Classification of Autism Spectrum Disorder based on Brain Image using Deep Learning models

Jongwoo Choi (jc4816), Qianqian Hu (qh2185), Ziyu Chen (zc2393)

Abstract

Autism Spectrum Disorder (ASD) is a developmental disorder that affects social skills and behaviors. In neuroimaging studies, machine learning or deep learning methods are applied to diagnosis of ASD which may assist physicians or psychiatrists. In this project, we explored classification of the brain images for ASD using different classifiers such as logistic regression, auto encoders with multilayer perceptrons (AE + MLP) and convolutional neural networks (CNN), which is based on resting-state functional MRI (rs-fMRI) and structural MRI data. Our model was tested on ASD brain imaging data from Paris-Saclay Center for Data Science. Results of our classification accuracy is 73%. Accuracy is limited by current data size, and correlation feature extraction. Fingerprints feature extraction and 3D-CNN could be tried to improve accuracy.

1 Introduction

1.1 Overview

Diagnostic and Statistical Manual for Mental Disorders (DSM), published by the American Psychiatric Association (APA). The first version (DSM-I) was published in 1952 and the most recent version (DSM-5) was published in 2013. "According to the DSM-5, a guide created by the APA used to diagnose mental disorders, people with ASD have:

– Difficulty with communication and interaction with other people.

– Restricted interests and repetitive behaviors.

– Symptoms that hurt the person's ability to function properly in school, work, and other areas of life."

Autism spectrum disorders (ASD) are developmental disorder in which restricts interests and has difficulty in communication (DSM-5). The ASD was reported to affect about one in 59 children (DSM-5). Since ASD usually has other neurological and psychiatric disorders, such as anxiety and

mood disorders, and has complexity of phenotypes, it is challenge to find clinically useful diagnostic biomarkers (Andrew et al. 2018).

1.2 Research Question

Can we objectively diagnose patients with mental disorders? All medical conditions except mental disorders can be objectively diagnosed by instruments. For example, heart disease can be diagnosed by electrocardiograms (ECG/EKG). Similarly, blood sugar levels can be used for diagnosing patients with diabetes. Then, can a patient with mental disorders be objectively diagnosed by brain scans? Brain scans are usually used to identify brain biomarkers which would be used for differentiating healthy controls from patients although they have not been translated into objective and clinically useful biomarkers yet.

Then we'd like to read our mind through brain scans, brain encoding and decoding. Our goal is to turn our brain back to normal if there are any abnormal activity, theoretically it is doable using brain computer interface/interaction like real-time fMRI for treating mental disorder patients such as Autism because of plasticity of our brain.

Recently, the pattern classification of ASD using machine learning and deep learning has increased dramatically for diagnosis and predictions. For example, results from recent study using Deep Neural Network achieved 70% accuracy in identification of ASD verses controls in the ABIDE (Autism Brain Imaging Data Exchange) dataset. (Heinsfiled et al, 2018).

1.3 Magnetic Resonance Imaging (MRI) data

In this section, we introduce basic terminology of MRI data. Especially we focus on functional Magnetic Resonance Imaging (fMRI) image. Figure 1 shows the raw image of fMRI and sMRI respectively.

In general, voxel means volumetric pixel, which is the smallest distinguishable element of 3D image. Slice means one piece of a 3D image which slice order could be sequential ascending or descending and interleaved ascending or descending. Volumes mean scans. Repetition time (TR) is the time required to collect one volume (consists of many slices), which determines the temporal resolution. For example, given TR=2s, the duration of one run of the fMRI task is 5 mins, we have 5*60s/TR = 150 scans. Figure 2 shows the general format of MRI data.

