Classification of Parkinson's disease patients using Bayesian deep learning based on fMRI data

Maryam Amirmazlaghani[a,*], Amin Amini[a]

[a]Department of Computer Engineering and Information Technology, Amirkabir University of Technology, Tehran, Iran

ARTICLE INFO

Keywords:
Functional magnetic resonance imaging (fMRI)
Resting-state
Disease diagnosis
Bayesian deep learning
Convolutional neural network (CNN)
Parkinson's disease

ABSTRACT

The goal of resting-state functional magnetic resonance imaging (fMRI) is to investigate the brain's functional connections by using the temporal similarity between blood oxygenation level-dependent (BOLD) signals in different regions of the brain "at rest" as an indicator of synchronous neural activity. However having different images of patients' brains despite the sequential time-based information, may contain enough data to investigate their brains and enables us to recognize some diseases. In this paper, we propose a novel method to automatically extract this non-time sequence-based information using a Bayesian deep learning algorithm based on a convolutional neural network (CNN). Instead of using some predefined points of interest (POIs) we use the whole data in the training phase so those points of the brain which do not contain related information about the disease will be ignored automatically by the trained model. Moreover, this method does not make any assumptions about the disease, patients, etc., makes it a possible universal disease diagnosis approach to differentiate diseases having an impact on brain functionality. This method is a supervised algorithm with a small number of calculations using three-dimensional CNN. Each fMRI scan (which contains t time slices of the brain) of patients will be divided into t different 3D images enabling us to make the dataset much bigger in number and calculations way simpler. Subsequently, all of these images are fed to a Bayesian network similar to LeNet-5 (but in three dimensions) to train our model. Then to determine if a person is suffering from Parkinson's or not, we test his/her t fMRI images and get t different results which leads to a fraction (probability) of how unhealthy his/her brain is and if that fraction is above 0.5 we can classify that sample as a Parkinson's patient.

1. Introduction

Intro

2. Materials and methods

2.1. Participants

We had 30 subjects with ACER over 85 and MMSE over 27, comprising 15 healthy controls `[HC, age 63.33 ± 5.25, 8 females ]` and 15 patients diagnosed with Parkinson's disease `[PD, age 70.73 ± 4.80, 8 females ]`. Significant differences in age and ACER aong two groups were found `(age: p=0.0012 , ACER: p=0.015)`. However no significant difference in MMSE were found `(MMSE: p=0.315)`.

2.2. Data acquisition

fMRI scans were obtained from the OpenfMRI project which was acquired on a 3T Siemens Verio with repetition time (TR) = 2.5 s, echo time (TE) = 30ms, flip angle = 80°. Each sample contains 198 slices of $39 \times 64 \times 64$ voxels from the brains of patients. Subjects were instructed to close their eyes and to rest quietly during the scan session [7].

2.3. Data pre-processing

The fMRI data were pre-processed using the standard modules of FMRIB Software Library v5.0 [5]. The pre-processing steps for the anatomical data involved motion correction (MCFLRIT), skull stripping, and spatial smoothing (Gaussian kernel of 5-mm FWHM). Low level noise was removed using a high-pass temporal filtering. The functional images were then aligned to the individual's high-resolution T1-weighted scans, which were subsequently registered to the Montreal Neurological Institute standard space (MNI152) using affine linear registration. and resampled at 2mm cubic voxels. The product of the preprocessing step was used in the experiments.

2.4. Three dimensional convolutional neural networks

3D Convolutional Neural Networks which are inspired by human visual system are similar to classic neural networks. This architecture has been particularly designed based on the explicit assumption that raw data are three-dimensional that enables us to encode certain properties and also to reduce the amount of hyper parameters. The CNN topology utilizes spatial relationships to reduce the number of parameters which must be learned and thus improves upon general feed-forward back propagation training. Equation (1) shows how value of each voxel is calculated from previous layer based on a fixed kernel `(size = P×Q×R)` and a bias for `j`th feature in `i`th layers where v is value, f is the activation function and w is filter weight [2].

$$v_{ij}^{xyz} = f\left(b_{ij} + \sum_{m} \sum_{p=0}^{P_i-1} \sum_{q=0}^{Q_i-1} \sum_{r=0}^{R_i-1} w_{ijm}^{pqr}\, v_{(i-1)m}^{(x+p)(y+q)(z+r)}\right) \quad (1)$$

Since fMRI images are 3D time series and our experiment tries to find out if images contain any useful informa-

*Corresponding author
mazlaghani@aut.ac.ir (M. Amirmazlaghani); amin-amini@aut.ac.ir (A. Amini)
ORCID(s):

