance $\sigma_i = 1/\sqrt{2}$ for each of the visible units.

For generative training we used six-step contrastive divergence, with a variable learning rate that was initialized to 10^{-6} and adjusted during training according to the similarity of successive increment vectors, always remaining between 7×10^{-8} and 10^{-5}. We smoothed successive increments by using a momentum constant: at each step, the previous increment is multiplied by this constant and added to the current increment. The momentum constant is adjusted during training but is typically between 0.9 and 0.99. For discriminative training, we used gradient descent, with a learning rate that decreases on a fixed schedule from 2×10^{-5} to 2×10^{-6}, and a momentum that increases on a fixed schedule from 0.5 to 0.9. We had originally used conjugate gradient minimization (described in section 2.4.4) for discriminatively trained RBMs. However, we found that gradient descent ran faster and gave better results. We have not investigated this, beyond numerical verification of our gradients, but it suggests that care should be taken using conjugate gradient with very high-dimensional data.

For both generative and discriminative training, all of the weights w_{ij} were Cauchy regularized, meaning that for both RBMs, the following term was added to the loss function,

$$-\sum_{i,j} \log\left(\gamma / \left(\pi\left(\gamma^2 + w_{ij}^2\right)\right)\right),$$

where γ is an additional hyperparameter that we set to 30 during generative training and 10 during discriminative training. In informal experiments, we found that L1 regularization gave similar but slightly poorer results, while L2 regularization gave poorer results.

The implementation of restricted Boltzmann machines involves many hyperparameters, including the number of hidden units, the regularization hyperparameter, and the learning rates. Given the high computational costs (see Table 3, as well as Figure 9 in the Supplemental Material) it was not feasible to optimize these parameters using a formal validation procedure.[1] Instead, we experimented informally using all data from two subjects (S103 and S044) in order to empirically decide on hyperparameter settings that were then used on all subjects. We included these two subjects in our final test results, thus contaminating the desired training-validation-test set separation. It is reassuring that our final test results for these two subjects were not better than average (see Figures 6, 7, and 8 in the Supplemental Material).

2.4.6 K-Nearest Neighbors. A very simple classification method is to assign every test volume to the same class as the training volume that is nearest to it, with respect to a chosen metric in image space. We use the L2 (Euclidean) norm, applied in the 10,499-dimensional space of all preprocessed voxels. Rather than using the single nearest neighbor, we use K neighbors. The hyperparameter K, chosen from the range $1, \ldots, 10$, was tuned using the validation sets. The median value of K chosen was 3 for the

[1]Supplemental material referred to throughout the article is available online at http://www.mitpressjournals.org/doi/suppl/10.1162/NECO_a_00024.

Table 3: Median Out-of-Sample Classification Accuracy.

Method	All Tasks Pooled	E/L	H/I	F/W	Method Type	Run Time
RBF-SVM	0.944	0.969	0.875	0.944	Nonlinear	38 seconds
Quadratic Discriminant	0.938	1.000	0.912	0.884	Nonlinear	49 seconds
RBM-GT	0.938	1.000	0.887	0.884	Nonlinear	7.4 hours
RBM-DT	0.925	0.950	0.850	0.938	Nonlinear	2.7 hours
Quadratic SVM	0.912	0.975	0.831	0.894	Nonlinear	55 seconds
Linear SVM	0.900	0.863	0.841	0.950	Linear	102 seconds
K-nearest neighbors	0.881	0.906	0.762	0.897	Nonlinear	0.84 seconds
Linear discriminant	0.875	0.831	0.816	0.944	Linear	42 seconds
Logistic regression	0.859	0.819	0.803	0.931	Linear	5.7 minutes
NB (distinct variance)	0.722	0.850	0.700	0.625	Nonlinear	0.99 seconds
NB (pooled variance)	0.713	0.722	0.713	0.706	Linear	0.99 seconds

Notes: The first numerical column shows the median accuracy of each classification method, pooling results from all three classification tasks and all nine subjects; this column is used to sort the rows in the table. The next three columns show the median accuracy, still for all subjects pooled, but for each of the three tasks separately: early versus late, healthy versus impaired, and finger versus wrist. The entries in these three columns are the same values as those shown by the diamond-shaped markers in Figure 3. The "method type" column indicates whether a method is linear or nonlinear. The final column shows run times for a single trial, on one task and one subject, for fixed parameter values.

healthy-versus-impaired task, and 1 for the other two tasks. Let $\mathbf{x}_1, \ldots, \mathbf{x}_K$ be the nearest neighbors, in increasing order of distance from the test volume. Each class is given a score by the following formula:

$$\text{Score for class } C = \sum_{k=1}^{K} \left(1 + \epsilon^k\right) m(k, C),$$

where $m(k, C)$ equals 1 if $\mathbf{x}_k$ is a member of class C, and 0 otherwise. The test volume is assigned to the class that has the highest score. Thus, the decision function for this method is

$$D_{KNN} = \begin{cases} 1, & \text{if class 1 has the highest score,} \\ -1, & \text{if class 0 has the highest score.} \end{cases}$$

The value of ϵ is irrelevant as long as it is less than or equal to $\frac{1}{2}$, which implies that $\sum_{i=k}^{K} \epsilon^k < 1$. The factor ϵ^k is included as a tie-breaker: it guarantees that there will always be a unique highest score, and it does so by giving more weight to nearer neighbors.

