Journal of Neuroscience Methods 239 (2015) 238–245

Contents lists available at ScienceDirect

Journal of Neuroscience Methods

journal homepage: www.elsevier.com/locate/jneumeth

Basic Neuroscience

An empirical solution for over-pruning with a novel ensemble-learning method for fMRI decoding

Satoshi Hirose[a,*], Isao Nambu[b], Eiichi Naito[a,c]

[a] CiNet, National Institute of Information and Communications Technology, CiNet Bldg., 1-4 Yamadaoka, Suita, Osaka 565-0871, Japan
[b] Department of Electrical Engineering, Nagaoka University of Technology, 1603-1, Kamitomioka, Nagaoka, Niigata 940-2188, Japan
[c] Graduate School of Medicine, Osaka University, 1-4 Yamadaoka, Suita, Osaka 565-0871, Japan

HIGHLIGHTS

- We propose a solution for over-pruning called Iterative Recycling (iRec).
- iRec is a novel ensemble learning method for sparse algorithms.
- In iRec, classifiers are trained iteratively by recycling over-pruned voxels.
- In our experiments, iRec rectified over-pruning in sparse logistic regression.
- iRec is applicable to any kind of sparse method.

ARTICLE INFO

Article history:
Received 22 August 2014
Received in revised form 17 October 2014
Accepted 24 October 2014
Available online 4 November 2014

Keywords:
Multi-voxel pattern classification
Ensemble-learning
Over-pruning
Sparse logistic regression
Iterative Recycling (iRec)
iSLR

ABSTRACT

Background: Recent functional magnetic resonance imaging (fMRI) decoding techniques allow us to predict the contents of sensory and motor events or participants' mental states from multi-voxel patterns of fMRI signals. Sparse logistic regression (SLR) is a useful pattern classification algorithm that has the advantage of being able to automatically select voxels to avoid over-fitting. However, SLR suffers from over-pruning, in which many voxels that are potentially useful for prediction are discarded.
New method: We propose an ensemble solution for over-pruning, called "Iterative Recycling" (iRec), in which sparse classifiers are trained iteratively by recycling over-pruned voxels.
Results: Our simulation demonstrates that iRec can effectively rectify over-pruning in SLR and improve its classification accuracy. We also conduct an fMRI experiment in which eight healthy volunteers perform a finger-tapping task with their index or middle fingers. The results indicate that SLR with iRec (iSLR) can predict the finger used more accurately than SLR. Further, iSLR is able to identify a voxel cluster representing the finger movements in the biologically plausible contralateral primary sensory-motor cortices in each participant. We also successfully dissociated the regularly arranged representation for each finger in the cluster.
Conclusion and comparison with other methods: To the best of our knowledge, ours is the first study to propose a solution for over-pruning with ensemble-learning that is applicable to any sparse algorithm. In addition, from the viewpoint of machine learning, we provide the novel idea of using the sparse classification algorithm to generate accurate divergent base classifiers.

1. Introduction

In recent years, many functional Magnetic Resonance Imaging (fMRI) researchers have become interested in extracting information from patterns of brain activity using machine learning techniques, instead of voxel-by-voxel functional mapping using general linear models (GLM; Friston et al., 1995; Worsley et al., 2002). This approach (fMRI decoding) allows us to predict (or "decode") the contents of sensory and motor events or participants' mental states (= labels) from multi-voxel patterns of fMRI signals (Haynes and Rees, 2006; Norman et al., 2006; Formisano et al., 2008; Pereira et al., 2009).

However, in fMRI decoding, the number of available voxels ("features," in machine learning terminology) tends to be greater than 30,000; whereas, for practical reasons, usually only a few

* Corresponding author. Tel.: +81 80 9045 9529; fax: +81 06 7174 8612.
E-mail addresses: satoshi.hirose@nict.go.jp (S. Hirose), inambu@vos.nagaokaut.ac.jp (I. Nambu), eiichi.naito@nict.go.jp (E. Naito).

http://dx.doi.org/10.1016/j.jneumeth.2014.10.023

hundred trials ("samples") are available. The greater number of features relative to the small number of samples frequently causes the classifier to identify spurious relationships between patterns of feature values and labels in the training dataset. This problem is known as over-fitting.

To prevent over-fitting, researchers have recently started applying sparse classification algorithms (SCA) to whole-brain fMRI data (e.g., De Martino et al., 2008; Yamashita et al., 2008; Ryali et al., 2010; Ng et al., 2010; Ng and Abugharbieh, 2011). The major advantage of SCA is its automatic selection of a small subset of given features during classifier training. This feature-selection ability helps to alleviate over-fitting, because SCA is able to ignore features that are actually irrelevant to labels but incidentally have spurious relations with the labels in the finite samples of the training dataset. On the other hand, because of its strong tendency to select fewer features, some SCAs have a high likelihood of removing relevant features even when they are potentially useful for improving accuracy in label prediction (over-pruning; Yamashita et al., 2008; De Martino et al., 2008). Over-pruning not only deteriorates prediction accuracy but also makes it difficult to determine whether selected voxels are biologically plausible or not. For example, when over-pruning occurs, SCA often selects only scattered voxels in a certain brain region, even when all voxels in that region actually contain useful label information. This makes it difficult to conclude whether that brain region is indeed of physiological significance or the scattered voxels are merely selected by chance.

For example, sparse logistic regression (Tibshirani, 1996; Yamashita et al., 2008) is a group of typical SCA algorithms that has the serious drawback of over-pruning (Yamashita et al., 2008; De Martino et al., 2008). This is because the priors used in sparse logistic regression (e.g., Laplacian prior, Tibshirani, 1996; ARD priors, MacKay, 1992) force the selection of the smallest number of voxels possible.

Solutions for over-pruning in the sparse logistic regression with Laplacian prior have previously been proposed. For example, over-pruning in the algorithm was rectified by incorporating priors that encourage the selection of a group of features among which the pairwise correlations are very high (Elastic Net; Zou and Hastie, 2005; De Martino et al., 2008) or a group of voxels that are spatially close together (Graph Net; Grosenick et al., 2013, Generalized Sparse Classifiers; Ng et al., 2010; Ng and Abugharbieh, 2011, Total Variation regularized Logistic Regression; Michel et al., 2011). When these solutions were applied to fMRI datasets, they successfully solved the over-pruning problem and facilitated improved prediction accuracy and/or prominently identified large cluster structures in task-relevant brain regions.

