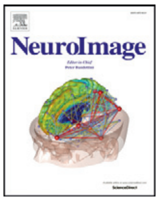
NeuroImage 145 (2017) 314–328

Contents lists available at ScienceDirect

NeuroImage

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



Task-specific feature extraction and classification of fMRI volumes using a deep neural network initialized with a deep belief network: Evaluation using sensorimotor tasks

Hojin Jang [a], Sergey M. Plis [b], Vince D. Calhoun [b,c,*], Jong-Hwan Lee [a,**]

[a] Department of Brain and Cognitive Engineering, Korea University, Seoul, Republic of Korea
[b] The Mind Research Network & LBERI, Albuquerque, NM, USA
[c] Department of Electrical and Computer Engineering, University of New Mexico, Albuquerque, NM, USA

ARTICLE INFO

Article history:
Received 15 November 2015
Revised 10 March 2016
Accepted 1 April 2016
Available online 11 April 2016

ABSTRACT

Feedforward deep neural networks (DNNs), artificial neural networks with multiple hidden layers, have recently demonstrated a record-breaking performance in multiple areas of applications in computer vision and speech processing. Following the success, DNNs have been applied to neuroimaging modalities including functional/structural magnetic resonance imaging (MRI) and positron-emission tomography data. However, no study has explicitly applied DNNs to 3D whole-brain fMRI volumes and thereby extracted hidden volumetric representations of fMRI that are discriminative for a task performed as the fMRI volume was acquired. Our study applied fully connected feedforward DNN to fMRI volumes collected in four sensorimotor tasks (i.e., left-hand clenching, right-hand clenching, auditory attention, and visual stimulus) undertaken by 12 healthy participants. Using a leave-one-subject-out cross-validation scheme, a restricted Boltzmann machine-based deep belief network was pretrained and used to initialize weights of the DNN. The pretrained DNN was fine-tuned while systematically controlling weight-sparsity levels across hidden layers. Optimal weight-sparsity levels were determined from a minimum validation error rate of fMRI volume classification. Minimum error rates (mean $\pm$ standard deviation; %) of 6.9 ($\pm$3.8) were obtained from the three-layer DNN with the sparsest condition of weights across the three hidden layers. These error rates were even lower than the error rates from the single-layer network (9.4 $\pm$ 4.6) and the two-layer network (7.4 $\pm$ 4.1). The estimated DNN weights showed spatial patterns that are remarkably task-specific, particularly in the higher layers. The output values of the third hidden layer represented distinct patterns/codes of the 3D whole-brain fMRI volume and encoded the information of the tasks as evaluated from representational similarity analysis. Our reported findings show the ability of the DNN to classify a single fMRI volume based on the extraction of hidden representations of fMRI volumes associated with tasks across multiple hidden layers. Our study may be beneficial to the automatic classification/diagnosis of neuropsychiatric and neurological diseases and prediction of disease severity and recovery in (pre-) clinical settings using fMRI volumes without requiring an estimation of activation patterns or ad hoc statistical evaluation.

1. Introduction

Since the development of an efficient weight initialization scheme for a deep belief network (DBN) using the restricted Boltzmann machine (RBM) (Hinton, 2002; Hinton and Salakhutdinov, 2006) and the stacked auto-encoder (SAE) (Bengio et al., 2007), artificial deep neural networks (DNNs) have been gainfully utilized for the feature extraction and/or classification of multimedia data, such as image (Krizhevsky et al., 2012) and speech (Hinton et al., 2012a) data, and have shown unprecedented classification performance. Recently, the convolutional variants (convolutional neural networks or CNNs) trained on millions of natural images have been used to model internal representations of human visual cortices along the ventral pathway (Güçlü and van Gerven, 2015; Khaligh-Razavi and Kriegeskorte, 2014). The biological plausibility of the hidden representations of the trained CNNs has been shown using a linear mapping function between the hidden representations and fMRI brain activity (Güçlü and van Gerven, 2015; Van Gerven et al., 2010) and using representational similarity/dissimilarity analysis (Khaligh-Razavi and Kriegeskorte, 2014).

Using an RBM as a building block of the DBN, Hjelm et al. (2014) presented a comprehensive picture of the RBM application to fMRI data, in

* Correspondence to: V.D. Calhoun, The Mind Research Network, 1101 Yale Blvd NE, Albuquerque, NM, USA.
** Correspondence to: J.H. Lee, Department of Brain and Cognitive Engineering, Korea University, Anam-dong 5ga, Seongbuk-gu, Seoul 02841, Republic of Korea.
E-mail addresses: vcalhoun@mrn.org (V.D. Calhoun), jonghwan_lee@korea.ac.kr (J.-H. Lee).

http://dx.doi.org/10.1016/j.neuroimage.2016.04.003
1053-8119/© 2016 Elsevier Inc. All rights reserved.

which the RBM was applied to both simulated and real auditory oddball task fMRI and compared the RBM to conventional single-matrix factorization methods, including independent component analysis (ICA). The results showed the RBM's ability to extract both temporal and spatial patterns from the 4-D fMRI volumes and to obtain slightly superior temporal features than the ICA (Hjelm et al., 2014). More recently, DNNs have directly been applied to neuroimaging data to obtain efficient feature representations (Kim et al., 2016; Plis et al., 2014; Suk et al., 2014; Zhang et al., 2015b). Zhang et al. (2015a, 2015b) employed CNNs to extract hidden feature representations from anatomical MRI data, such as T1 and T2 images, and a fractional anisotropy map from the diffusion-weighted images. The gray matter, white matter, and cerebrospinal fluid regions were automatically segmented using the data from infants with an isointense phase (i.e., almost identical gray and white matter intensities) (Zhang et al., 2015b). In another recent study, Plis et al. (2014) used structural MRI data as the input for a DBN that consisted of RBMs as building blocks. The authors performed (1) classification to discriminate schizophrenia patients from healthy controls and (2) the prediction of the disease severity of patients with Huntington's disease. The resulting classification accuracy and prediction capacity were substantially increased when the three-layer DBN, rather than one-layer or two-layer DBNs, was used for classification and prediction (Plis et al., 2014). More recently deep learning has also been used to implement non-linear ICA applied to structural brain imaging data (Castro et al., 2015) and even to perform multimodal data fusion of brain structure and function (Amin et al., 2015).

Despite these recent investigations of the feature extraction and classification of MRI/fMRI data using deep networks, no study has explicitly employed whole-brain fMRI volume as an input for the DNN and blindly extracted hidden features from the fMRI data. Schmah et al. (2008) developed an RBM model-based classifier and applied them to fMRI data to classify (1) either finger or wrist movement and (2) either the early or the late post-stroke period for stroke-recovery patients. However, the input fMRI data were not from the whole brain but, rather, one slice or a few slices due to the small number of samples compared to input dimensions, and the classifier consisted of a single-layer RBM rather than a multiple-layer DBN (Schmah et al., 2008). In our study, we were motivated to deploy the DBN to initialize the hidden features of whole-brain fMRI volumes via the representation of non-linear hidden layers. Consequently, classification of a single fMRI volume would be enabled using the fully connected feedforward DNN (denoted by DNN) initialized with the same number of hidden layers as the pretrained DBN but with the addition of an output layer with class indices in the fine-tuning phase (Kim et al., 2016; Plis et al., 2014). We anticipated that the representations of the non-linear hidden layers from the trained DNN would provide significantly task-specific feature representations. Using hidden representations, the classification performance was evaluated by deploying classifiers such as a softmax and a support vector machine (SVM) (Cortes and Vapnik, 1995; Lee et al., 2009).

The curse of dimensionality was evident when the DNN with tens of thousands of input nodes (i.e., number of voxels; more than approximately 70,000 voxels within a whole brain with a 3-mm isotropic voxel size from a single fMRI volume) was trained on a limited number of input samples. For example, single subject fMRI collection at 2 s TR (time of repetition) for 1 h results merely in 1800 fMRI volumes, which is significantly smaller than the number of voxels per volume. To address this issue, explicit control of the weight-sparsity level via L1-norm regularization of the DNN weights has been shown to be beneficial (Kim et al., 2016). In the current study, the DBN was adopted for greedy layer-wise pretraining of the DNN. In the fine-tuning phase of the training the sparsity levels of the weight parameters for each of the hidden layers were automatically adapted to maintain the desired percentage of active (non-zero) weights (Kim et al., 2016). The efficacy of our proposed approach to classifying a single fMRI volume using the DBN pretrained DNN with an explicit weight-sparsity control scheme is presented using the fMRI data acquired from four sensorimotor tasks.

2. Materials and methods

2.1. Overview

Fig. 1 illustrates a flow diagram of our overall study. Raw blood-oxygenation-level-dependent (BOLD) fMRI data were preprocessed, and in-brain voxels were included in subsequent analyses. The percentage signal change with respect to baseline BOLD intensity was calculated, and task-related fMRI volumes were considered as input to the DNN. Based on a leave-one-subject-out cross-validation scheme from a total of 12 subjects, training data (i.e., data from 11 training subjects) were used for the DBN pre-training followed by the DNN fine-tuning across all combinatorial scenarios of weight-sparsity levels across all hidden layers. The trained DNN was validated using the data from one remaining validation subject by calculating the classification error of the softmax output layer or of the SVM classifier using the top hidden layer output. Based on the DNN with a minimum error rate, the learned weights and representation of hidden layer output were interpreted. A nested cross-validation framework was also employed to optimize the target percentage of non-zero weights in the three-layer DNN.

2.2. Participants and fMRI data acquisition/preprocessing

An institutional review board (IRB) at Korea University approved the study protocol. Participants provided written consent forms and were compensated for their participation as denoted in the IRB documents. Twelve right-handed, young, and healthy volunteers without any neuropsychiatric or neurologic problems participated (age $= 25.0 \pm 2.0$ years; handedness $= 88.3 \pm 10.8$ using the Edinburgh handedness inventory (Oldfield, 1971)). Participants performed each of the four sensorimotor tasks, including left-hand clenching (LH), right-hand clenching (RH), auditory attention (AD), and visual stimulus (VS) tasks, and one fMRI run per task was acquired (Kim et al., 2012). Each fMRI run (150 s) included a rest period (30 s) in the beginning and three blocks of task periods (20 s per block) alternating with a rest period (20 s). In the LH and RH tasks, participants clenched their left or right hand approximately twice per second. In the AD task, the participants listened to sounds of 1 kHz during the task period and 900 Hz during the rest period. In the VS task, the participants watched an alternating black and white checkerboard with 8 Hz during the task period and watched a black screen during the rest period. Task instructions were delivered to participants during the task period via MR-compatible goggles (NordicNeuroLab Inc., Bergen, Norway).

BOLD fMRI data were acquired using a standard gradient-echo echo-planar imaging pulse sequence (TR/TE $= 2000/30$ ms; flip angle $= 90°$; in-plane voxels $= 64 \times 64$; field-of-view $= 240 \times 240$ mm^2; 36 axial slices; 4 mm slice thickness without a gap; voxel size $= 3.75 \times 3.75 \times 4.0$ mm^3) and a 3-T Tim Trio MRI scanner with a 12-channel head coil (Siemens, Erlangen, Germany). For each fMRI run, the first five volumes (10 s) were discarded to equilibrate the T1 effect. The remaining 70 fMRI volumes (i.e., 140 s) were preprocessed using the SPM8 software toolbox (www.fil.ion.ucl.ac.uk/spm) with standard options, including slice timing correction, motion correction, spatial normalization to the Montreal Neurological Institute space with a 3-mm isotropic voxel size, and spatial smoothing using an 8-mm full-width at half-maximum Gaussian kernel. To evaluate the quality of the acquired fMRI data and preprocessing, an individual activation map from a general linear model (GLM) was obtained using the onset and duration of the task paradigm convolved with a canonical hemodynamic response function, as implemented in SPM8. Then, a one-sample t-test was performed to calculate the group inference of the GLM activation maps across the 12 subjects. Multiple comparisons were addressed using a family-wise-error (FWE) corrected p-value

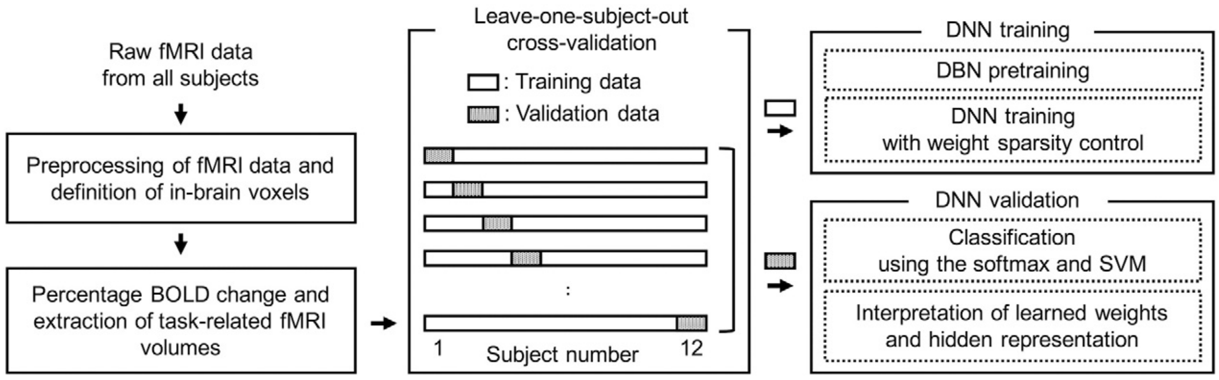

Raw fMRI data from all subjects
Preprocessing of fMRI data and definition of in-brain voxels
Percentage BOLD change and extraction of task-related fMRI volumes
Leave-one-subject-out cross-validation
: Training data
: Validation data
1
Subject number
12
DNN training
DBN pretraining
DNN training with weight sparsity control
DNN validation
Classification using the softmax and SVM
Interpretation of learned weights and hidden representation

Fig. 1. Flow diagram of our overall study. fMRI, functional magnetic resonance imaging; BOLD, blood-oxygenation-level-dependent; DNN, deep neural network; DBN, deep belief network; SVM, support vector machine.

estimated from a random field theory as implemented in SPM8 if applicable.

