Deep Representational Similarity Learning for analyzing neural signatures in task-based fMRI dataset

Muhammad Yousefnezhad · **Jeffrey Sawalha** · **Alessandro Selvitella** · **Daoqiang Zhang**

Abstract Similarity analysis is one of the crucial steps in most fMRI studies. Representational Similarity Analysis (RSA) can measure similarities of neural signatures generated by different cognitive states. This paper develops Deep Representational Similarity Learning (DRSL), a deep extension of RSA that is appropriate for analyzing similarities between various cognitive tasks in fMRI datasets with a large number of subjects, and high-dimensionality — such as whole-brain images. Unlike the previous methods, DRSL is not limited by a linear transformation or a restricted fixed nonlinear kernel function — such as Gaussian kernel. DRSL utilizes a multi-layer neural network for mapping neural responses to linear space, where this network can implement a customized nonlinear transformation for each subject separately. Furthermore, utilizing a gradient-based optimization in DRSL can significantly reduce runtime of analysis on large datasets because it uses a batch of samples in each iteration rather than all neural responses to find an optimal solution. Empirical studies on multi-subject fMRI datasets with various tasks — including visual stimuli, decision making, flavor, and working memory — confirm that the proposed method achieves superior performance to other state-of-the-art RSA algorithms.

Keywords fMRI Analysis · Representational Similarity Analysis · Deep Representational Similarity Learning

M. Yousefnezhad and D. Zhang are with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China.
M. Yousefnezhad and J. Sawalha are with the Deparment of Computing Science and the Deparment of Psychiatry, University of Alberta, Edmonton T6G 2R3, AB, Canada.
A. Selvitella is with the Department of Mathematical Sciences, Purdue University Fort Wayne, 2101 E Coliseum Blvd, Fort Wayne, 46805 United States.
E-mail: {myousefnezhad, jsawalha}@ualberta.ca; aselvite@pfw.edu; dqzhang@nuaa.edu.cn

1 Introduction

One of the most significant challenges in both neuroscience and machine learning is comprehending how the human brain works [17, 21]. Indeed, we have long been fascinated by the process of conscious thought, which translates to an interest in better understanding human brains. We anticipate this will offer methods for diagnosing and treating mental health disorders, which could have tremendous benefits [13]. The neural activities can be analyzed at different levels, but a crucial step is knowing what the similarities (or differences) between distinctive cognitive tasks are [13, 33, 35]. It is like a spotlight that allows us to facilitate other areas of brain studies. Since task-based functional Magnetic Resonance Imaging (fMRI) can provide better spatial resolution in comparison with other modalities of measurement, most of the previous studies employed fMRI datasets [13, 17, 21, 33, 35].

As one of the fundamental approaches in fMRI analysis, Representational Similarity Analysis (RSA) [7, 21, 22] evaluates the similarities (or distances) between distinctive cognitive tasks [32]. In practice, classical RSA [7, 21] can be mathematically formulated as a multi-set (group) regression problem — i.e., a linear model for mapping between the matrix of neural activities and the design matrix [4, 21]. Classical RSA employs basic linear approaches — e.g., Ordinary Least Squares (OLS) [21] or General Linear Model (GLM) [7].

Recent studies have shown that these methods could not provide accurate performances on large real-world datasets — such as datasets with a broad Region of Interest (ROI) or whole-brain fMRI data [4, 17, 18, 34]. There are several problems in these classical approaches [8, 32]. First, classical RSA methods may not provide accurate similarity analysis — especially when the covariance of the neural activities has a low rate of Signal-to-Noise Ratio (SNR)

[4, 8, 9]. In other words, classical RSA calculates the inverse of the covariance matrix *without a regularization term* [7, 8, 21, 22], whereas most fMRI datasets may not be full rank — i.e., the number of voxels are greater than the time points [13, 17, 34]. The next problem is that these techniques need all of the data points in memory at the same time in order to apply the similarity analysis. As a result, they are not computationally efficient for large datasets [11, 25].

The existing efforts on developing RSA techniques demonstrate some promising results. However, there are still several long-standing challenges. Some of modern approaches focused on the regularization issue [3, 11, 25, 28] — such as Octagonal Shrinkage and Clustering Algorithm for Regression (OSCAR) [3] or Ordered Weighted $\ell1$ (OWL) [11, 25]. However, regularized approaches always consider that the relation between the feature space and the design matrix is linear. Alternatively, other algorithms have utilized Bayesian techniques [4, 8, 9, 12] — such as Bayesian RSA (BRSA) [4] or Pattern Component Model (PCM) [9]. Although Bayesian approaches may significantly handle the SNR issue and even improve some nonlinear datasets, it is limited to a restricted transformation function — i.e., Gaussian distribution of the hyper-parameter [28, 34]. Moreover, the time complexity of Bayesian methods is not suitable for high-dimensional datasets because the hyper-parameter must be calculated for each dimension separately [28].

This paper proposes Deep Representational Similarity Learning (DRSL) as a new deep extension of the RSA method to solve the previously mentioned challenges. The main contributions of this paper are summarized as follows:

- The regular, nonlinear approaches such as the Gaussian kernel suffer from a limited latent space which is caused by a fixed, non-parametric kernel function. Unlike these nonlinear methods, DRSL utilizes a deep network as the parametric kernel function — including multiple stacked layers of nonlinear transformations. This parametric kernel function allows for a more flexible latent space. Thus, DRSL is not limited to restricted transformation functions.
- DRSL also employs a new regularization term that can improve the quality of the results. The proposed regularization function steeply penalizes neural responses with a low rate of SNR.
- Most nonlinear similarity approaches (with non-parametric kernels) can significantly increase the computational time of the analysis. Alternatively, DRSL uses a gradient-based optimization method that can generate the neural signatures in each iteration by using a batch of instances instead of utilizing all samples. Consequently, it can provide a time-efficient analysis for evaluating high-dimensional fMRI images — such as whole-brain datasets.

- We also demonstrate DRSL's effectiveness by showing that it can distinguish what task the subject is pursuing, even for a novel subject (not in training-set), without a prior calibration phase.
- In some sense, the transformation function employed in DRSL is similar to the one utilized in the Deep Canonical Correlation Analysis (DCCA) for multi-view representational learning [1, 2] or Deep Hyperalignment (DHA) for aligning neural activities of multi-subject data across subjects [34]. However, these supervised classification approaches cannot be used for similarity analysis.

The rest of this paper is organized as follows: Section 2 briefly introduces some related works. DRSL is proposed in Section 3. Section 4 reports the empirical studies; Section 5 presents the conclusions and some lines of future work.

2 Background

In general, a similarity analysis approach maps the neural activities generated based on different cognitive tasks to a set of neural signatures, where each signature belongs to a unique category of stimuli [13]. Encoding Analysis (EA) [8, 23] is utilized for analyzing the various type of neural signatures such as movement stimuli [27], low-level visual stimuli [16], and semantic maps [15, 23]. The classical encoding approach assumes the neural signatures belong to an individual subject as the basis of a feature space. Here, the neural responses can be characterized as a linear combination of this space [23, 24].

Representational Similarity Analysis (RSA) is another approach for analyzing similarity across different categories of stimuli [8, 13, 21]. Classical RSA can be formulated as a multi-set regression problem, where a set of neural signatures are generated by using Ordinary Least Squares (OLS) [21] or General Linear Model (GLM) [4, 7, 21]. Recent studies showed that the classical similarity techniques — such as EA and RSA — could increase the danger of overfitting mostly on high-dimensional datasets [4, 8].

Some of the modern RSA methods used regularization functions to control overfitting and improve the performance. Among the prevalent approaches are Ridge Regression [14], Least Absolute Shrinkage and Selection Operator (LASSO) [28, 36], Elastic Net [37], and OSCAR [3]. Oswal et al. proposed Representational Similarity Learning (RSL) as a modified version of Ordered Weighted ℓ_1 (OWL) [11] that can produce better performance for analyzing similarity of fMRI datasets in comparison with other sparse techniques — such as LASSO, Elastic Net, or OSCAR [25]. Sheng et al. also proposed Gradient-Based RSA (GRSA) for applying RSA with LASSO regularization. While GRSA can significantly reduce the time complexity of analysis on

