On the benefits of self-taught learning for brain decoding

Elodie Germani, Elisa Fromont, Camille Maumet

ON THE BENEFITS OF SELF-TAUGHT LEARNING FOR BRAIN DECODING

Elodie Germani
Univ Rennes, Inria, CNRS, Inserm
Rennes, France
elodie.germani@irisa.fr

Elisa Fromont
Univ Rennes, IUF, Inria, CNRS, IRISA
Rennes, France

Camille Maumet
Univ Rennes, Inria, CNRS, Inserm
Rennes, France

ABSTRACT

We study the benefits of using a large public neuroimaging database composed of fMRI statistic maps, in a self-taught learning framework, for improving brain decoding on new tasks. First, we leverage the NeuroVault database to train, on a selection of relevant statistic maps, a convolutional autoencoder to reconstruct these maps. Then, we use this trained encoder to initialize a supervised convolutional neural network to classify tasks or cognitive processes of unseen statistic maps from large collections of the NeuroVault database. We show that such a self-taught learning process always improves the performance of the classifiers but the magnitude of the benefits strongly depends on the number of data available both for pre-training and finetuning the models and on the complexity of the targeted downstream task.

***K*eywords** Self-taught Learning · Brain Decoding · Autoencoder · Convolutional Neural Network · Deep Learning

1 Introduction

In the past few years, deep learning (DL) approaches have achieved outstanding performance in the field of neuroimaging [1] due to their ability to model complex non-linear relationships in the data. Functional Magnetic Resonance Imaging (fMRI) data, a noninvasive neuroimaging technique in which brain activity is recorded during specific experimental protocols probing different mental processes and giving a big picture on cognition, are often used as input data to these models. These can be used for different purpose, such as disease diagnosis [2] or brain decoding (*i.e.* identifying stimuli and cognitive states from brain activities) [3], with a common goal: linking a target with highly variable patterns in the data and ignoring aspects of the data that are unrelated to the learning task. Researchers took advantage of the specific properties of fMRI data to build more and more sophisticated models [4, 5, 6, 7, 8, 9, 10, 11].

However, training effective DL models using fMRI data comes with many challenges [12]. Among these is the need for large amounts of training data to compensate for the large number of trainable parameters in DL models [13]. Performance of these models are limited by the high dimensionality and low sample size of conventional fMRI datasets [14, 15]. These are typically composed of 3D volumes with hundred thousand dimensions (or voxels) for each for a rather small number of subjects (typically 10-100). Although sample sizes in fMRI studies increased these past few years and despite the growing efforts in collecting large-scale fMRI datasets [16], these studies often focus on a small number of specialised tasks. Thus, each neuroimaging experiment only explore a few cognitive processes, making it difficult to build models that can effectively be applied to new studies.

To prevent overfitting and allow for generalizable statistical inference, researchers focused on methods to tackle this lack of training data [17, 18, 19]. For instance, [20] built a decoding model using data gathered from 35 studies and thousands of individuals that cover various cognitive domains. Despite the good performance of the models, these can

only be applied on restricted sets of studies, discriminating between few cognitive concepts. More annotated training data (*e.g.* using large public databases) would be required to map a wider set of cognitive processes.

This problem of small training sets is not limited to neuroimaging but is already known in the field of machine learning, where researchers extensively use deep transfer learning to improve classification and generalization performance of their models [21]. It consists in using the knowledge obtained from a model trained for a source task on a source dataset and applying it to a target task on a target dataset. Transfer learning proved its worth by using large, publicly available datasets [22] to pre-train DL models before fine-tuning them on smaller datasets of a related domain.

In neuroimaging, lots of studies were made on inductive transfer learning with labeled source data as defined in [21] (e.g. source task and target task are different, as well as source domain and target domain) [23, 24, 25]. For instance, [23] pre-trained two DL classifiers on a large, public whole-brain fMRI dataset of the HCP, fine-tuned them and evaluated their performance on another task on the same dataset and on a fully independent dataset. In another study, [24] used the ImageNet database [22], a large, public dataset containing naturalistic images from more than 1000 classes, to pre-train a model and adapt it to classify tasks from 2D fMRI data. Both these studies showed improved decoding accuracies as well as quicker learning and less training data required.

However, labeled databases are not always available in neuroimaging. Despite the growing effort in data sharing to build public databases [26], such as OpenNeuro for raw data [27] and NeuroVault for fMRI statistic maps [28]. The unconstrained annotations and the heterogeneity of tasks and studies make them difficult to use to pre-train a supervised deep learning model. To compensate this, weakly supervised learning techniques such as automatic labelling of data has proved its worth. For instance, [29] enriched NeuroVault annotations using the Cognitive Atlas ontology [30] and used these labeled data to train a multi-task decoding model that successfully decoded more than 50 classes of mental processes on a large test set.

A specific type of inductive transfer learning named *self-taught learning* [31, 32] showed strong empirical success in the field of machine learning. It does not require any labels as it consists in training models to autonomously learn latent representations of the data and using these to improve learning in a supervised setting. This approach is motivated by the observation that data from similar domains contain patterns that are similar to those of the target domain. By initializing the weights of a supervised classifier with the pre-trained weights of an unsupervised model trained on many images. The aim is to improve the model performance by placing the parameters close to a local minimum of the loss function and by acting as a regularizer [33].

In the field of neuroimaging, latent representations have recently been used in a task-relevant autoencoding framework. [34] used an autoencoder with a classifier attached to the bottleneck layer on a small fMRI dataset. This model outperformed the classifier trained on raw input data by focusing on cleaner, task-relevant representations. This suggests that a low-level representation of fMRI data, learned for a reconstruction task, could be helpful in a classification task, as in a self-taught learning framework.

In this work, we want to take advantage of a large public neuroimaging database in a self-taught learning framework. We pre-train an unsupervised deep learning model to learn a latent representation of fMRI statistic maps and we fine-tune this model to decode tasks or mental processes involved in several studies. In a first part, we leverage the NeuroVault database to select the most relevant statistic maps and train a convolutional autoencoder (CAE) to reconstruct these maps. In a second part, we use the final weights of the encoder to initialize a supervised convolutional neural network (CNN) to classify the cognitive processes, tasks or constrasts of unseen statistic maps from large collections of the NeuroVault database (an homogeneous collection of more than 18000 statistic maps and an heterogeneous one with 6500 maps). Our goal was to investigate how self-taught learning can be beneficial in the field of brain imaging for deep learning models.

2 Material & Methods

The code produced to run the experiments and to create the figures and tables of this paper is available in the Software Heritage public archive at: `https://archive.softwareheritage.org/browse/origin/https://gitlab.inria.fr/egermani/self_taught_decoding`. Derived data used to execute these notebooks are stored in Zenodo [35].

