Neurocomputing 434 (2021) 1–10

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Fast vertex-based graph convolutional neural network and its application to brain images

Chaoqiang Liu [a], Hui Ji [d], Anqi Qiu [a,b,c,e,*]

[a] Department of Biomedical Engineering, National University of Singapore, Singapore
[b] Smart Systems Institute, National University of Singapore, Singapore
[c] The N.1 Institute for Health, National University of Singapore, Singapore
[d] Department of Mathematics, National University of Singapore, Singapore
[e] Department of Biomedical Engineering, Johns Hopkins University, USA

ARTICLE INFO

Article history:
Received 30 August 2020
Revised 4 November 2020
Accepted 19 December 2020
Available online 7 January 2021
Communicated by Zidong Wang

Keywords:
Convolutional neural network
Graph-structured data
Dementia classification
Semi-triangulated meshes

ABSTRACT

This paper proposes a vertex-based graph convolutional neural network (vertex-CNN) for analyzing structured data on graphs. We represent graphs using semi-regular triangulated meshes in which each vertex has 6 connected neighbors. We generalize classical CNN defined on equi-spaced grids to that defined on semi-regular triangulated meshes by introducing main building blocks of the CNN, including convolution, down-sampling, and pooling, on a vertex domain. By exploiting the regularity of semi-regular meshes in terms of vertex connections, the proposed vertex-CNN keeps the inherent properties of classical CNN in a Euclidean space, such as shift-invariance and down-sampling at a rate of 2, 4, etc. We employ brain images from Alzheimer's Disease Neuroimaging Initiative (ADNI) (n = 6767) and extract cortical features (e.g., cortical thickness, surface area, curvature, Jacobian, sulcal depth, and volume) for the classification of healthy controls (CON), patients with mild cognitive impairment (MCI) and Alzheimer's disease (AD). Based on cortical thickness, we show that the proposed vertex-CNN is near 3 times faster and performs significantly better in the classification performance of CON, MCI, and AD than an existing graph CNN defined on the graph spectral domain given in Defferrard (2016). Moreover, we examine the robustness of a multi-channel implementation of vertex-CNN on 6 cortical measures for the MCI and AD classification. Finally, we show a promising finding of the prediction accuracy from MCI to AD as a function of years before the onset of AD. Our experiments demonstrate the fast computation and promising classification performance of the vertex-CNN.

1. Introduction

Machine learning has been widely used as one of the crucial techniques for medical image segmentation, registration, disease prediction, and classification. In these tasks, image data are defined on an equi-spaced grid in a Euclidean space, where geometric information of human organs may not explicitly be represented. Nevertheless, the geometry of human organs characterizes their intrinsic and complex anatomy, as well as physiological functions. For instance, the cerebral cortex is composed of gyri and sulci. A gyrus is a ridge of the cerebral cortex, while a sulcus is a groove of the cerebral cortex. Because of the way that gyri and sulci are curved, the cortex is thicker in gyral regions but thinner in sulcal regions. Hence, it is preferred to represent brain image data in a way that the underlying cortical geometry is encoded. One can express the cerebral cortex using triangulated meshes embedded in a 3D Euclidean space. Existing literature has demonstrated that such representation incorporates useful geometry information of the brain in machine learning for disease diagnosis and prediction (e.g., [44,16,1,36]).

In recent years, deep learning has tackled a wide range of challenging tasks in image analysis and classification. Among many architectures of deep neural networks, *convolutional neural network* (CNN) receives great attention for its successes in computer vision (e.g., [40,37,43,41,22,18]), medical imaging and diagnosis (e.g., [38,29,15,25]). Classical CNN contains three major blocks, including convolution with localized filters, rectified linear unit (ReLU, non-linear activation function), and pooling. These three blocks are sequentially concatenated to form a convolutional layer that represents a simple non-linear function. Then, the integration of

* Corresponding author at: National University of Singapore, 4 Engineering Drive 3, Block E4 04-08, Singapore 117583, Singapore.
E-mail address: bieqa@nus.edu.sg (A. Qiu).

https://doi.org/10.1016/j.neucom.2020.12.097
0925-2312/

many such convolutional layers forms a very complex non-linear function that can model highly non-linear intrinsic patterns of training data. In medical image applications, existing CNNs are developed for analyzing medical images defined in a Euclidean equi-spaced grid. The definitions of their main building blocks are based on the voxel-connectivity property of equi-spaced grids. Thus, a direct call of such CNNs to model structured data on graphs embedded in a Euclidean space certainly is not possible as the vertex connectivity of meshes is different from that of equi-spaced Euclidean grids. Indeed, it is not trivial to adapt the definition of convolution in equi-spaced grids to meshes, while keeping the essential property, such as shift-invariance. The same observation holds for defining the operation of dyadic sampling on meshes.