2.5 Performance Comparisons with Nonparametric Tests. The Friedman test (Conover, 1998) is a nonparametric test for comparing the performance of multiple methods on multiple data sets. The test addresses the hypothesis that all methods perform equally well. We used the implementation of the Friedman test in the Matlab Statistics Toolbox. For each task, method, and subject, we calculated the median classification accuracy over all 20 random splits of the data. For each task and subject, we ranked the methods from 1 to 10 according to their median accuracy, with rank 1 being the most accurate method. Whenever two methods were tied, say for ranks r and $r + 1$, they were assigned a common rank of $r + 0.5$. For each task, the Friedman test was performed on the resulting 10×9 matrix of ranks in which each row corresponds to a method and each column to a subject.

If the Friedman test rejects the hypothesis that all methods perform equally well, we then proceed to the post hoc pairwise Friedman test as described in Conover (1998) and Demšar (2006). This test is performed on the mean ranks for each method, pooling all subjects. Thus, for each task, we are comparing 10 mean ranks. Given a significance level, which we choose to be 0.05, the test results in a critical difference (CD). The interpretation of this number is such that whenever two mean ranks differ by less than the CD, the corresponding methods are not significantly different. Following Demšar (2006), the CD is defined in such a way as to take into account the 45 possible pairwise comparisons of methods: the probability of a type 1 error in one or more of the comparisons is less than 0.05. Note that this is a very conservative test, as it is debatable whether we are really interested in making 45 pairwise comparisons.

3 Results

The classification accuracies of all of the methods are summarized in Figure 3, which shows median out-of-sample accuracy for each method applied to each classification task, for both individual subjects and all subjects pooled. The pooled median accuracy for each task is also shown in Table 3, as is the overall median accuracy obtained when all subjects and all classification tasks are pooled. Per-subject results are shown in more detail, in box-whisker plots, in Figures 6 to 8 in the Supplemental Material. Note that the intrasubject spread of accuracies, as shown by interquartile ranges in Figures 6 to 8, is typically no larger than the intersubject range of medians shown in Figure 3, and is often much smaller, especially for the healthy-versus-impaired task.

The results shown for naive Bayes are for that method applied directly to voxels. There are two versions of this method: pooled variance (linear) and distinct variance (nonlinear). Results for both versions are shown in Table 3. Note that for the early-versus-late task, the distinct-variance version has the higher median accuracy, while for the other two tasks, the pooled-variance version has the higher accuracy. In Figure 3 and subsequent figures, results are shown only for the version that gives the highest per-task median accuracy.