Figure 1 illustrates the overall process used to implement our self-taught learning framework: a CAE is first trained to reconstruct the maps of a large dataset extracted from NeuroVault. Then, the encoder part of the CAE is fine-tuned to answer a classification problem on another dataset (with labels). After hyperparameters optimisation, performance of the pre-trained classifier are compared to those of a classifier initialized with a default algorithm. Details regarding the datasets (NeuroVault dataset and classification datasets) can be found in the next subsection. The models of the CAEs

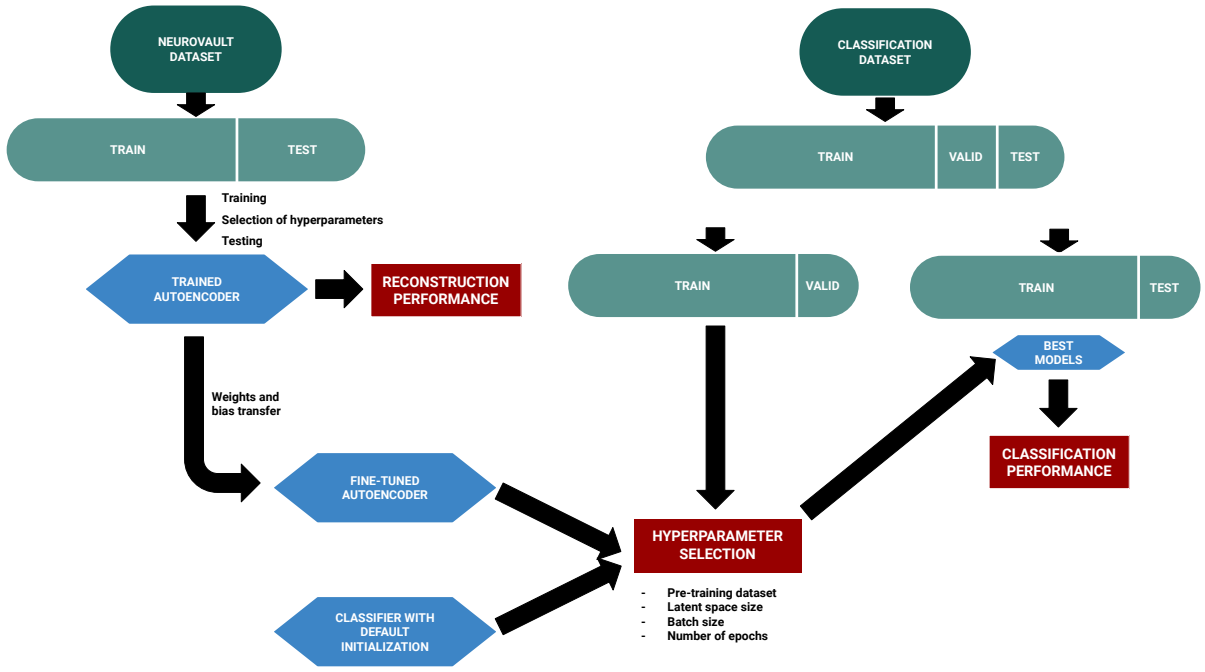

Figure 1: Flow diagram of the self-taught learning methodology.

and the CNNs are presented in Section 2.3. Further explanations on the workflow used to train the CAE and the CNN and to evaluate their performance are available in Sections 2.4 and 2.5 respectively.

2.1 Overview of the datasets

A summary of the different datasets can be found in Table 1. Details are given below.

Table 1: Overview of the datasets. For each dataset, number of statistic maps are presented, as well as the number of subjects, number of studies and the type of labels (if available).

Dataset	Maps	Subjects	Studies	Labels
NeuroVault	28,532	-	-	-
HCP	18,070	707	1	Tasks (*7*) Contrasts (*23*)
Large BrainPedia	6,448	133	29	Cognitive processes (*36*)
Small BrainPedia	1,536	72	9	Cognitive processes (*30*)

2.1.1 NeuroVault dataset

NeuroVault [28] (RRID:SCR_003806) is a web-based repository for statistic maps, parcellations and atlases produced by MRI and PET studies. This is currently the largest public database of fMRI statistic maps. NeuroVault has its own public Application Programming Interface (API) that provides a full access to all images (grouped by collections) and enables filtering of images or collections with associated metadata. At the time of experiment (19/01/2022), a total of 461,461 images in 6,782 collections were available. Among the available metadata, some were mandatory and specified for all maps such as the modality (e.g. "fMRI-BOLD" for Blood-Oxygen Level Dependent Functional MRI; diffusion

MRI, etc.), the type of statistic (e.g. "T map" or "Z map") or the cognitive paradigm (e.g. "Working memory" or "Motor fMRI task paradigm"), and others were optional and only available if additionally entered at the time of the upload.

From this large database, relevant maps were selected based on multiple criteria. First, we chose maps for which the modality was 'fMRI-BOLD' to exclude others modalities such as structural or diffusion MRI, etc. To get comparable maps, we set three additional inclusion criteria to select maps: 1/ for which all required metadata were provided ('is_valid' to True) and 2/ that were registered in MNI space ('not_mni' to False) – to ensure that anatomical structures were located at the same coordinates in each map – and 3/ referenced as 'T map' or 'Z map' – to exclude maps in which voxel values did not have the same meaning (P value maps, Chi-squared maps, etc.) –. Among these, thresholded statistic maps were excluded.

We found that some maps in our initial dataset, were wrongly referenced as T map or Z map. These misclassified maps were removed by filtering the 'filename' column of the dataframe to exclude *SetA_mean SetB_mean* (AFNI contrast maps), *con* (SPM contrast maps), *cope* (FSL contrast maps).

Using these criteria, a total of 28,532 statistic maps were selected from the NeuroVault database and constituted our 'NeuroVault dataset'. Most of these maps were unlabeled (*i.e.* cognitive processes or tasks performed described as 'None / Other') or not labeled in a standardized way (*i.e.* use of terms that are specific for a study instead of generic terms, *e.g.* some maps are labeled as 'word-picture matching task' for the cognitive paradigm whereas others in which a similar task is performed are referenced as 'working memory fMRI task paradigm' which is a label that includes other specific tasks).

2.1.2 HCP dataset (NeuroVault Collection nº4337)

[36] included 18,070 z-statistic maps, for base contrasts (task vs baseline), corresponding to 700 subjects of the HCP 900 release [16]. This collection was excluded from our pre-training dataset (see section 2.1.1) due to missing metadata (i.e. 'is_valid' was False).

All maps of the collection were collected in a dataset named 'HCP dataset'. Multiple labels were entered for each map including: mental concepts ('cognitive_paradigm_cogatlas'), tasks ('task') and contrasts ('contrast_definition'). We used the definitions provided by [30]. For each subject, 23 contrasts ('0BKBODY', '0BK-FACE', '0BKPLACE', '0BKTOOL', '2BKBODY', '2BK-FACE', '2BKPLACE', '2BKTOOL', 'CUE', 'FACES', 'LF', 'LH', 'MATCH', 'MATH', 'PUNISH', 'RANDOM', 'REL', 'REWARD', 'RF', 'RH', 'SHAPES', 'STORY', 'TOM') distributed in 7 tasks ('EMOTION', 'GAMBLING', 'LANGUAGE', 'MOTOR', 'RELATIONAL', 'SOCIAL', 'WM') were available. For more details on contrasts, tasks and mental concepts of this study, see [16].

2.1.3 BrainPedia large and small datasets (NeuroVault collection nº1952)

[37] [38], known as BrainPedia, contained fMRI statistic maps from about 30 fMRI studies in OpenNeuro [27], the Human Connectome Project [16] and data acquired at Neurospin research center, together they were chosen to map a wide set of cognitive functions.

This collection contained 6,573 statistic maps corresponding to 45 unique mental concepts derived from 19 sub-terms (*e.g.* 'visual, right hand, faces' for maps associated with the task of watching an image of a face and responding to a working memory task). These images were previously used to build a multi-class decoding model [38] and labels corresponded to the mental concepts associated with the statistic map, e.g., 'visual', 'language' or 'objects'. Here we excluded the nine classes that had less than 30 samples each, leaving 6,448 images corresponding to 36 classes. These 6,448 images formed the 'Large BrainPedia' dataset.