These extensions are not theoretically guaranteed to be implementable in other sparse algorithms. Specifically, to date, no theoretical or empirical evidence that these methods can be applicable to sparse logistic regression with ARD priors has been provided. In this paper, we propose another type of empirical solution, *I*terative *Rec*ycling (iRec), which can be applied to any kind of SCA to rectify the over-pruning problem. Our method takes the form of ensemble-learning (Kuncheva, 2004), in which multiple SCA classifiers (base classifiers) are iteratively generated with a training dataset that comprises only voxels that were not used by the previously generated classifiers. After training, these trained classifiers are united and one ensemble classifier is generated. The secondary base classifiers are able to utilize any features that were over-pruned by SCA in the generation of the first base classifier. Thus, by combining the secondary classifiers with the first one, we can rectify the over-pruning.

Although our solution is applicable to any kind of SCA, in the present study, we focus on improving the sparse logistic regression with ARD priors (SLR; Yamashita et al., 2008), because this algorithm can predict labels accurately without adjusting any hyper-parameters. Thus, this approach is able to reduce computational cost and eliminate the risk of selecting inappropriate parameters. In this paper, when we refer to SLR, we will be assuming this version with ARD priors.

The remainder of this paper is organized as follows. First, we outline the general procedure used by our ensemble-learning method (iRec). Then, we explain how SLR is extended with iRec. Next, we present the results of a simulation experiment conducted that show that iRec can improve label prediction in SLR by rectifying over-pruning. Finally, we demonstrate the utility of iRec on a real dataset obtained from an fMRI experiment in which normal volunteers performed a simple finger-tapping task with either their right index or middle finger.

2. Materials and methods

2.1. Definition of the problem

In our present study, we focus on binary classification; i.e., classification problems with labels of binary values (−1 and 1). The training dataset given for classifier training includes feature vectors $\boldsymbol{X}_{train} = \{\boldsymbol{x}^1_{train}, \ldots, \boldsymbol{x}^S_{train}\}$ and the corresponding labels $Y_{train} = \{y^1_{train}, \ldots, y^S_{train}\}$. S is the number of samples included in the training dataset and $\boldsymbol{x}^s_{train}$ is a D-dimensional vector containing the feature values in the sth sample; i.e., $\boldsymbol{x}^s_{train} = \left[x^{s,1}_{train}, \ldots, x^{s,D}_{train}\right]^T$, where $x^{s,d}_{train}$ denotes the value of the dth feature in the sth sample and $\boldsymbol{v}^T$ represents the transpose of vector $\boldsymbol{v}$. The goal of the classification analysis is to train a classifier that can accurately predict the label (y_{test}) of a novel feature vector ($\boldsymbol{x}_{test}$). The prediction accuracy is evaluated with a test dataset disjoint from the training dataset, which includes S_{test} feature vectors ($\boldsymbol{X}_{test} = \{\boldsymbol{x}^1_{test}, \ldots, \boldsymbol{x}^{S_{test}}_{test}\}$) and the corresponding labels ($Y_{test} = \{y^1_{test}, \ldots, y^{S_{test}}_{test}\}$).

2.2. iRec

In this section, we describe the general procedure used to apply iRec to SCA, which selects a subset of features during training and uses only the selected subset for label prediction. Fig. 1A illustrates the training algorithm for iRec. First (Panel 1 in Fig. 1A), the original feature vectors in the training dataset ($\boldsymbol{X}_{train}$) are defined as the feature vectors for the first iteration ($\boldsymbol{X}_{train(1)}$). Iterative training then commences. Each iteration (Panels 2–5 in Fig. 1A) comprises two actions: the actual training (Panels 2–3) and updating of the training dataset (Panels 4–5). In the nth iteration, first, SCA training is performed with the training dataset $\{\boldsymbol{X}_{train(n)}, Y_{train}\}$, generating a base classifier (yellow square), which uses a selected subset of features ("Selected features" in Panel 3). In the next step (Panels 4–5), features selected by the SCA (the black squares in "Selected Features") are excluded from the training dataset and the updated feature vectors are defined as the feature vectors for the next iteration, $\boldsymbol{X}_{train(n+1)}$ (Left side of Panel 5). For example, if the third, fifth, and seventh components are selected by SCA, each of $\boldsymbol{x}^s_{train(n+1)} \in \boldsymbol{X}_{train(n+1)}$ is the vector with all the components of $\boldsymbol{x}^s_{train(n)}$, except for the third, fifth, and seventh components. Note that the labels Y_{train} do not change. These two steps (SCA training and updating of the training dataset) are repeated N times, so that N classifiers are acquired.

Then, in the prediction (Fig. 1B), each SCA base classifier respectively predicts a label and the final outcome is achieved by voting among them.

We emphasize that iRec is applicable to any kind of SCA classifier for the following reasons. First, iRec does not require any other particular characteristics of the base algorithm except for sparseness. Second, iRec can be easily implemented without explosion of