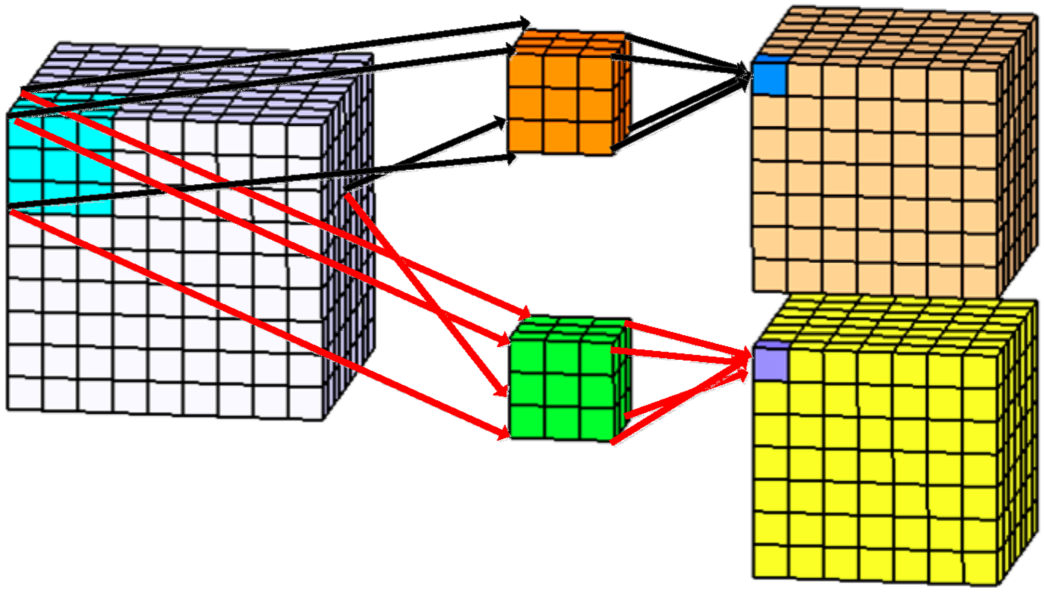

Figure 1: 3D CNN demonstration (kernel size = 3×3×3).

tion despite their sequences, three dimensional CNN can be used as our computational core. A simple 3D CNN can be seen in Figure 1

2.5. Bayesian deep learning

Deep learning has achieved significant success in many perception tasks including seeing (visual object recognition), reading (text understanding), and hearing (speech recognition). These are undoubtedly fundamental tasks for a functioning comprehensive artificial intelligence (AI) or data engineering (DE) system. However, in order to build a real AI/DE system, simply being able to see, read, and hear is far from enough. It should master the ability of thinking.

Take medical diagnosis as an example. Besides seeing visible symptoms (or medical images from MRI) and hearing descriptions from patients, a doctor has to look for relations among all the symptoms and preferably infer the corresponding etiology. Only after that can the doctor provide medical advice for the patients. In this example, although the abilities of seeing and hearing allow the doctor to acquire information from the patients, it is the thinking part that defines a doctor. Specifically, the ability of thinking here could involve causal inference, logic deduction, and dealing with uncertainty, which is apparently beyond the capability of conventional deep learning methods [6].

A neural network with arbitrary depth and non-linearities, with dropout applied before every weight layer, is mathematically equivalent to an approximation to the probabilistic deep Gaussian process [1].

Let $\hat{y}$ be the output of a NN model with L layers and a loss function $E(\cdot, \cdot)$ such as the softmax loss or the Euclidean loss (square loss).We denote by W_i the NN's weight matrices of dimensions $K_i \times K_{i-1}$, and by b_i the bias vectors of dimensions K_i for each layer $i = 1, ..., L$. We denote by y_i the observed output corresponding to input x_i for $1 \leq i \leq N$ data points, and the input and output sets as X, Y. During NN optimization a regularization term is often added. We often use $L2$ regularization weighted by some weight decay λ, resulting in a minimization objective (often referred to as cost) and it can written as Equation (2) [1].

$$\mathcal{L}_{dropout} := \frac{1}{N}\sum_{i=1}^{N} E(y_i, \hat{y}_i) + \lambda \sum_{i=1}^{L}(||W_i||_2^2 + ||b_i||_2^2) \quad (2)$$

Now of if we add dropout to the calculations each unit may remain the same with probability p_i or dropped out (its value is set to zero). We use the same values in the backward pass propagating the derivatives to the parameters. Assume we are given a covariance function of the form of Equation (3).

$$K(x, y) = \int p(w)p(b)\sigma(w^T x + b)\sigma(w^T y + b)dwdb \quad (3)$$

with some element-wise non-linearity $\sigma(\cdot)$ and distributions $p(w), p(b)$. It had been shown that a deep Gaussian process with L layers and covariance function $K(x, y)$ can be approximated by placing a variational distribution over each component of a spectral decomposition of the GPs' covariance functions.

Let W_i be a random matrix of dimensions $K \times K_{i-1}$ for each layer i, and write $\omega = \{W_i\}_{i=1}^{L}$. A priori,we let each row of W_i distribute according to the $p(W)$ above. In addition, assume vectors m_i of dimensions K_i for each GP layer. The predictive probability of the deep GP model given some precision parameter $\tau > 0$ can be parameterized as Equation (4).

$$p(y|x,X,Y) = \int p(y|x,\omega)p(\omega|X,Y)d\omega \quad (4)$$
$$p(y|x,\omega) = \mathcal{N}(y;\hat{y}(x,\omega),\tau^{-1}I_D)$$
$$\hat{y}(x,\omega=\{W_1,...,W_L\}) =$$
$$\sqrt{\frac{1}{K_L}}W_L\sigma(...\sqrt{\frac{1}{K_1}}W_2\sigma(W_1x+m_1)...)$$

The posterior distribution $p(\omega|X,Y)$ in Equation (4) is intractable. It had been shown that if we use a distribution over matrices whose columns are randomly set to zero, we can recover Equation (2) [1]. Note that previously mentioned distribution $q(\omega)$ can be defined as:

$$\begin{aligned} W_i &= M_i \cdot diag([z_{i,j}]_{j=1}^{K_i}) \\ z_{i,j} &= Bernoulli(p_i)\ \forall\ 1 \le i \le L\ ,\ 1 \le j \le K_{i-1} \end{aligned}$$

given some probabilities p_i and matrices M_i as variational parameters. The variable $z_{i,j} = 0$ corresponds then to unit j in layer $i-1$ being dropped out as an input to layer i. Subsequently if we use a dropout in 3D convolutional neural network, it would be a practice of Bayesian 3D CNN.