Table 1 Variables or Functions

Variable or Function	Description
$\mathbb{R}$	The set of real numbers.
i, j, ℓ, k, m	The indices.
T	The number of time points.
V_{org}	The number of voxels in the original space.
V	The number of mapped features in the linear embedded space.
P	The number of distinctive categories of stimuli.
S	The number of Subjects.
C	The number of deep network layers.
$\mathbf{X}^{(\ell)} = \left\{ x_{ij}^{(\ell)} \right\} \in \mathbb{R}^{T \times V_{org}}$	The original neural activities for ℓ-th subject.
$\mathbf{D}^{(\ell)} = \left\{ d_{ik}^{(\ell)} \right\} \in \mathbb{R}^{T \times P}$	The design matrix for ℓ-th subject.
$\mathbf{B}^{(\ell)} = \left\{ \beta_{kj}^{(\ell)} \right\} \in \mathbb{R}^{P \times V_{org}}$	The matrix of estimated regressors for ℓ-th subject.
$\widetilde{\mathbf{B}}$	The mean of the extracted neural signatures across subjects.
$\mathscr{D}(\widetilde{\beta}_{i.}, \widetilde{\beta}_{j.})$	The distance (or similarity) between i-th and j-th categories of stimuli by using the metric $\mathscr{D}()$.
$\theta^{(\ell)} = \left\{ \mathbf{W}_m^{(\ell)}, \mathbf{a}_m^{(\ell)}, 2 \leq m \leq C \right\}$	All network parameters for ℓ-th subject.
$\mathbf{h}_m^{(\ell)} = \mathrm{g}(\dots)$	The result of activation function for ℓ-th subject and m-th layer.
$U^{(m)}, 2 \leq m < C$	The number of units in m-th intermediate layer.
$\Psi^{(k,\ell)}$	A set of randomly selected time points related to k-th iteration and ℓ-th subject.
$\alpha \geq 1$	The scaling factor.
$\nabla_{\mathbf{B}}(J_R)$	The gradient of RSA objective function(J_R) with respect to $\mathbf{B}$.
$\nabla_{\theta}\left(f(\mathbf{x}_{i.}; \theta)\right)$	The gradient of network parameters.
η	The learning rate.
M	The maximum iteration.
N	The batch size.
$\mu_1, \mu_2, \varepsilon$	The Adam optimization parameters.
δ_j, γ_j	The moment vectors.
$r(\mathbf{B}^{(\ell)})$	The regularization function.
$\mathrm{sign}(\mathbf{B}^{(\ell)}) \in \left\{ -1, +1 \right\}^{P \times V}$	The sign function.
$\mathrm{g}()$	Nonlinear activation function, i.e., sigmoid, tanh, ReLU.

the datasets, its performance is not stable. Indeed, the output of GRSA analysis is related to the number of algorithm's iterations and the learning rate [28].

Similarity analysis for fMRI signals with a low rate of Signal-to-Noise Ratio (SNR) is a challenging problem. Cai et al. recently showed that classical RSA could not distinguish the real fMRI signals and randomly generated noises [4]. For solving this issue, they proposed Bayesian RSA (BRSA) considers the covariance matrix as a hyperparameter generative model [4]. However, the time complexity of BRSA is not suitable for large datasets [28]. Pattern Component Modeling (PCM) is another Bayesian approach for similarity analysis [9]. PCM assumes that the neural activities have a multivariate Gaussian distribution, and the model can be evaluated by using the marginal likelihood of the neural responses based on the assumed probability distribution [8, 9].

There are two paradigms for evaluating the neural signatures generated by a similarity method, viz, using a distance metric, and converting a similarity analysis to a linear classification problem [32]. Most of the earlier studies in similarity analysis compared the neural signatures by using a distance metric — such as Euclidean distance, correlation, the rank of generated neural signatures [7, 17, 21, 32]. How-

ever, using these metrics for comparing models generated by different similarity approaches sometimes need an extra normalization. For instance, the generated neural signatures must be scaled to a specific range that could be compared by using the Euclidean metric [8, 32]. As another example, the cosine metric can be only utilized to compare two sets of neural signatures (generated by two distinctive similarity techniques) when these signatures have the same origin in the vector space [32]. Alternatively, we can use the correlation metric that naturally provides normalized comparisons for all similarity approaches [32].

Another paradigm for evaluating the performance of a similarity method is converting the neural signatures into a classification model [8, 17, 32]. Like a regular classification problem, we first use the subject(s) out cross-validation for partitioning data to training and testing sets [13]. After that, the neural signatures are generated by applying a similarity approach to the training set [32]. Each pair of these signatures will be used for producing a linear hyperplane as a binary classification model that can distinguish the corresponding categories of stimuli [32]. Further, these binary models can be employed in an Error-Correcting Output Codes (ECOC) framework for predicting multi-class problems [33]. Finally, we can test the generality of the neu-
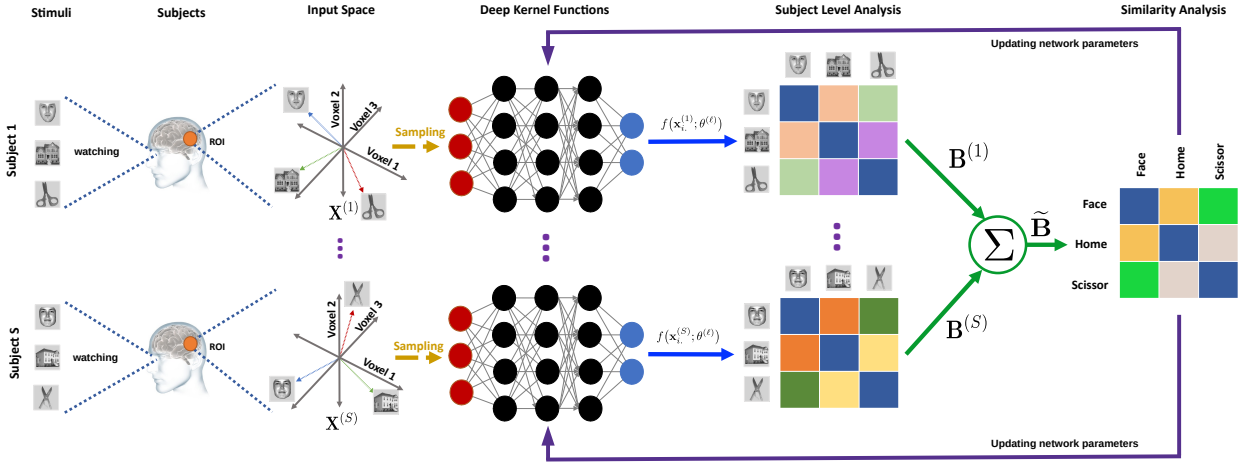
Fig. 1 The proposed Deep Representational Similarity Learning (DRSL)

ral signatures by evaluating the performance of the classification models on the testing set, where better signatures can generate more general hyperplane and consequently improve the classification accuracy [32, 33].

3 Deep Representational Similarity Learning (DRSL)

Table 1 illustrates all of the variables and functions that are employed in this paper. We will gradually introduce them in this section. During collecting a task-based fMRI dataset, we generate two elements, viz, functional brain images, and design matrices. fMRI functional image collected from ℓ-th subject can be denoted by $\mathbf{X}^{(\ell)} = \left\{ x_{ij}^{(\ell)} \right\} \in \mathbb{R}^{T \times V_{org}}, 1 \leq i \leq T, 1 \leq j \leq V_{org}$, where T is the number of time points, and V_{org} denotes the number of voxels in the original space. This paper assumes that the neural activities of each subject are column-wise standardized, i.e., $\mathbf{X}^{(\ell)} \sim \mathcal{N}(0,1)$. We can also consider this condition as a preprocessing step if the original data is not standardized. In addition, a design matrix can also be defined for each subject that describes the events related to each cognitive task [35]. In this paper, $\mathbf{D}^{(\ell)} = \left\{ d_{ik}^{(\ell)} \right\} \in \mathbb{R}^{T \times P}, d_{ik} \in \mathbb{R}, 1 \leq i \leq T, 1 \leq k \leq P$ defines the design matrix belonging to ℓ-th subject. Here, P denotes the number of distinctive categories of stimuli. Further, $\mathbf{d}_{.k}^{(\ell)} \in \mathbb{R}^{T}, 1 \leq k \leq P$ as the k-th column of the design matrix is the convolution of the onsets of k-th category ($\mathbf{o}_{.k}^{(\ell)} \in \mathbb{R}^{T}$) with Ξ as the Hemodynamic Response Function (HRF) signal ($\mathbf{d}_{.k}^{(\ell)} = \mathbf{o}_{.k}^{(\ell)} * \Xi$) [4, 35].

As Figure 1 depicted, DRSL maps neural activities to an information-rich space by using a transformation function, i.e. $\mathbf{x} \in \mathbb{R}^{V_{org}} \rightarrow f(\mathbf{x}) \in \mathbb{R}^{V}$, where $V_{org} \geq V$ denotes the number of mapped features in the linear embedded space. Although f can be any restricted fixed transformation function (such as Gaussian or Polynomial), this paper uses mul-

tiple stacked layers of nonlinear transformation function in $\mathbf{x}$ as follows:

$$f(\mathbf{x}; \theta) = \mathbf{W}_C \mathbf{h}_{C-1} + \mathbf{a}_C, \tag{1}$$

where the intermediate layers can be defined as follows:

$$\mathbf{h}_m = g(\mathbf{W}_m \mathbf{h}_{m-1} + \mathbf{a}_m), \tag{2}$$

where $2 \leq m < C$, $\mathbf{h}_1 = \mathbf{x}$, $C \geq 3$ is the number of deep network layers, $\theta = \left\{ \mathbf{W}_m, \mathbf{a}_m \text{ for } 2 \leq m \leq C \right\}$ denotes all network parameters, and g is a nonlinear function applied componentwise, i.e., Rectified Linear Unit (ReLU), sigmoid, or $tanh$ [4]. By considering $U^{(m)}, 2 \leq m < C$ as the number of units in m-th intermediate layer, the parameters of the output layer are denoted by $\mathbf{W}_C \in \mathbb{R}^{V \times U^{(C-1)}}$, $\mathbf{a}_C \in \mathbb{R}^{V}$. For $2 \leq m < C$, the parameters of intermediate layer are defined by $\mathbf{W}_m \in \mathbb{R}^{U^{(m)} \times U^{(m-1)}}$, $\mathbf{a}_m \in \mathbb{R}^{U^{(m)}}$ except $\mathbf{W}_2 \in \mathbb{R}^{U^{(2)} \times V_{org}}$.