2.3. Preparation of fMRI volumes as input to the DNN

The voxels whose BOLD intensities were greater than 20% of the maximum BOLD intensity for each preprocessed fMRI volume were defined as an in-brain mask from the fMRI volume. Then, the in-brain mask that overlapped across all fMRI volumes and all subjects was used to define the final set of input voxels (Kim et al., 2013; Kim and Lee, 2013), resulting in a total of 74,484 in-brain voxels. The BOLD intensities of the fMRI volumes within a task-related period (i.e., three blocks of 20-s task periods after a 6-s delay from the task onset for each fMRI run) were normalized to percent signal change relative to the average BOLD signal with the 80-s baseline period. The 30 task-related fMRI volumes (i.e., 60-s task-related period across the three task blocks with 2-s TR) for each fMRI run were used as the input of the DBN and DNN (dimension of one input sample = $74{,}484 \times 1$). A total of 1440 fMRI volumes (i.e., 30 volumes/run $\times$ 4 runs/subject $\times$ 12 subjects) were available across all tasks and all subjects.

2.4. Pretraining of the DNN using DBN

A DBN is a multi-layer directed network whose layers were greedy layer-wise trained using the unsupervised undirected RBM ((Hinton et al., 2006); see Appendix A for details of the DBN training algorithms). The DBN was used to initialize/pretrain the DNN as shown in Fig. 2 since it has been reported that pretraining the DNN using the DBN facilitates

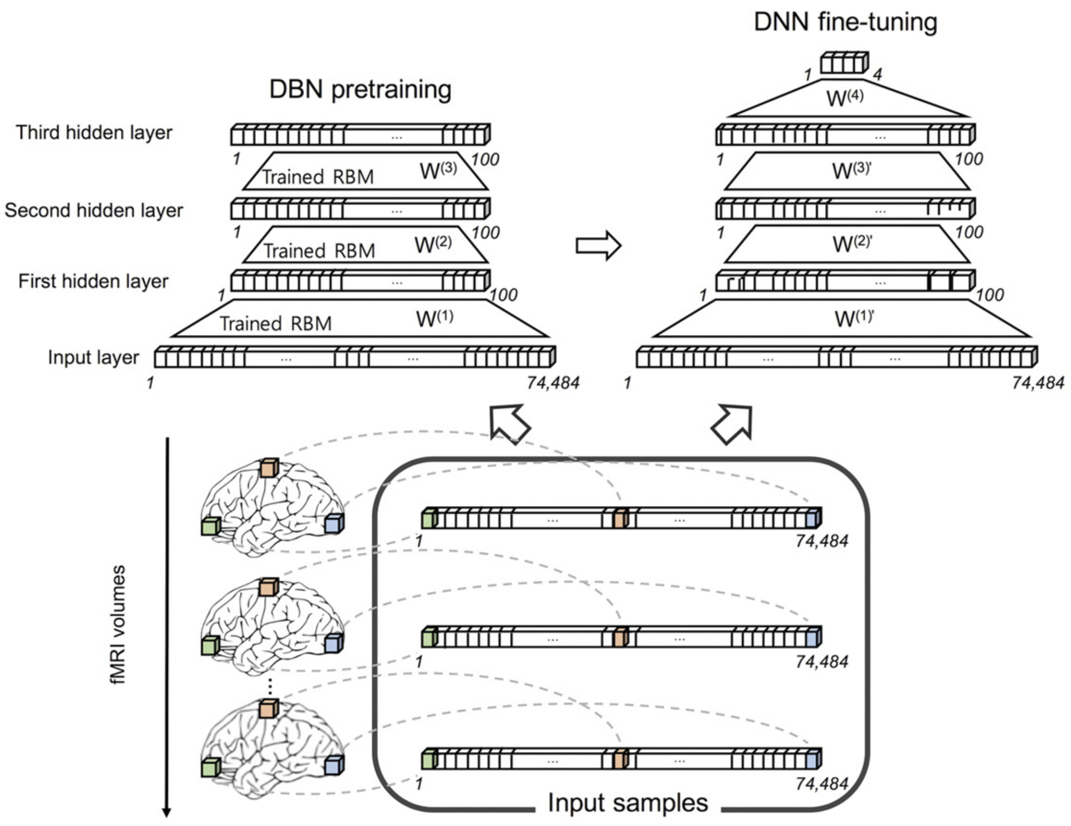


Fig. 2. Schematics of the DBN pretraining followed by the DNN fine-tuning exemplified for three hidden layers. The input layer consisted of 74,484 in-brain voxels of an fMRI volume, each hidden layer consisted of 100 hidden nodes, and the output layer consisted of four nodes to assign each of the four tasks (i.e., LH, RH, AD, and VS). The DBN and DNN were trained using 1320 fMRI volumes of input samples across 11 training subjects (i.e., 30 volumes per task for each subject), and the trained DNN was validated using 120 fMRI volumes from one remaining validation subject. LH, left-hand clenching; RH, right-hand clenching; AD, auditory attention; VS, visual stimulus.

the convergence of the DNN during fine-tuning (Erhan et al., 2010). The weights of the RBM were initialized using random values within the range of $\pm 2\sqrt{6/(n_{in}+n_{out})}$, where n_{in} and n_{out} are the numbers of nodes in the visible and hidden layers (Kim et al., 2016). The biases were initially zero. The weights and biases were trained over 300 epochs. The learning rates were set to 10^{-5}, 10^{-2}, and 10^{-2} for the first, second, and third layers, respectively. The momentum factor was 0.5 and the mini-batch size was 10. Once the Gaussian–Bernoulli RBM between the input layer and the first hidden layer was trained, the weight parameter, $\mathbf{W}^{(1)}$, was fixed and the output of the first hidden layer was used as the input (i.e., visible nodes) to a Bernoulli–Bernoulli RBM between the first and second hidden layers, after which the stacking of the Bernoulli–Bernoulli RBMs continued to all hidden layers of the DBN (Fig. 2). The hyperbolic tangent (tanh, (Hjelm et al., 2014)) and rectified linear unit (ReLU, (Dahl et al., 2013); see "Appendix C. *Rectified linear unit*" for details) were used as node functions.

2.5. Fine-tuning of the DNN with weight-sparsity control

The pretrained DBN weights were used as the initial values of the DNN (Fig. 2). An output layer with four nodes for the four tasks was added to the top hidden layer. Then, the DNN was fine-tuned using a back-propagation algorithm by minimizing the mean-squared error (MSE) between the four actual output values and the desired ones (see Appendix B for mathematical details). At each layer four target percentages of non-zero values of weights (i.e., $\rho = 0.2$, 0.4, 0.6, and 0.8) were used as candidate weight-sparsity levels. All combinatorial scenarios of these percentages of non-zero weights across the hidden layers were used to set the weight-sparsity levels of the DNN, and the corresponding classification accuracies were obtained. The overall learning rate $\alpha(t)$ was initially 10^{-3} and was gradually reduced after 100 epochs to 10^{-6}, μ was 10^{-3}, and ε was 10^{-3}. The momentum factor was set to 0.1. The DBN and DNN algorithms, including functions to implement feed-forward neural networks, were adopted from the DeepLearnToolbox implemented in the MATLAB environment (github.com/rasmusbergpalm/DeepLearnToolbox) and were modified to apply our proposed weight-sparsity control scheme. DNNs and DBNs with one, two, and three hidden layers were used. The number of hidden nodes at each hidden layer was 100 and the mini-batch size was 10. Tanh and ReLU functions were used as activation functions of hidden nodes by matching the activation function of the DBN pretraining.

2.6. Training the DNN and classification via a leave-one-subject-out cross-validation

Based on a leave-one-subject-out cross-validation scheme, 1320 fMRI volumes from 11 subjects were used to train the DBN and DNN, and then 120 fMRI volumes from one remaining validation subject were used to evaluate the classification performance using the trained DNN (Figs. 1 and 2). The softmax and SVM classifiers, applied to the output layer of the DNN and to the top hidden layer, respectively, were used to determine the target class label of the input fMRI volume from the validation subject. A linear SVM was adopted to classify the representations of the hidden layer of the trained DNN without an additional non-linear transformation (essentially, treating DNN as a data-driven feature-space projection method). The soft margin parameter *C* was selected by cross validating grid search on the interval 2^{-5}, …, 2^{15} with step of 2^2 in the leave-one-subject-out cross-validation. The SVM classifier implemented in the LIBSVM software toolbox in the MATLAB environment (www.csie.ntu.edu.tw/~cjlin/libsvm) was used. The error rates obtained across the four target percentages of non-zero weights for each layer were evaluated using one-way analysis-of-variance (ANOVA) and then an ad hoc paired *t*-test. A Bonferroni-corrected *p*-value was used for multiple comparison correction.

To impose further sparsity to the weight parameters between the input layer and the first hidden layer (i.e., the receptive fields of the voxels to the first hidden layer), additional target percentage of non-zero weights of 0.1 and 0.05 was adopted when the three-layer DNN was fine-tuned, and the classification performance was evaluated. For visualization, the representations of the hidden node output (100×1 vector) in the top hidden layer of the fine-tuned DNN were transformed into 2-D representations (2×1 vector) using the *t*-distributed stochastic neighbor embedding (*t*-SNE) method (Hinton and Salakhutdinov, 2006; Van der Maaten and Hinton, 2008).

2.7. Interpretation of weight parameters of the DNN

Performance of the DNN classifier is often attributed to the efficiency of data (in our case fMRI volumes) representation obtained at the top hidden layer. This representation is completely determined by the value of free parameters-also called receptive fields or features. In this work, high level features at level *l* (i.e., the *l*th hidden layer) were computed as a linear composition of weight-feature matrices from all lower layers as follows (Erhan et al., 2009):

$$\mathbf{F}_{\mathbf{W}}^{(l)} = \mathbf{W}^{(l)}\mathbf{W}^{(l\text{-}1)}\cdots\mathbf{W}^{(1)}, \tag{1}$$

$$\mathbf{W}^{(m)} = \left[\mathbf{w}_1^{(m),\ T};\ \cdots;\ \mathbf{w}_n^{(m),\ T}\right],\ \text{and}\ w_{ij}^{(m)} = \begin{cases} w_{ij}^{(m)}, & \text{if } \left|w_{ij}^{(m)}\right| \geq c_j^{(m)} \\ 0, & \text{otherwise} \end{cases}, \tag{2}$$

where the superscript and subscript denote the hidden layer index and node index, respectively; $c_j^{(m)}$ is the threshold value among the elements of $|\mathbf{w}_j^{(m)}|$ to include a subset of the elements of $\mathbf{w}_j^{(m)}$ with the largest magnitude. A subset of the elements (i.e., 5, 10, 20, 50, and 80) as well as all 100 elements for each weight vector $\mathbf{w}_j^{(m)}$ (i.e., the weight vector from the nodes at the $(m\text{-}1)$th layer to the *j*th node at the *m*th layer) were used to calculate the weight-feature representation. Thus, the robustness of the reconstructed weight-feature representation was evaluated when the elements with lower magnitude in the weight vectors were ignored.

Each column vector of $\mathbf{F}_{\mathbf{W}}^{(l)}$ represents the spatial pattern/filter obtained from the weight parameters in the *l*th layer ($M \times N$; *N* is the input dimension, i.e., 74,484; *M* is the number of hidden nodes, i.e., 100). The anatomical locations of each of the spatial patterns/filters were defined by the automated anatomical labeling (AAL) map (Tzourio-Mazoyer et al., 2002). More specifically, for each of the column vectors of weight-feature representations $\mathbf{F}_{\mathbf{W}}^{(l)}$, the corresponding elements were *z*-scored and the voxel locations of the elements that represented statistically significant intensities (uncorrected $p < 10^{-3}$ with a minimum of 5 connected voxels) were quantitatively evaluated using an overlap ratio with each of the AAL regions. For example, if the voxels fully occupied any of the AAL regions, the overlap ratio with the corresponding AAL region was 1. The overlap ratios across all 116 AAL regions were visualized as a 2-D image plot (Lee et al., 2012a).