2.6. Proposed method

As mentioned before, our experiment aims to find out if fMRI scans carry any hidden information without taking time-based sequences in action. Accordingly, we examined some previous researches in this field and an eye-catching example of this glance is Sarraf and Tofighi's adopted LeNet-5 architecture [4]. Their algorithm concatenates fMRI images based on z and t axis and does not make any assumptions about the disease or the data (except their unknown patient selection criteria), making it a potential applicable solution to different disease diagnosis tasks. In brief, each patient's scan is presented as a huge two-dimensional matrix which is then fed to a CNN network. Although this method has shown huge success in the case of Alzheimer's (based on large ADNI dataset), our experiments clarify that it is not suitable for Parkinson's disease diagnosis with a small dataset (such as OpenfMRI ds000245). In addition to that, Sarraf and Tofighi's method consumes a huge amount of memory and demands a powerful processor due to its large dimensions of data.

Without a doubt, fMRI scans are nothing but spatial and temporal connectivities. As we aim to focus on spatial relations and put temporal ones aside, our algorithm must try to keep all of that information in calculations. Besides, Sarraf and Tofighi's method loses a lot of these connectivities by flattening data toward the z axis. In addition to that, the mentioned data conversion will increase our model complexity and variance, especially in a case like Parkinson's disease dataset. In fact, by resizing 198 slices of $39\times64\times64$ voxels to a 2D 2496×12672 pixels, we are adding a lot of unnecessarily and incorrect spatial connectivities based on the t axis and losing some important ones in the z axis by misplacing them. As a result, it is predictable that flattening algorithm may lead to overfitting and compute-intensive calculations which can be seen in Figure 2. This experiment ran on an NVIDIA® GTX1080 and took $6h\ 47m\ 36s$.

In order to conquer the mentioned downsides of a conventional CNN in Parkinson's disease diagnosis, we will introduce our method which keeps spatial information and can overcome overfitting by reducing model variance. To do so, the proposed algorithm will keep the z index of data and feed them into a 3D Bayesian CNN. As discussed in Section 2.5 if we add dropouts before each convolution layer, our model will turn into a deep Bayesian convolutional neural network that can avoid overfitting. Dropouts remove some units (by setting their output values to zero) temporarily which enables other units to justify their weights. Consequently, remaining units in the training phase can precisely adjust their weights to more appropriate ranges if they already hold acceptable values, or keep away from unwanted boundaries noticeably if needed. In addition to that, Bayesian networks are dramatically less compute-intensive and can solve problems in even a single epoch. Hence, a mid-end CPU can be used instead of a high-end GPU in experiments.

As we meant to put temporal data aside, we can assume that the fMRI data for each participant is t different 3D images we had obtained from their brain (despite its time-based sequences). Correspondingly this assumption will multiply our dataset samples by $t(=198)$, enabling us to train our model faster and escape overfitting. On the other hand, in the test phase, our model won't just give binary predictions about a participant being a Parkinson's patient or a healthy control, but it can give us an approximation about the probability of a person being healthy or not. In this step, the trained network will give t different predictions for each person (based on t images) providing a ratio, which corresponds to a probability of healthiness, and we can set a criterion (like 0.5) to decide if someone belongs to HC class or not.

First of all, input data will be preprocessed using Algorithm 1, which removes noises and non-informational data parts (such as the skull). In addition to that, each patient's scan will be normalized to a standard space that boosts the learning phase's efficiency. This step is followed by dividing each sample (containing t 3D fMRI time-slices) to t different images. For instance, in Parkinson's dataset, 30 samples consisted of 198 time-slices of $39\times64\times64$ voxels will turn into 5940 3D images. We convert these data to TFRECORD format to improve performance. Furthermore, for each experiment, six participants (20% of all data) from both groups are randomly selected, and their t time-slices are set aside for the test phase. On the other hand, the rest of the data (which are 24×198 labeled time-slices) will be shuffled and fed to our 3D deep Bayesian convolutional neural network. We use the first 159 time-slices (80%) as training samples while the remnants will be our validation portion of data correspondingly.

Now we can start to train our model based on the proposed architecture which is illustrated in Figure 12. During

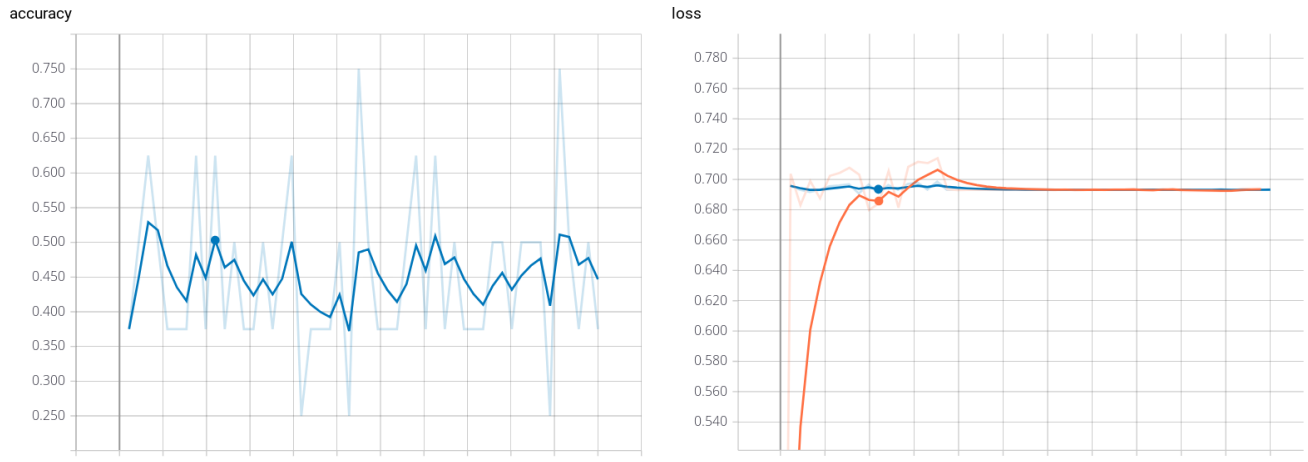

Figure 2: Parkinson's disease diagnosis accuracy and loss using Sarraf and Tofighi's algorithm.

input raw fMRI images

For each fMRI scan following pre-processing steps had been done.