By using the proposed transformation function, the DRSL's objective function can be denoted within subject-level as follows:

$$J_R^{(k,\ell)} = \sum_{i \in \Psi^{(k,\ell)}} \left\| f(\mathbf{x}_{i.}^{(\ell)}; \theta^{(\ell)}) - \mathbf{d}_{i.}^{(\ell)} \mathbf{B}^{(\ell)} \right\|_2^2 + r\left(\mathbf{B}^{(\ell)} \right), \tag{3}$$

where optimal results can be achieved by solving the following optimization problem:

$$\min_{\mathbf{B}^{(\ell)}, \theta^{(\ell)}} J_R^{(k,\ell)}. \tag{4}$$

Here, $\mathbf{B}^{(\ell)} = \left\{ \beta_{kj}^{(\ell)} \right\} \in \mathbb{R}^{P \times V}, \beta_{kj}^{(\ell)} \in \mathbb{R}, 1 \leq k \leq P, 1 \leq j \leq V$ denotes the matrix of estimated regressors — including the extracted neural signatures for ℓ-th subject. In addition, $\Psi^{(k,\ell)}$ is a set of randomly selected time points related to k-th iteration and ℓ-th subject. The size of this set ($|\Psi^{(k,\ell)}| < T$) is equal to the batch size. Further, $\mathbf{x}_{i.}^{(\ell)} \in \mathbb{R}^{1 \times V_{org}}$ denotes all voxels belonging to the ℓ-th subject and i-th time point (row)

Algorithm 1 Deep Representational Similarity Learning

Input: Dataset $\mathbf{X} = [\mathbf{X}^{(\ell)}, \ell = 1 \ldots S]$, Design $\mathbf{D} = [\mathbf{D}^{(\ell)}, \ell = 1 \ldots S]$, Maximum Iteration M_1 (default 10), and Algorithm 2 parameters.
Output: Neural signatures $\widetilde{\mathbf{B}}$
Method:
01. Initializing $\widetilde{\mathbf{B}} \sim \mathcal{N}(0,1)$.
03. **For** $j = 1:M_1$
04.   **For** $\ell = 1:S$
04.   Generating new $\mathbf{B}^{(\ell)}$ by using Algorithm 2 and $\widetilde{\mathbf{B}}$.
12.   **End For**.
12.   Updating $\widetilde{\mathbf{B}}$ by using (6).
13. **End For**.

---

of the neural activities $\mathbf{X}^{(\ell)}$, and $\mathbf{d}_{i.}^{(\ell)} \in \mathbb{R}^{1 \times P}$ is the i-th row of the design matrix $\mathbf{D}^{(\ell)}$ that belongs to ℓ-th subject.

Unlike other applications of deep transformation function [1, 2, 34], we consider a fixed structure of deep network layers for all subjects, including $f(\mathbf{x}_{i.}^{(\ell)}, \theta^{(\ell)})$ rather than $f_\ell(\mathbf{x}_{i.}^{(\ell)}, \theta^{(\ell)})$. This notation can improve the stability of generated results and also decrease the number of parameters that must be estimated for each RSA problem. Thus, we just need to estimate additional network parameters ($\theta^{(\ell)}, 1 \le \ell \le S$) for each problem in comparison with the classical RSA approaches.

DRSL regularization term is defined as follows:

$$r\left(\mathbf{B}\right) = \sum_{j=1}^{V} \sum_{k=1}^{P} \alpha \left| \beta_{kj} \right| + 10\alpha \left(\beta_{kj} \right)^2, \tag{5}$$

where α is the scaling factor. In this paper, we have used $\alpha = 10$ for all normalized data — i.e., $\mathbf{X}^{(\ell)} \sim \mathcal{N}(0,1)$.

The final neural signatures is also calculated as follows:

$$\widetilde{\mathbf{B}} = \frac{1}{S} \sum_{\ell=1}^{S} \mathbf{B}^{(\ell)}, \tag{6}$$

where S denotes number of subjects, and $\widetilde{\mathbf{B}}$ calculates the mean of subject-level neural signatures across all subjects. Indeed, $\widetilde{\beta}_{i.} \in \mathbb{R}^V$, $1 \le i \le P$ is the i-th row of the matrix $\widetilde{\mathbf{B}}$ denotes the extracted neural signatures that belong to i-th category of stimuli across all subjects. Similarity between two categories of stimuli can be calculated by comparing different rows of the matrix $\widetilde{\mathbf{B}}$. Here, $\mathscr{D}\left(\widetilde{\beta}_{i.}, \widetilde{\beta}_{j.}\right)$ denotes the distance (or similarity) between i-th and j-th categories of stimuli by using the metric $\mathscr{D}()$. We can use any common used metric in machine learning for comparing neural activities — such as Euclidean, cosine, correlation, covariance, etc. [8, 17, 18, 32]. For instance, $\mathbb{E}\left[\left(\widetilde{\mathbf{B}} - \mathbb{E}\left[\widetilde{\mathbf{B}}\right]\right)\left(\widetilde{\mathbf{B}} - \mathbb{E}\left[\widetilde{\mathbf{B}}\right]\right)^\top\right] \in \mathbb{R}^{P \times P}$ denotes the covariance comparison of the neural activities across different categories of stimuli [4].

3.1 Optimization

In this section, we implement a version of a Block Coordinate Descent (BCD) optimization approach for the DRSL problem. This method seeks an optimal solution (3) by using two different steps, which iteratively work in unison. By considering fixed network parameters ($\theta^{(\ell)}$), RSA objective function ($J_R^{(k,\ell)}$) is firstly optimized as follows by using the k-th mini-batch ($\Psi^{(k,\ell)}$) of neural activities:

$$\min_{\mathbf{B}^{(\ell)}} J_R^{(k,\ell)} =$$
$$\min_{\mathbf{B}^{(\ell)}} \left(\sum_{i \in \Psi^{(k,\ell)}} \left\| f(\mathbf{x}_{i.}^{(\ell)}; \theta^{(\ell)}) - \mathbf{d}_{i.}^{(\ell)} \mathbf{B}^{(\ell)} \right\|_2^2 + r\left(\mathbf{B}^{(\ell)}\right) \right). \tag{7}$$

Lemma 3.1 $J_R^{(k,\ell)}$ can be minimized with respect to $\mathbf{B}^{(\ell)}$ as follows:

$$\nabla_{\mathbf{B}^{(\ell)}} \left(J_R^{(k,\ell)} \right) = \alpha \operatorname{sign}\left(\mathbf{B}^{(\ell)}\right) + 20\alpha \mathbf{B}^{(\ell)}$$
$$- 2 \sum_{i \in \Psi^{(k,\ell)}} (\mathbf{d}_{i.}^{(\ell)})^\top \left(f(\mathbf{x}_{i.}^{(\ell)}; \theta^{(\ell)}) - \mathbf{d}_{i.}^{(\ell)} \mathbf{B}^{(\ell)} \right) \tag{8}$$

where $\operatorname{sign}\left(\mathbf{B}^{(\ell)}\right) \in \left\{ -1, +1 \right\}^{P \times V}$ is the sign function.
Proof. Please see the proof section.

As the next step, we denote the kernel objective function, where back-propagation algorithm [26] is applied to this function for updating the network parameters:

$$\min_{\theta^{(\ell)}} \sum_{i \in \Psi^{(k,\ell)}} \left\| f(\mathbf{x}_{i.}^{(\ell)}; \theta^{(\ell)}) - \mathbf{d}_{i.}^{(\ell)} \mathbf{B}^{(\ell)} \right\|_2^2. \tag{9}$$

In this step, we consider the regressors matrix ($\mathbf{B}^{(\ell)}$) is fixed. The network parameters ($\theta^{(\ell)}$) can be updated by considering the vector $\mathbf{d}_{i.}^{(\ell)} \mathbf{B}^{(\ell)}$ as the ground truth of $f(\mathbf{x}_{i.}^{(\ell)}; \theta^{(\ell)})$ and then using back-propagation algorithm to update the parameters. Since $f(\mathbf{x}_{i.}^{(\ell)}; \theta^{(\ell)})$ is a standard multilayer perceptron (MLP), we can update the network parameters by using $\nabla_{\theta^{(\ell)}} \left(f(\mathbf{x}_{i.}^{(\ell)}; \theta^{(\ell)}) \right)$, where the output of the optimized deep neural network has the lowest error in comparison with the vector $\mathbf{d}_{i.}^{(\ell)} \mathbf{B}^{(\ell)}$ (as the ground truth). By reducing this error, (9) will be also minimized. Please refer to [26] for technical information related to MLP and back-propagation algorithm.