The output layer extracts predominantly task-specific features to assure that the values at the output layer could be used to discriminate the class information of the input sample. Exploiting this, we quantified the degree of the task specificity by the overlap ratio between the voxels that represented the top 5 percentile intensities (1) from the group inference of the GLM activation maps from the 11 training subjects for each of the four tasks and (2) from the weight-feature representations at the output layer was calculated.

2.8. Interpretation of hidden node outputs of the DNN

The output of the hidden nodes in each hidden layer (i.e., activations with applied nonlinearities) provided hidden representations of the input fMRI volumes. Consequently, the hidden representations of the optimal DNN with the lowest validation error rate were used to interpret the DNN and to investigate an association with each of the tasks in the corresponding hidden layer. Specifically, the hidden layer output representations were calculated for each of the hidden layers for all fMRI volumes ($n = 1440$), that is, those of the 11 training subjects and the validation subject. For each task and for each subject, the average hidden layer output representation was obtained from 30 hidden representations of 30 input fMRI volumes. Then, for each subject, there were four average hidden output representations across the four tasks at each hidden layer, and a representational similarity analysis (RSA) (Kriegeskorte et al., 2008) was conducted using the hidden representations (100×1 vector for each hidden representation) across all four tasks and all 11 training subjects. Several significantly active nodes of the hidden layer or voxels of the input layer could dominate the overall patterns of the RSA matrices. Thus, the significantly active hidden or input nodes with values in the top 5 percentile for each task were excluded, when the corresponding RSA matrices were calculated.

In addition to the RSA, the average hidden representations or average input patterns from each of the four tasks were evaluated on training subjects. In detail, for each training subject, an average hidden representation pattern or input pattern of one of four tasks was selected and the closely matched average hidden representation or input pattern obtained from each of the 10 remaining subjects was identified among the four tasks by calculating Pearson's correlation coefficients. If the associated task labels of the two matched patterns were identical, they were considered to be correctly matched subjects (i.e., a maximum of 10 would be counted from the 11 training subjects for each task). Similarly, the hidden representations and input patterns from one validation subject were evaluated using those from the 11 training subjects (in this case, the maximum number of matched subjects would be 11). The numbers of correctly matched subjects were also calculated when significantly active hidden or input nodes with values in the top 5 percentile were excluded for each task.

2.9. Classification performance under various scenarios of hidden layer output from the DBN/DNN

To evaluate the efficacy of (a) the DNN fine-tuning and (b) the DBN pretraining, the hidden layer output of (1) the pretrained DBN and (2) the fine-tuned DNN with/without DBN pretraining were used as inputs to the softmax and SVM classifiers. In this way classification performance was measured under various scenarios of the DBN without DNN fine-tuning and of the DNN with/without DBN pretraining.

2.10. Classification test using the DBN-pretrained DNN from a nested cross-validation framework

To apply the DNN in real situation and obtain the classification performance of a test subject, a nested cross-validation needs to be performed to optimize target percentage of non-zero weights. To ascertain the classification performance with one test subject via a nested cross-validation, 11 sets of leave-one-subject-out cross-validation were performed using the data from the remaining 11 subjects for training/validation. For each of the 11 sets, the 1200 fMRI volumes of 10 out of the 11 subjects were used to train the DBN and DNN with candidate target percentages of non-zero weights across all the hidden layers. The 120 fMRI volumes of the validation subject were used to validate the classification performance of the trained DNN with each candidate target percentage of non-zero weights. This training and validation of the DNN with the candidate target percentage of non-zero weights was repeated for all 11 sets of 10 training subjects and one validation subject to obtain the average validation accuracy. The lowest average validation error rate was determined to identify the optimal target percentage of non-zero weights. The DNN was then re-trained using the optimal target percentage and the 1320 fMRI volumes from the 11 training/validation subjects and tested using the 120 fMRI volumes from the one remaining test subject to obtain test accuracy. The softmax and linear SVM classifiers were applied to the output of the output layer and to the output of the top hidden layer of the DNN. The three-layer DNN was used, and one subject whose error rate from a leave-one-subject-out cross-validation scheme was in the vicinity of the average error rate across all subjects was selected as the test subject.

3. Results

3.1. Group-level GLM activation maps of the four tasks

Fig. 3 shows the group inference of the GLM activation maps across all subjects for each of the four tasks. Note that the highly task-specific brain regions—such as the contralateral precentral gyri and ipsilateral cerebellum areas (for the LH and RH tasks); the superior temporal cortex, including the planum temporale/Heschl's gyri (for the AD task); and the lingual gyri and calcarine cortices (for the VS task)—showed activations at an FWE-corrected thresholds of $p < 10^{-2}$ for the LH, RH, and VS tasks and an uncorrected threshold of $p < 10^{-4}$ for the AD task. Table 1 summarizes the detailed information.

3.2. Classification results of DNN from leave-out-subject-out cross-validation

Fig. 4 shows the average error rates across all 12 validations for each of the combinatorial scenarios of the target percentages of non-zero weights and for each of the one-layer, two-layer, and three-layer DNNs. The softmax classifier of the fine-tuned DNNs produced minimum average error rates (± standard deviation; %) of 9.4 (±4.6), 7.4 (±4.1), and 6.9 (±3.8) with the one-layer, two-layer, and three-layer DNNs, respectively. The corresponding target percentages of non-zero weights were 0.2 for the one-layer DNN, 0.2 and 0.2 for the two-layer DNN, and 0.2, 0.2, and 0.2 for the three-layer DNN. The error rates across the four target percentages of non-zero weights were significantly different only in the first hidden layer of the three-layer DNN (one-way ANOVA; *d.f.* = 15, Bonferroni-corrected $p < 10^{-5}$). In the first hidden layer, ad hoc paired *t*-tests revealed that the error rates from the target percentage of non-zero weights of 0.2 were significantly lower than the error rates from the remaining target percentages of non-zero weights (i.e., 0.4, 0.6, and 0.8). The linear SVM classifier produced minimum average error rates of 7.9 (±5.0), 6.9 (±4.0), and 6.9 (±3.4) with the one-layer, two-layer, and three-layer DNNs, respectively, and the corresponding target percentages of the non-zero weights were 0.2 across all hidden layers. Further investigation revealed that the misclassified fMRI volumes were mainly from the AD task (22.2 ± 11.7%), in contrast to the LH (3.1 ± 6.3%), RH (1.4 ± 2.7%), and VS (0.8 ± 2.1%) tasks using the three-layer DNN. In the one-layer and two-layer DNNs, the error rates were slightly lower when using the linear SVM classifier than when using the softmax classifier. Fig. 4b illustrates the *t*-SNE plots of the representation of the hidden layer output from the top hidden layer of the one-layer, two-layer, and three-layer DNNs, in which the exclusively task-specific feature representations were obtained in the top hidden layer of the three-layer DNN compared to the one-layer DNN. The weight-feature vector representations at the output layer showed an increased overlap with the group inference of the GLM activation maps as the number of hidden layers of the DNN increased (Fig. 4c). This finding is in line with the superior classification performance of the three-layer DNN compared to the one-layer and two-layer DNNs. Fig. 5 shows the classification performance from the sparser (i.e., target percentage of non-zero weights = 0.1 or 0.05)

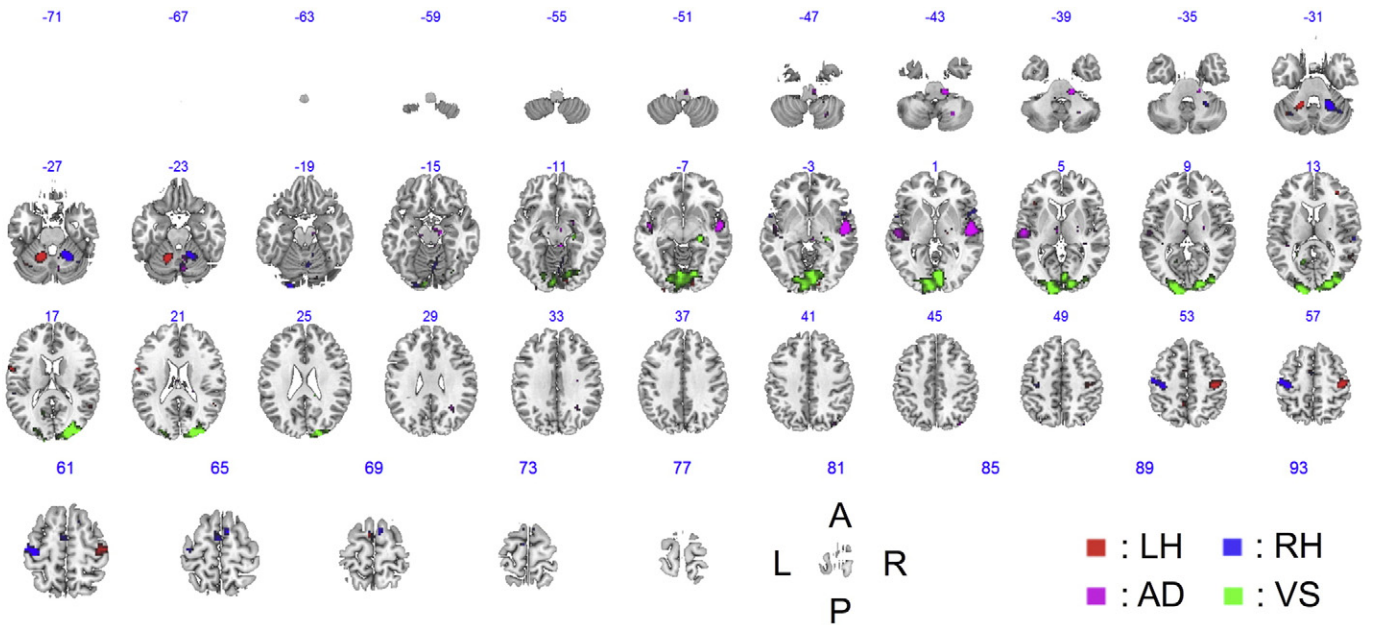


Fig. 3. Group inference of the GLM activation maps for each of the four tasks across all 12 subjects (uncorrected $p < 10^{-3}$). The number above an axial image denotes the z-coordinate (mm) in the MNI space. L, left; R, right; A, anterior; P, posterior; MNI, Montreal Neurological Institute.

weight parameters between the input layer and the first hidden layer. The minimum error rates were significantly reduced (one-way ANOVA; *d.f.* = 15, Bonferroni-corrected $p < 10^{-3}$) when the degree of weight sparsity was strengthened.

3.3. Interpretation of learned weight-feature representation

Fig. 6 shows the overlap ratio between (1) the weight-feature vector associated with each hidden node in each hidden layer and (2) each of the 116 AAL regions using the three-layer DNN with 0.2–0.2–0.2 weight-sparsity levels across the hidden layers. In the first hidden layer, each of the weight-feature vectors showed a moderate level of overlap with each of the 116 AAL regions. However, in the top hidden layer, the weight-feature vectors were highly task specific, with each weight-feature vector showing a substantial overlap with either (1) the left precentral/postcentral regions; (2) the right precentral/postcentral regions; (3) the bilateral superior/middle temporal regions; or (4) the bilateral calcarine, lingual, and superior occipital regions. The four weight-feature vectors in the output layer were extremely task specific (Fig. 6b), and the overlap ratios with the highly task-related AAL regions were markedly increased (bottom of Fig. 6a). Fig. 6c shows the overlap ratios between (1) the group inference of the GLM activation maps from training subjects and (2) the weight-feature representation at the output layer for several options to include a subset of weight values to calculate the weight-feature representation using Eqs. (1) and (2). The overlap ratios increased as more elements of weights were used to calculate the weight-feature representation, and the overlap ratios were saturated at approximately 50 elements of weights that were used to calculate the feature representation. Overall, the overlap ratios were greater in the VS task than in the LH, RH, and AD tasks, which is consistent with the corresponding classification performance.

3.4. Interpretation of hidden layer output

Fig. 7a shows the 2-D plots of the average pattern of the hidden layer output or the input layer for each task (i.e., average of patterns from 30 volumes for each task and for each subject) obtained from the three-layer DNN with 0.2–0.2–0.2 weight-sparsity levels across hidden layers. The average BOLD intensities of in-brain voxels in the input layer appeared to be similar across tasks and across subjects. The average pattern representations of the first hidden layer output were noisy, and it seemed that there was no clear distinction in these patterns across subjects within a task. However, the average pattern representations of the third hidden layer output clearly demonstrated the similarity structures across the subjects within a task and the

Table 1
Brain regions that presented significant activation patterns for each task (FWE-corrected $p < 10^{-2}$ for the LH, RH, and VS tasks; uncorrected $p < 10^{-4}$ for the AD task).