1. Intra-modal motion correction using **MCFLIRT**. it loads the time-series in its entirity and will default to the middle volume as an initial template image. A coarse 8mm search for the motion parameters is then carried out using the cost function specified followed by two subsequent searches at 4mm using increasingly tighter tolerances. All optimizations use trilinear interpolation.

2. Skull stripping using BET. Brain Extraction Tool deletes non-brain tissue from an image of the whole head. It can also estimate the inner and outer skull surfaces, and outer scalp surface, if you have good quality T1 and T2 input images.

3. Spatial Smoothing via Full-Width at half maximum (FWHM) with 5mm range which is carried out on each volume of the FMRI data set separately. This is intended to reduce noise without reducing valid activation; this is successful as long as the underlying activation area is larger than the extent of the smoothing.

4. Highpass temporal filtering uses a local fit of a straight line (Gaussian-weighted within the line to give a smooth response) to remove low frequency artifacts. Low level noises were removed using this filter with $t = 90sec$.

5. The functional images were then aligned to the individual's high-resolution T1-weighted scans, which were subsequently registered to the **Montreal Neurological Institute standard space (MNI152)** using affine linear registration and resampled at 2mm cubic voxels

Algorithm 1: Data preparation

the training phase, data gets split to 50 chunks (95 samples) which are fed sequentially to the network. Consequently, the model can be validated during the training to prevent overfitting and enables us to track the whole process. The proposed method can be briefly expressed as Algorithm 2 which $N_{i,p}$ indicates the number of predictions that instance i is a Parkinson's patient and $N_{i,hc}$ indicates a Healthy Control. $\hat{Y}_{i,t}$ is the network prediction for t'th time-slice of sample i whilst Y_i is the real class and $C_{i,t}$ is the correctness of that output. Finally, P_i is the predicted probability of how healthy a patient is, $\hat{Y}_i$ is the class based on P_i and C_i is the correctness of that prediction.

Furthermore, our model's accuracy can be obtained in two different aspects. The first one would be the correctness of each time-slice prediction ($\overline{C_T}$), while the second, is the accuracy of disease diagnosis ($\overline{C}$), and we are mostly interested in the later. Both of the mentioned statistics can be calculated using Equations (5) and (6).

$$\overline{C_T} = \frac{\sum_{i=1}^{m} \sum_{t=1}^{T(=198)} C_{i,t}}{m \times T} \tag{5}$$

$$\overline{C} = \frac{\sum_{i=1}^{m} C_i}{m} \tag{6}$$

As can be seen in Figure 3, we have added different dropout units between our convolution and pooling layers to avoid

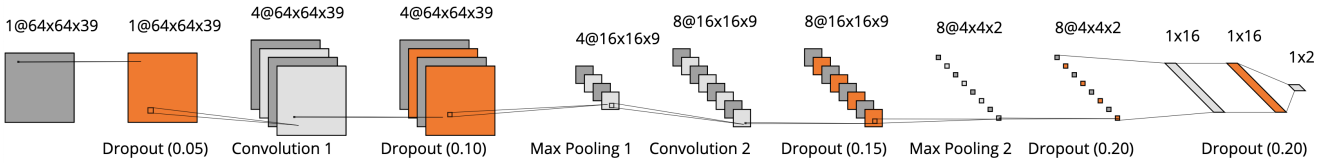

Figure 3: Proposed architecture

input processed data which is the output of Algorithm 1.

Shuffle scans
Put 6 scans (3 group each) aside as test data
Split time-slices of data and shuffle them
Divide train and validation data to 50 chunks
for all 50 chunks **do** Train network (Fig. 12)
  Input dropout ($p = 0.05$)
  3D convolution ($4@64 \times 64 \times 39$; $kernel[5, 5, 5]$)
  Dropout ($p = 0.10$)
  Max pooling ($4@16 \times 16 \times 9$; $pool[4, 4, 4]$)
  3D convolution ($8@16 \times 16 \times 9$; $kernel[4, 4, 4]$)
  Dropout ($p = 0.15$)
  Max pooling ($8@4 \times 4 \times 2$; $pool[4, 4, 4]$)
  Dropout ($p = 0.20$)
  Fully connected layer (16 *units*)
  Dropout ($p = 0.20$)
  Fully connected layer (2 *units*)
  Validate and emit network loss and accuracy
for all test data **do**
  $N_{i,p} \leftarrow 0$, $N_{i,hc} \leftarrow 0$, $C_{i,t} \leftarrow 0$
  for all time-slices **do**
    if $\hat{Y}_{i,t}$ = Parkinson's **then**
      $N_{i,p} \leftarrow N_{i,p} + 1$
    else
      $N_{i,hc} \leftarrow N_{i,hc} + 1$
    if $\hat{Y}_{i,t} = Y_i$ **then**
      $C_{i,t} \leftarrow 1$
    else
      $C_{i,t} \leftarrow 0$
  $P_i \leftarrow \frac{N_{i,hc}}{T_{(=198)}}$
  if $P_i \geq 0.5$ **then**
    $\hat{Y}_i \leftarrow$ Healthy Control
  else
    $\hat{Y}_i \leftarrow$ Parkinson's
  if $\hat{Y}_i = Y_i$ **then**
    $C_i \leftarrow 1$
  else
    $C_i \leftarrow 0$

Algorithm 2: Proposed method

overfitting, decrease variance and make convergence to appropriate weights faster. In addition to that, dropout probabilities increase in each layer because layers with less free parameters have a greater chance to train well even if they are less frequently updated. In other words, a huge number of parameters cannot tune acceptably with a small amount of data while getting omitted a lot. On the other hand, as there are only 16 weights in the first fully connected layer, higher dropout probabilities can speed up the training process.