Algorithm (1) shows how DRSL generates the neural signatures from fMRI responses. We first initiate the neural signatures with random numbers — including $\widetilde{\mathbf{B}} \sim \mathcal{N}(0,1)$. Then, we use Algorithm (2) for updating these signatures by using the neural responses belonging to each subject. As the first step in Algorithm (2), the network parameters are considered fixed, and then Stochastic Gradient Descent (SGD) [2, 20, 34] updates the regressors matrix $\mathbf{B}^{(\ell)}$ belonging to

---

Algorithm 2 Deep RSL for ℓ-th subject

Input: Data $\mathbf{X}^{(\ell)}$, Design $\mathbf{D}^{(\ell)}$, Number of layers C, Number of units $U^{(m)}$ for $m = 2{:}C$, Learning rate η (default 10^{-3}), Maximum Iteration M_2 (default 100), Batch Size N (default 50), Scaling parameter α (default 10), Adam optimization parameters $\mu_1 = 0.9$, $\mu_2 = 0.999$, $\varepsilon = 10^{-8}$ [19], Neural signatures $\widetilde{\mathbf{B}}$.
Output: Regressors matrix $\mathbf{B}^{(\ell)}$, and Parameters $\theta^{(\ell)}$
Method:
01. Initializing $\theta^{(\ell)} \sim \mathcal{N}(0,1)$.
02. Initializing $\mathbf{B}^{(\ell)} := \widetilde{\mathbf{B}}$.
03. $\delta_0 \leftarrow 0$ (Initializing 1st moment vector)
04. $\gamma_0 \leftarrow 0$ (Initializing 2nd moment vector)
05. **For** $k = 1{:}M_2$
06.   Creating $\Psi^{(k,\ell)}$ by selecting N samples from 1 to T.
07. $\widehat{\phi}_k = \sum_{i \in \Psi^{(k,\ell)}} \nabla_{\mathbf{B}^{(\ell)}} \left(J_R^{(k,\ell)} \right)$.
08.   Updating $\mathbf{B}^{(\ell)} \leftarrow \mathbf{B}^{(\ell)} - \eta \widehat{\phi}_k$.
09. $\phi_k = \sum_{i \in \Psi^{(k,\ell)}} \nabla_{\theta^{(\ell)}} \left(f(\mathbf{x}_{i.}^{(\ell)}; \theta^{(\ell)}) \right)$.
10. $\delta_k \leftarrow \mu_1 \delta_{k-1} + (1 - \mu_1) \phi_k$.
11. $\gamma_k \leftarrow \mu_2 \gamma_{k-1} + (1 - \mu_2) \phi_k^2$.
12. $\widetilde{\delta}_k \leftarrow \delta_k / (1 - \mu_1^k)$.
13. $\widetilde{\gamma}_k \leftarrow \gamma_k / (1 - \mu_2^k)$.
14.   Updating $\theta^{(\ell)} \leftarrow \theta^{(\ell)} - \eta \widetilde{\delta}_k / (\sqrt{\widetilde{\gamma}_k} - \varepsilon)$.
15. **End For**.

ℓ-*th* subject. As the second step, the regressors matrix is assumed fixed, and then the Adam [19] algorithm updates the deep network parameters.

This paper develops DRSL as a flexible deep approach for improving the performance of RSA method in fMRI analysis. For seeking an efficient analysis, DRSL uses a deep network (*multiple stacked layers of nonlinear transformation*) for mapping neural activities of each subject into a linear embedded space ($f : \mathbb{R}^{V_{org}} \rightarrow \mathbb{R}^V$). Unlike the previous nonlinear methods that used a restricted fixed transformation function, mapping functions in DRSL demonstrate flexibility across subjects because they employ multi-layer neural networks, which can implement any nonlinear function [1, 2, 34]. Therefore, DRSL does not suffer from disadvantages of the previous nonlinear approach. Finally, using a gradient-based method to optimize DRSL can provide acceptable time complexity for large datasets because it uses a batch of samples in each iteration rather than all neural responses to find an optimal solution.

4 Experiments

4.1 Datasets

Table 2 shows the 6 datasets that are employed in this paper for running empirical studies. These datasets are shared by Open Neuro[1]. In this paper, there are two groups of datasets — including ROI-based data, and the whole-brain data. We employ a prefix 'R' in the rest of this paper for denoting

---

ROI datasets — i.e., R105, R107, and R232. Further, a prefix 'W' will be used to define the whole-brain datasets — such as W005, W203, and W231. These datasets are listed as follows:

- **R105**: visual object recognition includes eight different categories of visual stimuli — including grayscaled photos of faces, houses, cats, bottles, scissors, shoes, chairs, and scramble patterns. The neural activities in temporal cortex (VT) is selected as the ROI for this dataset [13, 33].

- **R107**: word and object processing contains four categories of stimuli — i.e., words, consonants, objects, and scramble photos. ROI in this dataset is selected based on the original paper [10].

- **R232**: object-coding localizer task includes four tasks — including faces, objects, places, and scramble photos. We have used the original ROI that is presented in the main reference of this dataset [6].

- **W005**: mixed-gambles is the first whole-brain data that contains four decision making tasks. Each cognitive tasks in this dataset has the same change of selection (25%). The scheme of experiment is explained in the original paper [30].

- **W203**: visual imagery and false memory contains four categories of working memory tasks — including hit, omission, false alarm, and correct rejection. During the experiment, two groups of stimuli presented to each subject in random order — i.e., 45 words and 45. In the recall phase, subjects need to remember whether a picture of the item had been presented, or only a word. Please refer to the main reference for more information [29].

- **W231**: integration of sweet taste includes six different categories of flavor task — including 0 cal, 112 cal, 150 cal, rinse, tasteless, and control. The setup of experiment is presented in [31].

All datasets are separately preprocessed by FSL 6.0.1[2] — i.e., slice timing, anatomical alignment, normalization, smoothing. For the whole-brain datasets, we have registered all fMRI images to the *MNI152-T1-4mm* standard space, and then $V_{org} = 19742$ voxels are extracted from the standard whole-brain mask [4]. We have also provided a preprocessed version of these datasets in MATLAB format[3].

4.2 Performance analysis

We compared the performance of DRSL with 6 existing similarity methods. The hyper-parameters for each similarity technique are selected based on a grid-search. Here, we select the hyper-parameters for each method that generate

---

[1] Available at `https://openneuro.org/`

[2] Availabel at `https://fsl.fmrib.ox.ac.uk`

[3] Available at `https://easydata.gitlab.io`

Table 2 The datasets.

ID	Title	Task Type	S	P	T	V_{ROI}	Scanner	TR	TE
R105	Visual object recognition	visual	6	8	121	1452	G3T	2500	30
R107	Word and object processing	visual	49	4	164	722	S3T	2000	28
R232	Object-coding localizer task	visual	10	4	760	9947	S3T	1060	16
W005	Mixed-gambles	decision	16	4	714	–	S3T	2000	30
W203	Visual imagery and false memory	memory	26	4	534	–	S1.5T	2000	35
W231	Integration of sweet taste	flavor	9	6	1119	–	S3T	2000	30

S is the number of subject; P denotes the number of stimulus categories; T is the number of scans in unites of TRs (Time of Repetition); V_{ROI} denotes the number of voxels in ROI; 19742 voxels are extracted from *MNI152-T1-4mm* space [21] for all whole-brain datasets. Scanners include S=Siemens, or G = General Electric in 1.5, or 3 Tesla; TR is Time of Repetition in millisecond; TE denotes Echo Time in millisecond; Please see `https://openneuro.org/` for more information.

a lower error for the corresponding objective function. The number of maximum iterations for all algorithms is considered 1000. For the DRSL method, we have also selected 10 iterations for Algorithm 1 and 100 iterations for Algorithm 2 — i.e., the total iterations are $10 \times 100 = 1000$. As a non-parametric approach, this paper employs classical RSA with GLM optimization as a baseline [7, 21]. We also report the performance of the LASSO algorithm [36], where different values for balance factor $\alpha_{LASSO} = [0, 0.8, 0.9, 1, 1.1]$ are evaluated, and $\alpha_{LASSO} = 0.9$ generates the best results. As another regularized method, the performance of RSL is presented. To generate the results, we evaluate the performance of the RSL approach by using two regularization terms proposed in the original paper [25] — i.e., GrOWL-Lin and GrOWL-Spike. Here, GrOWL-Spike produces a better performance for similarity analysis.

The performance of Bayesian RSA (BRSA) [4, 5] and Pattern Component Modeling (PCM) [8, 9] are reported here. We evaluate the performance of these techniques by using different nuisance regressor methods — including Principal Component Analysis (PCA), Independent Component Analysis (ICA), Factor Analysis, and Sparse PCA. In all cases, PCA generates better performance and runtime. We also use different SNR prior — i.e., Gaussian, Log-Norm, and Uniform. The Gaussian prior generates better results. We also initiate 20 as the minimum number of iterations and consider a full rank of the covariance matrices in these approaches. We also compare the performance with Gradient RSA (GRSA) [28] — by using $\alpha_{GRSA} = [0, 0.8, 0.9, 1]$, the learning rates $[10^{-2}, 10^{-3}, 10^{-4}]$, and the batch sizes $[10, 30, 50, 80, 100]$. The best results are generated by using $\alpha_{GRSA} = 0.9$, the learning rate 10^{-3}, with a batch size of 50.