Task	Automated anatomical labeling (AAL)/Brodmann's area (BA)	Size	Foci (x/y/z) in mm	Peak *t*-score
LH	Cerebellum (L), Culmen, Dentate	48	−15/−51/−27	23.21
	PrCG (R), PoCG (R)/BA3, 4, 6	48	45/−18/57	15.25
RH	Cerebellum (R), Vermis, Dentate	73	24/−48/−30	25.48
	PrCG (L), PoCG (L)/BA3, 4, 6	74	−36/−21/60	22.56
	SMA (L/R), SFG, MFG, Cerebellum (L/R)/BA 6	11	9/3/66	14.63
	Cerebellum (L/R), Lingual (L), IOG, FG/BA 18	11	−21/−93/−18	14.30
	Cerebellum (R)	13	6/−63/−21	13.03
	SMA (L)	13	−3/−6/63	12.99
AD	STG, MTG/BA 21, 22	11	54/−12/−3	6.93
VS	Cerebellum (L/R), Lingual (L/R), Cuneus(L/R), Calcarine (LR) MOG(L/R)/BA 18	456	24/−90/18	25.15

Size denotes a number of voxels; L, left; R, right; LH, left-hand clenching; RH, right-hand clenching; AD, auditory attention; VS, visual stimulus; PrCG, precentral gyrus; PoCG, postcentral gyrus; SMA, supplementary motor area; SFG, superior frontal gyrus; MFG, medial frontal gyrus; IOG, inferior occipital gyrus; FG, fusiform gyrus; STG, superior temporal gyrus; MTG, middle temporal gyrus; MOG, middle occipital gyrus.
Brain regions of the clusters were identified using the xjView (www.alivelearn.net/xjview8).

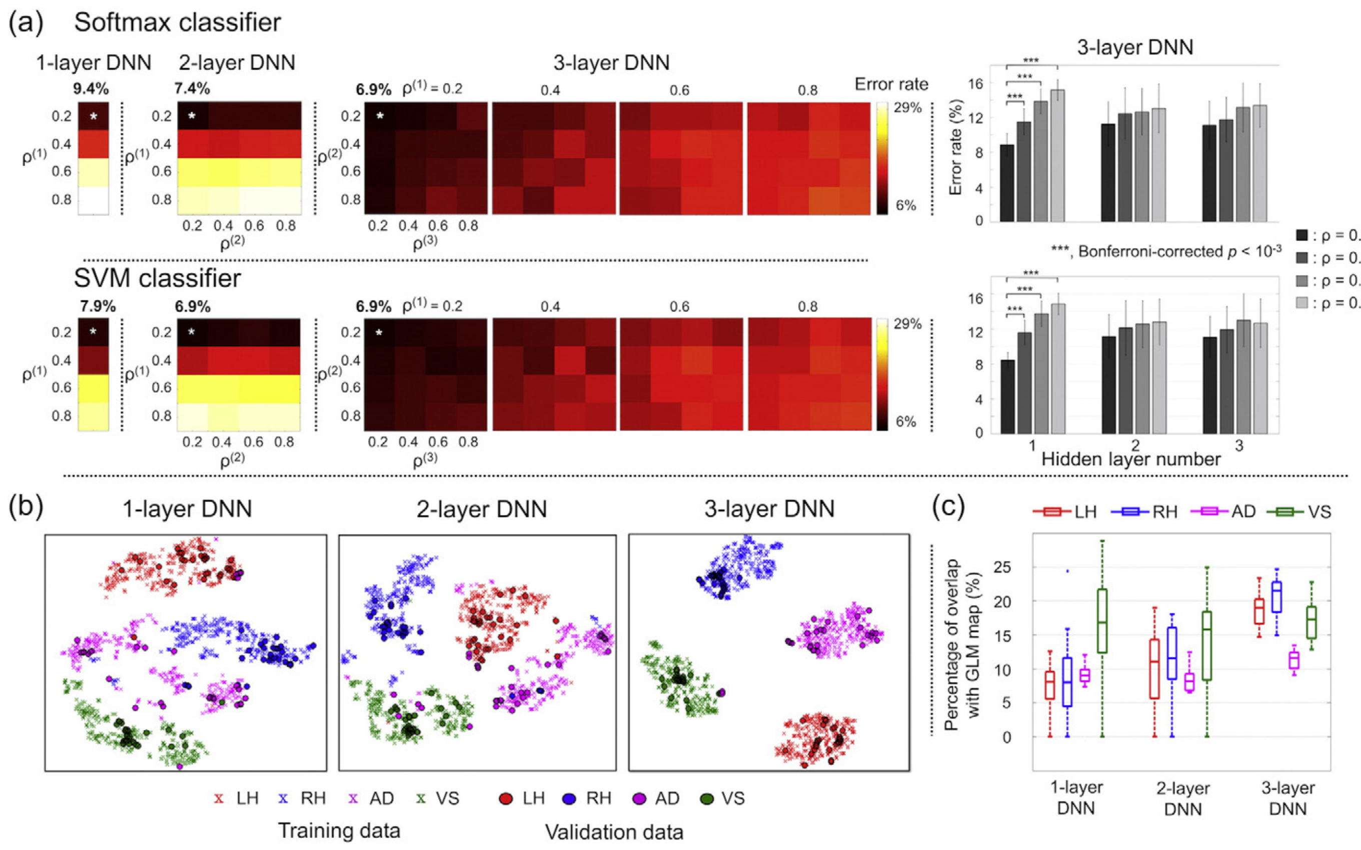


Fig. 4. (a) Classification performance of the 1-, 2-, and 3-layer DNNs from the softmax (top) or SVM (bottom) classifiers across all combinatorial scenarios of four target percentages of non-zero weights (i.e., 0.2, 0.4, 0.6, and 0.8) across the hidden layers to systematically control the weight-sparsity level in each of the hidden layers. $\rho^{(l)}$ denotes the target percentage of non-zero values of weight parameters between the (l-1)th and lth hidden layers (the input layer is equivalent to the 0th hidden layer). Error rates across the four target percentages of non-zero weights are shown as boxplots for each of the three hidden layers of the three-layer DNN. The statistical significance was evaluated using one-way ANOVA followed by ad hoc paired *t*-tests. (b) The *t*-SNE representations of the top hidden layer output (100-dimensional space) of the three DNNs on a 2D plane. All 1320 training data samples (cross) and 120 validation data samples (circle) are presented. (c) Box-whisker plots of the percentage overlap between (1) the voxels corresponding to the top 5 percentile values of the z-scored weights in the output layer for each task and (2) the voxels corresponding to the top 5 percentile values of the group inference of the GLM activation maps of the task across 11 training subjects. ANOVA, analysis of variance; *t*-SNE, *t*-distributed stochastic neighbor embedding; LH, left-hand clenching; RH, right-hand clenching; AD, auditory attention; VS, visual stimulus; GLM, general linear model.

dissimilarity structure across the four tasks. These similarity and dissimilarity structures of the input layer and each of the three hidden layers were clearly shown in the RSA matrices, as shown in Fig. 7b. Fig. 7c shows that the number of correctly determined tasks among subjects increased from the lower to the top hidden layers. The performance of the AD task was inferior to that of the remaining three tasks.

Fig. 8a shows the average pattern representations from one validation subject, which is similar to the representations of the 11 training subjects in Fig. 7a. Fig. 8b shows that the distinctness of the pattern representations is increased from lower to higher layers. At the 3rd hidden layer of the three-layer DNN, the hidden layer representations of each of the four tasks were perfectly distinguished despite the top 5 percentile values of the hidden node outputs of each task being excluded (Fig. 8b right).

3.5. Classification from DBN without DNN fine-tuning and DNN with/without DBN pretraining

Fig. 9 shows the error rates obtained by varying the inputs to the softmax/SVM classifiers. First, the hidden layer output from the DBN pretraining (white bars) shows the highest error rates across all conditions and these error rates were greater than the error rates from the BOLD intensities across the whole brain as input to the classifiers (dotted lines; baseline performance). Second, the error rates, particularly of the two-/three-layer DNNs, decreased when DBN pretraining was used to initialize the weights of the DNNs. Error rates were further reduced by using the ReLU function rather than the tanh function. Table 2 summarizes the performance improvement obtained

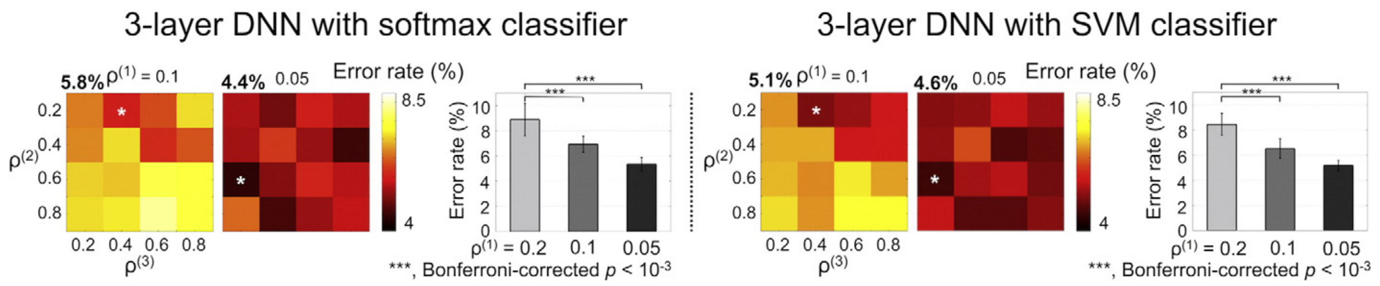


Fig. 5. Error rates of the three-layer DNN when the percentage of non-zero values of weights between the input and hidden layers (i.e., $\rho^{(1)}$) was 0.1 or 0.05. $\rho^{(2)}$ and $\rho^{(3)}$ values were selected to be 0.2, 0.4, 0.6, or 0.8. Error rates (mean $\pm$ standard deviation) when $\rho^{(1)} = 0.2$, 0.1, and 0.05 are shown as bar plots.

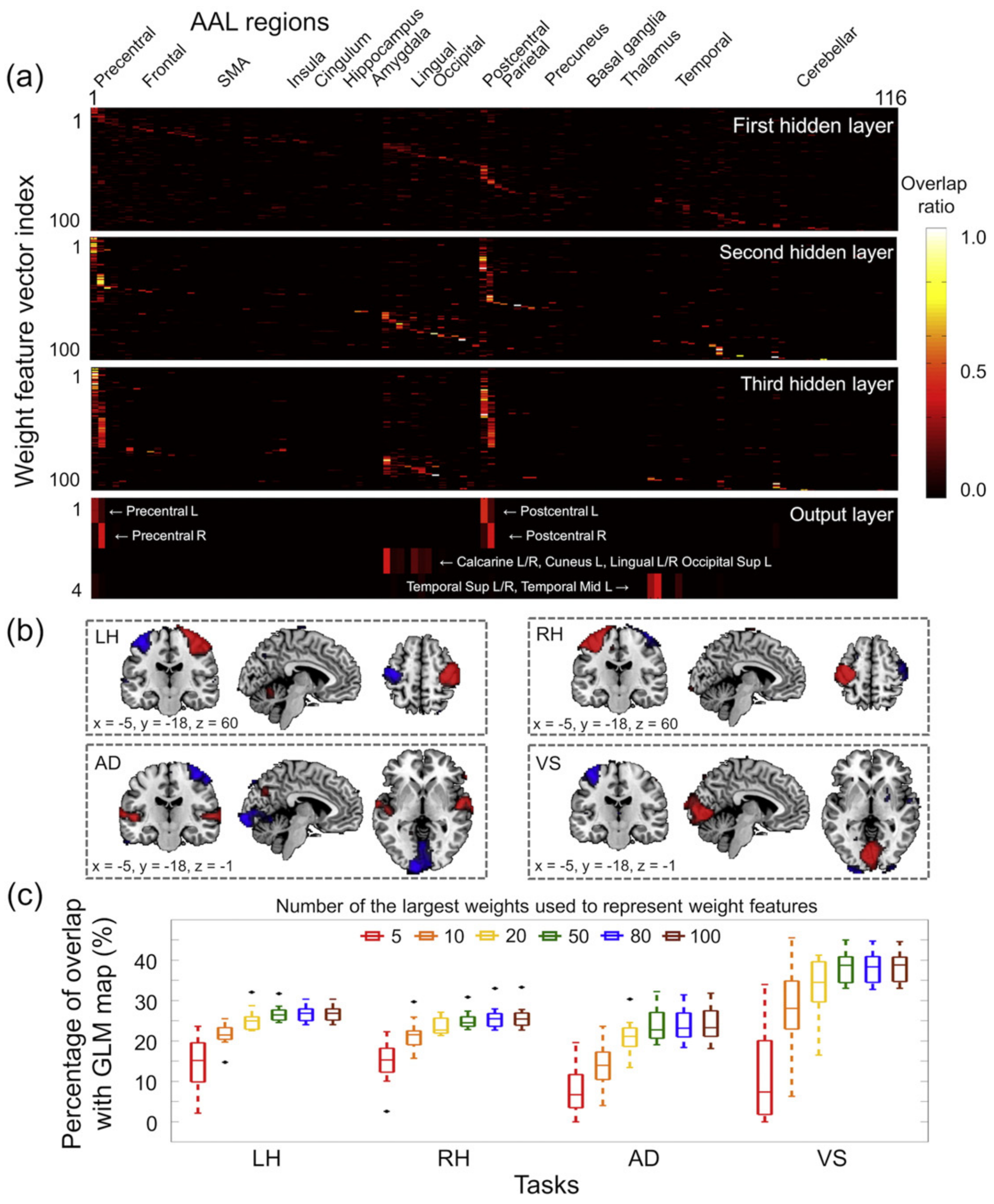


Fig. 6. Interpretation of weight parameters learned from the three-layer DNN with the percentage of non-zero weights of 0.2, 0.2, and 0.2 in the first, second, and third layers, respectively. (a) Overlap ratios between each of the weight feature vectors in each layer and each of the 116 AAL regions (see the labels of the 116 AAL regions at http://neuro.imm.dtu.dk/wiki/Automated_Anatomical_Labeling). (b) Four weight-feature vectors in the output layer were pseudo z-scored to have a zero-mean and unit-variance for visualization (absolute z-score >3.29 or $p < 0.001$; positive values in red and negative values in blue). (c) Box-whisker plots illustrate the overlap ratios between the weight-feature vector representations in the output layer and the group inference of the GLM activation maps for each task (uncorrected $p < 0.001$). Weight-feature vector representations in the output layer were calculated for varying numbers of large values from the weight vector representation in each of the hidden layers using Eqs. (1) and (2). AAL, automated anatomical labeling; Sup, superior; Mid, middle.

by using the hidden representations of the DNNs with the ReLU activation function rather than the BOLD intensities as input.