This paper aims to extend classical CNNs from equi-spaced grids to meshes and develop a vertex-based graph CNN (vertex-CNN) for analyzing structured data on semi-regular triangulated meshes. The connectivity of most vertices on a semi-regular triangulated mesh is 6. A semi-regular triangulated mesh has certain similarities to equi-spaced grids in a Euclidean space. When image data are defined on a semi-regular triangulated mesh, a direct call of a generic CNN for a mesh certainly is sub-optimal, as it discards specific connectivity regularity of the mesh. Indeed, the ordering property of a semi-regular triangulated mesh allows us to better mimic classical convolution. The most desired property of classical convolution remains the same for the convolution on meshes, that is, the dimension of the parameter space of convolutional filters is the same as the number of vertices involved in each shift. Moreover, down-sampling a semi-regular mesh is more flexible in terms of a sampling rate and can be done more efficiently than down-sampling a general graph. We employ brain images from Alzheimer's Disease Neuroimaging Initiative (ADNI) (n = 6767) and extract cortical features (e.g., cortical thickness, surface area, curvature, Jacobian, sulcal depth, and volume) for the classification of healthy controls (CON), patients with mild cognitive impairment (MCI) and Alzheimer's disease (AD). Based on cortical thickness data, we compare the computation time and the classification performance of CON, MCI, and AD between the vertex-CNN and an existing graph CNN based on the graph spectral domain in Defferrard (2016). Moreover, we examine the robustness of a multi-channel implementation of vertex-CNN on 6 cortical measures for the MCI and AD classification. Finally, we show a promising finding of the prediction accuracy from MCI to AD as a function of years before the onset of AD.

This paper contributes to.

- Unlike existing graph neural networks, we exploit the regularity of semi-regular meshes and develop a fast vertex-CNN that mimics the convolution and pooling operations of classical CNN for analyzing structured data on meshes.
- The proposed vertex-CNN keeps the inherent properties of classical CNN in a Euclidean space, including shift-invariance and down-sampling at the rate of 2, 4, etc.
- The proposed vertex-CNN is near 3 times faster than graph CNN defined on the graph spectral domain in Defferrard (2016).
- We proposed a multi-channel vertex-CNN that incorporated 6 cortical measures and achieved the outstanding classification accuracy of MCI and AD.
- We provided a unique finding on the prediction of AD as a function of years before the onset of AD.

2. Review on relevant work

Graph convolutional neural networks (graph CNNs) are deep learning techniques that apply to graph-structured data. Graph-structured data are in general complex, which imposes significant challenges on existing convolutional neural network algorithms. Graphs are irregular and have variable number of unordered vertices with different topology at each vertex. This makes important algebraic operations such as convolutions and pooling challenging to apply to the graph domain. Hence, existing research on graph CNN has been focused on defining convolution and pooling operations.

For convolution, there are two types of approaches for defining convolution on mesh/graph, one is in a vertex domain and the other is in a spectral domain. Existing vertex-based definitions of convolution focus on how to process the vertices whose neighborhood has different sizes and connections. Several methods have been proposed in the past, including Diffusion-Convolutional Neural Networks (DCNNs) [2], PATCHY-SAN [32,13], Gated Graph Sequential Neural Networks [24], DeepWalk [35], and so on. An alternative approach is to map individual patches of a mesh to a representation that is more amenable to convolution. Here are some examples for such a representation, including 2D polar coordinate representation [28], local windowed spectral representation [4], anisotropic variants of heat kernel diffusion filters [6,5], Gaussian Mixture-model kernels [30,31], and so on.

