Data Representations and Ensemble Deep Learning Networks for Functional Neuroimaging Datasets

Morgan Cambareri[1,2(✉)] and Farshid Alizadeh-Shabdiz[3]

[1] Biomedical Engineering Department, College of Engineering, Boston University, Boston, MA, USA
mcambareri@mgh.harvard.edu

[2] Center for Neurotechnology and Neurorecovery, Massachusetts General Hospital, Boston, MA, USA

[3] Computer Science Department, MET College, Boston University, Boston, MA, USA

Abstract. This project was designed to test the predictive accuracy of combining two separate data representations of resting state functional magnetic resonance imaging (rs-fMRI) data into an ensemble deep learning architecture. Three main data representations of the same neuroimaging dataset were tested by building associated deep learning architectures and testing their accuracy in predicting if the neuroimaging data originated from healthy controls or from individuals diagnosed with autism spectrum disorder (ASD). The three data representations were 2D correlation matrices derived from time courses extracted from the blood-oxygen-level-dependent (BOLD) signal within the brain, a graph tensor representation of the same connectivity data, and a 3D profile of the posterior cingulate cortex's (PCC) connectivity across the brain. These data representations were fed into a 2D Convolutional Neural Network (2D-CNN), a Graph Convolutional Neural Network (GCN), and a 3D Convolutional Neural Network (3D-CNN) respectively. Finally, the 2D-CNN and the 3D-CNN were chosen to combine into a single ensemble model to test the hypothesis that the combination of two different representations of the same data can improve upon the individual models. This ensemble model performed better than both the 2D-CNN and 3D-CNN models individually when validated using 5-fold cross-validation and 5 × 2-fold cross validation. However, this improvement was only statistically significant for the comparison with the 3D-CNN model ($p = 0.0224$). This result suggests that using combinations of multiple data representations may improve model accuracy when using functional neuroimaging data in deep learning applications.

Keywords: Functional Neuroimaging · Ensemble Learning · Deep Learning

T. Zlateva and G. Tuparov (Eds.): CSECS 2023, LNICST 514, pp. 204–211, 2023.
https://doi.org/10.1007/978-3-031-44668-9_16

1 Introduction

Deep learning applications for functional neuroimaging datasets, such as resting state functional magnetic resonance imaging (rs-fMRI) data, come with a variety of challenges. These datasets are generally large, have high dimensionality, complex spatiotemporal dynamics, and have a low sample size, which makes training deep learning models more challenging. Despite these challenges deep learning applications for neuroimaging datasets are of great interest in psychiatry and neurology for the potential of these models to identify patient diagnoses, predict prognoses, and individualize treatment [1, 2].

Currently machine learning applications for functional neuroimaging datasets have not demonstrated sufficient accuracy to make their way into clinical practice. To achieve this goal, more research must be done to understand the optimal representations of the input data and model architecture for improving prediction accuracy. Input of the full 4-dimensional rs-FMRI datasets into a deep learning model is not feasible, therefore it is important to find ways to dimensionally reduce these datasets to a more usable format for training while preserving important features.

In this project we investigated three different data representations of resting state functional magnetic resonance imaging (rs-fMRI) data taken from the Autism Brain Imaging Data Exchange (ABIDE) as input into their associated deep learning architectures [3]. We assessed each architecture's accuracy in classifying brain scans into a healthy control (HC) group or an Autism Spectrum Disorder (ASD) group. We then combined two of these models to test the hypothesis that an ensemble deep learning model with inputs from two representations of the same data will result in an overall binary classification accuracy that improves upon the individual models.