3.6. Classification using the DBN-pretrained DNN from a nested cross-validation framework

Using the nested cross-validation scheme, the classification errors of the test subject were 7.5% using the softmax classifier and 5.8% using the SVM classifier. The corresponding optimal percentages of non-zero weights were 0.2, 0.2, and 0.4 across the first, second, and third hidden layers. Note that the error rates of this subject from a leave-one-subject-out cross-validation scheme were 6.7% for both the softmax and SVM classifiers and that the corresponding percentages of non-zero weights were 0.2, 0.2, and 0.2 across the first, second, and third hidden layers.

4. Discussion

4.1. Summary of the study

In this study, we presented the utility of the deep neural network for the classification of a single fMRI volume. The restricted Boltzmann machine was used as a building block to initialize the weight parameters of each layer of the DNN, and the DNN was fine-tuned for the classification of fMRI volumes obtained from the four overt sensorimotor task

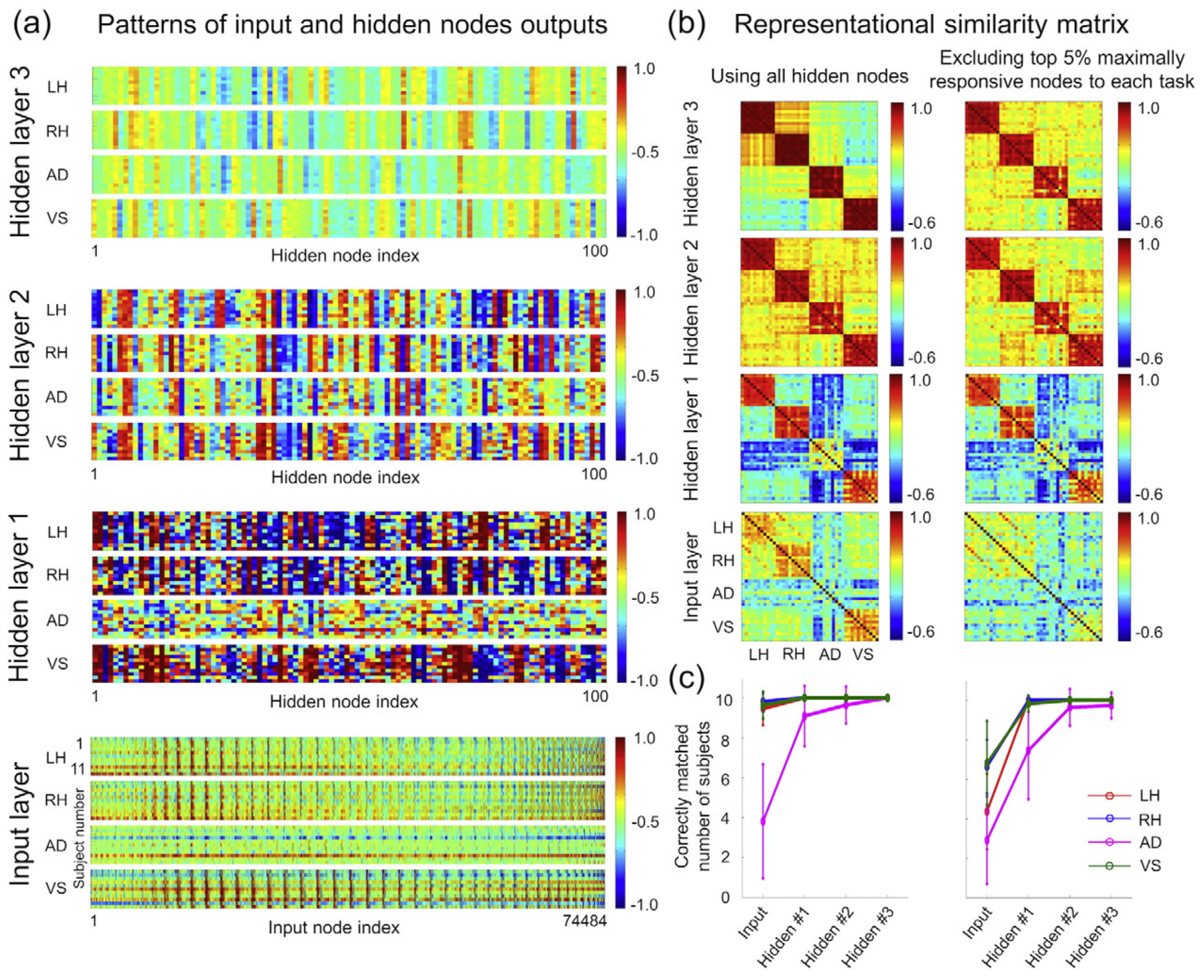


Fig. 7. (a) Hidden layer output patterns and the input sample patterns for each task and for each training subject from the three-layer DNN with percentage of non-zero weights of 0.2, 0.2, and 0.2 in the first, second, and third layers, respectively. The hidden layer output and input sample patterns across the 30 volumes for each task and for each subject were averaged. (b) The RSA matrices using these patterns revealed a markedly enhanced similarity structure across subjects within a task and a dissimilarity structure among the four tasks. These RSA matrices were not substantially altered when the top 5% of the hidden or input nodes per task that showed greater values than the remaining nodes were excluded (right column). (c) The correctly matched numbers of subjects in the training data were calculated by comparing the hidden representation pattern from one subject with the patterns from the 10 remaining subjects in the training set and are visualized for each of the four tasks. RSA, representational similarity analysis; Hidden #1, the first hidden layer; Hidden #2, the second hidden layer; Hidden #3, the third hidden layer.

paradigms. To circumvent the paucity of input samples available to train the DNN, an explicit control of the weight-sparsity level of each hidden layer of the DNN was adopted. The adaptive control of L1 regularization strength facilitated obtaining distinct sparsity levels of weights depending on the target percentages of non-zero values of weights (Fig. S2a). The fixed L1 regularization strength was unable to achieve distinct levels of weight sparsity as exemplified in Fig. S2b; an L1 regularization strength of 0.01 and 0.001 reached approximately the same levels of weight sparsity for both the kurtosis and Hoyer's sparseness measure (Hoyer, 2004). Weight sparsity levels insensitive to the learning rate can be achieved using sparseness measures such as kurtosis and/or Hoyer's sparseness as target weight sparsity levels instead of the computationally efficient percentage of non-zero weights.

The classification accuracies were evaluated across all combinatorial scenarios of weight-sparsity levels across all hidden layers via a leave-one-subject-out cross-validation scheme. The lower error rates were obtained from the linear SVM than from the softmax classifier, which was used for DNN fine-tuning and this may be explained by the superior optimization/generalization capability of the SVM classifier based on maximization of the margin between support vectors (a subset of training samples) and the hyperplane when compared to that of the softmax classifier based on cross-entropy minimization using all the samples (Tang, 2013; Zhang et al., 2015a). For example, (a) the lower cross-entropy value obtained by the softmax classifier training led to a higher classification error and (b) the deep network that was initialized using the weights of the trained deep network with the SVM output layer (11.9% error) showed an increased classification error (14%) when the network was fine-tuned using a softmax output layer (Tang, 2013). This limited evidence suggests that the linear SVM may have better optimization and generalization capability than the softmax classifier. The weight-feature representations and hidden layer output explained well the trained DNN and the corresponding classification performance.

4.2. Current methods used for fMRI data analysis

In conventional fMRI data analysis, a model-based general linear model (GLM, (Worsley and Friston, 1995)) and a data-driven independent component analysis (ICA, (Calhoun et al., 2001; McKeown and Sejnowski, 1998)) have gainfully been applied to estimate neuronal activations from fMRI measurements. In the context of fMRI data classification using estimated neuronal activations, multivoxel/multivariate pattern analysis (Davis et al., 2014; Haynes and Rees, 2006; Kriegeskorte et al., 2006) has been widely appreciated as a promising approach due to its representational capability in high-dimensional space when compared to the voxel-wise mass-univariate approach. Our presented method is advantageous compared to the searchlight-

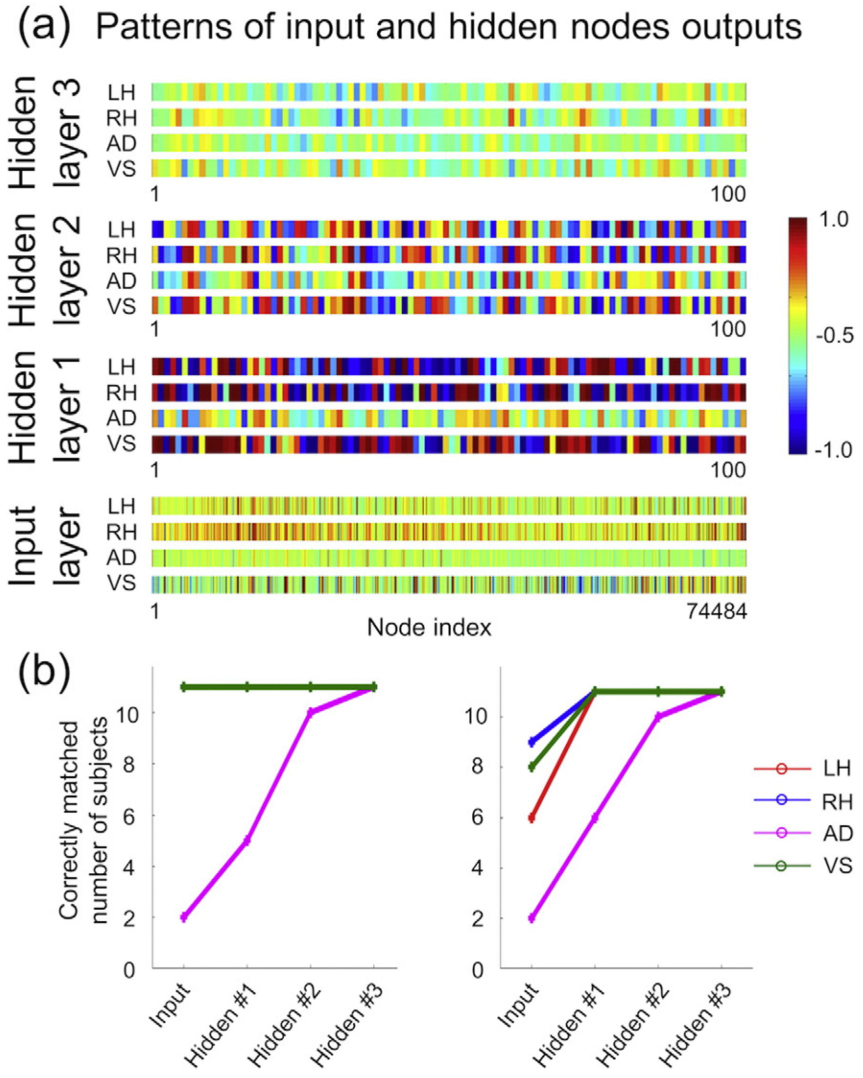


Fig. 8. (a) Hidden layer output patterns and input sample patterns from one validation subject and (b) the correctly matched numbers of subjects in the training sets were calculated by comparing hidden representation patterns from the training subjects with those of the validation subject for each of the four tasks. In the right graph the top 5 percentile values of the hidden node outputs for each task were excluded. Please refer to the legend of Fig. 7 for more detailed information.

based multivoxel pattern analysis approach because it enables (1) the use of raw BOLD intensities from a single fMRI volume rather than the activation patterns from the GLM/ICA that utilize all the fMRI volumes during the task-period, and (2) extraction of the hierarchical features of the neuronal activations of the whole-brain across multiple hidden layers of the DNN. The classification error obtained over 48 runs (i.e.