There have been several methods proposed for the definition of convolution in a spectral domain (see e.g., [7,9,19,21,45,23,39]). Based on spectral graph theory, Bruna et al. [7] proposed a CNN for graph-structured data in a spectral domain in which the convolution is defined as the multiplication operation of a diagonal matrix in graph Fourier transform derived from a normalized graph Laplacian. As such, the convolution may not be localized and hence the diagonal entries of the multiplication matrix need to be regularized with a smooth prior for possible localization. To avoid the cost of calculating a graph Laplacian and have a convolution with better localization, Defferrard et al. [9] introduced Chebyshev polynomial approximation such that the resulting convolution operator is a polynomial of the adjacency matrix of a graph. Kipf and Welling [21] further simplified the approximation using the linear polynomial of the adjacency matrix of a graph and used the resulting CNN as the tool for semi-supervised learning. In [9,39], it is shown that a k-order polynomial formation of the graph Laplacian performs a k-ring filtering operation. Representing polynomial filters as linear combinations of Chebyshev polynomial recurrence relation allows rapidly apply these filters in the spatial domain.

Nevertheless, the convolution built on the polynomials of the adjacency matrix is quite different from classical convolution on Euclidean equi-spaced grids, especially on the aspect of expression power determined by the dimension of a parameter space. To be more specific, the localized convolution for Euclidean equi-spaced grids whose dimension of a parameter space, the size of filters, is indeed the same as the number of vertices it covers for each shift. However, when considering a convolutional filter derived from the polynomial degree of k, it is parameterized by $(k+1)$ parameters. The support of such a convolution, i.e., the number of the vertices it covers for each shift, is $3k^2+3k+1$ vertices for a regular triangular mesh whose vertices have 6 connected neighbors. The main idea of CNN is that it uses filters of small size to collect local information. Then, in a multi-scale manner, it gradually increases the filter width and down-sizes the features to represent more global and high-level information of image data. Hence, a convolutional filter that covers a large number of vertices with few parameters, such as a convolutional filter derived from the polynomial degree of k, is likely to miss important local features which can be helpful for modeling image data.

Moreover, how to down-sample images in CNN is also an important operation that is used in both convolution with stride >1 and pooling to abstract more high-level information in a multi-scale manner. The graph coarsening procedure used for pooling in [9] is implemented by calling a weighted graph cut

method in [10]. From the coarsest to the finest level, fake vertices, i.e., disconnected vertices, are added to pair with singletons such that each vertex has two children. The fake vertices artificially increase the dimensionality of the graph, which increases the computational cost even when the number of singletons from multi-level clustering algorithms is not very large.

In the following, we describe the vertex-based CNN and tackle both convolution and pooling operations. While taking an advantage of the regularity of a semi-regular triangulated mesh, we are able to define these two operations in an efficient manner.

3. Methods

3.1. Motivation from CNN defined in Euclidean spaces

Classical CNN is designed to extract features from input images and to perform image recognition and classification. Images used in classical CNN are defined on equi-spaced grids in a Euclidean space. There are 3 essential operations to form a convolutional layer in classical CNN, including convolution with localized filters, rectified linear units (ReLU), and pooling.

Consider an input image f defined on Euclidean equi-spaced grids $\{k\}_{k\in\mathbb{Z}^2}$, and a finite filter h supported on a finite subset $\Omega \subset \mathbb{Z}^2$. The convolution between f and h is defined as

$$(f \otimes h)[m] = \sum_{n\in\mathbb{Z}^2} f[m-n]h[n]. \tag{1}$$

It can be seen that the value of $(f \otimes h)[m]$ is the weighted average of f over the neighbors of m, whose weights are specified by h and neighbors are determined by the configuration of Ω.

ReLU, such as max, sigmoid, or tanh function, introduces non-linearity into the convolutional layer and operates at each pixel. For the pooling operation in classical CNN, spatial pooling is considered as a general practice. It is equivalent to down-sampling that reduces the dimensionality of images and retains important information. When a down-sampling rate is 2, we define it as a stride of 2 in which the filter moves 2 vertices at a time. In general, when stride is k, then the convolution in Eq. (1) can be rewritten as

$$(f \otimes h)[m] = \sum_{n\in\mathbb{Z}^2} f[m-kn]h[kn]. \tag{2}$$

Both convolution and pooling in classic CNN are operated on Euclidean equi-spaced grids, which makes spatial shift and down-sampling straightforward. In the following, we generalize these two operations from Euclidean equi-spaced grids to a triangulated mesh but retain the same properties, e.g., spatial shift-invariance property of convolution and down-sampling.