This paper also reports the performance of the proposed method using linear and deep transformation functions. Linear RSL (LRSL) utilizes the objective function (3) and the optimization Algorithm 1, but the transformation function is considered linear — i.e., $f(\mathbf{x}) = \mathbf{x}$. Here, we illustrate how much DRSL improves the performance of similarity analysis with the proposed deep transformation function. We have generated results in both LRSL and DRSL by using different values of $\alpha = [1, 5, 10, 20, 50, 100]$. In all datasets with normalization $\mathbf{X}^{(\ell)} \sim \mathcal{N}(0, 1)$, $\alpha = 10$ produces a better performance in comparison with other values. Further, the Adam optimization parameters are set optimally based on the original paper [19], including $\mu_1 = 0.9$, $\mu_2 = 0.999$, and $\varepsilon = 10^{-8}$. We also evaluate the performance of LRSL and DRSL by using different learning rates $[10^{-2}, 10^{-3}, 10^{-4}]$ and batch sizes $[10, 30, 50, 80, 100]$. Like GRSA, the best results are produced by using the batch size 50 and learning rate 10^{-3}.

In DRSL, we need to define the structure of deep neural networks — i.e., multiple stacked layers of nonlinear transformations. Since we want to use the deep network as the kernel function in a multi-set regression problem, the network structure must satisfy some criteria — i.e., number of layers, number of units, and the nonlinear activation functions. These criteria can affect both the performance and runtime of the analysis. As the previous studies showed [1, 2, 34], deep networks with more than two layers can act as nonlinear kernel functions for accurately modeling complex real-world datasets. Since the number of samples is limited in most fMRI datasets, we keep the number of layers in the deep network minimal [34]. Thus, we have employed two hidden layers for DRSL — i.e., $C = 4$. Another component of the deep network is the number of units. Here, we present the number of units for the hidden layers and the output layer as follows, $[HL1, HL2, OUT]$. As an example, $[1000, 700, 500]$ represents $U^{(1)} = 1000$ as the number of the first hidden layer, $U^{(2)} = 700$ as the number of the second hidden layer, and $V = 500$ as the output layer. We evaluate the DRSL by using different numbers of units in each layer — including $[2000, 1500, 1000, 700, 500, 200, 100, 50]$. The best results for all datasets except *R*107 are achieved by using $[1000, 700, 500]$ settings. For *R*107, a network with $[700, 500, 200]$ setting is generated the best results (perhaps because the number of voxels in this data is less than 1000). The last component in the network structure is the nonlinear activation function. Since the deep network is used in the form of a multi-set regression problem, the output layer must have a linear activation function [1, 2, 34]. So, we only need to define the activation functions for the hidden layers. We

Table 3 Between-class correlation analysis (max±std)

Datasets	RSA [22]	LASSO [25, 36]	RSL [25]	BRSA [4]	PCM [9]	GRSA [28]	LRSL	DRSL
R105	0.947±0.042	0.751±0.242	0.821±0.120	0.389±0.010	0.401±0.064	0.584±0.142	0.451±0.081	**0.372±0.016**
R107	0.922±0.053	0.715±0.147	0.531±0.123	0.458±0.076	0.357±0.041	0.426±0.099	0.142±0.092	**0.135±0.000**
R232	0.927±0.033	0.900±0.026	0.631±0.193	0.871±0.100	**0.490±0.057**	0.754±0.107	0.555±0.112	0.496±0.093
W005	0.891±0.035	0.823±0.036	0.699±0.076	0.519±0.045	0.591±0.083	0.392±0.069	0.361±0.021	**0.139±0.019**
W203	0.951±0.045	0.851±0.077	0.572±0.155	0.383±0.051	0.405±0.075	0.411±0.094	0.451±0.012	**0.270±0.039**
W231	0.888±0.113	0.761±0.114	0.421±0.273	0.521±0.082	0.487±0.107	0.267±0.243	0.231±0.063	**0.126±0.002**

Table 4 Accuracy of classification analysis (mean±std)

Datasets	RSA [22]	LASSO [25, 36]	RSL [25]	BRSA [4]	PCM [9]	GRSA [28]	LRSL	DRSL
R105	18.69±2.37	47.63±1.04	51.29±1.13	60.39±0.69	63.91±0.92	58.36±1.03	59.73±2.81	**78.13±0.19**
R107	33.57±3.06	38.18±1.27	37.97±2.08	65.24±0.58	69.11±0.73	66.05±1.76	72.31±0.04	**84.26±0.64**
R232	31.78±2.71	43.66±1.80	50.30±1.27	61.06±0.19	59.74±0.88	65.14±1.01	63.27±1.30	**70.31±0.55**
W005	27.10±1.02	31.03±0.71	46.17±1.36	66.58±0.73	68.24±0.93	61.43±0.92	60.00±0.72	**81.37±0.11**
W203	24.40±2.13	35.64±0.81	60.06±2.41	76.90±0.59	72.58±0.33	83.15±0.24	75.68±0.80	**91.40±0.24**
W231	18.63±3.37	29.43±0.96	**68.25±1.07**	50.26±1.00	49.51±0.98	52.76±1.61	51.68±1.42	65.74±1.00

evaluated the performance of DRSL by employing different nonlinear activation functions — including ReLU, sigmoid, *tanh*. In most of the normalized datasets, the sigmoid activation function generated better performance. Based on our empirical studies, the explained structure for the DRSL method was fit to our hardware limitation and provided an efficient trade-off between runtime and performance (please see the result sections).

All algorithms in this paper are implemented by Python 3 and run on a PC with certain specifications[4] by authors for generating the experimental results. The source codes of the proposed method and a GUI-based toolbox is also shared[5].

4.2.1 Correlation analysis

In this section, we use the mentioned methods for generating the neural signatures. Then, we used two performance tests for evaluating the quality of the generated signatures — including correlation and classification analysis. Suppose $\widetilde{\mathbf{B}} = \{\widetilde{b}_{ij}\} \in \mathbb{R}^{P \times V}$ denotes the generated neural signatures by using each of similarity approaches for P different categories of stimuli. Further, $\widetilde{\mathbf{b}}_{i.} \in \mathbb{R}^{V}$ represents the signature belonging to *i-th* category of stimuli. The primary assumption for similarity analysis is that better methods must generate independent neural signatures in distinctive categories [7, 8, 17, 18, 21, 32]. Thus, this paper analyzed the similarities of extracted signatures in different categories of stimuli by correlation metric. For evaluating the performance of each similarity technique, we first analyze between-class correlation — i.e., $\rho(\widetilde{\mathbf{B}}) = \max_{i=1}^{P} \max_{j=i+1}^{P} |\text{corr}(\widetilde{\mathbf{b}}_{i.}, \widetilde{\mathbf{b}}_{j.})|$. Here, a better similarity approach must generate more independent neural signatures and reduce the correlation among absolute

values simultaneously. As mentioned before, correlation is a normalized metric that can be used for evaluating signatures generated by using different similarity approaches [32].

Table 3 shows the maximum of the between-class correlation. The neural signatures generated by classical RSA are highly correlated. LASSO and RSL provided better signatures by applying regularization approaches. Here, RSL outperformed LASSO by using a customized regularization term that can handle sparsity and noise in comparison with the regular ℓ_1 norm. Next, BRSA and PCM generated neural signatures by estimating a Gaussian distribution on the fMRI datasets. While PCM could outperform other algorithms in R232, the Gaussian prior assumption cannot provide better performance in comparison with the proposed approach. As mentioned before, GRSA cannot provide stable analysis. Table 3 compares the variance of the neural signatures generated by GRSA with other techniques. Finally, DRSL has produced better performance in comparison with other methods. Indeed, it provides better feature representation by using deep neural networks, i.e., mapping the neural activities to a linear space.

4.2.2 Classification analysis

We first use one-subject-out cross-validation for partitioning each dataset into the training and testing sets. Then, training-set is used to generate the neural signatures ($\widetilde{\mathbf{B}} \in \mathbb{R}^{P \times V}$). As mentioned before, each row of these signatures belong to a unique category of stimuli. The goal is to generate a linear hyperplane by using each pair of these signatures where this hyperplane can be used as a binary classification model to distinguish two corresponding categories of stimuli. Suppose $\mathbf{a}_{ij}\widehat{\mathbf{x}} + z_{ij} = 0$ represents a linear hyperplane for classifying between *i-th* and *jk-th* classes, where $\mathbf{a}_{ij} = \left(\widetilde{\mathbf{b}}_{i.} - \widetilde{\mathbf{b}}_{j.}\right)\Sigma_{ij}$. Here, Σ_{ij} is used for normalization and denotes the square root of the residuals error — i.e., please refer to equation (4) in [32]. Further, $\widehat{\mathbf{x}} \in \mathbb{R}^{V}$ denotes a neural response and $z_{ij} \in \mathbb{R}$ is the offset from the origin of the vec-

---

[4] Main: Giga X399, CPU: AMD Ryzen Threadripper 2920X (24×3.5 GHz), RAM: 64GB, GPU: NVIDIA GeForce RTX 2080 SUPER (8GB memory), OS: Fedora 31, Python: 3.7.5, Pip: 19.3.1, Numpy: 1.16.5, Scipy: 1.2.1, Scikit-Learn: 0.21.3, MPI4py: 3.0.1, PyTorch: 1.2.0.