From this collection, another smaller dataset was extracted to test if the impact of the self-taught learning process was different depending on the number of samples. Indeed, researchers often want to study specific tasks or cognitive paradigms that are present on only a few number of studies with a small number of participants. This small dataset was designed to evaluate the performance of our framework in this context. Among the 30 studies of the Large BrainPedia dataset, those with less than 20 subjects were excluded and among the remaining studies, 20 subjects were randomly drawn per study. In the end, the Small BrainPedia dataset was composed of 1,536 maps, divided in 30 classes, from 9 studies and 180 different subjects.

2.2 Preprocessing

All statistic maps included in this study were downloaded from different collections of NeuroVault and therefore were obtained using different processing pipelines (see the original studies for more details [38], [16]). In addition, we resampled all maps to dimensions (48, 56, 48) using the MNI152 template available in Nilearn [39] (RRID:

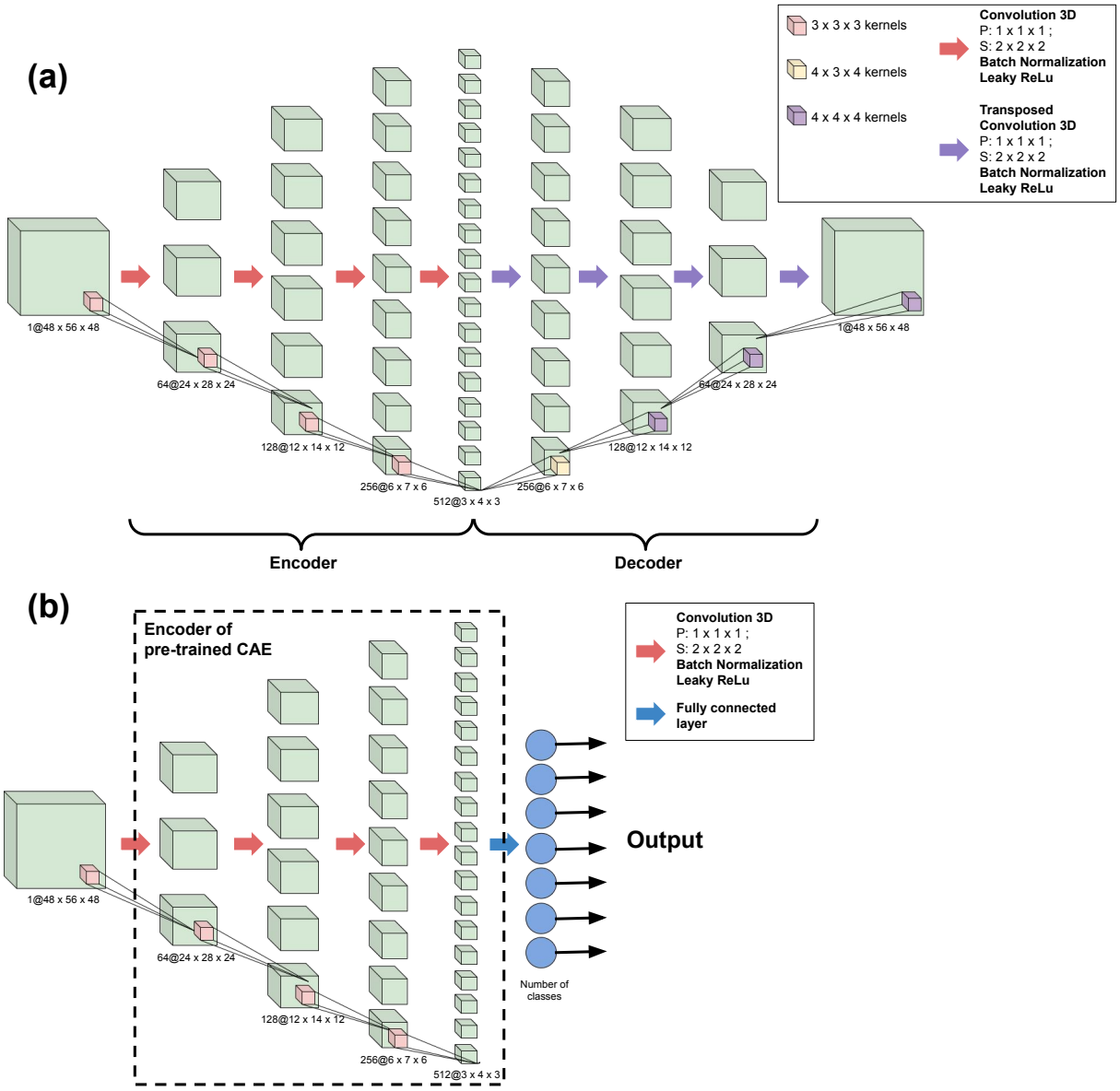

Figure 2: Schematic visualisation of the architectures of the Convolutional AutoEncoder (a) and Convolutional Neural Network (b) with 4 layers.

SCR_001362) as target image. A min-max normalization was also performed on all resampled maps to get statistical values between -1 and 1 and limit the impact of outlier values. Finally, the brain mask of the MNI152 template in Nilearn was used to exclude statistical values outside the brain in all statistic maps.

2.3 Model architectures

All models were implemented using PyTorch [40] v1.12.0 with CUDA [41] v10.2. For our model architectures, we chose to use 3D-convolutional feature extractors that take into account the three spatial dimensions of fMRI statistic maps. Schematic representations of the architectures are available in Figure 2 and Supplementary Figure S1.

2.3.1 Convolutional AutoEncoder (CAE)

The base architecture of our CAE was inspired from [19]. Two architectures were derived from this base: a 4 layers and a 5 layers architecture, respectively corresponding to the number of convolutional layers in each part of the CAE (encoder and decoder). In the 4-layer model, the encoder part consisted in four 3D convolutional layers with respectively 64, 128, 256 and 512 channels. Each layer had a kernel size of 3 x 3 x 3, a stride of 2 x 2 x 2 and a padding of 1 x 1 x 1. 3D batch normalization layers [42] followed each convolutional layers with respectively 64, 128, 256 and 512 channels and a leaky rectified linear unit (ReLU) activation function was used for all layers. The decoding part of the CAE was symmetric to the encoder, except that 3D transposed convolutional layers were used instead of classic convolutional layers. Transposing convolutions is a method to upsample an output using learnable parameters. It can be seen as an opposite process to classical convolutions. To keep the number of features symmetric at each layers output, the kernel size of the first layer was set to 4 x 3 x 4 and to 4 x 4 x 4 for all other transposed convolutional layers. Leaky ReLU activation function was also used for all layers except for the last one, *i.e.* the output one, for which a sigmoid function was used in order to obtain output values between -1 and 1. The latent space for this model was of size 512 x 3 x 4 x 3. A schematic representation of this architecture can be found in Figure 2(a).

In the 5-layer model, one convolutional layer was added at the beginning of the encoder with 32 channels and similar parameters as the other layers of the encoder. A transposed convolutional layer was also added at the end of the decoder with 32 channels. The kernel sizes in the decoder were also modified to maintain the feature map sizes: the first and second layers of the decoder had kernel sizes of 3 x 4 x 3 and 4 x 3 x 4 respectively. All other parameters, batch normalization layers and activation functions were the same. The latent space for this model was of size 512 x 2 x 2 x 2. A schematic representation of this architecture can be found in the Supplementary Figure S1 (a).