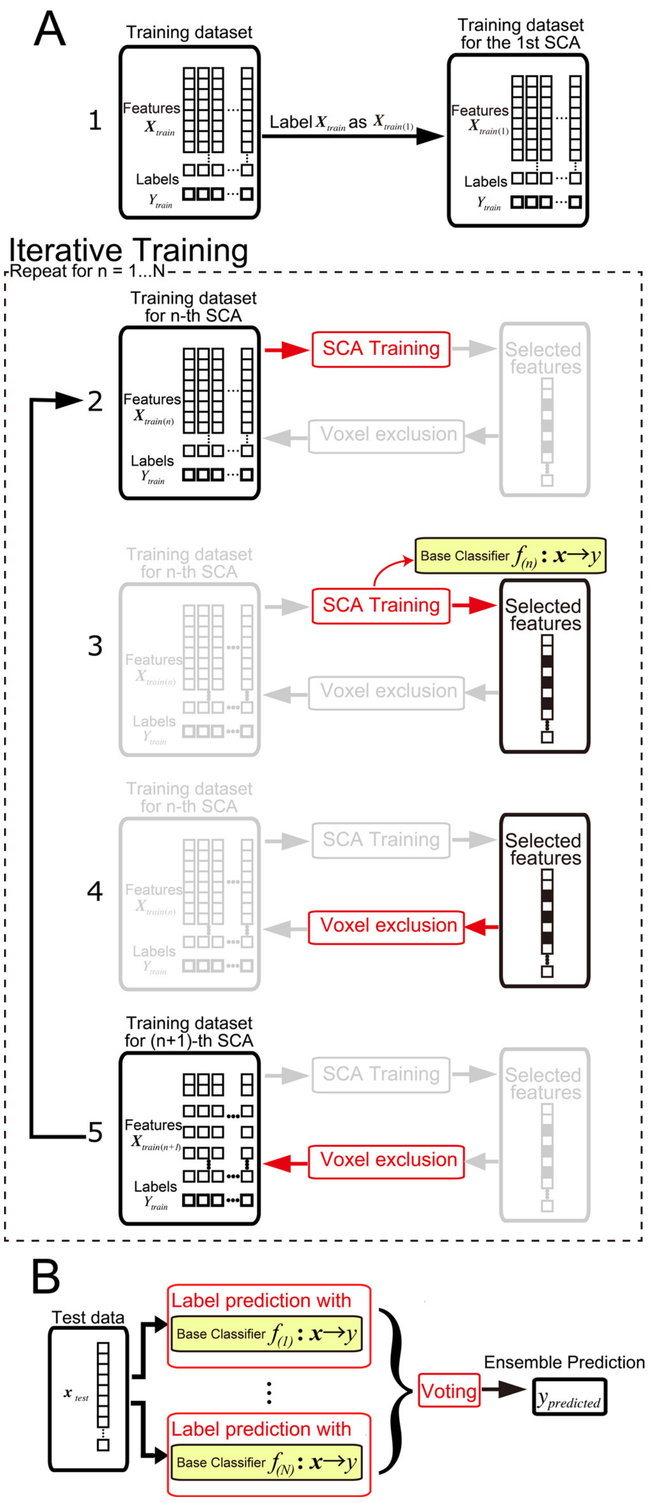

Fig. 1. iRec algorithm for training (A) and label prediction (B). (A) 1: First, feature vectors in the initial training dataset are labeled as $\boldsymbol{X}_{train(1)}$. Iterative training then commences (2–5). 2: In each iteration, training for a base classifier is first performed with sparse classification algorithm (SCA) and the training dataset $\boldsymbol{X}_{train(n)}$, Y_{train}. 3: After the training, we achieve a base classifier (yellow) and indices of features selected by SCA (black pixels in "Selected features"). 4–5: Next, the training dataset is updated by excluding the features that are selected by SCA. For example, if only the third, fifth, and seventh are selected by SCA, the corresponding features are excluded from the training dataset for the next iteration (training dataset for $(n+1)$th SCA). After N iterations, we obtain N SCA base classifiers. (B) The label is first predicted by each of the SCA base classifiers. Then, these predictions are combined to achieve ensemble prediction (Voting).

computational cost, because training of iRec consists of repetition of base SCA training and simple feature selection.

2.3. iSLR

Although iRec is applicable to any kind of sparse classifier, in this paper, we focus on iRec as a solution to over-pruning for SLR with ARD priors (Yamashita et al., 2008). Here, we briefly explain the SLR algorithm (please see the original paper by Yamashita et al., 2008 for more details).

2.3.1. SLR

In SLR, the label probability is modeled by the weighted sum of feature values through a logistic function:

$$P(y|\boldsymbol{x}, \boldsymbol{w}) = \frac{1}{1 + e^{\boldsymbol{w}^T \boldsymbol{x} \times y}} \tag{1}$$

where $\boldsymbol{x}$ and y denote the feature vector and its corresponding label, respectively. $\boldsymbol{w} = [w^1, \ldots, w^D]^T$ is a vector called the "weight vector," which includes the feature weights. The SLR classifier is trained by calculating the posterior mean of the weight given the training dataset, under the assumption that each component of $\boldsymbol{w}$ follows the ARD prior distribution. After the training, the SLR classifier can predict the label by inserting the corresponding feature vector and the trained weight vector into Eq. (1).

Here, we note two SLR characteristics presented in the original paper (Yamashita et al., 2008) that are essential for construction of iSLR. First, many components in the trained weight vector become zero and the features corresponding to the zero weights do not affect the prediction. Second, SLR outputs the probabilistic prediction of the label rather than the binary prediction.

2.3.2. iSLR

We refer to the overall algorithm comprising iRec incorporated with SLR as *iSLR*. The toolbox including iSLR source code can be downloaded from http://www2.nict.go.jp/cinet/bnc/hirose/mvpc/index.html. Pseudo-codes used by iSLR for training and label prediction are outlined in Fig. 2A and B, respectively. Because SLR provides weights with a value of zero for the unselected features, updating of the training dataset is carried out by choosing the features corresponding to weights with a value of zero (Line 4 in Fig. 2A). The weight vectors generated after the first iteration lack components corresponding to the features excluded in earlier iterations. On completion of the iterative training, the lacking components are replaced with zeroes to keep the dimension of the weight vectors equal to D.

In iSLR, we use the product rule (Kittele et al., 1998; Hinton, 2002), to combine the predictions of the base classifiers. Thus, the label prediction procedure is as follows. First, the label probabilities are estimated by each of the N base classifiers using the SLR prediction procedure described in Section 2.3.1. Specifically, the feature vector of the test data and each of the N weight vectors resulting from the training are inserted into Eq. (1) such that N probabilistic label predictions are acquired (Line 1 in Fig. 2B). The label that maximizes the product of these probabilities is then defined as the predicted label of iSLR (Line 2):

$$y_{predicted} = \arg\max_{y} \prod_{n=1}^{N} P(y|\boldsymbol{x}_{test}, \boldsymbol{w}_{(n)}) \tag{2}$$

By inserting Eq. (1) into Eq. (2) and transforming the equation, we find that iSLR is equivalent to a linear classification algorithm whose weight vector is the sum of the weight vectors of the base

```
A: Training

Inputs:  $N$ (number of base classifiers)
         $\{X_{train}, Y_{train}\}$ (training dataset)
Outputs: $w_{(n)}$; $n = 1, ..., N$ (weight of the SLR base classifiers)

1: Define $X_{train}$ as $X_{train(1)}$
2: Repeat with $n = 1, ..., N$
    3: Build $w_{(n)}$ using SLR training procedure with the dataset $\{X_{train(n)}, Y_{train}\}$
    4: Define all components of $X_{train(n)}$ assigned with 0 in $w_{(n)}$ as $X_{train(n+1)}$
5: End repeat
6: Replace the lacking components in $w_{(n)}$; $n = 2, ..., N$ with 0.

B: Label Prediction

Inputs:  $w_{(n)}$; $n = 1, ..., N$ (weight of the SLR base classifiers)
         $x_{test}$ (feature values)
Outputs: $y_{predicted}$ (predicted label)

1: Predict label with $w_{(n)}$; $n = 1,..., N$ and $x_{test}$, to obtain
    probabilistic predictions $P(y=1 | x_{test}, w_{(n)})$ and $P(y=-1 | x_{test}, w_{(n)})$
2: Compute the product of the predictions,
    i.e. $\prod_{n=1}^{N} P(y=1 | x_{test}, w_{(n)})$ and $\prod_{n=1}^{N} P(y=-1 | x_{test}, w_{(n)})$
3: Predict the label as the one that provides the larger product.
```

Fig. 2. iSLR pseudo-codes for training (A) and label prediction (B).

classifiers (see Appendix A). In particular, the ensemble classifier can be redefined as follows:

$$y_{predicted} = \begin{cases} 1 & \text{if} \left(\sum_{n=1}^{N} \boldsymbol{w}_{(n)} \right)^T x_{test} > 0 \\ -1 & \text{otherwise} \end{cases} \tag{3}$$

In this paper, the weight of iSLR indicates $\sum_{n=1}^{N} w_{(n)}$ in accordance with this definition.