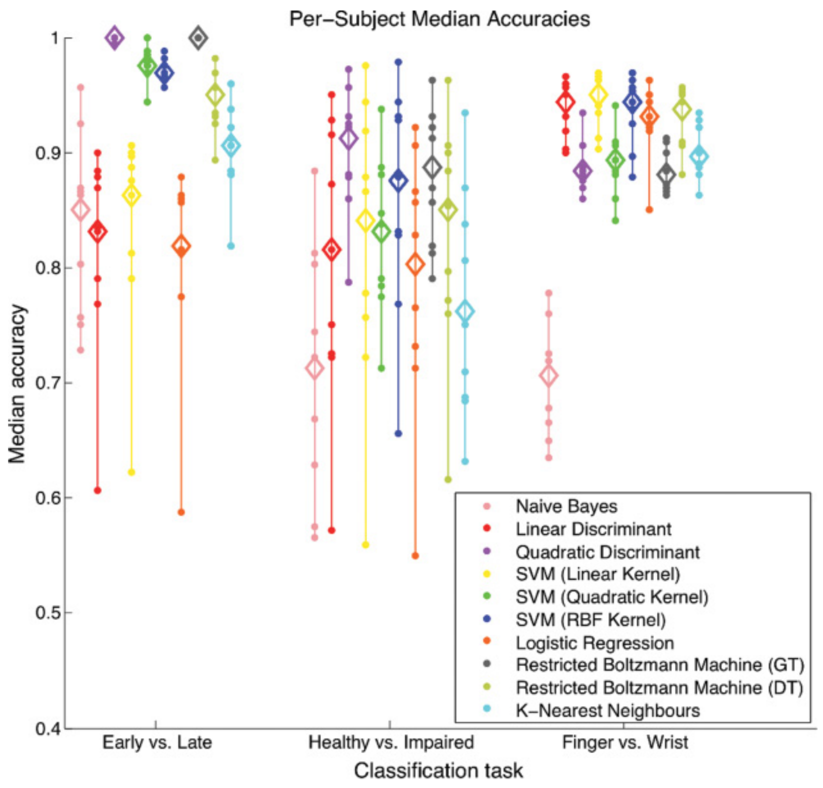

Figure 3: Out-of-sample classification accuracy of 10 different methods (listed in legend) for each of three classification tasks (shown on horizontal axis). For each method-task combination, the nine small colored dots, joined by a vertical line, show the median accuracy for individual subjects. The colored diamond on the same line shows the pooled median accuracy, calculated from pooled data from all nine subjects. Warmer colors are used for linear methods and cooler colors for nonlinear ones. For naive Bayes, the accuracies shown are the higher of two versions of the method. See the text for details.

We also tested "adaptive" versions of the naive Bayes method (i.e., naive Bayes applied to a projection of the data onto an optimal number of principal components, as explained in section 2.4.2). For all tasks, the pooled-variance (linear) version of the method outperformed the distinct-variance (nonlinear) version. The per-task median accuracies for the pooled-variance version were as follows: early versus late, 0.838; healthy versus impaired, 0.819; and finger versus wrist, 0.938. Since these values are all within 0.007 of the corresponding values for adaptive linear discriminant, we have not shown the results in either the tables or the figures. Note that for the healthy-versus-impaired and finger-versus-wrist tasks, the adaptive naive Bayes method performed much better than naive Bayes applied directly to voxels; however, this was not true for the early-versus-late task, where naive Bayes was slightly more accurate when applied directly to voxels. The latter result seems counterintuitive but may be due to the fact that when the method was applied directly to voxels, the classifier was trained on an enlarged training set ("training" plus "validation" sets merged), which was not done for the adaptive version of the method, since the validation set was used for tuning the number of principal components used.

Adaptive linear discriminant, linear SVM, and logistic regression (another linear method) all perform better on the finger-versus-wrist task than on either of the other tasks, which is not true of any of the nonlinear methods. This pattern of relative performance can be seen in Table 3, but is most easily seen from the relative position of the markers in Figure 9 in the Supplemental Material. The pattern fails only for the pooled-variance version of naive Bayes, a linear method for which performance on all tasks is similar.

To further compare the methods, we applied the nonparametric Friedman test, as described in section 2.5. We consider gaussian naive Bayes to be one method: for each task, we use the best-performing of the two variants of the method. For all three tasks, the p value was less than 10^{-5}, so the methods are significantly different. Specifically, the Friedman test rejects the hypothesis that all 10 methods have the same performance. We then proceeded to the post hoc pairwise Friedman test, also described in section 2.5. The results are summarized in Figure 4, in the form of critical difference diagrams as proposed in Demšar (2006). There are many significant differences among the methods, but the pattern is not consistent across the three tasks. In the early-versus-late task, the two best methods are adaptive quadratic discriminant and generatively trained restricted Boltzmann machines (RBM), with no significant difference between them. The healthy-versus-impaired task is similar except that support vector machines with an RBF kernel (RBF-SVM) are included in the group of top performers. For the finger-versus-wrist task, the top-performing method is linear SVM.

For all three tasks, linear SVM performed better than adaptive linear discriminant, though the difference was only significant for two of the tasks (early versus late and finger versus wrist). In contrast, adaptive quadratic discriminant was significantly more accurate than support vector machines with a quadratic kernel in the early-versus-late and healthy-versus-impaired tasks, with no significant difference in the finger-versus-wrist task. The results for logistic regression were never significantly different from the results for adaptive linear discriminant.