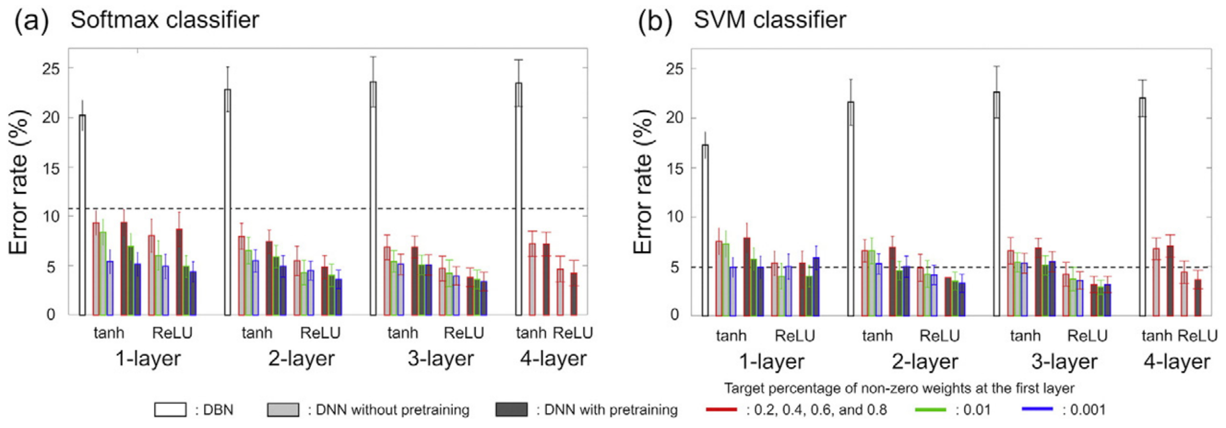


Fig. 9. Classification errors (%) from the (a) softmax and (b) SVM classifiers using the hidden layer output of the DBN without DNN fine-tuning (white box), DNN without DBN pretraining (light gray box), and DNN with DBN pretraining (dark gray box). The performance of the hidden activation function using tanh and ReLU is also compared. The dashed lines indicate the classification performance using fMRI volume (from the input layer) as an input to each classifier. tanh, hyperbolic tangent; ReLU, rectified linear unit; SVM, support vector machine; DBN, deep belief network; DNN, deep neural network.

Table 2
Improvement of classification performance at different network depths using the ReLU activation function when the target percentage of non-zero weights was 0.001 at the first layer (i.e., this condition showed the lowest error rate). The percentage improvement in classification error of the one- to three-layer DNNs over the classification performance achieved with the zero-layer DNN (i.e., with the input layer only) are given in parentheses.

Classifier	0-Layer	Pretraining	1-Layer	2-Layer	3-Layer
Softmax	10.8	Without	4.9 (+54.6%)	4.5 (+58.3%)	4.0 (+63.0%)
		With	4.4 (+59.3%)	3.6 (+66.7%)	3.4 (+68.5%)
SVM	4.9	Without	5.0 (−2.0%)	4.2 (+14.3%)	3.6 (+26.5%)
		With	5.9 (−20.0%)	3.3 (+32.7%)	3.2 (+34.7%)

DNN, deep neural network; ReLU, rectified linear unit; SVM, support vector machine.

12 subjects with four tasks/runs each) with the softmax classifier was 4.2% using the activation patterns of the GLM due to two miss-classified AD runs and 2.1% using those of the ICA due to one miss-classified AD run and with the SVM classifier was 0% using the activation patterns of the GLM and 6.3% using those of the ICA due to three miss-classified AD runs. These results are described in the supplementary material ("*Classification of activation patterns from GLM/ICA*"). All miss-classified runs were AD task runs due to the aberrant time courses of the auditory areas from the experimental paradigm as shown in Fig. S3b. It is worth noting that our presented method is based on an fMRI volume-by-volume classification and thus the variability of fMRI volumes must be predominantly greater than that of the estimated spatial patterns from the GLM/ICA.

4.3. A single fMRI volume classification from the adopted tasks

The error rates from the AD task using the DNNs were greater than the error rates of each of the remaining three tasks. This difference may have arisen because the subjects found it difficult to concentrate on the sinusoidal sounds that distracted them from the background noises, such as MR gradient switching and a cryogenic pump artifact (Kim et al., 2015), during fMRI data acquisition. The fact that the time course of the IC spatial pattern of the auditory area deviated from the experimental paradigm would also support this distraction during the AD task (Fig. S3b). Consistent with this, the statistical significance of the group inference of the GLM activation maps was relatively lower for the AD task than for the other tasks. Notably, the weight parameters of the output node associated with the AD task assigned negative values to the brain regions related to the LH and VS tasks (Fig. 6b), which enhanced the classification performance of the AD task. These negative values of weight parameters were also observed from the weight vectors of the output nodes associated with the LH and RH tasks (Fig. 6b). In our study, overt sensorimotor tasks were used to evaluate the capacity of the DNN toward a single fMRI volume classification. The efficacy of the DNN for the fMRI volume classification acquired from covert/imagery tasks (Boccia et al., 2015; Lee et al., 2009) is important because it can be extended to brain decoding without any overt task paradigm (Boccia et al., 2015; Haynes and Rees, 2006; Lee et al., 2012b).

4.4. Classification performance depending on hidden node function

The hyperbolic tangent (tanh) function rather than a sigmoid function was used as a node function of the hidden layer in our reported findings since the error rates were slightly elevated when the sigmoid node function was used (data not shown). The fact that the tanh function was symmetrical to zero and provided a broader dynamic range than the sigmoid function (Hjelm et al., 2014) may have affected classification performance. The performance was substantially improved using the ReLU activation function instead of the tanh function and this is possibly due to the ReLU function preventing the vanishing gradient problem of the stochastic gradient descent learning algorithm (Maas et al., 2013; Nair and Hinton, 2010).

4.5. Classification performance depending on the depth of the DNN and corresponding sparsity levels

Error rates decreased as the number of hidden layers of the DNN increased, and the representations at the top hidden layer were more distinct and specific to the tasks from the three-layer DNN compared to the one-layer or two-layer DNNs. Using the four candidate target percentages of non-zero weights (i.e., 0.2, 0.4, 0.6, and 0.8), the sparsest set (i.e., 0.2–0.2–0.2) of the three-layer DNN showed the lowest error rate (i.e., 6.9%) across the 12 validation subjects using the linear SVM classifier. Further analysis using the error rates obtained from all combinatorial scenarios of target percentages of non-zero weights revealed that the sparse weight parameters reduced the error rates in only the first hidden layer. The error rates (%) were further reduced when the target percentages of non-zero weights at the first hidden layer were 0.1 (6.9 ± 0.7 and 6.5 ± 0.8 from the softmax and SVM classifiers, respectively) and 0.05 (5.4 ± 0.5 and 5.2 ± 0.4) compared to the 0.2 target percentage of non-zero weights (8.9 ± 1.3 and 8.4 ± 0.9) across all four (0.2, 0.4, 0.6, and 0.8) target percentages of non-zero weights in the second and third hidden layers of the three-layer DNN. The massive number of input nodes (i.e., the number of in-brain voxels) caused the substantial number of elements of weight parameters between the input and the first hidden layers. Thus, the corresponding weight parameters would have achieved successful training using a limited number of samples with the help of a sparsity constraint and consequently the stringent sparsity level at the first layer (i.e., 0.01 or 0.001 target percentage) further reduced the error rates (Figs. 5 and 9). It appears that the error rates were also reduced as the sparsity of weight parameters in the second and third hidden layers increased. However, the error rates across the four target percentages of non-zero weights of weights parameters were not statistically significant (uncorrected $p = 0.3$ for the second hidden layer and uncorrected $p = 0.053$ for the third hidden layer). The error rates (%) from the four-layer DNN by adding one more layer to the three-layer DNN (with 0.2, 0.2, and 0.2 target percentages of non-zero weights from the first to third hidden layers) were 7.4 (± 3.9), 7.2 (± 4.1), 7.2 (± 4.2), and 7.8 (± 3.8) using the softmax classifier and 7.1 (± 3.9), 7.2 (± 4.4), 7.5 (± 4.5), and 7.7 (± 3.6) using the SVM classifier when the target percentages of non-zero weights were 0.2, 0.4, 0.6, and 0.8 in the fourth hidden layer, respectively. The lower performance of the four-layer DNN compared to the three-layer DNN may be due to the limited number of samples in our study, which could have led to the overfitting of the training data.

4.6. Spatial features/filters learned from weight parameters across the hidden layers of the DNN

The greater intensities of the learned weight parameters in the first layer were localized in brain regions across the whole brain (i.e., overlapping with most of the AAL regions), and the weight parameters in the output layer showed greater intensities within brain regions highly specific for each of the four tasks (i.e., overlapping with the motor, auditory, and visual areas). This finding is analogous to our earlier report, in which we observed the hierarchy of the learned weight parameters of the DNN using the whole-brain functional connectivity (FC) of resting-state fMRI data acquired from schizophrenic patients and healthy controls (Kim et al., 2016). In that study, abnormal FC patterns from pairs of brain regions were extracted from the weight parameters in the lower layer, whereas abnormal FC patterns across the whole brain distinguishing the schizophrenic patients from the healthy controls were learned from the weight parameters in the higher hidden layer (Kim et al., 2016). These results may indicate that the weight parameters of the multi-layer DNN would capture low-level features (i.e., localized and distributed brain regions across a brain area defined in the input nodes or FC patterns from pairs of brain regions) in the lower hidden layer and high-level features (i.e., highly task-specific brain regions or FC patterns across a whole brain) in the higher hidden layer. For example, using the BOLD intensities of in-brain voxels of fMRI volumes collected during face and place recognition tasks (Haxby et al., 2001; Kanwisher, 2010) as input samples, the multi-layer DNN may learn the spatial features/filters that are significantly more active in the primary visual areas in the lower hidden layer. The spatial features/filters in the higher hidden layer may primarily be active in the ventral temporal regions, such as the fusiform gyri and parahippocampal place areas, in which the multi-layer DNNs may ultimately be applicable to decode visual stimuli using only brain fMRI activity (Kay et al., 2008; Miyawaki et al., 2008; Naselaris et al., 2009; Schoenmakers et al., 2013; Thirion et al., 2006).

It is worth noting that the estimation of input patterns via the maximization and/or minimization of output/hidden nodes of the trained multi-layer DNNs (Erhan et al., 2009; Simonyan et al., 2013) would be a useful alternative way to characterize the trained DNN (Erhan et al., 2009; Lee et al., 2008). For example, in the activation maximization method, the estimated input pattern x^* is obtained by maximizing the activation of output and/or hidden node: $\mathbf{x}^* = \text{argmax}_{\mathbf{x}} \{h_k^{(l)}(\theta, \mathbf{x})\}$, where x is the input pattern and θ are the DNN parameters. This activation maximization method would be particularly useful for characterizing the learned features of deep convolutional neural networks which consist of several pooling layers (Simonyan et al., 2013) which would not be straightforward using the linear combination of weight matrices. When the DNN is trained to predict the target value of an output node in the regression framework, the trained DNN features that increase/decrease an output value can be obtained from estimated input patterns by maximizing/minimizing an output value.

4.7. Hidden representations in the hidden layers of the DNN

From the three-layer DNN, the error rates obtained from the softmax and SVM classifiers were comparable. This finding indicates that the representations of the third hidden layer were readily distinct and separable when the three-layer DNN was fine-tuned. The RSA using the hidden node outputs also showed that the patterns in the top hidden layer output were remarkably task specific; the similarity was higher within a task, whereas the dissimilarity was higher between pairs of the four tasks. The RSA using the hidden node outputs were comparable (1) when all nodes were used and (2) when highly task-specific nodes with values in the top 5 percentile were excluded (Fig. 7b). This finding indicates that the hidden layer representations can be treated as patterns and that several maximally active hidden nodes to the task would not critically alter the structure of the patterns, as similarly shown by Haxby et al. (2001) for the multivoxel-pattern analysis of visual object recognition using BOLD fMRI activity. It would be worth investigating whether techniques such as the random zeroing of input nodes adopted in a denoising auto-encoder (Vincent et al., 2008) and the dropout of hidden nodes (Hinton et al., 2012b; Srivastava et al., 2014) would facilitate the learning of the robust representation of the hidden layers against confounding artifacts of fMRI data such as head motion and physiological noises such as cardiac and respiratory signals (Allen et al., 2012; Power et al., 2015) and thus would further enhance the classification performance.

4.8. Classification performance with/without the DBN pretraining

Our results showed that the pretraining improved the classification performance of the DNN particularly when the stringent weight sparsity level was used in the first layer (i.e., 0.01 or 0.001 percentage of non-zero weights). The feature representations obtained from the DBN were not distinct to the target task, as evidenced by the classification performance shown in Fig. 9. The DNN with DBN pretraining showed better classification performance than the DNN without the DBN pretraining for both the tanh and ReLU activation functions. Unsupervised pretraining using the DBN has been interpreted as a network regularizer of the DNN and has enhanced classification performance when the network is deep and the number of hidden nodes is large, in which case the pretraining helps to prevent settling on a poor local minimum of the complex network (Erhan et al., 2010). Our finding that the pretraining enhanced both the convergence speed and classification performance of the three-layer DNN for both the training and validation phases (Fig. S1) also indicates that the pretraining helped find a good local minimum of the DNN with the adopted numbers of hidden nodes. Note, however, that purely supervised training without pretraining has also proved successful when using a large quantity of labeled training data with proper initialization and a choice of activation function (Bengio et al., 2013).