2.3.3. Determining the number of base classifiers

Before training, the number of iterations (the number of base classifiers that will be used: N) must be determined. This is accomplished by cross validation within the training dataset (i.e., "nested cross validation"). Specifically, N is determined as the number of SLR base classifiers that maximizes the prediction accuracy of the ensemble classifier evaluated by cross validation within the training dataset.

In the simulation and real fMRI analysis conducted in the present study, we performed 10-fold cross validation and leave-one-session-out (9-fold) cross validation, respectively, within the training dataset. When two or more numbers provide the best accuracy, we use the smallest number among them as N. After determining the number of iterations, the training procedure outlined above is performed with the determined N, except in cases where all features have been excluded in the N'th iteration, before reaching the Nth iteration ($N' < N$). In this exceptional case, N is redefined as N' to avoid training without any features.

Other candidates for the determination, e.g., determination based on the base classifier's performance or on characteristics of features selected in the iteration, will be evaluated in future studies.

2.4. Simulation

We conducted a simulation to demonstrate that iSLR can rectify over-pruning in SLR. In the simulation, we prepared two types of features, relevant and irrelevant features. The values of both the relevant and irrelevant features followed the normal distribution with standard deviations of one. The mean varied depending on the label (-0.25 or 0.25) in relevant features, whereas it was always zero on irrelevant features. Half of the relevant features had a positive mean (0.25) when the label was one and a negative mean (-0.25) when the label was -1, with the roles reversed for the other half:

$$x^{s,d} \sim \begin{cases} N(-0.25 \times y^s, 1) & \text{for } d \le \dfrac{D_{relev}}{2} \\ N(0.25 \times y^s, 1) & \text{for } \dfrac{D_{relev}}{2} < d \le D_{relev} \\ N(0, 1) & \text{for } D_{relev} < d \le D \end{cases} \tag{4}$$

where d and s are the indices of a feature and a sample, respectively, $N(\mu, \sigma^2)$ is a Gaussian distribution with mean μ and variance σ^2, and D_{relev} is the number of relevant features. The number of features (D) was fixed at 30,000 and the number of relevant features at 200. Both the training and test datasets contained 200 samples.

We repeated the simulation 100 times and evaluated the mean prediction accuracies of iSLR and SLR.

2.5. fMRI experiment

Next, we conducted an fMRI experiment to examine the utility of iSLR when it is applied to real data.

2.5.1. Participants

Eight right-handed (L.Q. > 80) (Oldfield, 1971) healthy participants participated in the experiment. The Ethical Committee of the National Institute of Information and Communications Technology (NICT) approved the study, and all the participants provided their written informed consent prior to the experiment. The experiment was carried out according to the principles and guidelines of the Declaration of Helsinki (1975).

2.5.2. Task procedure

The participants rested comfortably in the supine position in the fMRI scanner. Their right arms were orientated parallel to their torsi, and their forearms pronated and supported by a cushion, allowing them to relax their arms. During the scan, participants were only allowed to move their right fingers without moving their wrists.

Visual stimuli were projected onto a screen in the scanner and the participants could see the stimuli via a mirror in front of their eyes. Throughout the experiment, a fixation cross was presented in the center of the screen, and the participants were instructed to maintain their gaze on this point to avoid unnecessary eye movements during the experiment. At the beginning of each trial, we presented either the digit "1" or "2" for 0.5 s just above the fixation cross. When the digit 1 was presented, the participants prepared to perform repetitive extension-flexion movements with their right index finger. When the digit 2 was presented, they performed the same movements with their right middle finger. These movements were initiated once the digit disappeared and the fixation cross turned red. The movements were paced by 4-Hz auditory cues (duration = 50 ms, pitch = 1000 Hz) and lasted for 3.5 s (= one trial). To alert the participants at the beginning of a new trial, two 4-Hz auditory cues with a different pitch (500 Hz) were provided during the visual presentation of the digit.

Each session consisted of 20 trials comprising 10 index finger trials and 10 middle finger trials in random order, with inter-trial intervals of 8 s. Further, each session included an additional 18-s resting period before the first trial (pre-resting period) and another 10-s resting period after the last trial (post-resting period). In total, we collected 130 functional images in each session, including pre- and post-resting periods (TR = 2000 ms). Each participant completed 10 sessions; i.e., a total of 200 trials were completed.

2.5.3. *fMRI measurement and preprocessing*

We used a 3.0-T SIEMENS scanner (Trio Tim) with a head-coil to obtain T1-weighted anatomical images and functional T2*-weighted echo-planar images (EPI: 64×64 matrix; pixel size: $3.0\,\text{mm} \times 3.0\,\text{mm}$; TE: 30 ms). A functional image volume comprised 30 slices of thickness 4 mm with a 1-mm gap, which ensured that the entire brain was within the field of view (FOV) with dimensions $192\,\text{mm} \times 192\,\text{mm} \times 150\,\text{mm}$.

We excluded the first five functional images in each session from the analysis to allow for magnetization equilibrium, realigned the remaining images to correct for head movements, and co-registered them to each participant's anatomical image. We then calculated the percentage increase in the fMRI signal from the mean for each voxel in each session. This calculation was performed to minimize differences in the magnitude of the fMRI voxel values across sessions. Finally, a temporal high-pass filter (Butterworth filter) with a cutoff frequency of 1/128 Hz was applied to the data obtained in each session to remove low frequency drift. We then averaged the two images obtained in the time window of 2–6 sec after the end of each trial. These images were deemed most likely to include the signals corresponding to the finger movement events because the Blood Oxygen Level-Dependent (BOLD) signal normally has a delay and peaks at approximately 6 s after the occurrence of neuronal activity (Aguirre et al., 1998). Finally, we extracted the values of each voxel belonging to the brain (segmentation). The number of voxels used in this analysis was $33{,}270 \pm 1{,}570$ (mean ± S.D. across participants). The trials in which participants used their index finger were associated with the label -1, while those in which they used their middle finger were associated with the label 1.