We also compared the speed of all classification methods. They were timed on a dual-core Xeon 3 GHz processor, with 16 GB RAM, running Ubuntu Linux and Matlab version 7.6. For each method, we recorded the run time (CPU time only) for a single random split of the data, using

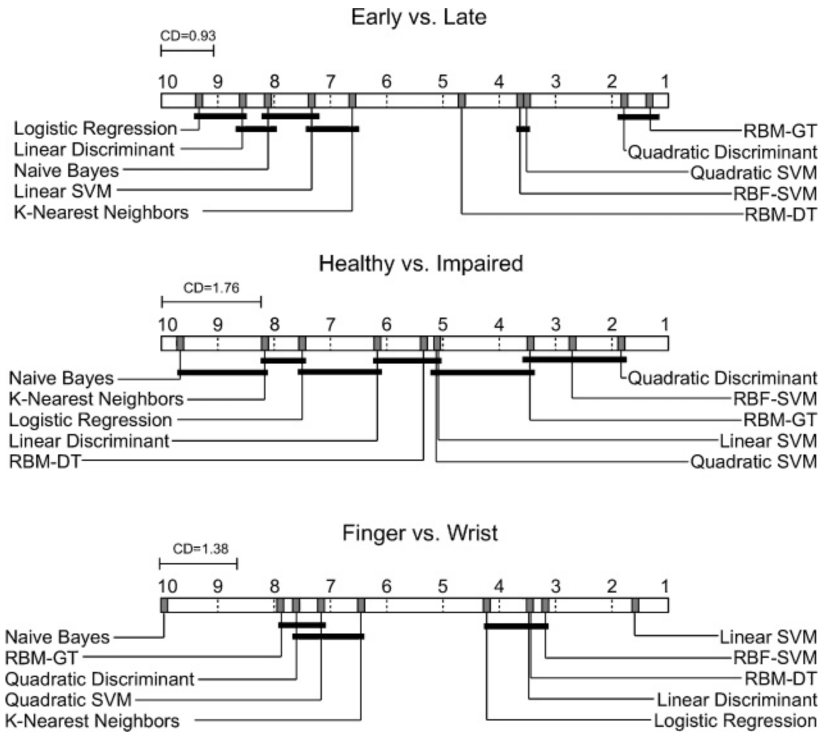

Figure 4: Critical difference diagrams for the post hoc pairwise Friedman tests. For each classification task, the mean rank of each method is shown, with rank 1 indicating the most accurate method. Horizontal line segments group together methods with ranks that are not significantly different. The critical difference (CD) for each task is shown in the upper left of each diagram.

an enlarged training set comprising the training plus validation subsets. The method was applied to only one task (healthy versus impaired) and one subject (S054) for fixed hyperparameter values. The hyperparameter values used were the optimal values found by the procedures described in section 2.3. The pooled-variance version of the naive Bayes method was used, applied directly to voxel space. The resulting run times are shown in Table 3, and also in Figure 9 in the Supplemental Material. Note that the full study used 20 random splits of the data, 9 subjects and 3 tasks, so the total time per method is larger than the times reported here by a factor of at least 540. In those methods that use validation sets to fit a hyperparameter, the actual time per method is longer still. For most methods, run time is expected to be a variable function of subject, task, hyperparameters, and the random splitting of the data. All times recorded are averaged over at least two trials, each trial corresponding to a different random splitting of the data.

The methods based on restricted Boltzmann machines are considerably slower than the other methods, which is to be expected as they have many more parameters. Of the two fastest methods, naive Bayes (applied directly to voxel space) is the least accurate overall; however K-nearest neighbors gave moderately good accuracy on the early-versus-late and finger-versus-wrist tasks. The run times of the remaining methods are similar: all are between 38 seconds and 6 minutes.

In practice, the time required to apply a classification method depends more on the ease of implementation of the method and the amount of parameter tuning done than on the running times reported in Table 3 and in Figure 9 in the Supplemental Material. In this respect, support vector machines with a fixed kernel are fast to use, given the availability of standard packages and the fact that a large fixed value of the penalty factor C works well; parameter tuning, for example, for the RBF kernel, can still take a lot of time. Naive Bayes, linear discriminant, and quadratic discriminant are also relatively fast methods to implement and use, though preceding them with principal component analysis and an optimal choice of PC subspace, as we have done, slows the process by a large factor. K-nearest neighbors is very fast to implement and use. Logistic regression is fairly easy to implement, as long as a standard optimization method is used; there is a choice of regularization methods, and for most standard methods, there is a single regularization parameter to be tuned. Methods based on restricted Boltzmann machines are significantly more difficult and time-consuming to implement and tune.

Though the main focus of this article is classification accuracy, classifiers are also of interest because of information they contain about the spatial distribution of neural activity. We discuss this briefly, though a thorough exploration of this subject is beyond the scope of this article. The most fundamental summary of the information contained in a classifier is an activation map that shows the relative importance of each voxel in the classification process. For linear classifiers, all defined by a weight vector $\mathbf{w}$ and a bias constant b, the standard way to produce an activation map is to set each voxel's intensity equal to the corresponding component of the weight vector. Applying this method to the gaussian naive Bayes classifier—the pooled-variance (linear) version, applied directly to voxels—yields the standard activation map associated with the general linear model (GLM). Figure 5 shows activation maps for the finger-versus-wrist classification task, for one random split of the data for subject S059, corresponding to four different linear methods: gaussian naive Bayes (GNB), adaptive linear discriminant (LD), logistic regression (LogReg) with L1 regularization, and linear support vector machines (L-SVM). All hyperparameter values were the same as those used for the classification accuracy tests. For LD, the weight vector is initially in principal component space but is transformed back into the