2 Methods

2.1 Data Preprocessing

For this analysis the ABIDE I dataset was directly accessed through the Nilearn Python package [4]. The dataset used contains aggregated data from 403 ASD subjects and 468 healthy controls obtained at 16 different scanning sites and was preprocessed using the configurable pipeline for the analysis of connectomes (cpac). Each scan was represented as a 4D dataset (3D volumetric scans with 2 mm^3 isotropic resolution taken every 2 s). The steps taken to ensure clean data included slice timing correction, realignment to correct for motion, and down sampling to 3 mm^3 isotropic resolution. Additional signal cleaning steps like global signal regression, and a low pass filtering of the signal to below 0.1 Hz were also performed. After preprocessing the rs-fMRI data was then transformed into different representations to act as the inputs to three separate deep learning architectures.

2.2 Data Representations

2D - Correlation Matrix Data Representation: For this representation, the data was transformed into 2D correlation matrices as input into a 2D convolutional neural network

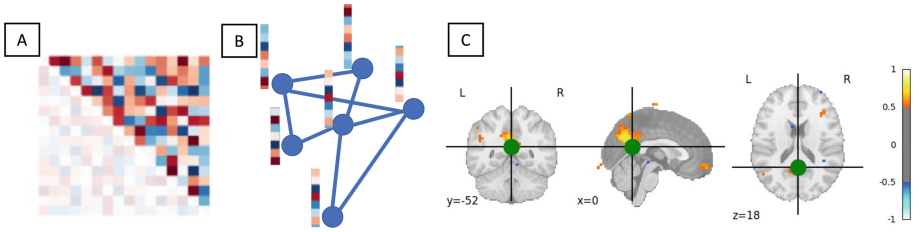

Fig. 1. Data representations of the rs-fMRI data. A) The correlation matrix representation where the brain signals from each ROI are correlated pairwise with all other ROIS to create a 2D correlation matrix of Pearson correlation values from -1 to 1. B) Graph tensor input where each subject scan is represented by a graph composed of edges, nodes and node features. C) A 3D Pearson correlation distribution with the PCC which is known to have altered functional connectivity in individuals with ASD [5].

(CNN) architecture. The 2D correlation matrices are a spatially and temporally simplified form of a rs-fMRI dataset which is created by performing pairwise Pearson correlations for every defined ROI time course (Fig. 1A). This was performed by first obtaining time courses from the resting state volumes using the Harvard oxford probabilistic atlas which defines 48 regions of interest (ROIs) across the cortical surface of the brain [5]. This 2D-CCN was later combined with the 3D-CNN to form an ensemble model.

Graph Tensor Data Representation: In this data representation the previous correlation matrices were converted to a graph data structure which consist of nodes and edges (Fig. 1B). The nodes and edges are defined from the correlation matrix where each ROI in the matrix is represented as a node with the correlation values for that ROI as the node features. The edges (or node connections) were made from the top 15 most correlated values for each node. Each subject's rs-fMRI data was therefore represented by a graph with 48 nodes, each having 47 features and 15 edges. This data was used as input into a graph convolutional network architecture [7].

3D Correlation Distribution Data Representation: For the third data representation the signal within the posterior cingulate cortex (PCC) in the brain was correlated with every other voxel in the volume. This gave us a spatial distribution of the functional connectivity of the PCC in the brain which is known to be altered in individuals with ASD [5] (Fig. 1C). Each dimension in this data (originally $60 \times 72 \times 60$ voxels) was downsampled by two to $30 \times 36 \times 30$ voxels due to memory constraints. This data was used as an input into a 3D-CNN architecture.

3 Results

3.1 2D-CNN Branch Architecture

For this branch the hyperparameters were tuned using the KerasTuner library's random search method. Initial hyperparameter tuning was performed to optimize the number of CNN layers. The architecture that performed the best after 15 epochs was a relatively

simple CNN with two 2D-CNN layers with ReLU activation each with 64 filters followed by batch normalization, a max pooling layer and then finally two dense layers with dropout (Fig. 3). The model with the best validation accuracy after 15 epochs had a learning rate of 3.293×10^{-5} with Adam as the chosen optimizer. 5-fold cross validation was performed to test the generalizability of the accuracy. An early stopping callback was added to minimize overfitting. The average accuracy for the 5-fold cross validation was 63.49% with a standard deviation of 1.72% (Fig. 2).