We performed realignment, coregistration, and segmentation with a statistical parametric mapping software application (SPM5; http://www.fil.ion.ucl.ac.uk/spm; Wellcome Department of Cognitive Neurology, UCL, London). The other preprocessing steps were performed with in-house scripts in the MATLAB programming environment.

2.5.4. *Evaluation of prediction accuracy*

To evaluate the prediction accuracy of each algorithm for each participant, we performed a leave-one-session-out (10-fold) cross validation. More specifically, we trained the classifiers using the 180 trials from nine sessions (training dataset), and used the 20 trials of the remaining session (test dataset) to evaluate prediction accuracy. This validation was performed for all 10 possible session combinations (= 10 validation folds). Before classifier training in each validation fold, we normalized the voxel values. That is, voxel values in the training dataset were transformed into z-scores, and the mean and S.D. of each voxel in the training dataset were used to normalize the voxel values in the test dataset. We regarded the mean prediction accuracy across 10 validation folds as the evaluated accuracy of each algorithm in each participant. Finally, we used paired t-tests for statistical evaluation of the differences in accuracy between iSLR and SLR (significance threshold: $p < 0.05$, uncorrected ($t_7 > 2.36$)).

2.5.5. *Spatial distribution of selected voxels*

To demonstrate the usefulness of iSLR for functional mapping, we identified voxels that were selected at least once across the 10 validation folds (= selected voxels). The images representing the selected voxels were then superimposed onto anatomical images of each participant.

2.5.6. *Finger representations*

Using iSLR, we were able to find the largest cluster around the hand section of the contralateral (left) primary sensory-motor cortices in all participants (see Section 3.2.2). To further demonstrate the utility of iSLR for neuroimaging, we performed post hoc analysis of the voxels included in the cluster. Because iSLR is a linear classification algorithm, we can interpret the meaning of each voxel by checking whether its corresponding weight is positive or negative. Specifically, higher values of positively weighted voxels push the ensemble classifier toward predicting label 1, i.e., that the middle finger was used. Conversely, higher values of negatively weighted voxels push the ensemble classifier toward predicting label -1, i.e., that the index finger was used.

On examining the weight vectors obtained from the 10 validation folds in each participant, we found two types of voxels: (1) voxels positively weighted at least once in all validation folds, and (2) voxels negatively weighted at least once. We defined the former voxels as "middle finger voxels" and the latter as "index finger voxels" and checked the spatial location of these voxels within the largest cluster around the hand section of the primary sensory-motor cortices. No voxels were weighted negatively in a validation fold and positively in another.

3. Results

3.1. *Simulation*

Fig. 3A and B is typical examples of the simulation results obtained as intuitive explanation of the iSLR behavior. Because SLR selected only a small proportion of the relevant features (solid yellow line in Fig. 3A), many relevant features (> 68) were available for SLR training (dashed yellow line in Fig. 3A) in the earlier iterations ($N < 12$). As a result, in these iterations the base classifiers could select sufficient relevant features (solid yellow line in Fig. 3A) to provide above-chance performance (yellow line in Fig. 3B). By combining them, the ensemble classifier could select many relevant features (red line in Fig. 3A) and provide better prediction accuracy (red line in Fig. 3B).

However, after a certain point, approximately $N = 15$, the ensemble classifier failed to improve or even degraded the prediction accuracy (red line in Fig. 3B), because the base classifiers were able to select only a few or no relevant features and provided prediction virtually by chance (yellow line in Fig. 3B).

Because we were able to successfully identify the number of iterations around the peak accuracy in the ensemble classifier ($N = 12$; black vertical line in Fig. 3A and B) with cross validation, iSLR (99.5%) successfully improved the prediction accuracy of SLR (84.0%).

This performance improvement was consistently observed in all 100 simulations and the mean prediction accuracy (Fig. 3C) was significantly ($p < 0.001$; $t_{99} = 55.26$) better in iSLR ($98.1 \pm 1.1\%$) than in SLR ($81.0 \pm 3.0\%$).

The simulation results demonstrate that in iSLR, multiple better-than-chance SLR base classifiers are generated in the earlier iterations because sufficient relevant features are available for these SLR base classifiers due to the over-pruning in the previous iterations. Then, by combining these SLR classifiers, iSLR provides more accurate predictions. We confirmed that the performance improvement was also observed when we increased the number of relevant features ($D_{relev} = 2{,}000$ and 20,000). Only when the number of relevant features was very small ($D_{relev} = 20$), did iSLR not

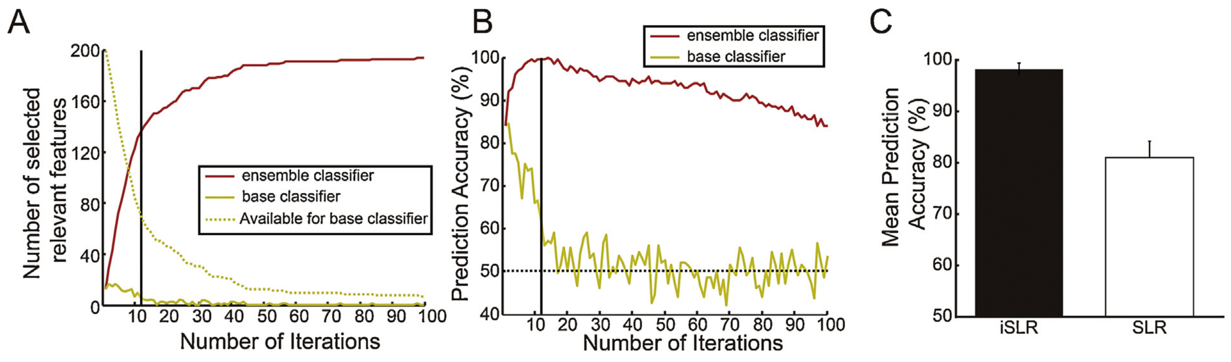

Fig. 3. Results of the simulation: (A) The number of selected relevant features in the base classifiers generated in each iteration (solid yellow line), in the ensemble classifiers (red line), and the number of relevant features available for each base classifier (dashed yellow line) are plotted against the number of iterations. The vertical black line indicates the number of iterations determined by cross validation within the training dataset. (B) Prediction accuracies in base classifiers generated by each iteration (yellow) and in the ensemble classifiers (red). The vertical black line indicates the number of iterations determined by cross validation and the horizontal dashed black line indicates the chance level (50%). (C) Mean prediction accuracies (percentage correct) across 100 simulations in iSLR (black filled circles) and SLR (black open circles).

improve the SLR performance. This could be because the risk of over-pruning relevant features might have originally been small in SLR (Supplementary Fig. 1).