Table 1
Average accuracies for different tests

j test no.	$\overline{C_{i,t}}$ (%) mean time-slice accuracy	$\overline{C_i}$ (%) mean accuracy
1	100	100
2	97.90	100
3	74.83	83.33
4	96.30	100
5	97.14	100
6	100	100
7	98.65	100
8	85.35	100
Average	$\mathbf{C_t} = 93.77$	$\mathbf{C} = 97.92$

3. Experiments and results

As discussed in Section 2.6, our method does not demand high processing power so we can use a CPU for our experiments. All of the tests had been run on an Intel® 4710HQ CPU. Averagely, the training phase takes $1h\ 12m\ 4s$ and the test phase finishes in $6m\ 41s$. Also it worth mentioning that the application consumes 270 megabytes of RAM thanks to the TFRECORD file format.

Table 1 shows a summary of 8 different tests. We can define the average of mean accuracies as Equations (7) and (8) in order to use them as a measure that shows how the model is performing in the case of accuracy.

$$\mathbf{C_t} = \frac{\sum_{j=1}^{J} \overline{C_T}}{J} \tag{7}$$

$$\mathbf{C} = \frac{\sum_{j=1}^{J} \overline{C}}{J} \tag{8}$$

As we can see, the proposed method has shown huge success in the task of Parkinson's disease diagnosis in an affordable time. Also, the fascinating point about the experiments is the average mean time-slice accuracy, which is around 92.96%. Accordingly, it means the model is able to judge only by a single time-slice of the brain while the prediction is acceptably accurate. We mostly care about the average mean accuracy, which is the average accuracy of the disease diagnosis task and as long as the whole process can predict someone's health situation precisely, we can ignore

Table 2
Test results for experiment 1. HC (Positive) stands for Health Control while P (Negative) means Parkinson's disease.

patient	Y_i	N_{hc}	N_p	$C_{i,t}(\%)$	P_i	$\hat{Y}_i$	$C_i(\%)$
CTL04	HC	198	0	100	1	HC	100
CTL05	HC	198	0	100	1	HC	100
CTL02	HC	198	0	100	1	HC	100
ODP04	P	0	198	100	0	P	100
ODP05	P	0	198	100	0	P	100
ODP02	P	0	198	100	0	P	100
			Mean time-slice accuracy $\overline{C_{i,t}}$	100	Mean accuracy $\overline{C_i}$		100

True positive (TP)	3	False positive (FP)	0
False negative (FN)	0	True negative (TN)	3
Time-slice TP	594	Time-slice FP	0
Time-slice FN	0	Time-slice TN	594
Time-slice sensitivity	1	Sensitivity	1
Time-slice specificity	1	Specificity	1
Time-slice PPV	1	PPV	1
Time-slice NPV	1	NPV	1

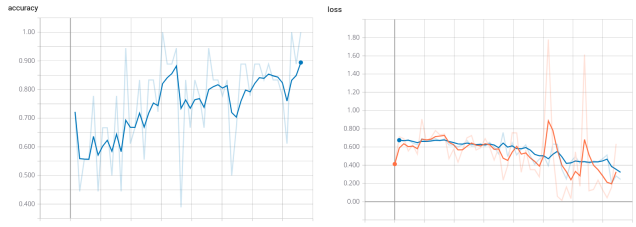

Figure 4: Experiment 1. Accuracy and loss of network for training and validation data. Accuracy is based on validation data and loss diagram contains train data loss in blue and validation loss in orange.

Table 3
Test results for experiment 2.

patient	Y_i	N_{hc}	N_p	$C_{i,t}(\%)$	P_i	$\hat{Y}_i$	$C_i(\%)$
CTL08	HC	191	7	96.46	0.9646	HC	100
CTL12	HC	195	3	98.48	0.9848	HC	100
CTL01	HC	198	0	100	1	HC	100
ODP08	P	1	197	99.49	0.0050	P	100
ODP12	P	2	196	98.99	0.0101	P	100
ODP01	P	12	186	93.94	0.0606	P	100
			$\overline{C_{i,t}}$	97.90		$\overline{C_i}$	100

True positive (TP)	3	False positive (FP)	0
False negative (FN)	0	True negative (TN)	3
Time-slice TP	584	Time-slice FP	15
Time-slice FN	10	Time-slice TN	579
Time-slice sensitivity	0.983164983	Sensitivity	1
Time-slice specificity	0.974747475	Specificity	1
Time-slice PPV	0.974958264	PPV	1
Time-slice NPV	0.983022071	NPV	1

the time-slice based results. So, we should mostly focus on average mean accuracy and use that to compare the proposed method to other models. Experiment results can be seen in Tables 2 to 9 and Figures 4 to 11.

Also we tested two other methods for comparison, First Sarraf and Tofighi's method [4] which had been described in the beginning of Section 2.6 which required a powerful GPU for calculations, had a lot of small assumptions about the data but faced overfitting in all 4 different experiments (Accuracies: [66.67% , 50% , 50% , 16.67%]). Second method is a SVM based method proposed by Rashid et al [3] for

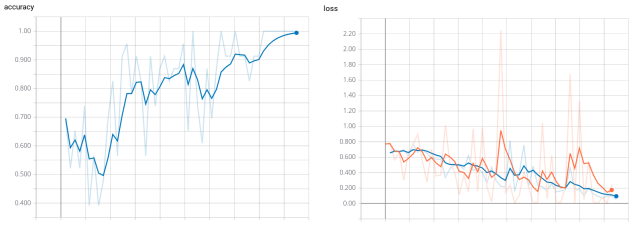

Figure 5: Experiment 2. Accuracy and loss of network for training and validation data.