3.2. Convolution in the vertex domain

Consider a triangulated mesh

$$T = \left(\{x_i\}, \{\Sigma_{ijk}\}\right), \quad i,j,\ k \in \{1,\dots,N\},$$

where $\{x_i\}$ denotes the set of vertex coordinates, $\{\Sigma_{ijk}\}$ is the set of triangles, and N represents the total number of vertices on the mesh. Each simplex Σ_{ijk} is a 3-tuple of vertices $(i,j,k), i,j,k \in \{1,\dots,N\}$ that forms a triangulated face with three vertices x_i, x_j and x_k. The three vertices are one-ring neighbors of each other.

The shift-invariant convolution in Euclidean equi-spaced grids in Eq. (1) cannot be easily achieved on T due to its irregularity. In this study, we consider T as a semi-regular triangulated mesh whose vertex has 6 neighbors. In this case, for vertex, $x_i \in T$, the set of its neighboring vertices can be denoted as $\mathbf{P}_i \subset \{x_i\}$:

$$\mathbf{P}_i = \{P[i,1], P[i,2], \dots, P[i,6]\}.$$

In Euclidean equi-spaced grids, the order of neighbors of each pixel can be easily defined in a clockwise or counter clockwise sequence. For instance, for a pixel of a 2D image, $x_{i,j}$, the order of its neighbors in a clockwise order can be written as $x_{i+1,j}, x_{i+1,j+1}, x_{i,j+1}, x_{i-1,j+1}, x_{i-1,j}, x_{i-1,j-1}, x_{i,j-1}, x_{i+1,j-1}$. However, for vertex, $x_i \in T$, the order of its neighboring vertices, $\mathbf{P}_i$, does not come naturally. In this study, we define the order of neighboring vertices as follows. As illustrated in Fig. 1, we first define a sphere, $\mathscr{S}_i$ (blue in Fig. 1), that passes x_i and approximates the mesh formed by $\mathbf{P}_i$. The tangent plane (orange in Fig. 1) of x_i on T is defined as the tangent plane of x_i on $\mathscr{S}_i$. The xyz-coordinate system of the tangent plane of x_i (red in Fig. 1) is the translation and rotation of the coordinate system of $\mathscr{S}_i$ (black in Fig. 1). We then order these six vertices in a clockwise sequence, where $P[i,1]$ is defined as the vertex whose projection is the closest to the x-axis of the tangent plane of the vertex, x_i. Notice that this ordering method is one definition for mesh orientation. Any other mesh orientation approach can be used to define the order of neighborhood vertices.

We can now define the convolution operation on T analogous to Eq. (1). Consider a 1-ring filter $h \in \mathbb{R}^7$:

$$h = [h[0], h[1], h[2], \dots h[6]]^\top.$$

Then, at vertex x_i, the value of a signal f defined on T convolved by h is computed as

$$(f \otimes h)[i] = \sum_{j=0}^{6} \tilde{f}[i,j]h[j], \tag{3}$$

where $\tilde{f}[i,j]$ denotes the value of f at vertex $P[i,j]$ and $P[i,0] = x_i$.

In a matrix form, we define a matrix $\mathscr{F} \in \mathbb{R}^{7,N}$ as

$$\mathscr{F} = \begin{bmatrix} \tilde{f}[1,0] & \cdots & \tilde{f}[i,0] & \cdots & \tilde{f}[N,0] \\ \tilde{f}[1,1] & \cdots & \tilde{f}[i,1] & \cdots & \tilde{f}[N,1] \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \tilde{f}[1,6] & \cdots & \tilde{f}[i,6] & \cdots & \tilde{f}[N,6] \end{bmatrix}$$

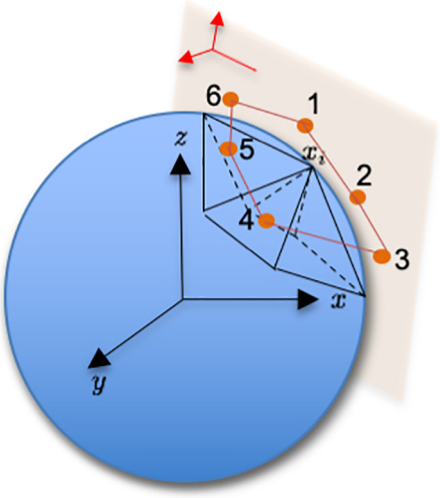

Fig. 1. The ordering of the 1-ring neighbors. The mesh, T, is represented in black. The sphere (blue) approximates the neighbors of vertex x_i and the tangent plane at x_i is in orange. The points colored in orange are the projected points of the neighbors of vertex x_i on its tangent plane.