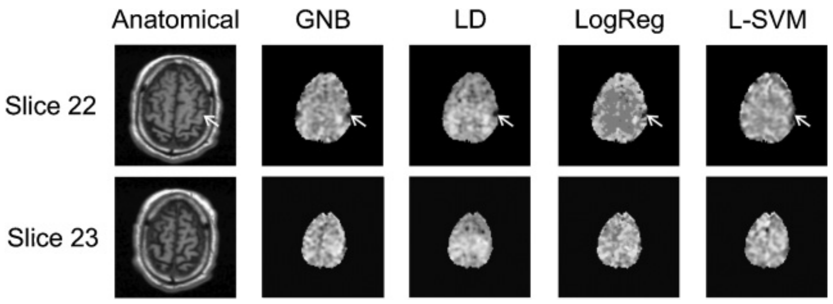

Figure 5: Activation maps for the finger-versus-wrist classification task, corresponding to four different linear methods (from left to right): gaussian naive Bayes, adaptive linear discriminant, logistic regression with L1 regularization, and linear support vector machines. All maps are for the same random split of the data for subject S059. For display purposes, all maps have been histogram-matched to the GNB one. The arrows indicate activations in the right motor cortex, which are discussed in the main text. A color version of this figure appears in Figure 10 in the Supplemental Material.

original voxel space by left-multiplying it by the matrix of eigenimages obtained from the singular value decomposition. (This method is valid for any linear classification method applied to principal components.) For display purposes, all maps have been histogram-matched to the GNB one. Only two axial slices are shown: slices 22 and 23 (with 24 being the top of the head). Color versions of the same maps appear in Figure 10 in the Supplemental Material.

The different methods all produce different activation maps. Those for GNB, LD, and LogReg have some broad similarities, especially in slice 22, while the L-SVM map appears less similar to the other three. The "pixellated" look of the LogReg map is attributable to the histogram-matching process, together with the fact that the logistic regression weight vector has a large number of zero values due to the L1 regularization. Note that all of the activation maps show a strong, negative activation (a dark spot in Figure 5, which is blue in Figure 10) in the area on the right of slice 22 that is marked with an arrow in each of the maps. This is in the motor cortex, contralateral to this subject's lesion. The negative activation means that this area is more active during finger movement than wrist movement, which is consistent with previous data from healthy individuals on finger and wrist movement.

4 Discussion

Our aim has been to determine which classification methods are appropriate for longitudinal studies of fMRI data from the point of view of classification accuracy. We recognize that there may be no general answer to this question, and specifically that the relative performance of various methods may depend on the nature of the data and the classification task, and the degree of generalization required. Further studies, using different data sets and a variety of different tasks, will be needed before confident general conclusions about the relative suitability of different methods can be given.

In this article, we have studied three moderately complex classification tasks, all based on motor tasks in a stroke recovery study. We have constructed the performance tests so as not to require strong generalization across subjects or across recording sessions. In particular, since in every random split of the data we include volumes from each run in each of the training, validation, and test sets, the prediction accuracies we report are likely to be upper bounds on those that would be obtained with training and test sets from separate runs or sessions.

We have chosen binary classification tasks for simplicity. All of the classification methods studied here can be adapted to arbitrary numbers of classes, though there are choices to be made in doing so, and this case deserves further consideration.

One difficulty in applying many classification methods in this domain is the relative paucity of data: the number of volumes available with which to train a model is typically small relative to the data dimensionality (i.e., the number of voxels). This is true especially of the single-subject models that this article considers, and especially if one does not want to restrict the input a priori to subsets of the voxels. The relatively small number of training examples can lead to overfitting of models, resulting in poor generalization. The risk of overfitting is greatest for complex nonlinear models with large numbers of parameters; however, regularization of the parameters can often compensate for this. Thus, it is not clear, before doing numerical experiments, whether nonlinear models will outperform linear models in fMRI applications and other contexts with relatively small numbers of training samples. Indeed, as first explored in the neuroimaging context by Morch et al. (1997), the answer may depend strongly on the number of training samples.