[5] Available at `https://easyfmri.gitlab.io`

(a) R105   (b) R107   (c) R232
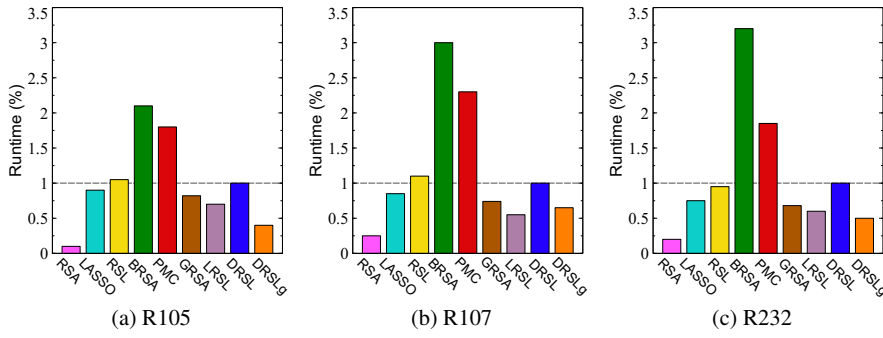
Fig. 2 Runtime Analysis
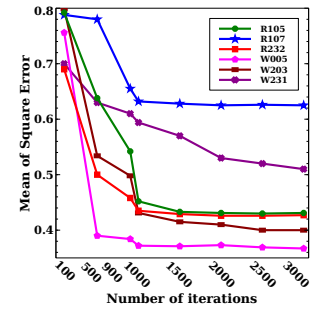
Fig. 3 Analyzing different number of iterations in DRSL
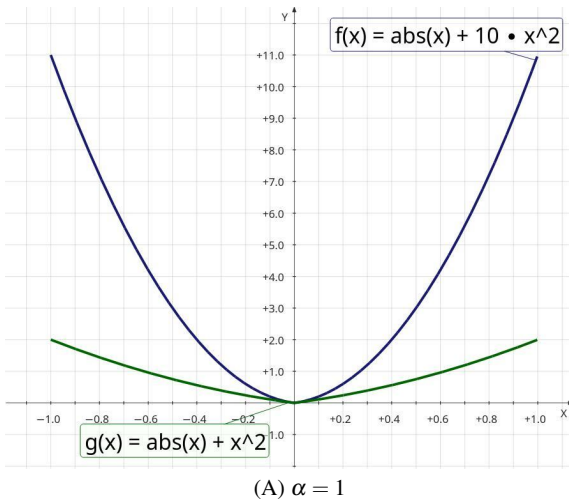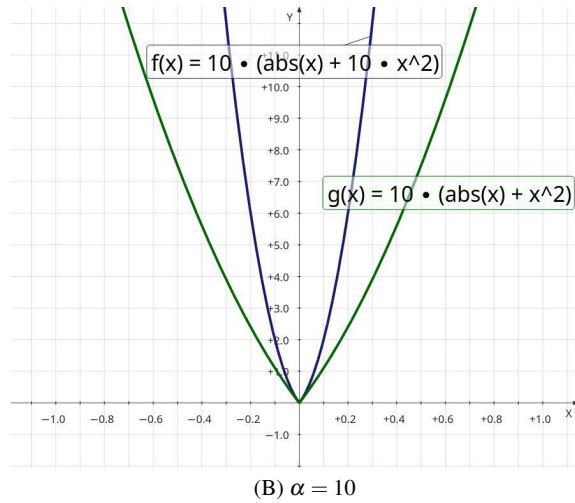
(A) $\alpha = 1$ (B) $\alpha = 10$

Fig. 4 Regularization function $r(\mathbf{B})$ by using $\alpha = 1$ and $\alpha = 10$.

tor space, which its optimal value calculates by applying the training-set to the hyperplane [8, 32]. After generating the binary classifiers, we use these models in an ECOC framework as a multi-class approach, where labels for testing-set are assigned to the closest Hamming distance [33]. To summarize, we first generate the neural signatures and then the binary and multi-class models by using the training-set. After that, we use the testing-set to evaluate the accuracy of these models. It is worth noting that the testing-set is entirely isolated from training-phase. In addition, the DRSL's network parameter for testing-set is estimated only by using the design matrices in the testing data — i.e., we use (3) in testing-phase when only testing-set is employed for calculating network parameters.

Table 4 illustrates the classification analysis. In this analysis, the performance of RSA (without regularization term) is akin to random sampling. The results of LASSO and RSL showed the positive effect of regularization on the performance of the linear models. By fitting a Gaussian distribution to the data manifold, BRSA and PCM provided better performance in comparison with the method that only control the quality of results via a regularization function. By comparing the linear LRSL with the proposed DRSL, we can understand the positive effect of the nonlinear transfor-

mation function on the performance of the analysis. DRSL provides better performance in most of the fMRI datasets because of two critical components: 1) a flexible nonlinear kernel, 2) an effective regularization approach.

4.3 Runtime analysis

This section analyzes the runtime of the proposed method and other similarity methods by employing ROI-based datasets. As mentioned before, all results in this paper are generated by using a PC with certain specifications. Further, the runtime of DRSL is evaluated by using both hardware, i.e., CPU (DRSL) and GPU (DRSLg). Figure 2 demonstrates the runtime of the mentioned methods, where runtime of other algorithms are scaled based on DRSL. In other words, the runtime of the proposed method is considered as a unit. As illustrated in this figure, BRSA and PCM generated the worse runtime because they must estimate a wide range of hyper-parameters for high-dimensional datasets. Further, the runtime of DRSL is similar to the regularized methods (LASSO and RSL), while those algorithms did not utilize any transformation function. Since GRSA and LRSL (same as DRSL) employ a min-batch of time-points, they produce better runtime in comparison with the regularized
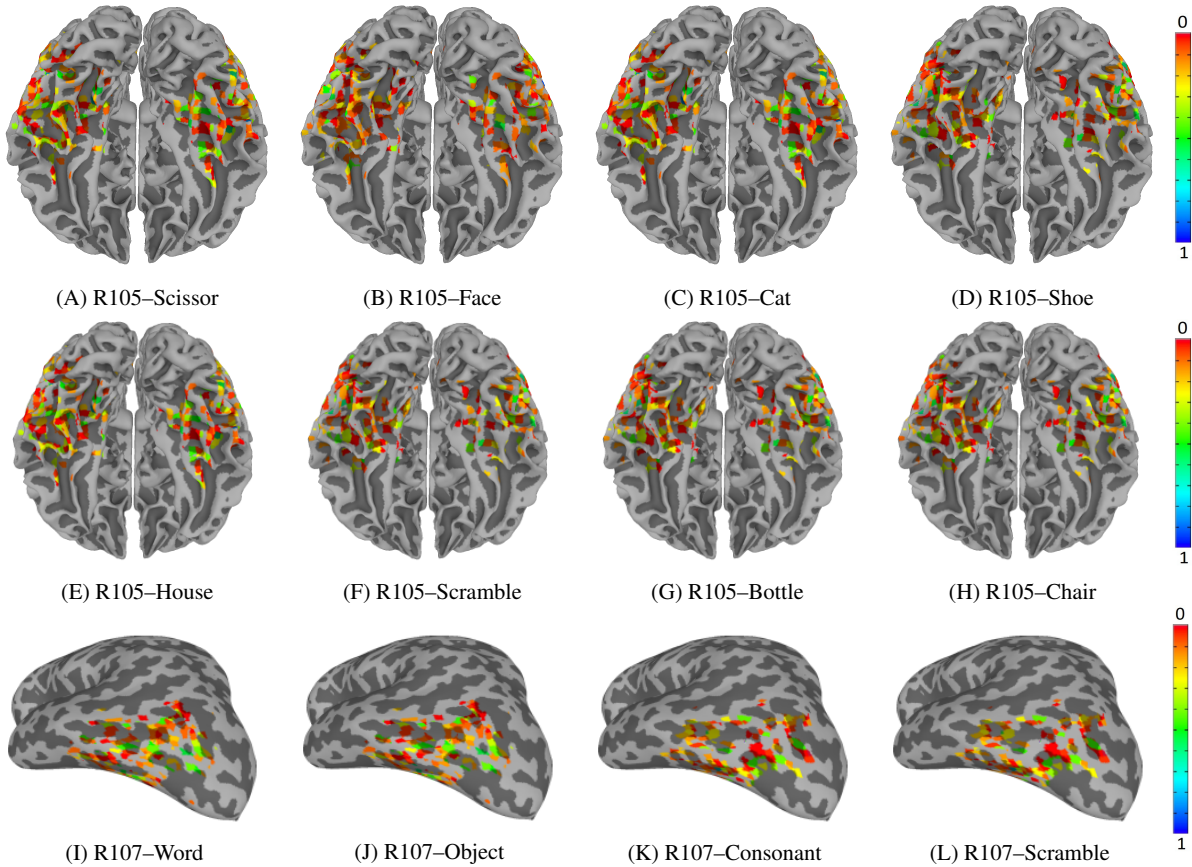
Fig. 5 Representing neural activities in the linear embedded space by using R105 and R107 datasets

methods. By considering the performance of the proposed method in the previous section, DRSL generates acceptable runtime. As mentioned before, the proposed method utilizes gradient-based approaches that can rapidly reduce the time complexity of the optimization procedure. It is worth noting that runtime of the whole-brain datasets has the same tendency.