Then, the convolution defined in Eq. (3) can be expressed in the form of matrix multiplication:

$$f \otimes h: \quad f \in \mathbb{R}^N \to h^\top \mathscr{F} \in \mathbb{R}^N. \tag{4}$$

The corresponding columns of vertices with valence < 6 have non-defined entries. Analogous to classical convolution for finite signals, the values of these non-defined entries can be defined using boundary extension. For example, these entries can be assigned with zero, which is the same as zero-padding boundary extension in classical convolution.

By the same procedure, we can define a 2-ring convolution and more. In general, a k-ring convolution can be parameterized by $3k(k+1)+1$ parameters, and its support covers the same number of vertices. Based on Eq. (3), the convolution on a semi-regular triangulated mesh is consistent with the behavior of classical convolution on equi-space image grids. Localization, parametrization, and shift-invariance properties enable the convolution in the vertex domain to have fast computation and extract local features of the data on a mesh. Such ring-type convolution has also been exploited in wavelet transform for surface denoising (e.g., [11]).

3.3. Rectified linear unit

For classical CNN, a *rectified linear unit* (ReLU) can be represented by many non-linear activation functions. The activation function is a map from $\mathbb{R}$ to $\mathbb{R}$, which does not involve any geometrical property of a triangulated mesh. In our proposed vertex-CNN on a semi-regular mesh, we adopt the well-known ReLU:

$$f(x) = \max\{0, x\}, \quad x \in \mathbb{R}.$$

3.4. Up-sampling of a triangulated mesh

Down-sampling is one important operation in CNN to control the dimensionality of features when building a multi-scale representation of signals via multiple layers. Down-sampling can see its usage in pooling and convolution with a stride > 1. The key of down-sampling signals on a mesh is about how to define a series of hierarchical triangulated meshes:

$$\left\{T^{(0)}, T^{(1)}, T^{(2)}, \ldots, T^{(L)}\right\}$$

such that $T^{(j)}$ at level j contains all vertices of $T^{(j+1)}$ and new vertices, and $T^{(0)}$ is the original mesh T at the finest level. Down-sampling of a triangulated mesh is not straightforward, especially when retaining the property of semi-regularity.

Nevertheless, there are many approaches for up-sampling of triangulated meshes. In this study, we adopt the up-sampling approach proposed in [27] which recursively uses a subdivision scheme to generate new vertices. In details, we subdivide a triangle in $T^{(j+1)}$ with 3 vertices (v_0, v_1, v_2) into 4 smaller ones by introducing 3 new vertices (w_0, w_1, w_2). (w_0, w_1, w_2) are the midpoints of three edges of this triangle. The four new triangles in mesh $T^{(j)}$ are given by

$$(v_0, w_2, w_1), \quad (v_1, w_0, w_2), \quad (v_2, w_1, w_0), \quad (w_0, w_1, w_2).$$

Fig. 2 illustrates this subdivision scheme for up-sampling of a triangulated mesh.

In practice, we start with an initial semi-regular mesh at the coarsest level and recursively apply the subdivision scheme above, which leads to a series of hierarchical semi-regular triangular meshes,

$$T^{(L)} \to T^{(L-1)} \to T^{(L-2)} \to \cdots \to T^{(1)} \to T^{(0)}.$$

The mesh at level $j+1$ is up-sampled by factor of 4 to obtain the mesh at level j. It can be seen that any vertex $x_i^{(j+1)}$ at the $(j+1)$th level mesh $T^{(j+1)}$ remains in the (j)-th level mesh $T^{(j)}$, and its 1-ring neighbors in $T^{(j)}$ are 6 new vertices not in $T^{(j+1)}$.