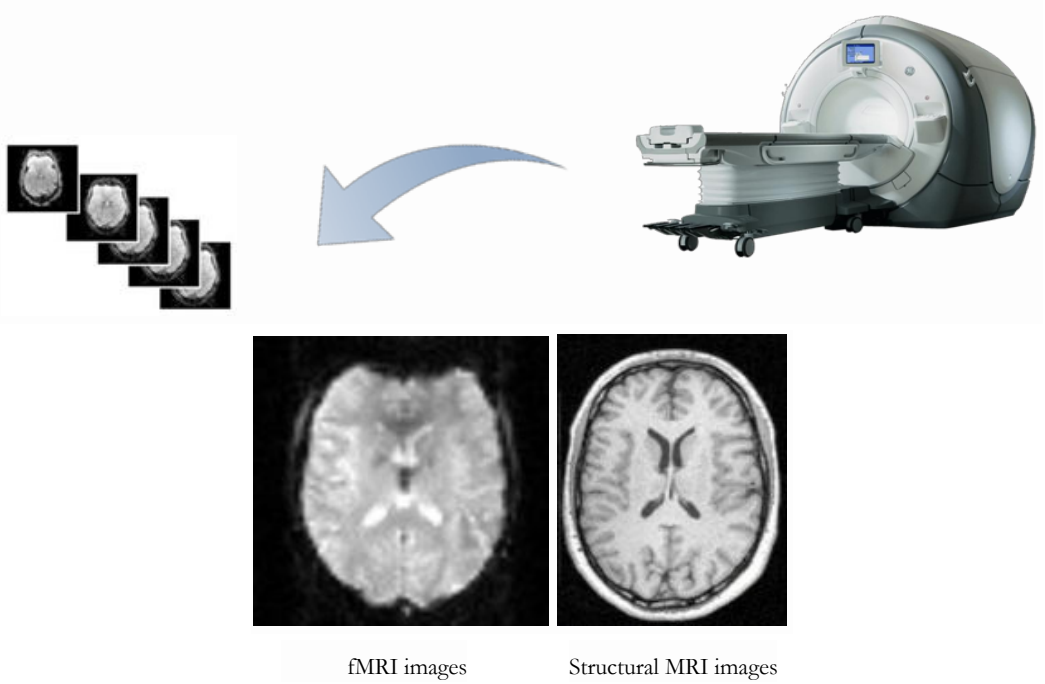

Figure 1. Raw fMRI and sMRI images

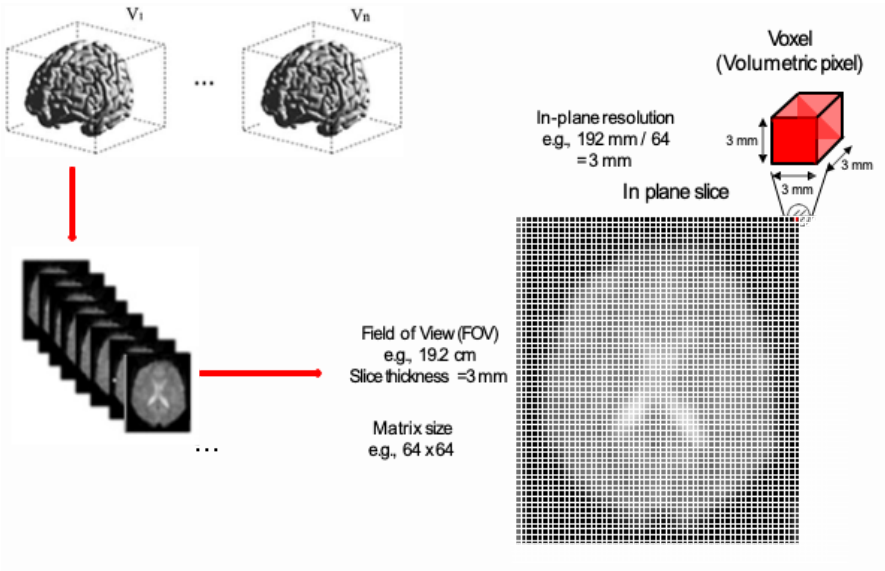

Figure 2. MRI data

fMRI data is functional in the sense that is measured continuously over time, so we measure the same brain volume multiple times across time. Each of these brain volumes consists of roughly 100,000 different voxels which are cubic volumes that span the three-dimensional space of the brain. Each voxel has a number associated with it, but also a spatial location. If we take the same voxel across time, we are actually studying what is going on and how the intensity is changing across that

voxel in that spatial location, By doing this, we can extract the time series of these intensities and study to see whether or not there's something in that time series that's related to the task that we performed. We are dealing with about 100,000 different time series that we are studying and looking for at a task-related behavior.