4.4 DRSL iteration analysis

In this section, we analyze the performance of DRSL by using different numbers of iterations. As the first step, we must calculate the Mean of Square Error (*MSE*) for DRSL technique, where better models generate lower error — i.e., $MSE = \frac{1}{TSV} \sum_{\ell=1}^{S} \sum_{i=1}^{T} \sum_{j=1}^{V} \left(x_{ij}^{(\ell)} - \sum_{k=1}^{P} d_{ik}^{(\ell)} \beta_{kj}^{(\ell)} \right)^2$. Figure 3 illustrates the performance of DRSL by using MSE, where different number of iterations are used — i.e., $100, 500, 900, 1000, 1500, 2000, 2500,$ and 3000 iterations. Here, the network structure for DRSL is considered fixed and same as the previous section. As Figure 3 depicted, MSE in most of the datasets is reduced between 100 iterations to 1000 iterations. However, MSE does not have significant fluctuation in most of the datasets (except W231

dataset) after 1000 iterations. Another advantage of using DRSL with 1000 iterations in this paper was the trade-off between runtime and the performance — please refer to Figure 2, Table 3, and Table 4. Based on these results, we have used $10 \times 100 = 1000$ iterations as the default value to generate all of the results in the previous sections.

4.5 Analyzing the proposed regularization function

In this section, we present some insights into how the proposed regularization function works. As the primary advantage, the regularization term $r(\mathbf{B}) = \sum_{j=1}^{V} \sum_{k=1}^{P} \alpha |\beta_{kj}| + 10\alpha(\beta_{kj})^2$ is convex, and it is consequently suitable for optimization.

As Figure 4 depicted, the coefficient $\alpha = 10$ can increase the sensitivity of the regularization term to handle small change of β values in comparison with the coefficient $\alpha = 1$. There are two advantages to use $\alpha = 10$. First, since the neural activities are highly correlated, the difference of β values in different categories of stimuli is indeed small. Second, it can remove the effect of random noises on the performance of analysis — specifically on a dataset with a low rate of SNR.

4.6 Representing neural signatures

In this section, we visualize some samples of neural signatures that are produced by DRSL — based on $\widetilde{\mathbf{B}}$ matrix — for datasets R105 and R107. Figure 5 shows these generated neural signatures. Here, we demonstrate the left side of the brain for R107 because there is no ROI region on the right side for this dataset [17]. As depicted in this figure, while some of the neural activities are significantly distinctive across categories of stimuli, the rest of them are highly correlated. For instance, word stimuli and object category have a strong correlation as the meaningful concepts in R107, whereas the neural signatures of these categories are completely different with non-meaningful concepts, i.e., consonant and scramble stimuli. Similarly, the non-meaningful concepts also have a strong correlation in this dataset. As another alternative, we can compare the meaningful concepts in R105 with the neural signature generated for scramble stimuli.

5 Discussion and Conclusion

Similarity analysis is one of the crucial steps in most fMRI studies. The existing efforts on developing similarity approaches to analyze neural responses show some promising results. However, there are still several long-standing challenges. First, classic similarity methods such as RSA (with OLS and GLM solutions) may not provide stable analysis — especially when the covariance of the neural activities has a low rate of SNR. The proposed method employed a new regularization term that steeply penalized neural responses to handle the SNR issue. Second, most regularized similarity approaches such as LASSO, RSL, and GRSA use a linear method for generating the neural signatures. In this paper, we used a parametric, nonlinear transformation function to map the neural responses to a lower-dimensional latent space. The proposed method enabled us to discover more complex feature representations for similarity analysis. Third, most nonlinear similarity approaches such as BRSA and PCM cannot provide a time-efficient analysis on data with a large number of subjects, a broad ROI, or even whole-brain fMRI images. Alternatively, the proposed method utilized a gradient-based optimization technique that rapidly reduced the runtime for analyzing high-dimensional fMRI datasets.

The main difference between the proposed deep model and a regular network model lies in its application. A regular deep neural network is mostly used in the form of a supervised structure — i.e., it maps input space to supervised labels. However, our method employed the unsupervised deep neural network (i.e., multiple stacked layers of nonlinear transformations) as the parametric kernel function that maps the voxel space to a lower-dimension, information-rich feature space. In some sense, the transformation function employed in DRSL is similar to the one utilized in the Deep Canonical Correlation Analysis (DCCA) for multi-view representational learning or Deep Hyperalignment (DHA) for aligning neural activities of multi-subject data across subjects. Nevertheless, DCCA and DHA cannot be used for similarity analysis. The primary objective functions in these approaches map neural activities of each subject to a shared space — where the transformed features can improve the performance of a classification model. Unlike DCCA and DHA, our method is formulated as a nonlinear multi-set regression problem — where it must find better regressors to map the matrix of neural responses and the design matrix.

In summary, this paper introduces a deep learning approach for Representational Similarity Analysis (RSA) in order to provide accurate similarity (or distance) analysis in multi-subject fMRI data. Deep Representational Similarity Learning (DRSL) can handle fMRI datasets with noise, sparsity, nonlinearity, high-dimensionality (broad ROI or whole-brain data), and a large number of subjects. DRSL utilizes gradient-based optimization approaches and generates an efficient runtime on large datasets. Further, DRSL is not limited by a restricted fixed representational space because the transformation function in DRSL is a multi-layer neural network, which can separately implement any nonlinear function for each subject to transfer the neural activities into an embedded linear space. To evaluate the performance of the proposed method, multi-subject fMRI datasets with various tasks — including visual stimuli, decision making, flavor, and working memory — are employed for running the empirical studies. The results confirm that DRSL achieves superior performance to other state-of-the-art RSA algorithms for evaluating the similarities between distinctive cognitive tasks. In the future, we will plan to use the proposed method for improving the performance of other techniques in fMRI analysis, e.g., multi-modality, and hub detection.

6 Proof of Lemma 3.1

Step 1:

Suppose $\mathbf{X}$ and $\mathbf{Y}$ are matrices of any size. And there is a differentiable mapping such that $\mathbf{Y} = f_1(\mathbf{X})$. If f_2 is a real-valued function differentiable with respect to $\mathbf{Y}$, then we have:

$$\frac{\partial f_2(\mathbf{Y})}{X_{ij}} = \mathrm{tr}\left[\frac{\partial \mathbf{Y}}{\partial X_{ij}}\left(\frac{\partial f_2(\mathbf{Y})}{\partial \mathbf{Y}}\right)^{\top}\right].$$

A simple application of the chain rule gives:

$$\frac{\partial f_2(\mathbf{Y})}{\partial X_{ij}} = \sum_k \sum_l \frac{\partial f_2(\mathbf{Y})}{\partial Y_{kl}} \frac{\partial Y_{kl}}{\partial X_{ij}}$$

$$= \sum_k \left(\frac{\partial \mathbf{Y}}{\partial X_{ij}}\right)_{k.} \left(\frac{\partial f_2(\mathbf{Y})}{\partial \mathbf{Y}}\right)^\top_{.k}$$

$$= \mathrm{tr}\left[\frac{\partial \mathbf{Y}}{\partial X_{ij}} \left(\frac{\partial f_2(\mathbf{Y})}{\partial \mathbf{Y}}\right)^\top\right].$$

<u>Step 2:</u>

We can proof the following equality for any matrix $\mathbf{A}$, $\mathbf{X}$, $\mathbf{B}$ in $\mathbb{R}$ space:

$$\frac{\partial \|\mathbf{AX} + \mathbf{B}\|^2_F}{\partial \mathbf{X}} = 2\mathbf{A}^\top(\mathbf{AX} + \mathbf{B}).$$

Let $\mathbf{D}^{(ij)}$ denote $\frac{\partial \mathbf{AX}}{\partial X_{ij}}$ and $\mathbf{Y}$ denote $\mathbf{AX} + \mathbf{B}$.

Firstly, we have

$$D^{(ij)}_{kl} = \frac{\partial (AX)_{kl}}{\partial X_{ij}} = \frac{\partial \sum_m A_{km} X_{ml}}{\partial X_{ij}} = \begin{cases} 0, & l \neq j \\ A_{ki}, & \text{otherwise} \end{cases}$$

According to Step 1, we have

$$\frac{\partial \|\mathbf{AX} + \mathbf{B}\|^2_F}{\partial X_{ij}} = \mathrm{tr}\left[\frac{\partial (\mathbf{AX} + \mathbf{B})}{\partial X_{ij}} \left(\frac{\partial \|\mathbf{AX} + \mathbf{B}\|^2_F}{\partial (\mathbf{AX} + \mathbf{B})}\right)^\top\right]$$

$$= 2\mathrm{tr}\left(\mathbf{D}^{(ij)} Y^\top\right)$$

$$= 2\sum_k \sum_l D^{(ij)}_{kl} Y_{kl}$$

$$= 2\sum_k D^{(ij)}_{kj} Y_{kj}$$

$$= 2\sum_k A_{ki} Y_{kj}$$

$$= 2(A^\top Y)_{ij}.$$

<u>Step 3:</u>

By considering Steps 1 and 2, we have:

$$\nabla_{\mathbf{B}^{(\ell)}}(J_R^{(k,\ell)})$$

$$= \sum_{i \in \Psi^{(k,\ell)}} \frac{\partial}{\partial \mathbf{B}^{(\ell)}} \|f(\mathbf{x}_{i.}^{(\ell)}; \theta^{(\ell)}) - \mathbf{d}_{i.}^{(\ell)} \mathbf{B}^{(\ell)}\|^2_2 + \frac{\partial}{\partial \mathbf{B}^{(\ell)}} r(\mathbf{B}^{(\ell)})$$

$$= -2\sum_{i \in \Psi^{(k,\ell)}} \left(\mathbf{d}_{i.}^{(\ell)}\right)^\top \left(f(\mathbf{x}_{i.}^{(\ell)}; \theta^{(\ell)}) - \mathbf{d}_{i.}^{(\ell)} \mathbf{B}^{(\ell)}\right) + \frac{\partial}{\partial \mathbf{B}^{(\ell)}} r(\mathbf{B}^{(\ell)})$$