4.9. Potential issues of the DNN and application of convolutional NN

The training of the DBN and the DNN is computationally expensive. In our study, a nested cross-validation was performed on one test subject using three-layer DNN trained/optimized from 11 training/validation subjects. The resulting error rates and optimally chosen percentage of non-zero weights were comparable to these from the leave-one-subject-out cross-validation scheme. Using this nested cross-validation scheme, the training/optimization of the three-layer DNN took approximately 8 days since one of the 11 sets of training/validation from the 11 training/validation subjects took approximately 17 h to complete. Graphic processing unit (GPU) installed hardware running a MATLAB environment (Intel i7-4790 3.6 GHz; Nvidia GeForce GTX980; 32 GB RAM; Fedora version 20) was used. The error rate was obtained using the one remaining test subject. The classification performance under the nested cross-validation scheme with the remaining test subjects could have been obtained at a computational cost. The built-in commands to utilize the GPU cores (i.e., 'gpuArray.m' and 'gather.m') during computation in the MATLAB environment were used to minimize the effort to transform the MATLAB codes of the DeepLearnToolbox into the Compute Unified Device Architecture (CUDA) programing (www.nvidia.com/cuda). This CUDA implementation, the Python based DNN toolboxes such as TensorFlow (tensorflow.org) and Theano (deeplearning.net/software/theano), and Torch (torch7.org) would be of benefit to optimization of DNNs, including weight-sparsity control across the hidden layers with fine-grained intervals and much deeper DNN architectures than the three-layer or four-layer DNNs used in this study.

In addition, the convolutional DNN (Krizhevsky et al., 2012) can be gainfully utilized to learn shift-invariant features from the fMRI volumes when voxel locations are distorted due to spatial misalignment during the spatial normalization step of the fMRI volume preprocessing. The shift-invariant features can be learned from convolutional kernel weights followed by a pooling layer rather than fully connected weights between the input and the first hidden layer. A 3D convolutional kernel that was developed to extract the temporal dynamics of a video sequence (Ji et al., 2013) can be modified for application to the fMRI volume and to implement a dependence of voxels across the whole brain. Additionally, the sparsity constraint to the convolutional kernel weight parameters could be deployed to reduce a number of parameters to train, which would reduce hardware resources while maintaining performance (Collins and Kohli, 2014).

5. Conclusion

To our knowledge, our study is the first to show the training of multi-layer DNNs that are pretrained using the DBN with BOLD intensities of in-brain voxels of fMRI volumes and the consequent classification of a single fMRI volume within a cross-validation scheme. The presented method can classify a label of the task using input fMRI volumes obtained from more challenging tasks, such as covert/imagery tasks, than the presented overt sensorimotor tasks. Our adopted explicit control of weight-sparsity levels can be gainfully utilized to learn weight parameters from a limited number of available input samples for training and can be optimized using a cross-validation scheme. Abundant neuroimaging data from data-sharing initiatives (Calhoun, 2015) and from the generation of realistic synthetic neuroimaging data (Castro et al., 2015) would tremendously benefit the application of the DNN to neuroimaging data while minimizing potential overfitting. A development of the DNN algorithms, including a pretraining algorithm such as the DBN, into a software toolbox implemented using the CUDA and/or Python codes with a user-friendly interface would drastically extend the utility of the DNN to fMRI data analysis and applications.

Conflicts of interest

The authors have no conflicts of interest regarding this study, including financial, consultant, institutional, or other relationships.

Sources of support

This work was supported by the National Research Foundation (NRF), MSIP of Korea (2015R1A2A2A03004462), in part by a grant of the Korean Health Technology R&D Project, Ministry of Health & Welfare, Korea (HI12C1847), and in part by National Institutes of Health (NIH) grants R01EB005846 & P20GM103472 and by National Science Foundation (NSF) grant #1539067.

Appendix A. Deep belief network (DBN) pretraining

As a building block of the DBN (Fig. 2; left), the RBM is a single-layer bipartite network model with visible nodes, $\mathbf{v}\in\{0,1\}^{N_v}$, and hidden nodes, $\mathbf{h}\in\{0,1\}^{N_h}$, where N_v and N_h are the numbers of visible and hidden nodes, respectively. For example, in the RBM, to link the input and the first hidden layer of a DBN, the input nodes are the visible nodes and the nodes at the first hidden layer are hidden nodes. The visible and hidden nodes of the RBM have undirected connections, and there is no connection either between the visible nodes or between the hidden nodes (Hinton, 2002). The likelihood that the RBM assigns values in the visible nodes using network parameters is represented using the joint probability of the visible and hidden nodes (i.e., $p(\boldsymbol{v},\boldsymbol{h})$) and its energy function (i.e., $E(\boldsymbol{v},\boldsymbol{h})$):

$$p(\mathbf{v})=\sum\nolimits_{\mathbf{h}}p(\mathbf{v},\mathbf{h})=\sum\nolimits_{\mathbf{h}}\frac{1}{Z}e^{-E(\mathbf{v},\mathbf{h})}=\frac{1}{Z}\sum\nolimits_{\mathbf{h}}e^{-\mathbf{v}^T\mathbf{b}-\mathbf{h}^T\mathbf{c}-\mathbf{v}^T\mathbf{W}\mathbf{h}} \quad (\text{A.1})$$

where $Z=\sum_{\mathbf{v}',\mathbf{h}'}e^{-E(\mathbf{v}',\mathbf{h}';\mathbf{W})}$ is the partition function; $\mathbf{W}\in R^{N_v\times N_h}$ is the weight matrix of network, and $\mathbf{b}\in R^{N_v}$ and $\mathbf{c}\in R^{N_h}$ are visible and hidden node biases, respectively (Hinton, 2002; Hinton et al., 2012a). The gradient descent scheme to the negative log-likelihood of visible nodes with respect to weight parameters can be used to derive a learning rule as follows:

$$\Delta w_{ji}\propto\frac{1}{N}\sum_{n=1}^{N}\frac{\partial\log p(\mathbf{v}^n)}{\partial w_{ji}}=\langle v_ih_j\rangle_{\text{data}}-\langle v_ih_j\rangle_{\text{model}}, \quad (\text{A.2})$$

where w_{ji} is the weight value between the ith visible node v_i and the jth hidden node h_j, N is the number of training samples and $\langle\cdot\rangle$ is an expectation operator under the distribution specified by the subscript (Hinton, 2002; Hinton et al., 2012a). Due to the absence of direct connections either between hidden nodes or between visible nodes in this Bernoulli–Bernoulli RBM, the calculation of unbiased samples of $\langle v_ih_j\rangle_{\text{data}}$ is straightforward:

$$p(h_j=1|\mathbf{v})=f\left(c_j+\sum_i w_{ji}v_i\right),\text{ and } p(v_i=1|\mathbf{h})=f\left(b_i+\sum_j h_jw_{ji}\right), \quad (\text{A.3})$$

where $f(\cdot)$ is a node function. To obtain an unbiased sample of $\langle v_ih_j\rangle_{\text{model}}$, a contrastive divergence (CD) scheme was adopted to estimate the reconstructed hidden and visible values with the probability given by Eq. (A.3). The CD-1 is a single step of full Gibbs sampling to update all visible nodes in parallel after an initial update of the hidden nodes and, consequently, to update all the hidden nodes in parallel (Hinton, 2002; Hinton et al., 2012a). As a result, the learning rules of the parameters when using the CD-1 scheme are given as:

$$\begin{aligned}\Delta w_{ji}&\propto\frac{\partial c\log p(\mathbf{v})}{\partial w_{ji}}=v_i^0h_j^0-v_i^1h_j^1,\\ \Delta b_i&\propto\frac{\partial\log p(\mathbf{v})}{\partial b_i}=v_i^0-v_i^1,\\ \Delta c_j&\propto\frac{\partial\log p(\mathbf{v})}{\partial c_j}=h_j^0-h_j^1,\end{aligned} \quad (\text{A.4})$$

where the superscript of visible or hidden nodes indicates the number of steps of the Gibbs sampling chain (Hinton, 2002; Hjelm et al., 2014). For real-valued input, such as fMRI data, the energy function was modified as (Hinton et al., 2012a):

$$E(\mathbf{v},\mathbf{h})=\sum_i\frac{(v_i-\text{b}_i)^2}{2\sigma_i^2}-\sum_j c_jh_j-\sum_{i,j}\frac{h_jw_{ji}v_i}{\sigma_i}, \quad (\text{A.5})$$

where σ_i is the standard deviation of the Gaussian noise for the visible node v_i. In this Gaussian–Bernoulli RBM, the conditional distributions for CD-1 are as follows (Hinton et al., 2012a; Hjelm et al., 2014):

$$p(h_j|\mathbf{v})=f\left(c_j+\sum\nolimits_i\frac{w_{ji}v_i}{\sigma_i}\right),\text{ and } p(v_i|\mathbf{h})=N\left(b_i+\sigma_i\sum\nolimits_j h_jw_{ji},\ \sigma_i^2\right). \quad (\text{A.6})$$

The update rules for the real-value parameters are as follows:

$$\begin{aligned}\Delta w_{ji}&\propto\frac{1}{\sigma_i^2}v_i^0h_j^0-\frac{1}{\sigma_i^2}v_i^1h_j^1,\\ \Delta b_i&\propto\frac{1}{\sigma_i^2}v_i^0-\frac{1}{\sigma_i^2}v_i^1,\\ \Delta c_j&\propto h_j^0-h_j^1.\end{aligned} \quad (\text{A.7})$$

Appendix B. DNN fine-tuning with weight-sparsity control

A back-propagation algorithm was adopted to fine-tune the DNN by minimizing the mean-squared error (MSE) between the four actual output values (i.e., $\mathbf{y}$) and desired output values (i.e., $\mathbf{d}$) in Fig. 2 (right). A non-zero value 1 was assigned as desired output value only for an output node when the input fMRI volume was associated with the task assigned to the corresponding output node. The dimension of the input sample was substantially greater than the available number of input samples. Thus, the sparsity of the weight parameters via L1-norm regularization was added to the MSE cost function of an ordinary back-propagation algorithm as follows:

$$J=\underset{\mathbf{W}}{\operatorname{argmin}}\left\{(1-\beta)\|\mathbf{d}-\mathbf{y}\|_2^2+\beta\|\mathbf{W}\|_1\right\}, \quad (\text{A.8})$$

where β is an L1-norm regularization parameter of weights, which was auto-tuned based on the percentage of non-zero values of weights, and $\|\cdot\|_2$ and $\|\cdot\|_1$ are the L2- and L1-norms, respectively. Then, the weight update term was derived using a gradient descent scheme to minimize the cost function, J:

$$\Delta\mathbf{W}(t) = \alpha(t)((1-\beta(t))\Delta_{BP}\mathbf{W}(t) + \beta(t)sign(\mathbf{W}(t))), \tag{A.9}$$

where $\Delta_{BP}\mathbf{W}(t)$ is the weight update term of an ordinary back-propagation algorithm and is defined as the first-order derivative of the MSE between **d** and **y**; i.e., the first term of Eq. (A.8), t is the epoch number, $\alpha(t)$ is the overall learning rate, $\beta(t)$ is the L1-norm regularization parameter to control weight sparsity to the target percentage of non-zero weights at the tth epoch, and $1-\beta(t)$ is multiplied by $\Delta_{BP}\mathbf{W}(t)$ to control the relative contribution of the MSE and the weight sparsity. A momentum factor can also be applied to accelerate the gradient descent learning (Bishop, 1995; Kim et al., 2016).

To explicitly control the level of weight sparsity, the degree of the L1-norm regularization parameter was adjusted to achieve the target percentage (0–1) of non-zero values of weights:

$$\Delta\beta(t) = \mu sign(pnz(\mathbf{W}(t))-\rho), \tag{A.10}$$

where μ is the learning rate of $\beta(t)$, ρ is the target percentage of non-zero values of weights, and $pnz(\cdot)$ is a function that accounts for the percentage of non-zero values whose absolute values are greater than threshold ε. $\beta(t)$ increases when the percentage of non-zero values of current weights $\mathbf{W}(t)$ is above the target percentage level; thus, L1-norm regularization is strengthened, leading to sparser weights (Kim et al., 2016). In contrast to our earlier study (Kim et al., 2016), the sign function was applied to Eq. (A.10) to facilitate the L1-norm regularization of weights to a target percentage of non-zero weights even when the difference between the target percentage and the percentage of non-zero values of current weights was negligible.