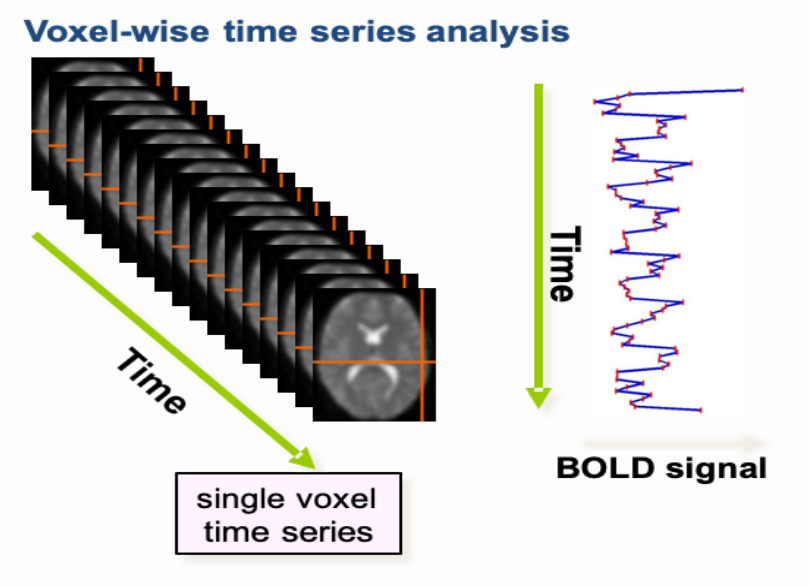

Figure 3. Voxel-wise time series analysis

2 Data and Preprocessing

2.1 Preprocessing

Preprocessing of the imaging data is not simple. Since our data is already preprocessed, we did not perform any preprocessing step of the raw imaging data. However, we simply introduce some crucial preprocessing steps to help reader's understanding. In general, SPM12 toolbox(Wellcome Trust Centre for Neuroimaging) or FSL (Free Surfer) is used for the analysis and visualization of sMRI and fMRI data. For temporal processing, slice timing helps to correct the time differences among the slides. For spatial processing, realignment is used to correct for head movement during the acquisition of fMRI data. Segmentation estimates a spatial transformation that can be used for spatially normalising images. In order to warps images to fit a standard template brain, normalization is used.

2.2 Understanding the data

Data we used for this project is available at https://paris-saclay-cds.github.io/autism_challenge/ (only `public set` is available) and initially published for competition: Imaging-psychiatry challenge:

predicting autism (IMPAC). The data contained 1150 subjects (601 health controls and 549 ASD patients) and the age range is 5 to 64.