3.2. *fMRI experiment*

3.2.1. *Prediction accuracies in the fMRI experiment*

Fig. 4 summarizes the mean prediction accuracies across participants. We found that the accuracy of iSLR was consistently better than that of SLR in all participants and that the mean prediction accuracy was significantly ($t_7 = 4.8$; $p < 0.05$) better in iSLR (86.5 ± 8.4%) than in SLR (81.1 ± 7.8%). This indicates that iSLR can efficiently recycle informative voxels over-pruned by SLR to improve prediction accuracy. Thus, iSLR can rectify SLR's over-pruning, as shown in the simulation (Fig. 3). We also confirmed that iSLR can provide prediction accuracy comparable to Elastic Net (84.1 ± 8.6%; Zou and Hastie, 2005; De Martino et al., 2008), which was also developed as a solution for over-pruning and recognized as a candidate for analyzing fMRI data (Niazi et al., 2014; Casanova et al., 2013; see Supplementary Material and Supplementary Fig. 2).

3.2.2. *Mapping of the selected voxels*

Fig. 5A shows the rendered images of the selected voxels for a representative participant. Each panel represents the result obtained from each algorithm. iSLR successfully identified a cluster structure of voxels in the sensory-motor cortices, which is a well-known central region for controlling fingers (yellow sections in the left panel in Fig. 5A). Conversely, this was not robustly detected by

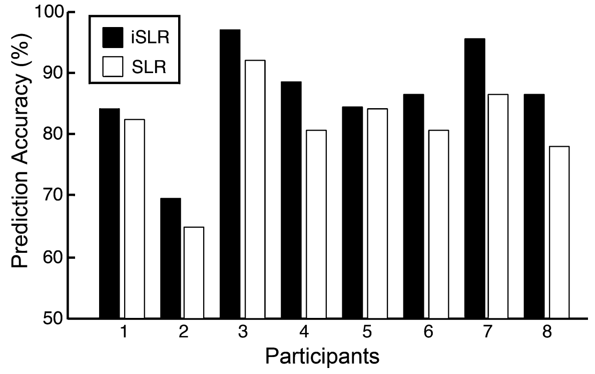

Fig. 4. Prediction accuracies in the fMRI experiment for each participant (1–8) using iSLR (black) and SLR (white).

SLR (yellow section in the left panel in Fig. 5A). Fig. 5B summarizes the sizes of the voxel clusters and their frequencies of appearance obtained from the same representative participant as in Fig. 5A. The size of the cluster (bars indicated with yellow arrows) was prominently greater than those of other clusters in iSLR. However, in SLR, only four voxels were included in the sensory-motor cluster and this size did not seem to be prominent when compared with other clusters comprising two three-voxel clusters.

To verify the consistency of these findings across participants, we compared the mean size of the sensory-motor cluster across participants obtained both from iSLR and from SLR (data not shown in figure). The mean size of the sensory-motor clusters was significantly ($t_7 = 6.5$, $p < 0.05$) larger in iSLR (40.0 ± 15.2 voxels) than in SLR (8.0 ± 1.9 voxels). The difference in size from the second-largest cluster was also significantly ($t_7 = 4.7$, $p < 0.05$) larger in iSLR (31.6 ± 16.5 voxels) than in SLR (3.6 ± 1.9 voxels). This may indicate that iSLR is superior in terms of its ability to identify a prominently larger cluster in the task-relevant brain region.

These results demonstrate the efficacy of iSLR for functional mapping. Specifically, even when much larger numbers of relevant voxels are actually concentrated in a certain brain region, SLR can identify only a few relevant voxels from the region because of over-pruning. This may increase the difficulty experienced in identifying the brain region that represents label information. Thus, by rectifying over-pruning with iRec, iSLR can be viewed as a robust functional mapping tool.

3.2.3. *Finger representations*

Another important finding from the fMRI experiment is that iSLR was able to identify the difference between the spatial representations of the index and middle fingers within the sensory-motor cluster (Fig. 5C). In a representative participant, the index finger voxels (green section in Fig. 5C; normalized coordinates of center-of-gravity (COG) in Montreal Neurological Institute (MNI) coordinates ((x, y, z) = (−45.0 mm, −27.2 mm, 53.8 mm)) were concentrated more inferiorly compared to the middle finger voxels (blue section in Fig. 5C; (x, y, z) = (−28.9 mm, −32.7 mm, 70.5 mm)). Importantly, this finding was consistently observed in all participants. In fact, when we compared the z-coordinate of the COG for the index finger voxels with that for the middle finger voxels, the former was smaller than the latter in all participants. Furthermore, the spatial relationship found in the present study is consistent with previous findings that index finger representation is more inferiorly located in the primary sensory-motor cortices than the middle finger representation, as reported in electrophysiological (Penfield and Boldrey, 1937) and high-resolution neuroimaging (Dechent