2.3.2 Convolutional Neural Network (CNN)

The 3D CNNs used for classification followed the architecture of the encoder part of the CAEs. In the same way as for the CAEs, two CNN architectures were derived. For each one, we took the corresponding architecture of the encoder (4 or 5 layers) and added a fully connected layer at the end. The number of nodes in this layer varied depending on the number of classes. A softmax activation function was used for this output layer. Visual representation of the CNNs are available in Figure 2(b) and Supplementary Figures S1 (b).

2.4 CAE training

To train our CAEs to reconstruct the statistic maps of the NeuroVault dataset, we used an Adam optimizer [43] with a learning rate of 1e-04 and all other parameters with default values. The loss function was the Mean Squared Error (MSE: the squared L2 norm) which is the standard reconstruction loss.

2.4.1 Architecture comparison

To limit the computational cost of our experiments, we fixed some of the hyperparameters of the CAE and only compared those who were of interest for the later experiments. Here, we use the term model "hyperparameters", to distinguish with model "parameters", to represent the values that cannot be learnt during training, but are set beforehand e.g., the batch size or the number of hidden layers. Thus, a batch size of 32 and a learning rate of 1e-04 were chosen to train the CAE for a number of 200 epochs (*i.e.* values that are often used in experiments). The only hyperparameter for which different values were compared was the number of hidden layers of the model: 4 layers vs 5 layers for each part (encoder/decoder) of the model. Each model architecture was first trained on the NeuroVault dataset. The dataset was randomly split in two subsets: training and test with respectively 80% and 20% of the maps. The training set was used to train the CAE with the different architectures and the test set to assess the performance of the different models (with different hyperparameters).

2.4.2 Performance evaluation

To assess the performance of the CAEs, we estimated Pearson's correlation coefficient between the reconstructed statistic map and the original statistic map. The correlation coefficient was computed using numpy version 1.21.2 (RRID: SCR_008633)[44]. The closer to 1 the correlation coefficient was, the stronger the relationship between the maps was and the more accurate the reconstruction. Note that the MSE could have also been used in this context but its individual values (for each data point) are not easily interpreted.

2.5 Classifier training

We trained two types of classifiers for all the experiments:

- one initialized with the original algorithm from [45] (*i.e.* Kaiming Uniform algorithm for convolutional and fully-connected layers with a parameter of $\sqrt{5}$) and
- one initialized using the weights and bias of the convolutional layers of the CAE pre-trained on NeuroVault dataset.

The CNNs were trained using the Adam optimizer with different learning rates depending on the dataset to classify. We used the cross-entropy loss function for training the classifier. Both were implemented in PyTorch.

2.5.1 Hyperparameters optimisation

As described in Fig. 1, all classification datasets were split into three disjoint subsets: train, test and validation with respectively 60%, 20% and 20% of the subjects of the dataset to avoid any data leakage (see [46, 47]). To select the best hyperparameters for each dataset and each type of initialization, we evaluated the performance of each model using the train set and the validation set.

For each type of classifier (initialized with default algorithm and pre-trained), we refined and optimised the hyperparameters using the largest datasets (Large BrainPedia and HCP). However, the large amount of training data makes it computationally extremely costly to perform a full grid-search. We therefore limited our research to predefined values of batch sizes (32 or 64), number of epochs (200, 500 or 1000) and model architectures (4 layers or 5 layers). All batch sizes and architectures were tested for each type of classifier and each dataset. For choosing the number of epochs, we trained our classifier for 200 and 500 epochs on the HCP dataset and 500 and 1000 for the BrainPedia dataset, for which we realized after some tests with 200 epochs that the classification task required more training time. Regarding the learning rate, classifiers trained to classify HCP datasets were optimised with a learning rate of 1e-04. However, for BrainPedia datasets, we tried two different learning rates (1e-04 and 1e-05) due to the very low performance of some models.

2.5.2 Evaluation of performance

The models with optimal hyperparameters were then evaluated on the test set. The performance of each model was measured using several metrics: accuracy (Acc), precision (P), recall (R) and F1-macro score (F1). All metrics were implemented in scikit-learn [39] with default parameters, except for F1-score for which the "average" parameter was specified with "macro" to deal with multi-class classification.

2.6 Benefits of self-taught learning and impact of different factors

To investigate the benefits of self-taught learning for neuroimaging data, different brain decoding experiments were compared. In all situations, after optimizing the hyperparameters of the model with default initialization and those of the model with fine-tuned weights, we trained these optimized models using the train set and evaluated the performance with the test set.

The HCP dataset was used to compare the performance of the models for the task of decoding on a homogeneous dataset (*i.e.* from a single study). We studied the impact of two factors on the classification: sample size and number of classes (*i.e.* how we classified the data). For sample size, subsets of the global HCP dataset were created with different number of subjects: 50, 100, 200, 500. Each subset being a superset of the previous one. At each sample size, the performance of the models (default algorithm initialization or pre-trained CAEs) trained on the corresponding subset were compared by testing them on the global test set to get more comparable results.

For the classification task, 3 types of classification were investigated. First, the 'contrast classification' task consisted in the assignment of the contrast associated with the statistic map (23 different contrasts). Another classification task was the 'task classification' in which multiple statistic maps corresponding to different contrasts were gathered into tasks (7 tasks with multiple contrasts per task). The last classification task was the 'one contrast task classification'. This time, we selected one type of contrast per task and classified the tasks using only the statistic maps corresponding to the selected contrast (7 tasks with one contrast per task). The selected contrasts were '2BKPLACE', 'FACES', 'PUNISH', 'REL', 'RH', 'STORY' and 'TOM' representing respectively the tasks 'WM', 'EMOTION', 'GAMBLING', 'RELATIONAL', 'MOTOR', 'LANGUAGE', 'SOCIAL'. The dataset used for this classification task was thus smaller than the others (only one map per task per subject).

To study the benefits of self-taught learning on a heterogeneous dataset (i.e. from multiple studies), we used the BrainPedia datasets. For these datasets, only one classification task was performed: classification of mental concepts available in NeuroVault metadata. However, we studied the impact of sample size and heterogeneity by comparing the

performance of the models between the Large BrainPedia dataset (large and very heterogeneous dataset) and the Small BrainPedia dataset (smaller and less heterogeneous as this dataset included fewer studies).

3 Results

3.1 AutoEncoder performance

Reconstruction performance of the CAEs are presented in Table 2. When comparing the two CAE architectures (4-layers vs 5-layers) trained on NeuroVault dataset, the mean correlations between original and reconstructed maps are better for the 4-layers architecture (86.9% vs 77.8%). These results suggest that the reconstruction capabilities of the CAEs are highly dependant on the model architecture and the size of the latent space. Figure 3 shows the reconstruction of a statistic map randomly drawn from the NeuroVault test dataset with the two CAE architectures. With the 4-layers architecture, details of the map are better reconstructed than with the 5-layers architecture (see the green square on the map). This is due to the level of compression of the data that is higher in the 5-layers CAE and that learn only the most useful features without putting a lot of efforts in learning specific details. Both models were used as pre-trained model for classification to see if the benefits of the CAEs were related to their reconstruction performance.

Table 2: Reconstruction performance of the CAE depending on model architecture and training set. Values are the mean Pearson's correlation coefficients (standard error of the mean).

Model	**4-layers** *Latent space 18,432*	**5-layers** *Latent space 4,096*
Correlation (*std error*)	86.9 (*0.18*)	77.8 (*0.23*)

3.2 Hyperparameters optimisation for classifiers

The best hyperparameters and corresponding performance can be found on Table 3.

Table 3: Hyperparameters chosen for each dataset and corresponding performance of the classifier on the validation set of the dataset.