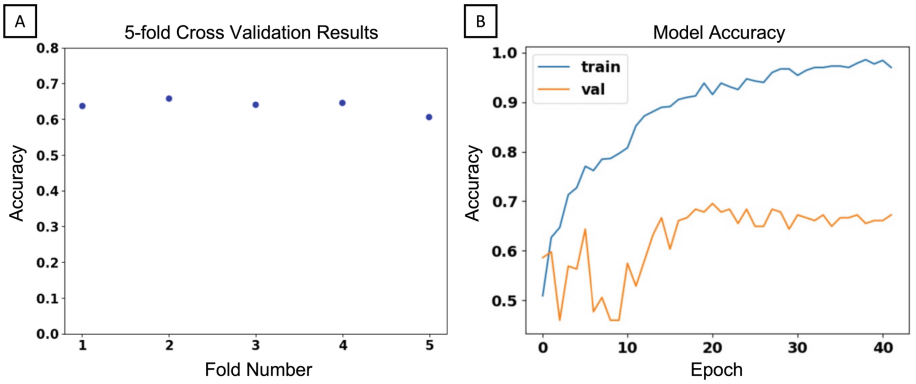

Fig. 2. Validation results of the 2D-CNN architecture. A) 5-fold cross validation accuracy across all folds. B) An example accuracy curve from one of the 5-fold cross validation iterations with early stopping. The average accuracy was found to be 63.49% with a standard deviation of 1.72%.

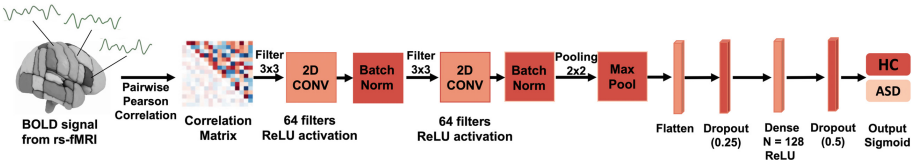

Fig. 3. 2D-CNN Architecture

3.2 GCN Branch Architecture

5-fold cross validation was used to choose the hyperparameters that had the best validation accuracy across all folds. From this tuning an architecture was chosen with 3 graph convolutional layers, each with ReLU activation and 64 layers followed by 2 dense layers both with ReLU activation and a final output layer with sigmoid activation (Fig. 5). Adam was chosen as the optimizer with a batch size of 150 samples and 0.0050 as the learning rate. These were all chosen by individually tuning the parameters in the architecture to achieve the highest validation accuracy. The average accuracy was found to be 60.30% with a standard deviation of 1.90% (Fig. 4).

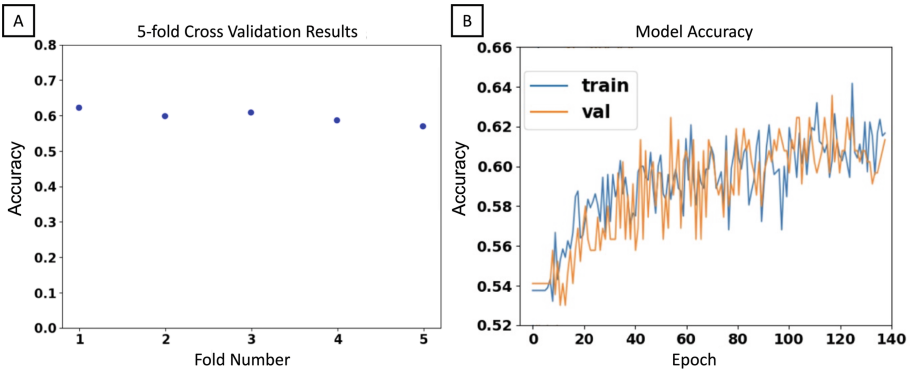

Fig. 4. 5-Fold cross validation results from the GCN model. A) The accuracy results from each of the 5-fold cross validations. B) An example accuracy curve from one of the 5-fold cross validation iterations with early stopping. The average accuracy across all folds was 60.30% with a standard deviation of 1.90%.