Table 4
Test results for experiment 3.

patient	Y_i	N_{hc}	N_p	$C_{i,t}(\%)$	P_i	$\hat{Y}_i$	$C_i(\%)$
CTL03	HC	198	0	100	1	HC	100
CTL13	HC	198	0	100	1	HC	100
CTL14	HC	198	0	100	1	HC	100
ODP03	P	53	145	73.23	0.2677	P	100
ODP13	P	198	0	0	1	HC	0
ODP14	P	48	150	75.76	0.2424	P	100
			$\overline{C_{i,t}}$	74.83		$\overline{C_i}$	83.33

True positive (TP)	3	False positive (FP)	1
False negative (FN)	0	True negative (TN)	2
Time-slice TP	594	Time-slice FP	299
Time-slice FN	0	Time-slice TN	295
Time-slice sensitivity	1	Sensitivity	1
Time-slice specificity	0.496632997	Specificity	0.666666667
Time-slice PPV	0.665173572	PPV	0.75
Time-slice NPV	1	NPV	1

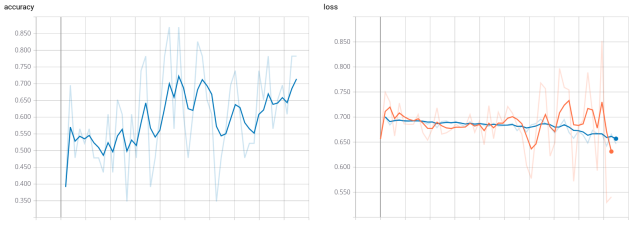

Figure 6: Experiment 3. Accuracy and loss of network for training and validation data.

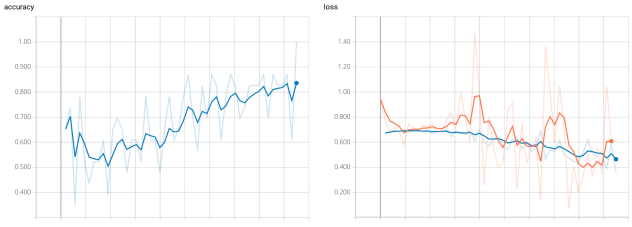

Figure 7: Experiment 4. Accuracy and loss of network for training and validation data.

classification of schizophrenia and bipolar patients based on functional network connectivities. This method takes some assumptions about the data and is not easily expandable to other diseases but I tried it as a competitor (Accuracies: [83.33% , 66.67% , 66.67% , 16.67%]). The comparison can be seen in Table 10.

Table 5
Test results for experiment 4.

patient	Y_i	N_{hc}	N_p	$C_{i,t}(\%)$	P_i	$\hat{Y}_i$	$C_i(\%)$
CTL12	HC	185	13	93.43	0.9343	HC	100
CTL06	HC	197	1	99.49	0.9949	HC	100
CTL11	HC	170	28	85.86	0.8586	HC	100
ODP12	P	0	198	100	0	P	100
ODP06	P	0	198	100	0	P	100
ODP11	P	2	196	98.99	0.0101	P	100
			$\overline{C_{i,t}}$	96.30		$\overline{C_i}$	100

True positive (TP)	3	False positive (FP)	0
False negative (FN)	0	True negative (TN)	3
Time-slice TP	552	Time-slice FP	2
Time-slice FN	42	Time-slice TN	592
Time-slice sensitivity	0.929292929	Sensitivity	1
Time-slice specificity	0.996632997	Specificity	1
Time-slice PPV	0.996389892	PPV	1
Time-slice NPV	0.933753943	NPV	1

Table 6
Test results for experiment 5.

patient	Y_i	N_{hc}	N_p	$C_{i,t}(\%)$	P_i	$\hat{Y}_i$	$C_i(\%)$
CTL15	HC	198	0	100	1	HC	100
CTL11	HC	198	0	100	1	HC	100
CTL07	HC	198	0	100	1	HC	100
ODP15	P	0	198	100	0	P	100
ODP11	P	34	164	82.83	0.1717	P	100
ODP07	P	0	198	1 00	0	P	100
			$\overline{C_{i,t}}$	97.14		$\overline{C_i}$	100

True positive (TP)	3	False positive (FP)	0
False negative (FN)	0	True negative (TN)	3
Time-slice TP	594	Time-slice FP	34
Time-slice FN	0	Time-slice TN	560
Time-slice sensitivity	1	Sensitivity	1
Time-slice specificity	0.942760943	Specificity	1
Time-slice PPV	0.945859873	PPV	1
Time-slice NPV	1	NPV	1

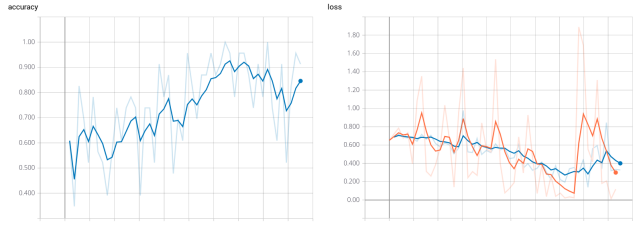

Figure 8: Experiment 5. Accuracy and loss of network for training and validation data.