Dataset	**Initialization**	**Model**	**Epochs**	**Batch**	**Accuracy (%)**	**F1 (%)**
HCP	Default algorithm	4-layers	500	64	91.3	91.2
	Pre-trained CAE	5-layers	200	64	91.5	91.6
BrainPedia	Default algorithm	4-layers	500	64	79.1	78.1
	Pre-trained CAE	5-layers	500	32	80.5	79.5

3.2.1 Choice of hyperparameters for HCP dataset

Performance of the different models trained with the different hyperparameters can be found in Supplementary Table S2. We selected one set of hyperparameters for each type of initialization (default and pre-trained) and used these for all subsequent experiments. For the default algorithm initialization, the best model had 4 layers and was trained with a batch size of 32 for 500 epochs. This model achieved an accuracy of 91.1% on the validation dataset. For the pre-trained CAE initialization, the best model had 5 layers and was trained with a batch size of 64 for 200 epochs (accuracy of 91.5%).

3.2.2 Choice of hyperparameters for BrainPedia dataset

Results for all sets of hyperparameters are available in Supplementary Table S3. For the default algorithm initialization, the model who achieved the best performance had 4 layers and a batch size of 32 for 1000 epochs with a learning rate of 1e-05. This model classified the validation subset of BrainPedia dataset with an accuracy of 75.5%. The performance of the pre-trained CAE was the best using a 5-layer architecture, a batch size of 64 and a training time of 1000 epochs with a learning rate of 1e-04.

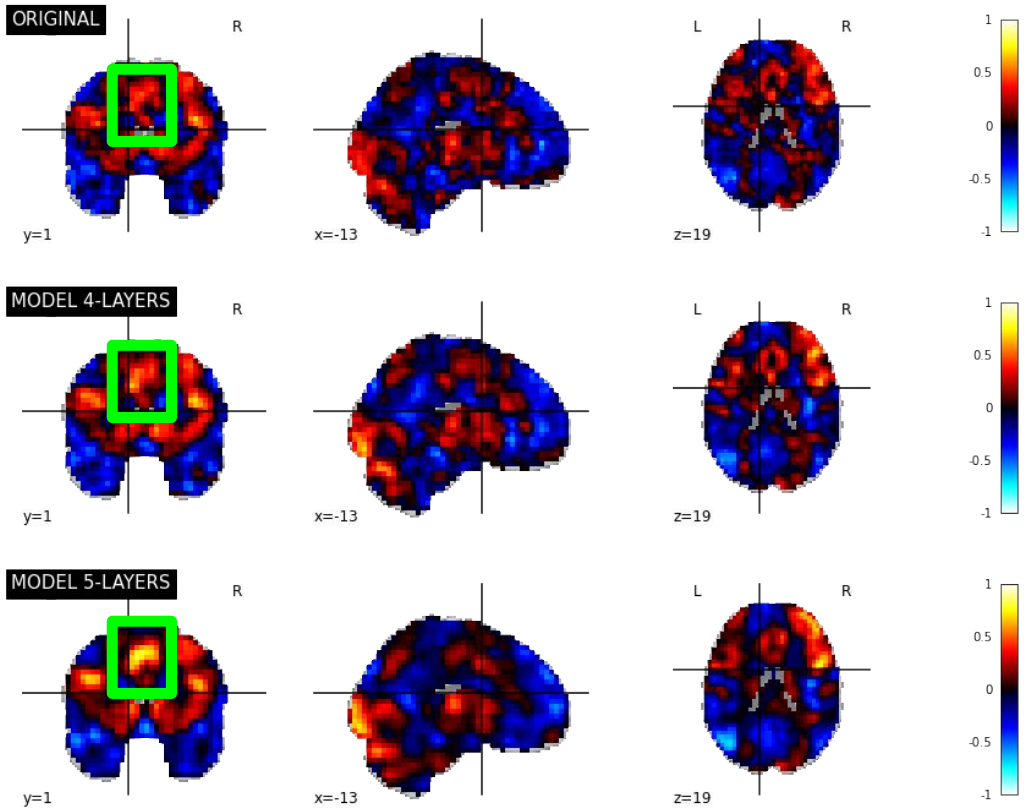

Figure 3: Original version and reconstruction of a randomly drawn statistic map of NeuroVault test dataset (image ID: 109) with the two CAEs (4-layers and 5-layers).

3.3 Benefits of self-taught learning on a homogeneous dataset

Benefits of self-taught learning on the HCP (homogeneous) datasets were assessed by comparing the performance on the test set of the best models selected in the previous section. Table 4 summarizes the results for the different classification experiments on the HCP datasets.

3.3.1 Impact of sample size

For all classification experiments, the size of the training set (in terms of number of subjects) had a strong impact on the benefits of self-taught learning. With 50 subjects, the performance of the pre-trained CAE outpassed the performances of the classifier initialized with the default algorithm in all our experiments (improvements of 0.9% to 5% of accuracies between default and pre-trained models). When sample size increased, this improvement reduced and there was sometimes no improvement at all. If we focus on contrast classification (Figure 4), which is the hardest classification task between the three presented here due to the higher number of classes, the difference between the performance of the two classifiers decreased with sample size (accuracies of 91.4% and 92.2% respectively for default initialization and pre-trained model respectively for N=500 which corresponds to an improvement of 0.8% compared to 3% for N=100 or 1.5% for N=200). For the global dataset (N=700), there was no more improvement of performance since default and pre-trained models had similar performance (0.2% of difference between accuracies).

3.3.2 Impact of the target classification task

For more simple classification experiments (*i.e.* with less classes to separate), pre-training was not always useful. In these experiments, performance were already nearly perfect (accuracies close to 1) and difficult to improve. For large sample sizes ($N \geq 100$), performance were close (difference between accuracies lower than 0.2%) between

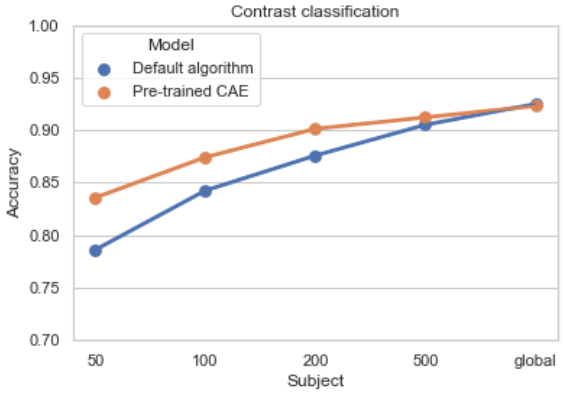

Figure 4: Accuracies on contrast classification with the HCP dataset for the models initialized with default algorithm (blue) and pre-trained CAE (orange). Pre-training improves contrast classification performance for small sample sizes and at a lower level of improvement, also for large sample sizes.

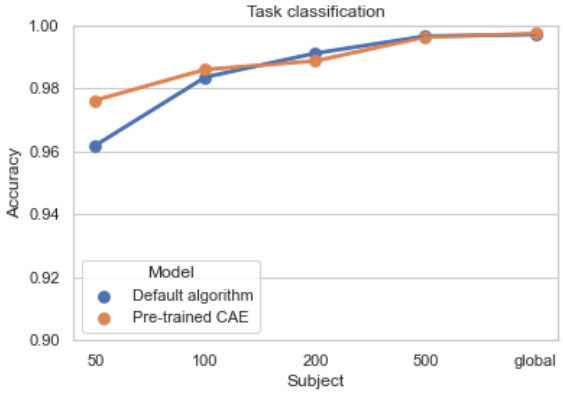

Figure 5: Accuracies on task classification with the HCP dataset for the models initialized with default algorithm (blue) and pre-trained CAE (orange). Pre-training improves task classification performance for all sample sizes but sample sizes does not have a huge influence on the level of improvement.