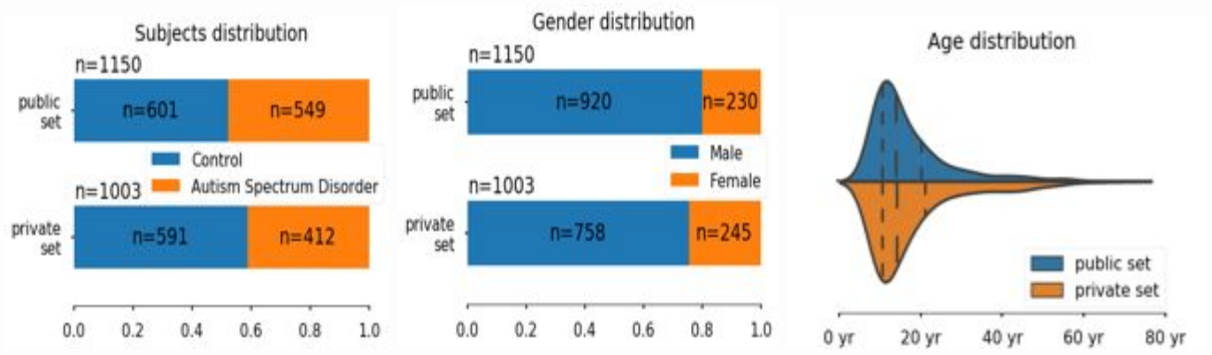

Figure 4. IMPAC data visualization

The pre-processed IMPAC dataset includes anatomy T1 features, rs-fMRI features and subject's site, age and sex. For Structural MRI data (T1 Anatomy), a set of structural features have been extracted for each subject with normalized brain volume computed using subcortical segmentation of FreeSurfer and cortical thickness and area for right and left hemisphere of Freesurfer. For rs-fMRI data, each subject comes with fMRI signals extracted on different brain parcellations and atlases, and a set of confound signals. Those brain atlases and parcellations are: (i) BASC parcellations with 64, 122, and 197 regions (Bellec 2010), (ii) Ncuts parcellations (Craddock 2012), (iii) Harvard-Oxford anatomical parcellations, (iv) MSDL functional atlas (Varoquaux 2011), and (v) Power atlas (Power 2011).

2.2 Feature selection: functional connectivity of ROIs

Since T1 features are structured, feature transformation is mainly done to fMRI features. The dataset provided has extracted time-series matrix per fMRI atlas per individual. The length of time series, however, is not equal for each individual. For example, the shape of time-series matrix for msdl, individual 1 is 200*39, individual 2 is 240*39, individual 3 is 120*39, etc.

Typical method of dealing with it is to calculate pairwise correlations and transform it into a correlation matrix, or 1d correlation vector (expansion of lower triangle matrix). This way, each fMRI atlas feature provides either a squared matrix (39*39 for msdl), or a fixed length vector (39*38/2 for msdl), which will be convenient to deal with afterwards. After basic standardization and detrending, we follow this procedure and are ready for a standard input data structure. Figure 1 shows the example correlation map of MSDL atlas.

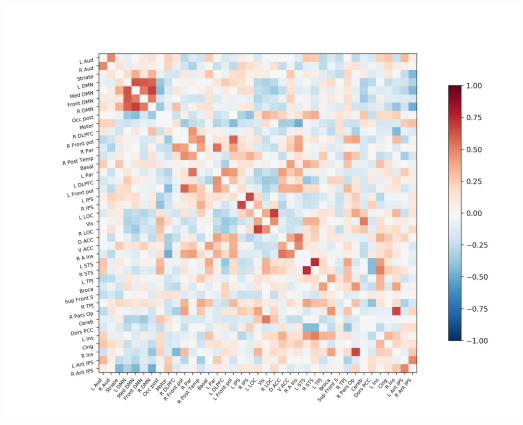

Figure 5. MSDL correlation matrix

We also followed Meszlenyi 2017 paper to extract DTW feature (a measure similar to edit distance) instead of correlation. However, it not only takes huge amount of time, but failed to beat correlation feature (far worse than correlation actually). So we exclude it here.

3 Methods

To have a unified version of test data so as to compare evaluation metrics, and to have enough data for reliability, we combine provided 1127 train and 23 test data and randomly re-split them into 920 train and 230 test data. Following models will train on these 920 individuals and test on 230. We did binary classification, 0 stands for healthy individual and 1 for ASD patient.

3.1 Failure of Logistic Regression

What appears most natural is to model this data with a logistic regression since we are trying to estimate a binary variable with our data. The generative process is:

$$P(y|x) \sim Bernoulli(\sigma(< v, x >- c))$$

for observations $x_{1, \ldots,} x_n$ and $y_i \in \{0, 1\}$. The function $\sigma(< v, x >- c)$ is a sigmoid ridge, where the ridge is orthogonal to the normal vector v, and c is an offset that shifts the ridge "out of the origin". The logistic regression model interprets $\sigma(< v, x >- c)$ as the probability of being in class y. (AML slides 1, 2018). We used L1-regularized logistic classifier preceded from a scaler that removed the mean and standard deviation computed on the training set.

We found that logistic regression was not an effective method for our dataset. Thus, we opted to continue the analysis with the deep learning model as we have seen in literature.