Table 7
Test results for experiment 6.

patient	Y_i	N_{hc}	N_p	$C_{i,t}(\%)$	P_i	$\hat{Y}_i$	$C_i(\%)$
CTL10	HC	198	0	100	1	HC	100
CTL03	HC	198	0	100	1	HC	100
CTL02	HC	198	0	100	1	HC	100
ODP10	P	0	198	100	0	P	100
ODP03	P	0	198	100	0	P	100
ODP02	P	0	198	100	0	P	100
			$\overline{C_{i,t}}$	100		$\overline{C_i}$	100

True positive (TP)	3	False positive (FP)	0
False negative (FN)	0	True negative (TN)	3
Time-slice TP	594	Time-slice FP	0
Time-slice FN	0	Time-slice TN	594
Time-slice sensitivity	1	Sensitivity	1
Time-slice specificity	1	Specificity	1
Time-slice PPV	1	PPV	1
Time-slice NPV	1	NPV	1

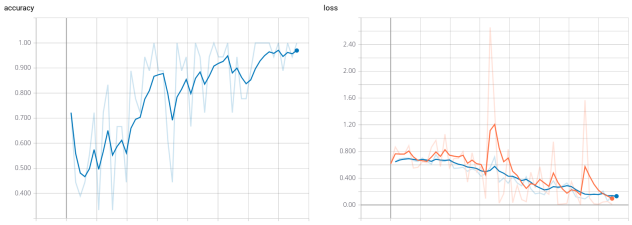

Figure 9: Experiment 6. Accuracy and loss of network for training and validation data.

Table 8
Test results for experiment 7.

patient	Y_i	N_{hc}	N_p	$C_{i,t}(\%)$	P_i	$\hat{Y}_i$	$C_i(\%)$
CTL05	HC	194	4	97.98	0.9798	HC	100
CTL04	HC	198	0	100	1	HC	100
CTL10	HC	186	12	93.94	0.9394	HC	100
ODP05	P	0	198	100	0	P	100
ODP04	P	0	198	100	0	P	100
ODP10	P	0	198	100	0	P	100
			$\overline{C_{i,t}}$	98.65		$\overline{C_i}$	100

True positive (TP)	3	False positive (FP)	0
False negative (FN)	0	True negative (TN)	3
Time-slice TP	578	Time-slice FP	0
Time-slice FN	16	Time-slice TN	594
Time-slice sensitivity	0.973063973	Sensitivity	1
Time-slice specificity	1	Specificity	1
Time-slice PPV	1	PPV	1
Time-slice NPV	0.973770492	NPV	1

4. Discussion

Our results suggest that the classification of Parkinson's disease patients using fMRI data can be done accurately without taking time-based information in calculations. Also, this is supported by statistics such as Accuracy, Sensitivity, Specificity, Positive and Negative Predictive Values. Moreover, it is obvious that predictions using only one time-slice can be very powerful and enables us to obtain a probability of how healthy a patient is. On the other hand, this method does not require to omit any time-slices from scans to unify them, unlike other methods that use this approach to find functional network connectivities. Additionally, the proposed method does not make any assumptions based on any information, and it might be able to be used to classify other diseases as well.

Test results show the method did not face overfitting thanks to the dropout layers. As discussed in Section 2.5, our network is a Bayesian 3D CNN that can learn from a small amount of data without memorizing them, just like other Bayesian networks. Experiment 3 is the only test that our model failed to predict the health status of samples precisely and did 1 mistake. However, our model did not overfit hugely, it just could not solve the problem acceptably, and if we take a look at Figure 6, it is obvious that the validation accuracy is less than 80%. As a result, the proposed method can train with low calculation costs and the trained model is reliable based on experiments.

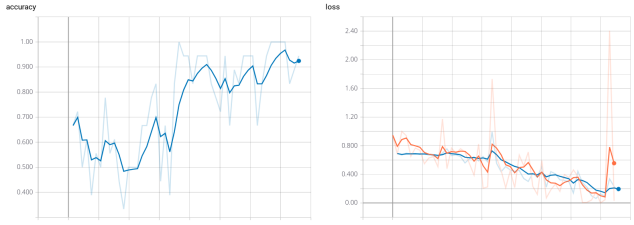

Figure 10: Experiment 7. Accuracy and loss of network for training and validation data.

Table 9
Test results for experiment 8.

patient	Y_i	N_{hc}	N_p	$C_{i,t}(\%)$	P_i	$\hat{Y}_i$	$C_i(\%)$
CTL09	HC	102	96	51.51	0.5151	HC	100
CTL12	HC	122	76	61.62	0.6162	HC	100
CTL06	HC	198	0	100	1	HC	100
ODP09	P	0	198	100	0	P	100
ODP12	P	2	196	98.99	0.0101	P	100
ODP06	P	0	198	100	0	P	100
			$\overline{C_{i,t}}$	85.35		$\overline{C_i}$	100

True positive (TP)	3	False positive (FP)	0
False negative (FN)	0	True negative (TN)	3

Time-slice TP	422	Time-slice FP	2
Time-slice FN	172	Time-slice TN	592

Time-slice sensitivity	0.71043771	Sensitivity	1
Time-slice specificity	0.996632997	Specificity	1
Time-slice PPV	0.995283019	PPV	1
Time-slice NPV	0.77486911	NPV	1

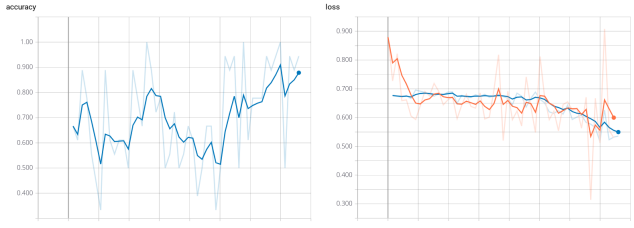

Figure 11: Experiment 8. Accuracy and loss of network for training and validation data.