Appendix C. Rectified linear unit activation function

The rectified linear unit (ReLU) activation function is defined as:

$$h_k^{(l)} = \max\left(\mathbf{w}_k^{(l),T}\mathbf{h}^{(l\text{-}1)}, 0\right), \tag{A.11}$$

where $h_k^{(l)}$ is the kth hidden node output at the lth hidden layer, $\mathbf{w}_k^{(l)}$ is the weight vector of the kth hidden node at the lth hidden layer from the (l-1)th hidden layer, and $\mathbf{h}^{(l\text{-}1)}$ is the node output at the (l-1)th hidden layer. The advantage of the ReLU function over the tanh and sigmoid activation functions is that it can minimize the potential vanishing gradient problem of the stochastic gradient descent learning algorithm (Maas et al., 2013).

To apply the ReLU function to the RBM model, the noisy rectified linear unit (NReLU) was introduced as follows (Dahl et al., 2013; Nair and Hinton, 2010):

$$h_k^{(l)} = \max\left(\mathbf{w}_k^{(l),T}\mathbf{h}^{(l\text{-}1)} + \varepsilon_k^{(l)}, 0\right), \tag{A.12}$$

where $\varepsilon_k^{(l)}$ is drawn from a normal distribution, $N(0, (1 + e^{-\mathbf{w}_k^{(l),T}\mathbf{h}(l-1)})^{-1})$. During the DNN fine-tuning phase, the ordinary ReLU in Eq. (A.11) was used (Dahl et al., 2013).

Appendix D. Supplementary data

Supplementary data to this article can be found online at http://dx.doi.org/10.1016/j.neuroimage.2016.04.003.

References

Allen, E.A., Damaraju, E., Plis, S.M., Erhardt, E.B., Eichele, T., Calhoun, V.D., 2012. Tracking whole-brain connectivity dynamics in the resting state. Cereb. Cortex bhs352.

Amin, F., Plis, S., Damaraju, E., Hjelm, D., Cho, K., Calhoun, V.D., 2015. A deep-learning approach to translate between brain structure and brain function. Pattern Recognition in NeuroImaging (PRNI), Palo Alto, CA.

Bengio, Y., Courville, A., Vincent, P., 2013. Representation learning: a review and new perspectives. Pattern Anal. Mach. Intell. IEEE Trans. 35, 1798–1828.

Bengio, Y., Lamblin, P., Popovici, D., Larochelle, H., 2007. Greedy layer-wise training of deep networks. Adv. Neural Inf. Proces. Syst. 19, 153.

Bishop, C.M., 1995. Neural Networks for Pattern Recognition. Oxford university press.

Boccia, M., Piccardi, L., Palermo, L., Nemmi, F., Sulpizio, V., Galati, G., Guariglia, C., 2015. A penny for your thoughts! Patterns of fMRI activity reveal the content and the spatial topography of visual mental images. Hum. Brain Mapp. 36, 945–958.

Calhoun, V., Adali, T., Pearlson, G., Pekar, J., 2001. A method for making group inferences from functional MRI data using independent component analysis. Hum. Brain Mapp. 14, 140–151.

Calhoun, V.D., 2015. A spectrum of sharing: maximization of information content for brain imaging data. GigaScience 4, 2.

Castro, E., Hjelm, D., Plis, S., Dinh, L., Turner, J., Calhoun, V.D., 2015. Independent component estimation of simulated structural magnetic resonance imaging data using deep learning. IEEE Machine Learning for Signal Processing Workshop, Boston, MA.

Collins, M.D., Kohli, P., 2014. Memory Bounded Deep Convolutional Networks (arXiv preprint arXiv:1412.1442).

Cortes, C., Vapnik, V., 1995. Support-vector networks. Mach. Learn. 20, 273–297.

Dahl, G.E., Sainath, T.N., Hinton, G.E., 2013. Improving deep neural networks for LVCSR using rectified linear units and dropout. Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on. IEEE, pp. 8609–8613.

Davis, T., LaRocque, K.F., Mumford, J.A., Norman, K.A., Wagner, A.D., Poldrack, R.A., 2014. What do differences between multi-voxel and univariate analysis mean? How subject-, voxel-, and trial-level variance impact fMRI analysis. NeuroImage 97, 271–283.

Erhan, D., Bengio, Y., Courville, A., Manzagol, P.-A., Vincent, P., Bengio, S., 2010. Why does unsupervised pre-training help deep learning? J. Mach. Learn. Res. 11, 625–660.

Erhan, D., Bengio, Y., Courville, A., Vincent, P., 2009. Visualizing higher-layer features of a deep network. Dept. IRO, Université de Montréal, Tech. Rep 4323.

Güçlü, U., van Gerven, M.A., 2015. Deep neural networks reveal a gradient in the complexity of neural representations across the ventral stream. J. Neurosci. 35, 10005–10014.

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, 2425–2430.

Haynes, J.-D., Rees, G., 2006. Decoding mental states from brain activity in humans. Nat. Rev. Neurosci. 7, 523–534.

Hinton, G., 2002. Training products of experts by minimizing contrastive divergence. Neural Comput. 14, 1771–1800.

Hinton, G., Deng, L., Yu, D., Dahl, G.E., Mohamed, A.-R., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T.N., 2012a. Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups. Signal Process. Mag. IEEE 29, 82–97.

Hinton, G., Osindero, S., Teh, Y.-W., 2006. A fast learning algorithm for deep belief nets. Neural Comput. 18, 1527–1554.

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

Hinton, G.E., Srivastava, N., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.R., 2012b. Improving Neural Networks by Preventing Co-Adaptation of Feature Detectors (arXiv preprint arXiv:1207.0580).

Hjelm, R.D., Calhoun, V.D., Salakhutdinov, R., Allen, E.A., Adali, T., Plis, S.M., 2014. Restricted Boltzmann machines for neuroimaging: an application in identifying intrinsic networks. NeuroImage 96, 245–260.

Hoyer, P.O., 2004. Non-negative matrix factorization with sparseness constraints. J. Mach. Learn. Res. 5, 1457–1469.

Ji, S., Xu, W., Yang, M., Yu, K., 2013. 3D convolutional neural networks for human action recognition. Pattern Anal. Mach. Intell. IEEE Trans. 35, 221–231.

Kanwisher, N., 2010. Functional specificity in the human brain: a window into the functional architecture of the mind. Proc. Natl. Acad. Sci. 107, 11163–11170.

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

Khaligh-Razavi, S.-M., Kriegeskorte, N., 2014. Deep Supervised, but not Unsupervised, Models may Explain IT Cortical Representation.

Kim, H.-C., Yoo, S.-S., Lee, J.-H., 2015. Recursive approach of EEG-segment-based principal component analysis substantially reduces cryogenic pump artifacts in simultaneous EEG–fMRI data. NeuroImage 104, 437–451.

Kim, J., Lee, J.-H., 2013. Integration of structural and functional magnetic resonance imaging improves mild cognitive impairment detection. Magn. Reson. Imaging 31, 718–732.

Kim, J., Calhoun, V.D., Shim, E., Lee, J.-H., 2016. Deep neural network with weight sparsity control and pre-training extracts hierarchical features and enhances classification performance: evidence from whole-brain resting-state functional connectivity patterns of schizophrenia. NeuroImage 124, 127–146.

Kim, J., Kim, Y.-H., Lee, J.-H., 2013. Hippocampus–precuneus functional connectivity as an early sign of Alzheimer's disease: a preliminary study using structural and functional magnetic resonance imaging data. Brain Res. 1495, 18–29.

Kim, Y.H., Kim, J., Lee, J.H., 2012. Iterative approach of dual regression with a sparse prior enhances the performance of independent component analysis for group functional magnetic resonance imaging (fMRI) data. NeuroImage 63, 1864–1889.

Kriegeskorte, N., Goebel, R., Bandettini, P., 2006. Information-based functional brain mapping. Proc. Natl. Acad. Sci. U. S. A. 103, 3863–3868.

Kriegeskorte, N., Mur, M., Bandettini, P., 2008. Representational similarity analysis—connecting the branches of systems neuroscience. Front. Syst. Neurosci. 2.

Krizhevsky, A., Sutskever, I., Hinton, G.E., 2012. Imagenet classification with deep convolutional neural networks. Adv. Neural Inf. Proces. Syst. 1097–1105.

Lee, H., Ekanadham, C., Ng, A.Y., 2008. Sparse deep belief net model for visual area V2. Adv. Neural Inf. Proces. Syst. 873–880.

Lee, J.H., Kim, J., Yoo, S.S., 2012a. Real-time fMRI-based neurofeedback reinforces causality of attention networks. Neurosci. Res. 72, 347–354.

Lee, J.-H., Marzelli, M., Jolesz, F.A., Yoo, S.-S., 2009. Automated classification of fMRI data employing trial-based imagery tasks. Med. Image Anal. 13, 392–404.

Lee, S.-H., Kravitz, D.J., Baker, C.I., 2012b. Disentangling visual imagery and perception of real-world objects. NeuroImage 59, 4064–4073.

Maas, A.L., Hannun, A.Y., Ng, A.Y., 2013. Rectifier Nonlinearities Improve Neural Network Acoustic Models. Proc, ICML.

McKeown, M.J., Sejnowski, T.J., 1998. Independent component analysis of fMRI data: examining the assumptions. Hum. Brain Mapp. 6, 368–372.

Miyawaki, Y., Uchida, H., Yamashita, O., Sato, M.-A., Morito, Y., Tanabe, H.C., Sadato, N., Kamitani, Y., 2008. Visual image reconstruction from human brain activity using a combination of multiscale local image decoders. Neuron 60, 915–929.

Nair, V., Hinton, G.E., 2010. Rectified linear units improve restricted boltzmann machines. Proceedings of the 27th International Conference on Machine Learning (ICML-10), pp. 807–814.

Naselaris, T., Prenger, R.J., Kay, K.N., Oliver, M., Gallant, J.L., 2009. Bayesian reconstruction of natural images from human brain activity. Neuron 63, 902–915.

Oldfield, R.C., 1971. The assessment and analysis of handedness: the Edinburgh inventory. Neuropsychologia 9, 97–113.

Plis, S.M., Hjelm, D.R., Salakhutdinov, R., Allen, E.A., Bockholt, H.J., Long, J.D., Johnson, H.J., Paulsen, J.S., Turner, J.A., Calhoun, V.D., 2014. Deep learning for neuroimaging: a validation study. Front. Neurosci. 8.

Power, J.D., Schlaggar, B.L., Petersen, S.E., 2015. Recent progress and outstanding issues in motion correction in resting state fMRI. NeuroImage 105, 536–551.

Schmah, T., Hinton, G.E., Small, S.L., Strother, S., Zemel, R.S., 2008. Generative versus discriminative training of RBMs for classification of fMRI images. Adv. Neural Inf. Proces. Syst. 1409–1416.

Schoenmakers, S., Barth, M., Heskes, T., van Gerven, M., 2013. Linear reconstruction of perceived images from human brain activity. NeuroImage 83, 951–961.

Simonyan, K., Vedaldi, A., Zisserman, A., 2013. Deep inside convolutional networks: visualising image classification models and saliency maps (arXiv preprint arXiv: 1312.6034).

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R., 2014. Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. 15, 1929–1958.

Suk, H.-I., Lee, S.-W., Shen, D., Initiative, A.S.D.N., 2014. Hierarchical feature representation and multimodal fusion with deep learning for AD/MCI diagnosis. NeuroImage 101, 569–582.

Tang, Y., 2013. Deep learning using linear support vector machines (arXiv preprint arXiv: 1306.0239).

Thirion, B., Duchesnay, E., Hubbard, E., Dubois, J., Poline, J.-B., Lebihan, D., Dehaene, S., 2006. Inverse retinotopy: inferring the visual content of images from brain activation patterns. NeuroImage 33, 1104–1116.

Tzourio-Mazoyer, N., Landeau, B., Papathanassiou, D., Crivello, F., Etard, O., Delcroix, N., Mazoyer, B., Joliot, M., 2002. Automated anatomical labeling of activations in SPM using a macroscopic anatomical parcellation of the MNI MRI single-subject brain. NeuroImage 15, 273–289.

Van der Maaten, L., Hinton, G., 2008. Visualizing data using t-SNE. J. Mach. Learn. Res. 9, 85.

Van Gerven, M.A., De Lange, F.P., Heskes, T., 2010. Neural decoding with hierarchical generative models. Neural Comput. 22, 3127–3142.

Vincent, P., Larochelle, H., Bengio, Y., Manzagol, P.-A., 2008. Extracting and composing robust features with denoising autoencoders. Proceedings of the 25th International Conference on Machine Learning. ACM, pp. 1096–1103.

Worsley, K.J., Friston, K.J., 1995. Analysis of fMRI time-series revisited—again. NeuroImage 2, 173–181.

Zhang, S.-X., Liu, C., Yao, K., Gong, Y., 2015a. Deep neural support vector machines for speech recognition. Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on. IEEE, pp. 4275–4279.

Zhang, W., Li, R., Deng, H., Wang, L., Lin, W., Ji, S., Shen, D., 2015b. Deep convolutional neural networks for multi-modality isointense infant brain image segmentation. NeuroImage 108, 214–224.