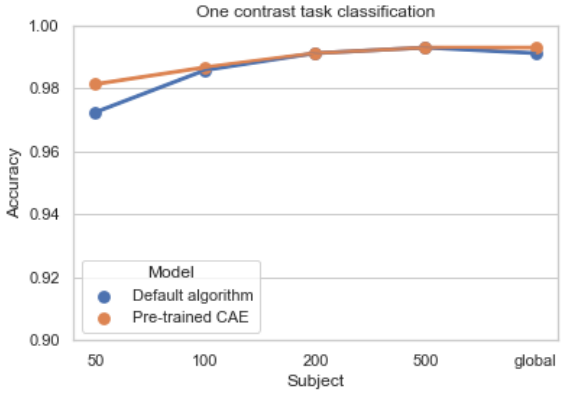

Figure 6: Accuracies on one contrast task classification with the HCP dataset for the models initialized with default algorithm (blue) and pre-trained CAE (orange). Pre-training does not always improve one-contrast task classification performance: for some sample sizes, pre-training and default initialization give very similar results.

Table 4: Classification performance on HCP datasets of models initialized with default algorithm vs with the weights of a pre-trained CAE. DA: Default Algorithm initialization ; PT: Pre-Training initialization.

Sample	**50**		**100**		**200**		**500**		**Global**	
Init.	**DA**	**PT**	**DA**	**PT**	**DA**	**PT**	**DA**	**PT**	**DA**	**PT**
Contrast classification (*23 classes*)										
Acc. (%)	78.6	83.5	84.3	87.4	88.5	90.1	91.4	92.2	92.5	92.3
Task classification (*7 classes, multiple contrasts per class*)										
Acc. (%)	96.2	97.6	98.3	98.6	99.1	98.9	99.6	99.6	99.7	99.7
One constrast task classification (*7 classes, one contrast per class*)										
Acc. (%)	97.2	98.1	98.6	98.6	99.1	99.1	99.3	99.3	99.1	99.3

models initialized with default algorithm and pre-trained models (see Figures 5 and 6). However, for smaller sample sizes (N=50), pre-training improved classification – similarly to what had been shown for more complex tasks – with accuracies of the pre-trained models higher than default models of 1.4% and 0.9% for task classification and one contrast task classification respectively. These results suggest that pre-training can be beneficial in complex situations with few training samples or difficult classification tasks.

3.4 Benefits of self-taught learning on a heterogeneous dataset

Table 5 summarizes the results for the classification of mental concepts on the small and the large BrainPedia datasets. These results are illustrated in Figure 7.

Table 5: Classification performances on BrainPedia datasets of models initialized with default algorithm vs with the weights of a pre-trained CAE. DA: Default Algorithm initialization ; PT: Pre-Training initialization

Dataset	**Small BrainPedia**		**Large BrainPedia**	
Init.	**DA**	**PT**	**DA**	**PT**
Acc.	72.1	72.6	79.8	81.2
F1-score	66.0	72.1	76.6	79.8

On a the small BrainPedia dataset, pre-training improved the performance of the classifier. When looking at the accuracy, respectively 72.1% and 72.6% for the classifier initialized with the default algorithm and the pre-trained classifier, the difference was small (0.5% of improvement). But in this case, the F1-score is a better metric to assess the performance. Indeed, this metric focuses more on classification errors and is a better indicator of performance when classes are imbalanced, which is the case in this dataset in which some classes are more represented than others (*e.g.* in the small BrainPedia training set, 200 maps correspond to the class "visual words, language, visual" whereas only 19 are in the class "left foot, visual"). When focusing on this metric, the pre-trained classifier performance were markedly higher than the ones of the classifier with default initialization (6.1% of improvement in F1-score).

On the large BrainPedia (heterogeneous) dataset, performance also increased with pre-training. Accuracy and F1-score were higher for the the pre-trained model (F1-score of 76.6% against 79.8% for the model with default initialization) even if the sample size of the dataset was higher and more classes were represented. Indeed, the classification task was also more complex for this dataset since data were separated into 36 classes instead of 30 for Small BrainPedia due to the presence of maps from other studies in the dataset.

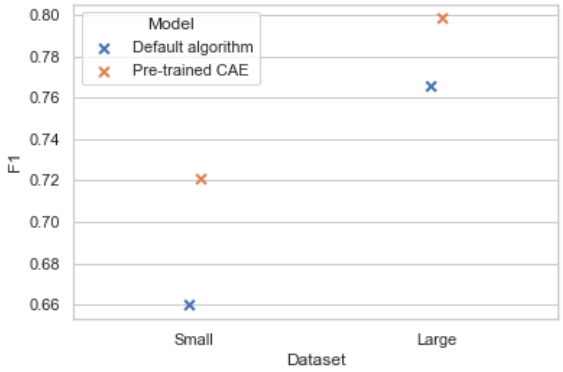

Figure 7: F1-scores of the classification of mental concepts on BrainPedia datasets (Small and Large) for the models initialized with default algorithm (blue) and pre-trained CAE (orange). Pre-training improves classification performance, in particular for the small dataset.

4 Discussion

4.1 Summary of the results

In this work, we showed the benefits of self-taught learning on two large public datasets with different sample sizes and classification tasks. In all cases, pre-training a classifier with an unsupervised task (in our case: reconstruction) was beneficial but the level of improvement varied depending on the classification task and the size of the training dataset.

When sample sizes were small, pre-training always improved the classification performance, regardless of whether the dataset was homogeneous or heterogeneous and of the complexity of the classification task. In medical imaging, where the dimensions of the data are often very large and few samples are typically available due to high financial and human costs, learning a good representation of the data can be very difficult [12, 48]. Unsupervised pre-training is thus helpful by initializing the weights of the CNN to preserve the (brain) structure learned by the autoencoder, and facilitate the learning process. However, when the sample size increased, benefits were less remarkable since the amount of available training data was probably sufficient to learn a good representation.

This observation can also be made for classification tasks. When trying to classify the data in a small number of classes, performance of the pre-trained classifier were better but not with a high improvement of performance, even for small sample sizes (*e.g.* 100 subjects for task classification). But when trying to separate data into more classes, for a more fine-grained classification, the representation learned during the pre-training was beneficial to improve the performance.

Another benefit of self-taught learning we found was the reduction of the training time. Performance of the pre-trained classifier were better even with less training epochs. This was the case for HCP dataset results which were computed for 500 epochs for the default algorithm and 200 epochs for the pre-trained model. This is confirmed by [49] in which researchers assumed that the pre-trained models remain in the same basin of the loss function when trained on new data and since the weights are already initialized close to a good representation of data, less epochs are necessary to adapt this representation for classification.

Architectures of the models also had an impact on the benefits of self-taught learning. With both datasets, pre-trained models performed better using the 5-layers architecture and default algorithm models with the 4-layers one. This effect was studied by [33] who showed that, while unsupervised pre-training helps for deep networks with more layers, it appears to hurt for too small networks. The size of the latent space of the CAE with 5-layers being almost 5 times smaller that the 4-layers one, it suggests that only a small subset of features of the input are relevant for predicting the class label.

However, the classification accuracies of the pre-trained models were not related to the reconstruction performance of the CAE since the 4-layers CAE reconstructs maps with better precision than the 5-layers one. This confirms that the features learned by the 4-layers CAE for reconstruction were not all useful for classification and focusing on a smaller number of features with the 5-layers one facilitates the learning process.