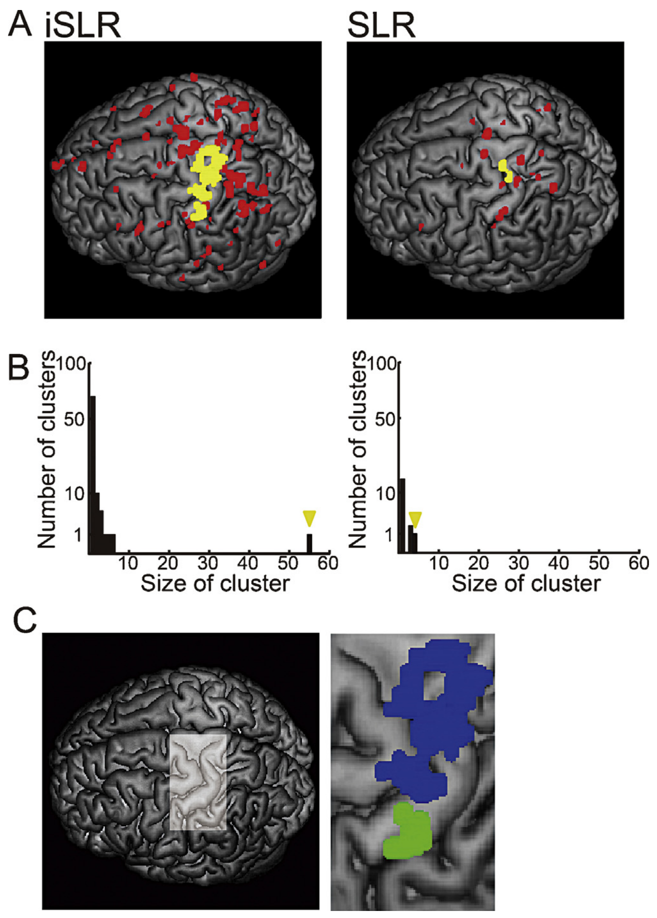

Fig. 5. Results of functional mapping in a representative participant: (A) Spatial locations of voxels selected by iSLR (left panel) and SLR (right panel). The yellow sections represent the largest clusters found around the hand section of the left primary sensory-motor cortices and the red sections represent the other selected voxels. For display purposes, the images representing selected voxels have been transformed into MNI standard brain space, and are rendered onto the surface of the template brain. (B) Histograms of the number of clusters formed by the selected voxels. The horizontal axis indicates the cluster size. The left and right panels represent the results from iSLR and SLR, respectively. In each panel, the vertical axis is log-scaled. A bar, denoted by a yellow arrow in each panel, indicates the largest cluster that was found in the primary sensory-motor cortices. (C) Spatial distributions of the index and middle finger voxels in the sensory-motor cluster identified by iSLR. The white square in the left panel indicates the region around the hand section of the left primary sensory-motor cortices, which is zoomed in the right panel. In the right panel, the green section corresponds to the index finger voxels and the blue section corresponds to the middle finger voxels.

and Frahm, 2003) studies. Thus, the present result demonstrates the utility of iSLR in terms of its capability to identify distinct multiple brain representations within the local brain region.

4. Discussion

In this paper, we proposed an ensemble solution (iRec) that can be applied to any kind of SCA to rectify over-pruning. The results of our simulation and real fMRI experiments demonstrate that iSLR, comprising iRec incorporated with SLR, can rectify over-pruning in SLR and thereby improve prediction accuracy. Further, iSLR is able to identify the brain region representing label information more clearly than SLR.

To the best of our knowledge, this is the first study to have proposed a solution for over-pruning using the ensemble-learning method and also demonstrate its usefulness both for improving label prediction and for mapping functional brain fields. Hereafter, we discuss the efficacy of iRec from the viewpoint of ensemble-learning theory.

In general, it is well known that the accuracy of the ensemble classifier highly depends not only on the accuracy but also the diversity of individual base classifiers (Jacobs, 1995; Tumer and Ghosh, 1995, 1996; Breiman, 2001; Kuncheva, 2004). In particular, ensemble learning can improve prediction accuracy only when multiple better-than-chance base classifiers are generated and these classifiers' predictions are stochastically independent. As demonstrated in the simulation, in iRec, multiple base classifiers can provide better-than-chance prediction accuracy, due to over-pruning in SCA for the training of base classifiers.

One way of increasing the diversity of the base classifiers is to generate base classifiers using disjoint feature subsets. For example, in random subspace ensemble (Ho, 1998), features are randomly separated into disjoint feature subsets and each of them is used to generate each base classifier. In the adaptive iterative learning algorithm based on feature selection and combination voting (AdaFSCV; He and Shenm, 2007), features are ranked based on usefulness for classification evaluated by univariate statistics and separated into disjoint subsets according to the rank (i.e., best group, second best group, . . ., worst group).iRec is also based on this idea. More specifically, in each iteration, we generate a new base classifier from the updated training dataset from which we excluded the previously selected features. Consequently, there is no overlap between the feature subsets used for label prediction in the base classifiers. Hence, we can presume that this updating rule increases, at least to some extent, the likelihood of independency of each base classifier.

A unique advantage of iRec is that it can perform both feature division and classifier training simultaneously by taking multivariate structure (i.e., correlation between features) into consideration. Thus, there is no risk of iRec corrupting the correlation structure of features during the feature separation procedure, whereas previous approaches (random separation or separation based on the univariate analysis) may carry such a risk. This is very important in fMRI decoding because fMRI signals normally tend to correlate highly across voxels and better prediction can be obtained when some subsets of voxels are used together even though each is not highly informative (Yamashita et al., 2008).

Acknowledgments

We would like to thank Dr. Matthew DeBrecht at the Center for Information and Neural Networks (CiNet), National Institute of Information and Communications Technology for helpful comments and discussions on earlier versions of this paper and also for his help with English phrasing. We are also grateful to Prof. Mitsuo Kawato at the Advanced Telecommunications Research Institute International (ATR) Computational Neuroscience Laboratories, and Prof. Masa-aki Sato and Prof. Okito Yamashita at the ATR Neural Information Analysis Laboratories for their many helpful suggestions. We would also like to thank Dr. Yu Shimizu at the Okinawa Institute of Science and Technology Graduate University for proofreading our documents. This work was supported by Scientific Research on Innovative Areas "Embodied-brain" (JSPS KAKENHI No. 26120003) to author EN, by Grant-in-Aid for Specially Promoted Research (No.24000012) to EN and JSPS Kakenhi (24800027) to author IN. Author SH was supported by a JSPS Research Fellowship for Young Scientists (No. 238768).

Appendix A. iSLR is a linear classifier.

In iSLR, the product rule is used when combining the base classifiers. Specifically, the ensemble prediction is defined as the label that maximizes the product of the associated probabilities predicted by the SLR sub-classifier in Eq. (2). This meta-classifier is equivalent to a linear classifier, whose weight vector is the sum of the sub-classifiers' weights (Eq. (3)). This is proven as follows.

Since y takes a value of 1 or −1, a different form of Eq. (2) is

$$y_{predicated} = \begin{cases} 1 & \text{if} \prod_{n=1}^{N} P(y = 1\,|\boldsymbol{x}_{test}, \boldsymbol{w}) - \prod_{n=1}^{N} P(y = -1\,|\boldsymbol{x}_{test}, \boldsymbol{w}) > 0 \\ -1 & \text{otherwise} \end{cases} \tag{s1}$$