For the last term $\frac{\partial}{\partial \mathbf{B}^{(\ell)}} r(\mathbf{B}^{(\ell)})$, we have:

$$\frac{\partial}{\partial \beta_{ab}^{(\ell)}} r(\mathbf{B}^{(\ell)}) = \sum_{j=1}^V \sum_{k=1}^P \alpha \frac{\partial |\beta_{kj}^{(\ell)}|}{\partial \beta_{ab}^{(\ell)}} + 10\alpha \frac{\partial (\beta_{kj}^{(\ell)})^2}{\partial \beta_{ab}^{(\ell)}}$$

$$= \alpha \frac{\beta_{ab}^{(\ell)}}{|\beta_{ab}^{(\ell)}|} + 20\alpha \beta_{ab}^{(\ell)}$$

$$= (\alpha \mathrm{sign}(\mathbf{B}^{(\ell)}) + 20\alpha \mathbf{B}^{(\ell)})_{ab}$$

Therefore, we obtain that $\frac{\partial}{\partial \mathbf{B}^{(\ell)}} r(\mathbf{B}^{(\ell)}) = \alpha \mathrm{sign}(\mathbf{B}^{(\ell)}) + 20\alpha \mathbf{B}^{(\ell)}$, completing the proof. ∎

Information Sharing Statement

In this paper we use the publicly available Open Neuro datasets that can be found via the site link: `https://openneuro.org/`. Further, we have shared a preprocessed version of datasets in MATLAB format at `https://easydata.learningbymachine.com/`. In addition, the proposed method can be accessed by using our GUI-based toolbox — i.e., available at `https://easyfmri.learningbymachine.com/`.

Compliance with Ethical Standards

Conflict of Interests

All authors declare that they have no conflict of interest.

Ethical Approval

This article does not contain any studies with human participants or animals performed by any of the authors.

References

1. Andrew G, Arora R, Bilmes J, Livescu K (2013) Deep canonical correlation analysis. In: International Conference on Machine Learning (ICML), pp 1247–1255

2. Benton A, Khayrallah H, Gujral B, Reisinger D, Zhang S, Arora R (2017) Deep generalized canonical correlation analysis. In: International Conference on Learning Representations (ICLR)

3. Bondell HD, Reich BJ (2008) Simultaneous regression shrinkage, variable selection, and supervised clustering of predictors with oscar. Biometrics 64(1):115–123

4. Cai MB, Schuck NW, Pillow JW, Niv Y (2016) A bayesian method for reducing bias in neural representational similarity analysis. In: Advances in Neural Information Processing Systems (NIPS), pp 4951–4959

5. Cai MB, Schuck NW, Pillow JW, Niv Y (2019) Representational structure or task structure? bias in neural representational similarity analysis and a bayesian method for reducing bias. PLoS Computational Biology 15(5):e1006299

6. Carlin JD, Kriegeskorte N (2017) Adjudicating between face-coding models with individual-face fmri responses. PLoS Computational Biology 13(7):e1005604

7. Connolly AC, Guntupalli JS, Gors J, Hanke M, Halchenko YO, Wu YC, Abdi H, Haxby JV (2012) The representation of biological classes in the human brain. Journal of Neuroscience 32(8):2608–2618

8. Diedrichsen J, Kriegeskorte N (2017) Representational models: A common framework for understanding encoding, pattern-component, and representational-similarity analysis. PLoS Computational Biology 13(4):e1005508

9. Diedrichsen J, Ridgway GR, Friston KJ, Wiestler T (2011) Comparing the similarity and spatial structure of neural representations: a pattern-component model. Neuroimage 55(4):1665–1678

10. Duncan KJ, Pattamadilok C, Knierim I, Devlin JT (2009) Consistency and variability in functional localisers. NeuroImage 46(4):1018–1026

11. Figueiredo M, Nowak R (2016) Ordered weighted l1 regularized regression with strongly correlated covariates: Theoretical aspects. In: Artificial Intelligence and Statistics (AAAI), pp 930–938

12. Hans C (2009) Bayesian lasso regression. Biometrika 96(4):835–845

13. Haxby JV, Connolly AC, Guntupalli JS (2014) Decoding neural representational spaces using multivariate pattern analysis. Annual Review of Neuroscience 37:435–456

14. Hoerl AE, Kennard RW (1970) Ridge regression: Biased estimation for nonorthogonal problems. Technometrics 12(1):55–67

15. Huth AG, De Heer WA, Griffiths TL, Theunissen FE, Gallant JL (2016) Natural speech reveals the semantic maps that tile human cerebral cortex. Nature 532(7600):453

16. Kay KN, Naselaris T, Prenger RJ, Gallant JL (2008) Identifying natural images from human brain activity. Nature 452(7185):352

17. Khaligh-Razavi SM, Kriegeskorte N (2014) Deep supervised, but not unsupervised, models may explain it cortical representation. PLoS computational biology 10(11):e1003915

18. Khaligh-Razavi SM, Henriksson L, Kay K, Kriegeskorte N (2017) Fixed versus mixed rsa: Explaining visual representations by fixed and mixed feature sets from shallow and deep computational models. Journal of Mathematical Psychology 76:184–197

19. Kingma DP, Ba J (2015) Adam: A method for stochastic optimization. In: International Conference on Learning Representations (ICLR)

20. Kiwiel KC (2001) Convergence and efficiency of subgradient methods for quasiconvex minimization. Mathematical Programming 90(1):1–25

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

22. Kriegeskorte N, Mur M, Bandettini PA (2008) Representational similarity analysis-connecting the branches of systems neuroscience. Frontiers in Systems Neuroscience 2:4

23. Mitchell TM, Shinkareva SV, Carlson A, Chang KM, Malave VL, Mason RA, Just MA (2008) Predicting human brain activity associated with the meanings of nouns. Science 320(5880):1191–1195

24. Naselaris T, Kay KN, Nishimoto S, Gallant JL (2011) Encoding and defcoding in fmri. Neuroimage 56(2):400–410

25. Oswal U, Cox C, Lambon-Ralph M, Rogers T, Nowak R (2016) Representational similarity learning with application to brain networks. In: International Conference on Machine Learning (ICML), pp 1041–1049

26. Rumelhart DE, Hinton GE, Williams RJ (1986) Learning representations by back-propagating errors. Nature 323(6088):533

27. Sergio LE, Hamel-Pâquet C, Kalaska JF (2005) Motor cortex neural correlates of output kinematics and kinetics during isometric-force and arm-reaching tasks. Journal of Neurophysiology 94(4):2353–2378

28. Sheng X, Yousefnezhad M, Xu T, Yuan N, Zhang D (2018) Gradient-based representational similarity analysis with searchlight for analyzing fmri data. In: Pattern Recognition and Computer Vision

29. Stephan-Otto C, Siddi S, Senior C, Muñoz-Samons D, Ochoa S, Sánchez-Laforga AM, Brébion G (2017) Visual imagery and false memory for pictures: a functional magnetic resonance imaging study in healthy participants. PLoS One 12(1):e0169551

30. Tom SM, Fox CR, Trepel C, Poldrack RA (2007) The neural basis of loss aversion in decision-making under risk. Science 315(5811):515–518

31. Veldhuizen MG, Babbs RK, Patel B, Fobbs W, Kroemer NB, Garcia E, Yeomans MR, Small DM (2017) Integration of sweet taste and metabolism determines carbohydrate reward. Current Biology 27(16):2476–2485

32. Walther A, Nili H, Ejaz N, Alink A, Kriegeskorte N, Diedrichsen J (2016) Reliability of dissimilarity measures for multi-voxel pattern analysis. Neuroimage 137:188–200

33. Yousefnezhad M, Zhang D (2017) Anatomical pattern analysis for decoding visual stimuli in human brains. Cognitive Computation pp 1–12

34. Yousefnezhad M, Zhang D (2017) Deep hyperalignment. In: Advances in Neural Information Processing Systems (NIPS), pp 1603–1611

35. Yousefnezhad M, Zhang D (2017) Multi-region neural representation: A novel model for decoding visual stimuli in human brains. In: Proceedings of the 2017 SIAM International Conference on Data Mining (SDM), SIAM, pp 54–62

36. Yuan M, Lin Y (2006) Model selection and estimation in regression with grouped variables. Journal of the Royal Statistical Society: Series B (Statistical Methodology) 68(1):49–67

37. Zou H, Hastie T (2005) Regularization and variable selection via the elastic net. Journal of the Royal Statistical Society: Series B (Statistical Methodology) 67(2):301–320