Table 10
Comparison of the proposed method with 2 other rivals

Method	Average accuracy (%)	Average time (min)	Test hardware
Proposed Method	97.92	72.06	GPU (GTX1080)
Sarraf and Tofighi	45.83	407.6	CPU (Intel4710HQ)
Rashid et al.	58.33	23.13	CPU (Intel4710HQ)

Without

A. My Appendix

Appendix sections are coded under `\appendix`.

`\printcredits` command is used after appendix sections to list author credit taxonomy contribution roles tagged using `\credit` in frontmatter.

CRediT authorship contribution statement

Amin Amini: Conceptualization of this study, Methodology, Software.

References

[1] Gal, Y., Ghahramani, Z., 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning, in: international conference on machine learning, pp. 1050–1059.

[2] Ji, S., Xu, W., Yang, M., Yu, K., 2013. 3d convolutional neural networks for human action recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence 35, 221–231. doi:10.1109/tpami.2012.59.

[3] Rashid, B., Arbabshirani, M.R., Damaraju, E., Cetin, M.S., Miller, R., Pearlson, G.D., Calhoun, V.D., 2016. Classification of schizophrenia and bipolar patients using static and dynamic resting-state fMRI brain connectivity. NeuroImage 134, 645–657. doi:10.1016/j.neuroimage.2016.04.051.

[4] Sarraf, S., Tofighi, G., 2016. Classification of alzheimer's disease using fmri data and deep learning convolutional neural networks arXiv:arXiv:1603.08631.

[5] Smith, S.M., Jenkinson, M., Woolrich, M.W., Beckmann, C.F., Behrens, T.E., Johansen-Berg, H., Bannister, P.R., De Luca, M., Drobnjak, I., Flitney, D.E., et al., 2004. Advances in functional and structural mr image analysis and implementation as fsl. Neuroimage 23, S208–S219.

[6] Wang, H., Yeung, D.Y., 2016. Towards bayesian deep learning: A framework and some existing methods. IEEE Transactions on Knowledge and Data Engineering 28, 3395–3408. doi:10.1109/tkde.2016.2606428.

[7] Yoneyama, N., Watanabe, H., Kawabata, K., Bagarinao, E., Hara, K., Tsuboi, T., Tanaka, Y., Ohdake, R., Imai, K., Ogura, R., Kato, Y., Masuda, M., Hattori, T., Ito, M., Maesawa, S., Mori, D., Katsuno, M., Sobue, G., 2018. Olfactory dysfunction and functional connectivity changes in cognitively normal parkinson's disease. https://openfmri.org/dataset/ds000245. This data was obtained from the OpenfMRI database. Its accession number is ds000245 [Online; accessed 25-September-2019].

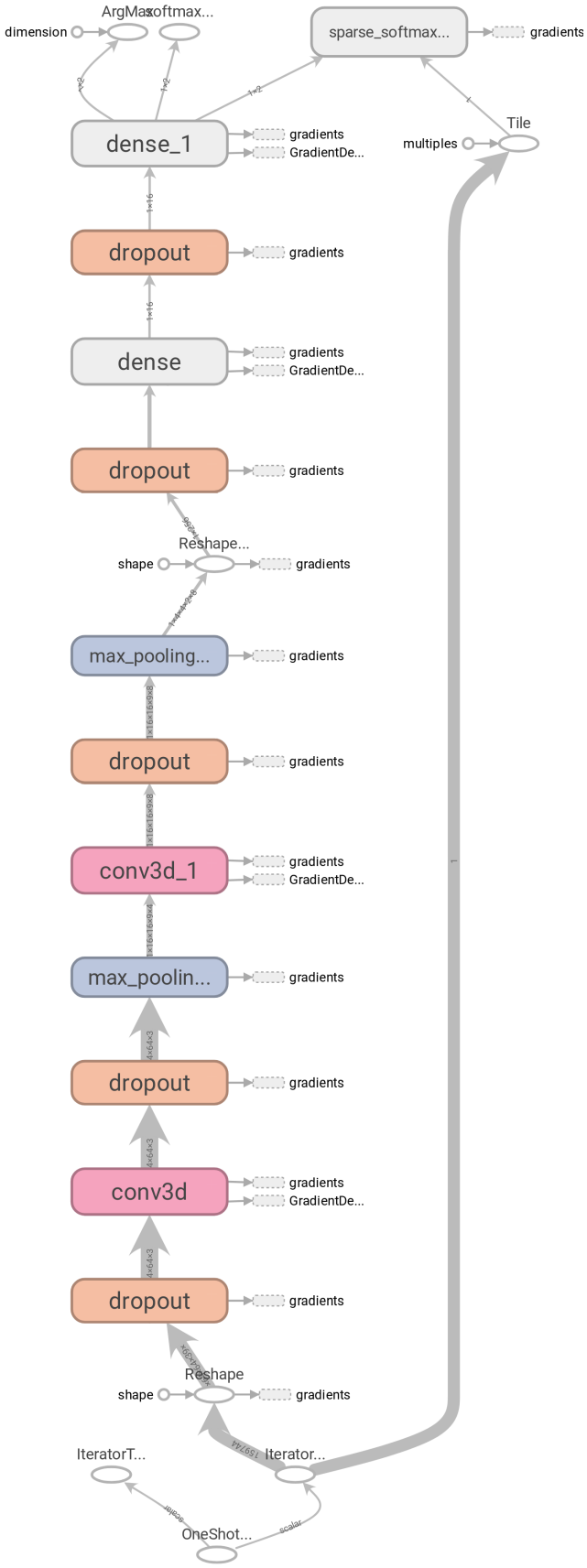

Figure 12: Proposed Bayesian 3D CNN architecture