4.2 Limitations and perpectives

Due to the high computational time required to train a model, we only compared two model architectures (4 and 5-layers). Other types of architectures with different number of fully-connected or convolutional layers could have been tested to see the effect of other latent space sizes as it was done in [33]. We also chose to evaluate the impact of whole transfer learning (*i.e.* transferring the weights of all convolutional layers) but it could be interesting to see if the effect is the same when transferring only the weights of the first layers, to see which part of the compressed data are the most impactful. For the same reason, we also chose to compare our models based on a single training set to identify the optimal hyperparameters. This procedure is not ideal to measure the ability to generalize to new data and there might be some bias in the performance related to the training and test sets chosen [50]. This could have been avoided by using a nested cross-validation scheme but the computational cost would have markedly inscreased.

The main limitation of our work is the classification experiments and datasets we chose. In fMRI, the number of possible labels and thus, classification tasks is very high due to the lack of consensus in the field [30]. For our experiments, we used the labels provided in NeuroVault metadata that were used in the original studies [16, 38]. We chose to compare multiple type of classification of the HCP dataset that could be used in the field or that were made in other studies [24, 23]. But for BrainPedia datasets, a multi-label decoding was performed in the original study since several concepts are relevant for most maps. Labels we had access to were then the list of labels associated with each map. To be able to compare our results with those of the homogeneous dataset (HCP), we chose to classify these as unique labels, which is less complex and less precise in practice. This type of issue is due to the lack of harmonization in the way tasks and cognitive processes are defined. Using ontologies such as Cognitive Atlas [30], NeuroVault annotations could be harmonized and enriched, as it was done by [29].

In neuroimaging, lots of sources of variability can impact the results of an experiment and the generalizability of these results. Here, we investigated the generalizability of our model by assessing the benefits of pre-training on a heterogeneous dataset (BrainPedia). While this dataset was heterogeneous in terms of the studies that were included, all maps were obtained using the same processing pipeline. Multiple studies have shown that the exact pipeline used to obtain an fMRI result can have a non-negligeable impact on fMRI statistic maps [51, 52]. In the future, using a more variable dataset with images from different studies but also processed using different pipelines would be of great interest. [10] tried to compare the performance of different classifiers trained on fMRI 3D volumes series obtained with various scenarios of minimal preprocessing pipelines. A similar experiment was recently made by [53] who found that preprocessing pipeline selection can impact the performance of a supervised classifier. Comparing the adaptation capacities of models on volumes preprocessed with different pipelines could be also interesting to evaluate the impact of analytical variability on deep learning with fMRI.

In this context, using unsupervised models could allow us to build a space capturing the similarities and differences of statistic maps, *i.e.* to learn a robust latent representation of the important features of statistic maps in a specific context. By adding other constraints to this latent space and/or choosing an adapted pre-training dataset, we could use this for other purposes than brain decoding. For instance, building a space that captures the analytical variability in statistic maps could help us understand the difference between the pipelines but also identify the more robust pipelines. Future works will focus on building such a space with specific constraints to evaluate distance between different pipelines.

5 Conclusion

In this study, we compared the benefits of a self-taught learning framework in the task of classifying 3D fMRI statistic maps. We showed that unsupervised pre-training improves the performance of classification in experiments with few training samples and complex classification tasks, which is a very common setup in fMRI studies.

6 Acknowledgements

This work was partially funded by Region Bretagne (ARED MAPIS) and Agence Nationale pour la Recherche for the programm of doctoral contracts in artificial intelligence (project ANR-20-THIA-0018). We thank Gregory Kiar who worked on a preliminary version of the autoencoder based on NeuroVault data.

References

[1] Anees Abrol, Zening Fu, Mustafa Salman, Rogers Silva, Yuhui Du, Sergey Plis, and Vince Calhoun. Deep learning encodes robust discriminative neuroimaging representations to outperform standard machine learning. *Nature Communications*, 2021.

[2] Wutao Yin, Longhai Li, and Fang-Xiang Wu. Deep learning for brain disorder diagnosis based on fMRI images. *Neurocomputing*, 469:332–345, 2022.

[3] Orhan Firat, Like Oztekin, and Fatos T. Yarman Vural. Deep learning for brain decoding. In *2014 IEEE International Conference on Image Processing (ICIP)*, pages 2784–2788. IEEE, 2014.

[4] Hanh Vu, Hyun-Chul Kim, and Jong-Hwan Lee. 3d convolutional neural network for feature extraction and classification of fMRI volumes. In *2018 International Workshop on Pattern Recognition in Neuroimaging (PRNI)*, pages 1–4, 2018.

[5] Jinlong Hu, Yuezhen Kuang, Bin Liao, Lijie Cao, Shoubin Dong, and Ping Li. A multichannel 2d convolutional neural network model for task-evoked fMRI data classification. *Computational Intelligence and Neuroscience*, 2019.

[6] Muhammad Bilal Qureshi, Laraib Azad, Muhammad Shuaib Qureshi, Sheraz Aslam, Ayman Aljarbouh, and Muhammad Fayaz. Brain decoding using fMRI images for multiple subjects through deep learning. *Computational and Mathematical Methods in Medicine*, 2022.

[7] Sotetsu Koyamada, Yumi Shikauchi, Ken Nakae, Masanori Koyama, and Shin Ishii. Deep learning of fmri big data: a novel approach to subject-transfer decoding. *arXiv preprint arXiv: 1502.00093*, 2015.

[8] Xiaoxiao Wang, Xiao Liang, Zhoufan Jiang, Benedictor A. Nguchu, Yawen Zhou, Yanming Wang, Huijuan Wang, Yu Li, Yuying Zhu, Feng Wu, Jia-Hong Gao, and Bensheng Qiu. Decoding and mapping task states of the human brain via deep learning. *Human Brain Mapping*, 2020.

[9] Xiaojie Huang, Jun Xiao, and Chao Wu. Design of deep learning model for task-evoked fMRI data classification. *Computational Intelligence and Neuroscience*, 2021.

[10] Hanh Vu, Hyun-Chul Kim, Minyoung Jung, and Jong-Hwan Lee. fMRI volume classification using a 3d convolutional neural network robust to shifted and scaled neuronal activations. *NeuroImage*, 2020.

[11] Kanghan Oh, Woosung Kim, Guangfan Shen, Yanhong Piao, Nam-In Kang, Il-Seok Oh, and Young Chul Chung. Classification of schizophrenia and normal controls using 3d convolutional neural network and outcome visualization. *Schizophrenia Research*, 212:186–195, 2019.

[12] Armin W. Thomas, Christopher Ré, and Russell A. Poldrack. Challenges for cognitive decoding using deep learning methods. *preprint arXiv:2108.06896*, 2021.

[13] Junghwan Cho, Kyewook Lee, Ellie Shin, Garry Choy, and Synho Do. How much data is needed to train a medical image deep learning system to achieve necessary high accuracy? *arXiv preprint arXiv:1511.06348*, 2015.

[14] Russell A. Poldrack, Chris I. Baker, Joke Durnez, Krzysztof J. Gorgolewski, Paul M. Matthews, Marcus R. Munafò, Thomas E. Nichols, Jean-Baptiste Poline, Edward Vul, and Tal Yarkoni. Scanning the horizon: towards transparent and reproducible neuroimaging research. *Nature Reviews Neuroscience*, 2017.

[15] Katherine S. Button, John P. A. Ioannidis, Claire Mokrysz, Brian A. Nosek, Jonathan Flint, Emma S. J. Robinson, and Marcus R. Munafò. Power failure: why small sample size undermines the reliability of neuroscience. *Nature Reviews Neuroscience*, 14, 2013.