Then, by inserting Eq. (1) and noting that $1/1 + e^{-\boldsymbol{w}_{(n)}^T \boldsymbol{x}_{test} \times (-1)} = e^{-\boldsymbol{w}_{(n)}^T \boldsymbol{x}_{test}} / 1 + e^{-\boldsymbol{w}_{(n)}^T \boldsymbol{x}_{test}}$, the conditional expression can be transformed as follows:

$$\begin{aligned} &\prod_{n=1}^{N} P(y = 1|\boldsymbol{x}_{test}, \boldsymbol{w}_{(n)}) - \prod_{n=1}^{N} P(y = -1|\boldsymbol{x}_{test}, \boldsymbol{w}_{(n)}) \\ &= \prod_{n=1}^{N} \frac{1}{1 + e^{-\boldsymbol{w}_{(n)}^T \boldsymbol{x}_{test}}} - \prod_{n=1}^{N} \frac{1}{1 + e^{-\boldsymbol{w}_{(n)}^T \boldsymbol{x}_{test} \times (-1)}} \\ &= \frac{1 - \prod_{n=1}^{N} e^{-\boldsymbol{w}_{(n)}^T \boldsymbol{x}_{test}}}{\prod_{n=1}^{N} 1 + e^{-\boldsymbol{w}_{(n)}^T \boldsymbol{x}_{test}}} \\ &= \frac{1 - e^{-\left(\sum_{n=1}^{N} \boldsymbol{w}_{(n)}^T \boldsymbol{x}_{test}\right)}}{\prod_{n=1}^{N} 1 + e^{-\boldsymbol{w}_{(n)}^T \boldsymbol{x}_{test}}} \end{aligned} \tag{s2}$$

Since the denominator in the last line of (s2) is always positive, and the numerator is positive when $\left(\sum_{n=1}^{N} \boldsymbol{w}_{(n)}^T \boldsymbol{x}_{test}\right)$ is positive, Eq. (s1) is equivalent to Eq. (s3):

$$y_{predicated} = \begin{cases} 1 & \left(\sum_{n=1}^{N} \boldsymbol{w}_{(n)}\right)^T \boldsymbol{x}_{test} > 0 \\ -1 & \text{otherwise} \end{cases} \tag{s3}$$

Appendix B. Supplementary data

Supplementary data associated with this article can be found, in the online version, at http://dx.doi.org/10.1016/j.jneumeth.2014.10.023.

References

Aguirre GK, Zarahn E, D'esposito M. The variability of human BOLD hemodynamic responses. NeuroImage 1998;8(4):360–9.
Breiman L. Random forests. Mach Learn 2001;45(1):5–32.
Casanova R, Hsu FC, Sink KM, Rapp SR, Williamson JD, Resnick SM, et al. Alzheimer's disease risk assessment using large-scale machine learning methods. PLOS ONE 2013;8(11):e77949.
Dechent P, Frahm J. Functional somatotopy of finger representations in human primary motor cortex. Hum Brain Mapp 2003;18(4):272–83.
De Martino F, Valente G, Staeren N, Ashburner J, Goebel R, Formisano E. Combining multivariate voxel selection and support vector machines for mapping and classification of fMRI spatial patterns. NeuroImage 2008;43:44–58.
Friston KJ, Holmes AP, Worsley KJ, Poline JP, Frith CD, Frackowiak RSJ. Statistical parametric maps in functional imaging: a general linear approach. Hum Brain Mapp 1995;2(4):189–210.
Formisano E, De Martino F, Valente G. Multivariate analysis of fMRI time series: classification and regression of brain responses using machine learning. Magn Reson Imaging 2008;26(7):921–34.
Grosenick L, Klingenberg B, Katovich K, Knutson B, Taylor JE. Interpretable whole-brain prediction analysis with GraphNet. NeuroImage 2013;72(C):304–21, http://dx.doi.org/10.1016/j.neuroimage.2012.12.062.
Haynes JD, Rees G. Decoding mental states from brain activity in humans. Nat Rev Neurosci 2006;7(7):523–34.
He H, Shenm H. Adaptive iterative learning for classification based on feature selection and combination voting. IJCNN 2007;2007:2800–5.
Hinton G. Training products of experts by minimizing contrastive divergence. Neural Comput 2002;14(8):1771–800.
Ho TK. The random space method for constructing decision forests. IEEE Trans Pattern Anal Mach Intel 1998;20(8):832–44.
Jacobs RA. Methods for combining experts' probability assessments. Neural Comput 1995;7:867–88.
Kittele J, Hatef M, Duin R, Matas J. On combining classifiers. IEEE Trans Pattern Anal Mach Intel 1998;20(3):226–39.
Kuncheva LI. Combining pattern classifiers. John Wiley & Sons; 2004.
MacKay D. Bayesian interpolation. Neural Comput 1992;4:415–47.
Michel V, Gramfort A, Varoquaux G, Eger E, Thirion B. Total variation regularization for fMRI-based prediction of behavior. IEEE Trans Med Imaging 2011;30(7):1328–40.
Ng B, Vahdat A, Hamarneh G, Abugharbieh R. Generalized sparse classifiers for decoding cognitive states in fMRI. Mach Learn Med Imaging 2010:108–15.
Ng B, Abugharbieh R. Modeling spatiotemporal structure in fMRI brain decoding using generalized sparse classifiers. In: International workshop on pattern recognition in neuroimaging (PRNI); 2011. p. 65–8.
Niazi AM, Broek P, Klanke S, Barth M. Online decoding of object-based attention using real-time fMRI. Eur J Neurosci 2014;39(2):319–29.
Norman KA, Polyn SM, Detre GJ, Haxby JV. Beyond mind-reading: multi-voxel pattern analysis of fMRI data. Trends Cogn Sci 2006;10(9):424–30.
Oldfield RC. The assessment and analysis of handedness: the Edinburgh inventory. Neuropsychologia 1971;9(1):97–113.
Penfield W, Boldrey E. Somatic motor and sensory representation in the cerebral cortex of man as studied by electrical stimulation. Brain: J Neurol 1937;60(4):389–443.
Pereira F, Mitchell T, Botvinick M. Machine learning classifiers and fMRI: a tutorial overview. NeuroImage 2009;45(1 Suppl.):S199–209.
Ryali S, Supekar K, Abrams DA, Menon V. Sparse logistic regression for whole-brain classification of fMRI data. NeuroImage 2010;51(2):752–64.
Tibshirani R. Regression shrinkage and selection via the lasso. J R Stat Soc Ser B (Methodol) 1996;58(1):267–88.
Tumer K, Ghosh J. Classifier combining: analytical results and implications. In: Proceedings of the AAAI-96 workshop on integrating multiple learned models for improving and scaling machine learning algorithms; 1995.
Tumer K, Ghosh J. Error correlation and error reduction in ensemble classifiers. Connect Sci 1996;8:385–404.
Worsley KJ, Liao CH, Aston J, Petre V, Duncan GH, Morales F, et al. A general statistical analysis for fMRI data. NeuroImage 2002;15(1):1–15.
Yamashita O, Sato M, Yoshioka T, Tong F, Kamitani Y. Sparse estimation automatically selects voxels relevant for the decoding of fMRI activity patterns. NeuroImage 2008;42(4):1414–29.
Zou H, Hastie T. Regularization and variable selection via the elastic net. J R Stat Soc 2005;67(2):301–20.