3.2 Model 1: AutoEncoder + MLP

Given the dimensionality of our data is so high and much higher than the sample size, dimensionality reduction techniques are necessary. One contemporary way of doing this is to use autoencoder. An autoencoder is a type of neural network that attempts to learn useful features from the original data. Inside, it has a hidden layer ***h*** that defines a code to represent the input. Therefore autoencoder can be thought of as a way to learn the internal representation of the data.
Usually an autoencoder consists of two parts: an encoder function ***h*** = f(x) that is used to represent the input and a decoder ***r*** = g(h) that produces a reconstruction. The learning objective can be formulated as finding such ***f*** and ***g*** that minimizes:

$$L(x,\ g(f(x))) \qquad (1)$$

Where ***L*** is a loss function such as the mean square error.

However, one problem of the most naïve autoencoder is that it may only learn the most obvious features of the data and thus fail to learn anything useful. For example, g(f(x)) = x everywhere and thus overfit the data. A way to avoid this is through applying regularization. In our project specifically, we applied denoising autoencoders. Instead of minimizing (1), a denoising autoencoder minimizes:

$$L(x,\ g(f(\hat{x}))) \qquad (2)$$

where $\hat{x}$ is a copy of x but has been manually corrupted by some form of noise. Many research show that in order to copying the input to output, the autoencoders will undo the corruption through learning the real distribution of the data, or in other words, the real important features of the data. Therefore through the use of autoencoder, our model will generalize well to novel data.

3.3 Model 2: Convolutional Neural Network (2D)

Convolutional Neural Network (CNN) is typically used in extracting features from image data. Our data, though not direct pixel images, are processed time series and correlation matrices from the original image. In addition to Auto-Encoder, we also wonder if convolution filters could perform the function of feature extraction. The convolution network will be directly applied on the 2d correlation matrices without extending to 1d.

At first, we tried several 2-4 layer networks, including 1 conv+softmax, 2 conv+softmax, 1 conv+hidden+softmax, 2 conv+hidden+softmax. All of them were using traditional square filters, (3*3, 5*5, 7*7, etc). These model didn't significantly improve the accuracy, though not bad compared to logistic regression - around 65-70%. Besides, because of the small size of filter, it took

hours to train the model even with only less than 1000 training individuals. There are actually quite a lot repeated information in the correlation matrix, and we decide to make use of them.

Inspired by Meszlenyi 2017, we turn to row-by-row and column-by-column filters, which form the first two convolution layers. Then add a fully connected layer followed by a output softmax layer that emits probability of classifying to 1. (Each of the first three layers is followed by a dropout layer.) Strictly speaking, this kind of convolution may not be useful on images, but could be a good trial for the correlation matrix structure. Since the correlation matrices are symmetric, extracting features from each row makes it possible to extract the relationship between one ROI and all others, which covers most of the information correlation provides. Also, the line-by-line filters largely accelerate the convolution process.

4 Experiments

4.1 Auto-Encoder + MLP

In the project, we implemented two stacked denoising autoencoders in the unsupervised pre-training stage to learn a lower dimensional representation of the data. We used mean squared error as our loss function. The data we have from all 7 atlas have 2016, 7381, 19306, 1128, 741, and 34716 dimensions respectively and altogether 96164 dimensions.

For the first autoencoder we set a corruption size of 25% of the data and code size of 6% of the original dimensionality. For the second autoencoder we set a corruption size of 20% of the data and code size of 50% of the dimensionality from the previous autoencoder. For the corruption, we applied a random size scaling from a uniform distribution on 0 to 1 minus the corruption size. For example, if the corruption size is 0.25, then 25% of the data are multiplied to a random number with uniform distribution on 0 to 0.75.