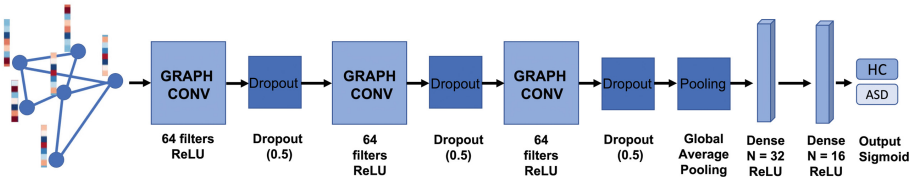

Fig. 5. Graph Convolutional Network Architecture

3.3 3D-CNN Branch

For this branch the hyperparameters were tuned using the KerasTuner library's random search method. An initial hyperparameter tuning was performed to decide on the number of 3D-CNN layers. We structured the beginning of the CNN model in blocks consisting of two 3D convolutional layers with 64 filters and $3 \times 3 \times 3$ kernels, followed by a MaxPool 3D layer and batch normalization. These blocks were followed by a flattening layer, 2 dense layers with ReLU activation functions, dropout, and a final sigmoid output layer (Fig. 7). Initial tuning was performed to choose the optimal number of CNN blocks and the type of optimizer based on validation accuracy of the model after 10 epochs. The model with the highest accuracy contained 1 block with stochastic gradient descent accelerated with Nesterov. Hyperparameter tuning continued for choosing the optimal learning rate, number of filters in the convolution blocks, the type of activation functions for CNN layers, dense layers, and the number of neurons in the dense layers. The final hyperparameters chosen were 15 filters for the first CNN block with ReLU activation, followed by a second CNN layer with 50 filters and ReLU activation. The dense layers contained 960 neurons and 630 neurons each with Tanh activation functions. The optimal learning rate was found to be 3.019×10^{-5}. 5-fold cross validation with early stopping was then performed to measure the final accuracy. The final 5-fold cross validation average accuracy was 57.06% with a standard deviation of 2.91% (Fig. 6).

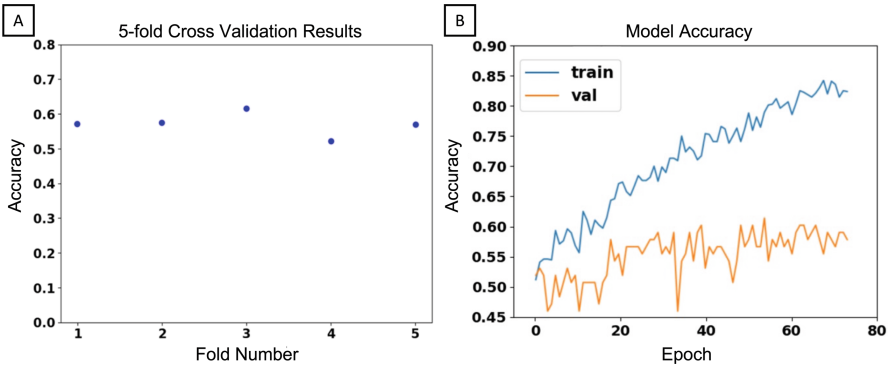

Fig. 6. 5-Fold Cross Validation Results from final 3D-CNN Branch. A) The accuracy results from each of the 5-fold cross validation iterations. B) An example accuracy curve from one of the 5-fold cross validation iterations with early stopping. The average accuracy across all folds was 57.06% with a standard deviation of 2.91%.