We have considered 10 classification methods. We note that all of these methods could possibly be refined to improve their performance in these tests, so the results reported here are only indicative. For example, any of our methods could be preceded by any kind of feature selection (see the discussion that follows). In particular, in place of the simple feature selection based on principal component analysis that we do prior to applying linear and quadratic discriminant, we could reduce the population variance prediction by applying the method of (Kjems, Hansen, & Strother, 2001), or we could use penalized discriminants with at least an L2 ridge penalty (see Kustra & Strother, 2001, which briefly describes how the choice of a principal component subspace can be viewed as discrete regularization). Wherever we have tuned hyperparameters, this tuning could have been finer. Support vector machines can be combined with any choice of kernel, and there are well-known kernels that we have not tried. Nor have we tuned all available hyperparameters or tried other implementations. The classifiers based on restricted Boltzmann machines (RBM) are very complex, with many hyperparameters and possible variations on the algorithms, which we have explored extensively but not exhaustively. In particular, variations of RBMs for modeling real-valued data are still under active development. Logistic regression is a straightforward method in principle, but there are choices to be made in the optimization method that could affect accuracy: the conjugate gradient implementation we used has several parameters, which we left set at their default values, and also we could have used a different optimization method such as gradient descent.

The methods were compared by median out-of-sample classification accuracy, with post hoc pairwise Friedman tests to determine the significance of differences in per-task median accuracies. No one classification method was consistently more accurate than the others in our experiments. In the early-versus-late task, the two best methods were generatively trained restricted Boltzmann machines (RBM) and adaptive quadratic discriminant, with no significant differences between them. In the healthy-versus-impaired task, the top methods were adaptive quadratic discriminant, support vector machines with an RBF kernel (RBF-SVM), and generatively trained RBMs, with no significant differences between them. For the finger-versus-wrist task, the most accurate method is linear SVM. Which method is best overall in these experiments depends on how this is judged. RBF-SVM had the best overall median accuracy when results from all three tasks were pooled (see Table 3), with adaptive quadratic discriminant and generatively trained RBMs sharing second place; however, when considering per-task median accuracies, adaptive quadratic discriminant outperformed RBF-SVM on two of three of the tasks, putting it in first place, with generatively trained RBMs a close second.

The least accurate method in our study was gaussian naive Bayes applied directly to voxels (see the previous section for results from the adaptive version of the method). It was significantly worse than all other methods for the finger-versus-wrist task, not significantly better than any other method for the healthy-versus-impaired task, and only significantly better than one other method (logistic regression) for the early-versus-late task. Though the distinct-variance version of naive Bayes performed better than logistic regression on the early-versus-late task, this is not true of the pooled-variance version of naive Bayes; thus, we do not reinforce the findings of Ng and Jordan (2002), who showed that pooled-variance naive Bayes can outperform logistic regression on a variety of data not including fMRI. The general observation of Ng and Jordan (2002) that generative training can improve discriminative performance is supported by our observation that generatively trained restricted Boltzmann machines outperformed the corresponding discriminatively trained models in two of our classification tasks. This issue was further explored in Schmah et al. (2009), which considered blends of generative and discriminative training of pairs of RBMs, using the same data set as this article, but did not find that a blend improved performance in this case. Our observation that K-nearest neighbors outperforms naive Bayes (significantly for two of the tasks, with no significant difference for the third) disagrees with the results of Mitchell et al. (2004). Naive Bayes was significantly worse than adaptive linear discriminant for two out of three of the tasks and not significantly different for the third, a finding in agreement with Zhang et al. (2008). Taken together, these observations illustrate the difficulty of trying to generalize prediction result and the need to apply multiple classifiers to a given domain and problem.

Most of the classification methods considered here are fairly easy to implement and use, including adaptive linear and quadratic discriminant, gaussian naive Bayes, and K-nearest neighbors. Implementing support vector machines can be complex; however, standard packages are available. Logistic regression can be moderately complex to implement, depending on the optimization method used. The classifiers based on RBMs are complex and computationally intensive, and no standard package implements the methods. Indeed, the methods used in this study are novel in their use of a pair of RBMs (see also Schmah et al., 2009). Methods based on RBMs were developed in contexts where large numbers of data points are typically available. We believe that this study, together with Schmah et al. (2009), represents the first application of these methods to biomedical data, in which the numbers of data points are often small relative to the dimensionality of the data.

Taking into account the accuracy, speed, and ease of implementation of the methods tested, the most promising methods for fMRI studies in the near future appear to be adaptive quadratic discriminant and support vector machines with various kernels. Though the methods based on restricted Boltzmann machines are significantly slower and not yet implemented in standard packages, their competitive performance in this study, and success in other contexts, suggest that further development of their applications to fMRI is warranted.