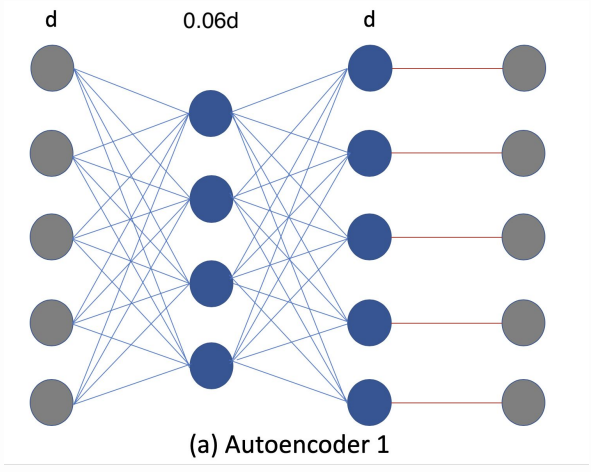

Figure 6: Illustration of 1st Auto-Encoder

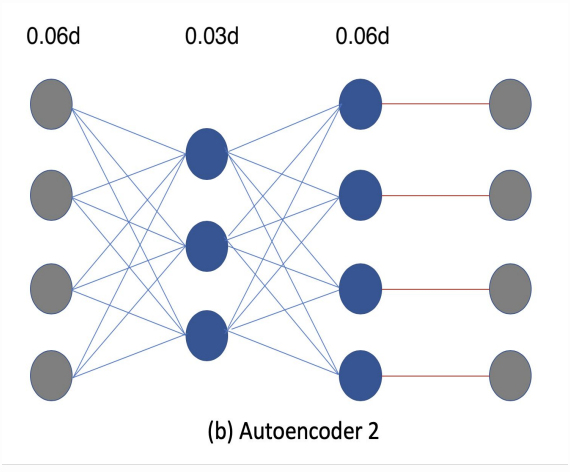

Figure 7: Illustration of 2nd Auto-Encoder

The 2 graphs above are illustrations of the two AutoEncoders we used. Assuming the original dimensionality of the data is d, the first AutoEncoder learns a code size of 0.06d and the second learns a code size of 0.03d.

After the unsupervised training of the two autoencoders, we then apply the learned weights of the 2 autoencoders to a MLP with 2 fully connected hidden layers. The first hidden layer with number of units and weights equal to the code size and weights of the first autoencoder respectively. The second one with number of hidden units and weights and weights equal to the code size of the second autoencoder respectively.

We tried two different setups of the model. In the first setup we trained the whole model on all the atlas features, which has 96164 dimensions and with the same configuration mentioned above. In the second setup, again with the same configuration, we trained the whole model separately on each atlas and made the predictions separately to give a final prediction using majority vote.

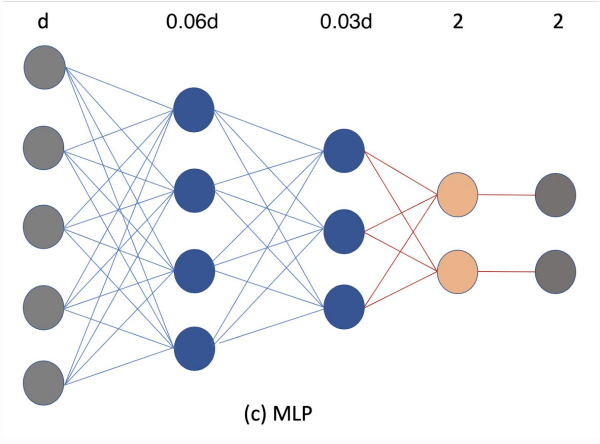

Figure 8: Illustration of Multilayer Perceptron

4.2 Convolutional Neural Network (2D)

We seek to use 2D-CNN on each fMRI atlas feature, then combine the results and do majority vote or calculate average logits. After experiments, it turns out that best results are gained by applying on basc064, basc122, basc197. Thus, we just combine these three to form an ensemble classifier.

The architecture of the 2D convolutional neural network is illustrated below. Take 'basc064' feature with 64*64 (row * col) correlation map as an example, we used 16 filters with kernel size of 1*row and col*1. The size of strides we used is (1,1) and the number of hidden layers are 64. Also, we used 80% dropout rate, 0.0001% of learning rate with batch size of 20.