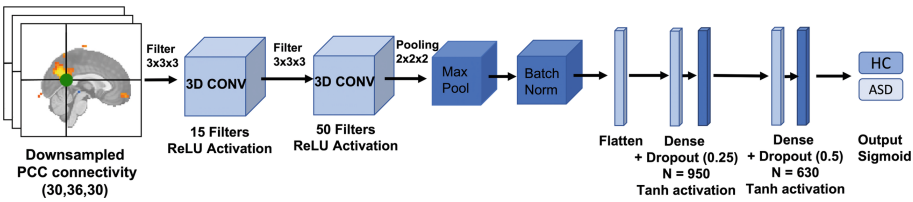

Fig. 7. 3D-CNN Architecture

3.4 Ensemble Architecture

Finally, the two CNN branches (2DCNN+3DCNN) were used to create an ensemble neural network which concatenated the outputs of the two models and added two dense layers before the final output layer. For each branch the hyperparameters were consistent with the original models, however the optimizer chosen was Adam and the learning rate was averaged between the two previous values (3.156×10^{-5}). 5-fold cross validation was run on a random sampling of the original data. The average final validation accuracy on all 5 folds was found to be to be 64.63% with a standard deviation of 1.62% (Figs. 8 and 9).

3.5 5 × 2-Fold Cross Validation

To test the significance between the ensemble learning model and the two original CNN models, a modified t-test was used to compare the accuracies of the models' 5 × 2-fold cross validation results [8]. Accuracies for each fold were acquired across the 2D-CNN, 3D-CNN and ensemble CNN models, and a modified student t-test was performed. The ensemble method outperformed the 2D-CNN model and the 3D-CNN model. However, only the comparison with the 3D-CNN model attained statistical significance. The 5 × 2-fold mean accuracies for the models were 61.66% with SD of 3.05% for the ensemble

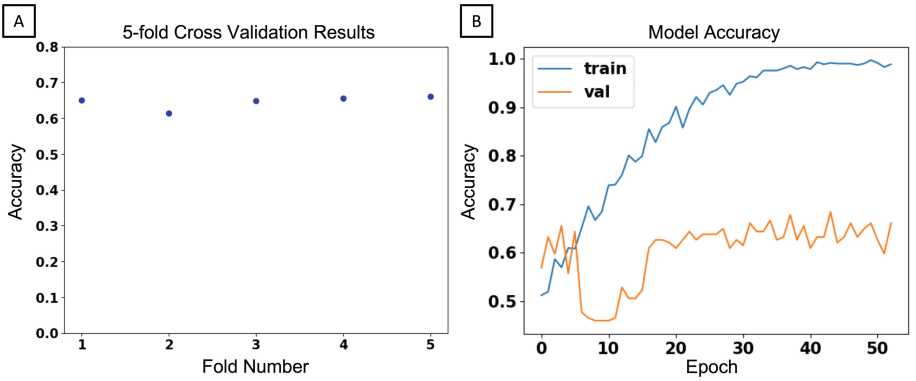

Fig. 8. Results from the combined 2DCNN+3DCNN combined ensemble model. A) Accuracy results from each of the 5-fold cross validations. B) An example accuracy curve from one of the 5-fold cross validation iterations with early stopping. The average accuracy across all folds was found to be 64.63% with a standard deviation of 1.62%.