[16] David C. Van Essen, Stephen M. Smith, Deanna M. Barch, Timothy E.J. Behrens, Essa Yacoub, and Kamil Ugurbil. The WU-minn human connectome project: An overview. *Mapping the Connectome*, 2013.

[17] Myriam Bontonou, Giulia Lioi, Nicolas Farrugia, and Vincent Gripon. Few-shot decoding of brain activation maps. *2021 29th European Signal Processing Conference (EUSIPCO)*, 2021.

[18] Sunao Yotsutsuji, Miaomei Lei, and Hiroyuki Akama. Evaluation of task fMRI decoding with deep learning on a small sample dataset. *Frontiers in neuroinformatics*, 2021.

[19] Peiye Zhuang, Alexander G Schwing, and Oluwasanmi Koyejo. Fmri data augmentation via synthesis. In *2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019)*, pages 1783–1787. IEEE, 2019.

[20] Arthur Mensch, Julien Mairal, Bertrand Thirion, and Gaël Varoquaux. Extracting representations of cognition across neuroimaging studies improves brain decoding. *PLOS Computational Biology*, 2014.

[21] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 2010.

[22] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[23] Armin W. Thomas, Ulman Lindenberger, Wojciech Samek, and Klaus-Robert Müller. Evaluating deep transfer learning for whole-brain cognitive decoding. *arXiv preprint arXiv: 2111.01562*, 2021.

[24] Yufei Gao, Yameng Zhang, Hailing Wang, Xiaojuan Guo, and Jiacai Zhang. Decoding behavior tasks from brain activity using deep transfer learning. *IEEE Access*, 2019.

[25] Michele Svanera, Mattia Savardi, Sergio Benini, Alberto Signoroni, Gal Raz, Talma Hendler, Lars Muckli, Rainer Goebel, and Giancarlo Valente. Transfer learning of deep neural network representations for fMRI decoding. *Journal of Neuroscience Methods*, 2019.

[26] Russell A Poldrack and Krzysztof J Gorgolewski. Making big data open: data sharing in neuroimaging. *Nature Neuroscience*, 2014.

[27] Christopher J Markiewicz, Krzysztof J Gorgolewski, Franklin Feingold, Ross Blair, Yaroslav O Halchenko, Eric Miller, Nell Hardcastle, Joe Wexler, Oscar Esteban, Mathias Goncavles, Anita Jwa, and Russell Poldrack. The OpenNeuro resource for sharing of neuroscience data. *eLife*, 2021.

[28] Krzysztof J. Gorgolewski, Gael Varoquaux, Gabriel Rivera, Yannick Schwarz, Satrajit S. Ghosh, Camille Maumet, Vanessa V. Sochat, Thomas E. Nichols, Russell A. Poldrack, Jean-Baptiste Poline, Tal Yarkoni, and Daniel S. Margulies. NeuroVault.org: a web-based repository for collecting and sharing unthresholded statistical maps of the human brain. *Frontiers in Neuroinformatics*, 2015.

[29] Romuald Menuet, Raphael Meudec, Jérôme Dockès, Gael Varoquaux, and Bertrand Thirion. Comprehensive decoding mental processes from web repositories of functional brain images. *Scientific Reports*, 2022.

[30] Russell A. Poldrack, Aniket Kittur, Donald Kalar, Eric Miller, Christian Seppa, Yolanda Gil, D. Stott Parker, Fred W. Sabb, and Robert M. Bilder. The cognitive atlas: Toward a knowledge foundation for cognitive neuroscience. *Frontiers in Neuroinformatics*, 2011.

[31] Rajat Raina, Alexis Battle, Honglak Lee, Benjamin Packer, and Andrew Y. Ng. Self-taught learning: transfer learning from unlabeled data. In *Proceedings of the 24th international conference on Machine learning*, pages 759–766. ACM Press, 2007.

[32] Hua Wang, Feiping Nie, and Heng Huang. Robust and discriminative self-taught learning. *Proceedings of the 30th International Conference on Machine Learning*, 2013.

[33] Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. Why does unsupervised pre-training help deep learning? *The Journal of Machine Learning Research*, 2010.

[34] Seyedmehdi Orouji, Vincent Taschereau-Dumouchel, Aurelio Cortese, Brian Odegaard, Cody Cushing, Mouslim Cherkaoui, Mitsuo Kawato, Hakwan Lau, and Megan AK Peters. " task-relevant autoencoding" enhances machine learning for human neuroscience. *arXiv preprint arXiv:2208.08478*, 2022.

[35] Elodie Germani, Elisa Fromont, and Camille Maumet. On the benefits of self-taught learning for brain decoding - data, 2022.

[36] Collection nº4337. NeuroVault Collection nº4337. `https://identifiers.org/neurovault.collection:4337`. Accessed: 2022-01-19.

[37] Collection nº1952. NeuroVault Collection nº1952. `https://identifiers.org/neurovault.collection:1952`. Accessed: 2022-01-19.

[38] Gaël Varoquaux, Yannick Schwartz, Russell A. Poldrack, Baptiste Gauthier, Danilo Bzdok, Jean-Baptiste Poline, and Bertrand Thirion. Atlases of cognition with large-scale human brain mapping. *PLOS Computational Biology*, 2018.

[39] Alexandre Abraham, Fabian Pedregosa, Michael Eickenberg, Philippe Gervais, Andreas Mueller, Jean Kossaifi, Alexandre Gramfort, Bertrand Thirion, and Gael Varoquaux. Machine learning for neuroimaging with scikit-learn. *Frontiers in Neuroinformatics*, 2014.

[40] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. 2019.

[41] Shane Cook. *CUDA Programming: A Developer's Guide to Parallel Computing with GPUs*. 2012.

[42] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015.

[43] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[44] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 2020.

[45] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[46] Gaël Varoquaux and Veronika Cheplygina. Machine learning for medical imaging: methodological failures and recommendations for the future. *npj Digital Medicine*, 2022.

[47] Sayash Kapoor and Arvind Narayanan. Leakage and the reproducibility crisis in ml-based science. *arXiv preprint arXiv:2207.07048*, 2022.

[48] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 843–852, 2017.

[49] Behnam Neyshabur, Hanie Sedghi, and Chiyuan Zhang. What is being transferred in transfer learning? *Advances in neural information processing systems*, 33:512–523, 2020.

[50] Gaël Varoquaux, Pradeep Reddy Raamana, Denis A. Engemann, Andrés Hoyos-Idrobo, Yannick Schwartz, and Bertrand Thirion. Assessing and tuning brain decoders: Cross-validation, caveats, and guidelines. *NeuroImage*, 2017.

[51] Joshua Carp. On the plurality of (methodological) worlds: estimating the analytic flexibility of fMRI experiments. *Frontiers in Neuroscience*.

[52] Rotem Botvinik-Nezer, Felix Holzmeister, Colin F Camerer, Anna Dreber, Juergen Huber, Magnus Johannesson, Michael Kirchler, Roni Iwanir, Jeanette A Mumford, R Alison Adcock, et al. Variability in the analysis of a single neuroimaging dataset by many teams. *Nature*, 582(7810):84–88, 2020.

[53] Xinhui Li, Alex Fedorov, Mrinal Mathur, Anees Abrol, Gregory Kiar, Sergey Plis, and Vince Calhoun. Pipeline-invariant representation learning for neuroimaging. *arXiv preprint arXiv:2208.12909*, 2022.