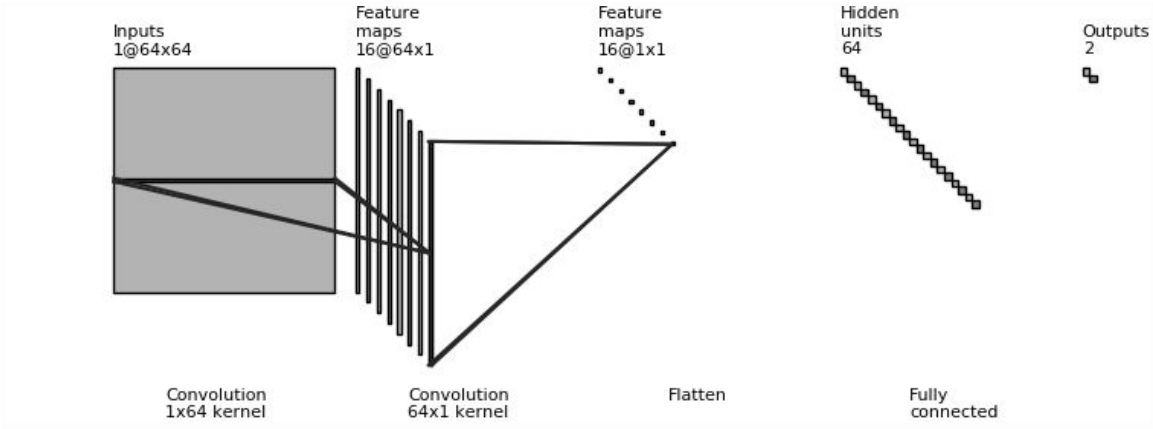

Figure 9. 2D CNN network illustration for basc064

4.3 Results

In the results section, we describe classification results with three performance metrics: accuracy, area under the receiver operator characteristic curve (AUC) and F1 score. Table 1 shows our evaluation results - the simple logistic regression, auto encoder + MLP and 2D convolutional neural networks.

	Logistic Regression	AE + MLP	2D-CNN
Accuracy	0.68	**0.73**	0.72
AUC	0.69	**0.73**	**0.73**
F1	0.68	**0.72**	**0.72**

Table 1: evaluation results

Logistic regression is used with all part of fMRI feature. As we mentioned before, logistic regression leads to failure. It achieved lower classification performance than other models.

The results of the two setups of AutoEncoder are quite interesting. When we trained on all the atlas features combined, we achieved an accuracy rate of 0.74 and an AUC score of 0.78. When we trained them separately and combined them through majority vote, we achieved an accuracy rate of 0.72 and an AUC score of 0.72. However the atlas feature basc064 along gives an accuracy rate of 0.73 and an AUC score of 0.73 and all other atlas features with accuracy rate and AUC score of below 0.7. These results here suggest that the ground truth model might be sparse and the atlas feature basc064 has great significance in predicting autism. 2D CNN are used with part of fMRI features; basc064, basc122 and basc197 with 2D vector for classification.

Basc series feature also work well for CNN model. Each basc feature provides unstable predictions depending on the split of training and validation set, but ensembling makes the result robust. Also, for each basc feature, Accuracy and F1 are between 65-70%, only AUC exceeds 70%. But by ensembling, all of them successfully go beyond 70%. The evaluation results from CNN model are quite close to those from AE+MLP.

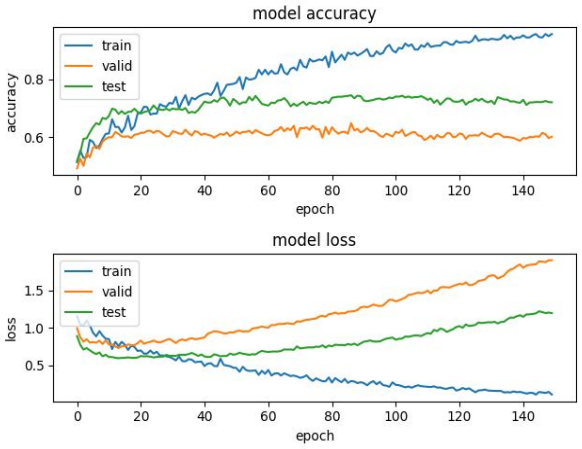

Figure 10. Training progress of AE+MLP under 'basc064'