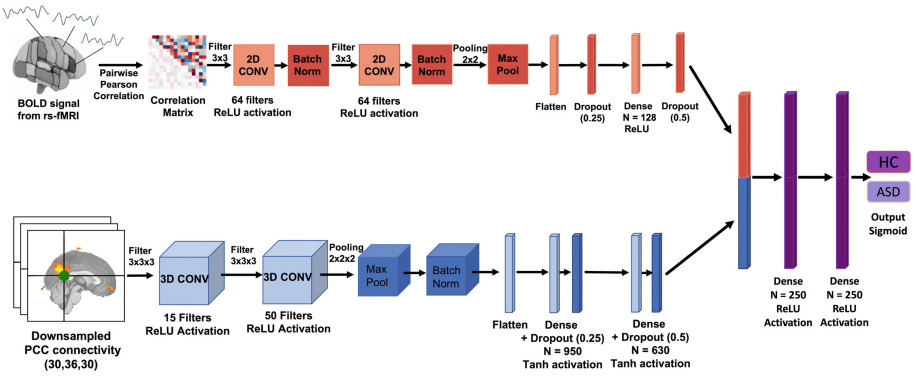

Fig. 9. Combined Ensemble 2D-CNN+3D-CNN Architecture

CNN model, 59.18% with an SD of 3.47% for the 2D-CNN model, and 54.54% with an SD of 2.10%. There was no significant difference between the 2D-CNN and the ensemble model accuracies ($p = 0.747$). There was a significant improvement between the 3D-CNN and ensemble model accuracies ($p = 0.0224$).

4 Conclusion

The performance of the ensemble model shows an increase over the two individual models when 5-fold cross validation and 5 × 2-fold cross validation was performed. This improvement is statistically significant when comparing this model to the individual 3D-CNN model ($p = 0.0224$), but not to the 2D-CNN model ($P = 0.747$). The improvement in performance of this ensemble learning model suggests that including multiple representations of functional neuroimaging may increase the information that the model can

learn from. The non-significant result could be due to the low predictive accuracy of the 3D-CNN branch, which may add limited predictive power to the ensemble model. The PCC connectivity data for this stream was downsampled, possibly eliminating some key features which reduced the accuracy. Further optimization of the preprocessing steps, the ROIs chosen for time course selection, and seed region selections for the 3D connectivity data would likely improve model quality. It may be easier to see improvements in an ensemble model with higher accuracy in both branches.

It is important to emphasize that the accuracies of the individual models reported here do not support the idea that one data representation is better than the others. The data types themselves contain different information, and the quality of information they contain is dependent upon how they are represented. For example, the correlation matrices were created using the Harvard-Oxford cortical atlas, therefore the matrix would contain different information if time courses were extracted from ROIs defined in a different atlas. The 3D-PCC connectivity data was only generated using one seed area; other seed areas might prove more useful, but they were not tested for this project. Therefore, a direct comparison of the three data types is not possible to determine based on the scope of this project.

References

1. Smucny, J., Shi, G., Davidson, I.: Deep learning in neuroimaging: overcoming challenges with emerging approaches. Front. Psych. **13**, 912600 (2022)
2. Vieira, S., Pinaya, W.H., Mechelli, A.: Using deep learning to investigate the neuroimaging correlates of psychiatric and neurological disorders: methods and applications. Neurosci. Biobehav. Rev. **74**(Pt A), 58–75 (2017)
3. Craddock, C., et al.: The neuro bureau preprocessing initiative: open sharing of preprocessed neuroimaging data and derivatives. Neuroinformatics **7**, 5 (2013)
4. Nilearn Library (2010–2022) GitHub Repository. http://nilearn.github.io/stable/auto_examples/index.html. Accessed 20 June 2023
5. Hull, J.V., Dokovna, L.B., Jacokes, Z.J., Torgerson, C.M., Irimia, A., Van Horn, J.D.: Resting-state functional connectivity in autism spectrum disorders: a review. Front. Psych. **7**, 205 (2017)
6. Rushmore, R.J., Bouix, S., Kubicki, M., Rathi, Y., Yeterian, E., Makris, N.: HOA2.0-ComPaRe: a next generation harvard-oxford atlas comparative parcellation reasoning method for human and macaque individual brain parcellation and atlases of the cerebral cortex. Front. Neuroanat. **16**, 1035420 (2022)
7. CSIRO's Data61, StellarGraph Machine Learning Library GitHub Repository (2018). https://github.com/stellargraph/stellargraph. Accessed 20 June 2023
8. Dietterich, T.G.: Approximate statistical tests for comparing supervised classification learning algorithms. Neural Comput. **10**, 1895–1923 (1998)