Many other classification methods merit further study in this context. Of those that have already been applied successfully to fMRI, we mention relevance vector machines (RVMs) (see Lukic et al., 2007), feedforward neural networks (see Hanson et al., 2004), and soft-penalty linear discriminants with elastic-net-like regularizers (see Carroll, Cecchi, Rish, Garg, & Rao, 2009). In addition, there are many promising methods from the field of machine learning that have not yet been applied to fMRI, including, for example, deep belief nets (Hinton, Osindero, & Teh, 2006).

Many authors have found that an initial feature selection step can improve classification accuracy (Hanson & Halchenko, 2008; De Martino et al., 2008; Yamashita, Sato, Yoshioka, Tong, & Kamitani, 2008; Mitchell et al., 2004). These feature selection methods are typically voxel selection methods and are often (but not always) based on either prior knowledge or a univariate analysis. We have not used an initial voxel selection step in our comparison. Our main reason for this is that there are several candidate methods, and exploring them thoroughly, in combination with the several classification methods we have studied, would lead to an unwieldy number of combinations. We note, however, that for three of our methods—linear and quadratic discriminant and naive Bayes (the adaptive version)—we used an initial principal component analysis step and then applied the classification method to an optimal-dimension PC subspace. The selection of a PC subspace is a kind of feature selection. Note that some kind of feature selection is required before computing linear and quadratic discriminants, since we have fewer training examples than voxels. For completeness, it would be interesting to try applying some or all of the classifiers evaluated here to an optimized PC subspace; however, the dimension optimization would need to be done separately for every classification method, which would be prohibitively time-consuming for the slower methods.

Although the focus of this article is classification accuracy, there remains the important related goal of extracting and interpreting information about neural representations contained in the trained classifiers. As illustrated in Figure 5, activation maps produced from different linear classifiers may look quite different from one another. For nonlinear classifiers, there is the added challenge of no single standard way to produce activation maps, though the sensitivity maps of Kjems et al. (2002) provide a useful general procedure. More generally, there may be useful information that can be extracted from the classifiers that cannot necessarily be presented in a single activation map. The use of classifiers to provide information about neural representations will be the subject of a future study.

Acknowledgments

We thank Natasa Kovacevic for coregistering and motion-correcting the fMRI data used in this study. This work was supported by the Brain Network Recovery Group through a grant from the James S. McDonnell Foundation (No. 22002082) and a grant from the National Institutes of Health (No. R01 DC7488). S.C.S. is partially supported by the Centre for Stroke Recovery of the Heart and Stroke Foundation of Ontario.

References

Bengio, Y., Lamblin, P., Popovici, D., & Larochelle, H. (2007). Greedy layer-wise training of deep networks. In B. Schölkopf, J. Platt, & T. Hoffman (Eds.), *Advances in neural information processing systems, 19* (pp. 153–160). Cambridge, MA: MIT Press.

Carroll, M. K., Cecchi, G. A., Rish, I., Garg, R., & Rao, A. R. (2009). Prediction and interpretation of distributed neural activity with sparse models. *NeuroImage*, *44*, 112–122.

Conover, W. J. (1998). *Practical nonparametric statistics*. New York: Wiley.

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

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *JMLR*, *7*, 1–30.

Friston, K., Ashburner, J., Kiebel, S., Nichols, T., & Penny, W. (Eds.). (2007). *Statistical parametric mapping: The analysis of functional brain images*. Orlando, FL: Academic Press.

Hansen, L. K. (2007). Multivariate strategies in functional magnetic resonance imaging. *Brain and Language*, *102*, 186–191.

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

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

Hinton, G. E. (2002). Training products of experts by minimizing contrastive divergence. *Neural Computation, 14*(8), 1711–1800.

Hinton, G. E. (2007a). Boltzmann machine. *Scholarpedia, 2*, 1668.

Hinton, G. E. (2007b). To recognize shapes, first learn to generate images. In P. Cisek, T. Drew, & J. Kalaska (Eds.), *Computational neuroscience: Theoretical insights into brain function*. New York: Elsevier.

Hinton, G. E., Osindero, S., & Teh, Y. W. (2006). Fast learning algorithm for deep belief networks. *Neural Computation, 18*, 1527–1554.

Hinton, G. E., & Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *Science, 313*(5786), 504–507.

Joachims, T. (1999). Making large-scale SVM learning practical. In B. Schölkopf, C. Burges, & A. Smola (Eds.), *Advances in kernel methods: Support vector learning*. Cambridge, MA: MIT Press.

Kjems, U., Hansen, L. K., Anderson, J., Frutiger, S., Muley, S., Sidtis, J., et al. (2002). The quantitative evaluation of functional neuroimaging experiments: Mutual information learning curves. *NeuroImage, 15*, 772–786.

Kjems, U., Hansen, L. K., & Strother, S. C. (2001). Generalizable singular value decomposition for ill-posed datasets. In T. K. Leen, T. G. Dietterich, & V. Tresp (Eds.), *Advances in neural information processing systems, 13* (pp. 549–555). Cambridge, MA: MIT Press.