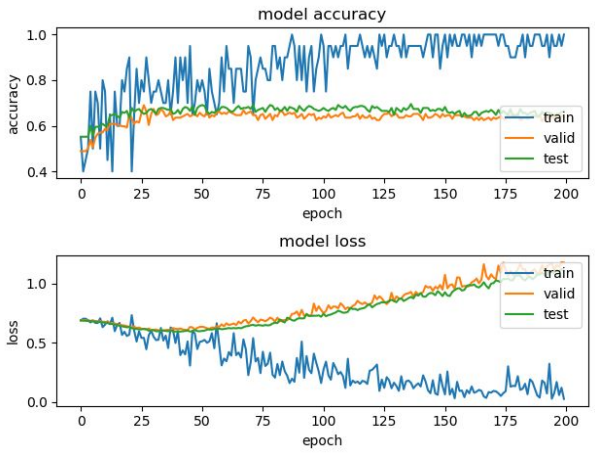

Figure 11. Training progress of CNN under 'basc064'

From Figure 10, we observe that both validation and test accuracy are not going up after certain point. The training accuracy goes to above 90% while validation and test accuracy stays around 60% to 70%. The training loss keeps going down with each epoch, just as expected. However, the validation and test loss go up again after around 30 epoch and finally even higher than the initial stage.

Training accuracy and loss are fluctuating under CNN model. We can see validation and test set follow very similar path in both accuracy and loss. The accuracies converge soon after several epochs, and stay stable afterwards. But losses continue to grow even when training loss is going down, which indicates the fit is not so well. We still see the overfit phenomenon in Figure 11.

5 Discussion

One thing can be inferred from the results of our Auto-Encoder model (from Figure 10) is that since the validation loss goes up again just in early stage, this phenomenon suggests that our model overfits. A possible reason for the overfitting model is that our autoencoders are not learning much useful information from the data and are potentially copying inputs to outputs, even we applied some regularization measures, for example, denoising autoencoder, dropouts, etc. However after

adjusting for many different combinations of learning rates, dropout rates, code sizes, and added shuffling, the performance still only improved slightly. This may further suggest that the nature of our data is actually quite sparse.

All of the three evaluation metrics are quite close between AE+MLP and 2D-CNN, indicating similar information is extracted from different network. It may also be upper limit for deep learning model. We did see from literatures that the best accuracy was around 70%. During our model training, overfitting was always warping around. We tried to prevent it, including regularization, dropout, batch normalization, simplifying network, etc. But they didn't quite work in this case. Data itself may be the main reason.

Brain imaging data is also sensitive to transients such as head movements, swallowing and eye movement especially for young patients. There also can be physiological signal fluctuation or resting state signal fluctuation. Sample size of brain imaging data is usually limited because MRI is too expensive. Although 1,000 patients are not small dataset in medical research, results can vary a lot for different split of training and validation dataset.

It's still worth noticing that our data is not direct scanned images, which may contain much more information. Instead, we are dealing with preprocessed time-series data provided by the website. The preprocessing may lose track of some information at the beginning. If we have access to the original image, more ways to improve the accuracy could be tried. For example, in feature extraction part we could try connectivity "fingerprints" which are represented as a multi-channel 3D volume. With these features, we could work with 3D CNN that allows to characterize connectivity with spatial relationship between voxels.

6 References

1. Heinsfeld, A. S., Franco, A. R., and Craddock, R. C. (2018). Identification of autism spectrum disorder using deep learning and the ABIDE dataset. Neuroimage 17, 16–23. doi: 10.1016/j.nicl.2017.08.017
2. Andrews DS, Marquand A, Ecker C, McAlonan G (2018) Using pattern classification to identify brain imaging markers in autism spectrum disorder. doi: 10.1007/7854_2018_47
3. Meszlényi, R. J., Buza, K., and Vidnyánszky, Z. (2017). Resting state fMRI functional connectivity-based classification using a convolutional neural network architecture. Front. Neuroinformatics 11:61. doi: 10.3389/fninf.2017.00061
4. American Psychiatric Association (2013). *Diagnostic and Statistical Manual of Mental Disorders* (Fifth ed.). Arlington, VA: American Psychiatric Publishing. pp. 5–25.
5. Advanced Machine Learning slides part 1 (2018). STAT 5242, Columbia University.