Kustra, R., & Strother, S. C. (2001). Penalized discriminant analysis of [^{15}O]-water PET brain images with prediction error selection of smoothness and regularization hyperparameters. *IEEE Trans. Med. Imaging, 20*, 376–387.

LaConte, S., Strother, S., Cherkassky, V., Anderson, J., & Hu, X. (2005). Support vector machines for temporal classification of block design fMRI data. *NeuroImage*, *26*(2), 317–329.

Larochelle, H., & Bengio, Y. (2008). Classification using discriminative restricted Boltzmann machines. In *ICML '08: Proceedings of the 25th International Conference on Machine Learning*. New York: ACM.

Lukic, A. S., Wernick, M. N., Tzikas, D. G., Chen, X., Likas, A., Galatsanos, N. P., et al. (2007). Bayesian kernel methods for analysis of functional neuroimages. *IEEE Trans Med Imaging, 26*, 1613–1624.

Mardia, K., Kent, J., & Bibby, J. (1979). *Multivariate analysis*. Orlando, FL: Academic Press.

McIntosh, A. R., & Lobaugh, N. J. (2004). Partial least squares analysis of neuroimaging data: Applications and advances. *NeuroImage*, *23*(S1), S250–S263.

Mitchell, T. M., 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.

Morch, N., Hansen, L. K., Strother, S. C., Svarer, C., Rottenberg, D. A., Lautrup, B., et al. (1997). Nonlinear versus linear models in functional neuroimaging: Learning curves and generalization crossover. In J. Duncan, & G. Gindi (Eds.), *Information processing in medical imaging* (pp. 259–270). New York: Springer-Verlag.

Ng, A., & Jordan, M. (2002). On discriminative vs. generative classifiers: A comparison of logistic regression and naive Bayes. In T. G. Dietterich, S. Becker, & Z. Ghahramani (Eds.), *Advances in neural information processing systems, 14* (pp. 841–848). Cambridge, MA: MIT Press.

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.

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

Pereira, F., Mitchell, T., & Botvinick, M. (2009). Machine learning classifiers and fMRI: A tutorial overview. *NeuroImage, 45*, S199–S209.

Pessoa, L., Japee, S., Sturman, D., & Ungerleider, L. G. (2006). Target visibility and visual awareness modulate amygdala responses to fearful faces. *Cerebral Cortex, 16*, 366–375.

Press, W., Teukolsky, S., Vetterling, W., & Flannery, B. (1992). *Numerical recipes in C* (2nd ed.). Cambridge: Cambridge University Press.

Schmah, T., Hinton, G. E., Zemel, R. S., Small, S. L., & Strother, S. (2009). Generative versus discriminative training of RBMs for classification of fMRI images. In D. Koller, D. Schuurmans, Y. Bengio, & L. Bottou (Eds.), *Advances in neural information processing systems, 21* (pp. 1409–1416). Cambridge, MA: MIT Press.

Small, S. L., Hlustik, P., Noll, D. C., Genovese, C., & Solodkin, A. (2002). Cerebellar hemispheric activation ipsilateral to the paretic hand correlates with functional recovery after stroke. *Brain, 125*(7), 1544–1557.

Strother, S. C. (2006). Evaluating fMRI preprocessing pipelines. *IEEE Eng. Med. Biol. Mag., 25*, 27–41.

Strother, S. C., LaConte, S., Hansen, L. K., Anderson, J., Zhang, J., Pulapura, S., et al. (2004). Optimizing the fMRI data-processing pipeline using prediction and reproducibility performance metrics: I. A preliminary group analysis. *NeuroImage*, *23*(S1), S196–S207.

Welling, M., Rosen-Zvi, M., & Hinton, G. E. (2005). Exponential family harmoniums with an application to information retrieval. In L. K. Saul, Y. Weiss, & L. Bottou (Eds.) *Advances in neural information processing systems, 17* (pp. 1481–1488). Cambridge, MA: MIT Press.

Woods, R. P., Grafton, S. T., Holmes, C. J., Cherry, S. R., & Mazziotta, J. C. (1998). Automated image registration: I. General methods and intrasubject, intramodality validation. *Journal of Computer Assisted Tomography, 22*, 139–152.

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

Zhang, J., Liang, L., Anderson, J. R., Gatewood, L., Rottenberg, D. A., & Strother, S. C. (2008). A Java-based fMRI processing pipeline evaluation system for assessment of univariate general linear model and multivariate canonical variate analysis-based pipelines. *Neuroinformatics, 6*, 123–134.

Received June 7, 2009; accepted April 17, 2010.