With such a hierarchical triangular mesh in hand, one can define the convolution with a stride of $2, 4$, and etc. We take the convolution with stride 2 as example. Let $f^{(j)}$ denote the signal defined on the mesh $T^{(j)}$. Then, the convolution with a stride of 2 is defined by

$$\left(f^{(j)} \otimes h\right)_{s=2} = f^{(j+1)} \otimes h.$$

This is similar to the one in Euclidean equi-spaced grids in Eq. (2).

3.5. Pooling

Pooling in classical CNN, viewed as a non-linear or linear down-sampling operation, aims to reduce the dimensionality of representation and thus to reduce the number of parameters. This facilitates memory usage, increases computational efficiency, and controls over-fitting. In general, pooling is achieved by either taking the maximum or taking the averaged value of the neighbors of the vertices lying on a coarser grid/mesh.

We now define the pooling operation for the vertex-CNN. Suppose that we have a series of hierarchical semi-regular triangulated meshes constructed in the previous section:

$$T^{(L)} \to T^{(L-1)} \to T^{(L-2)} \to \cdots \to T^{(1)} \to T^{(0)}.$$

Recall that any vertex $x_i^{(j+1)}$ at the $(j+1)$-th level mesh $T^{(j+1)}$ remains in the (j)th level mesh $T^{(j)}$, and its 1-ring neighbors in $T^{(j)}$ are 6 new vertices not in $T^{(j+1)}$. Let $\Omega_i^{(j)}$ denote these new 1-ring neighbors. Then, the pooling operator with a stride of 2 is defined as

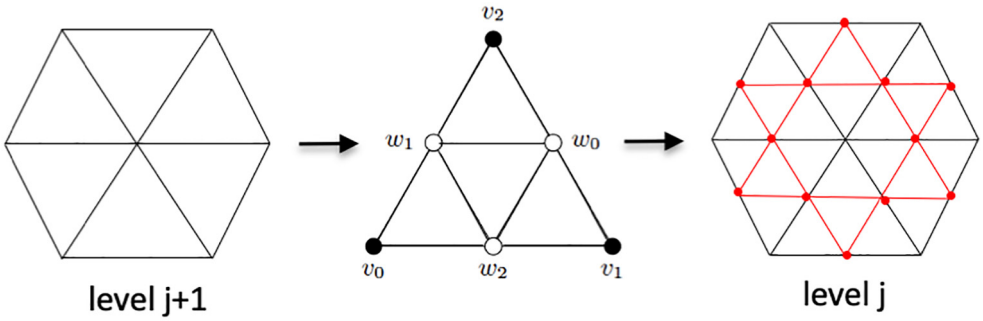

Fig. 2. The subdivision scheme of a triangulated mesh. A triangle of (v_0, v_1, v_2) is divided into four triangles by introducing the midpoint of each edge of this triangle, w_0, w_1, w_2, and their edges w_0w_1, w_0w_2, w_1w_2.

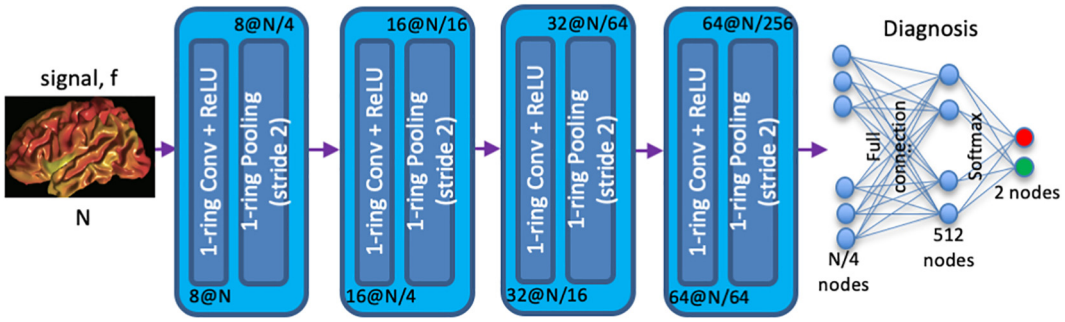

Fig. 3. The architecture of the vertex-CNN on a semi-regular triangulated mesh. This vertex-CNN network contains four layers that respectively include 8, 16, 32, and 64 channels. $N = 163842$ represents the number of vertices on the input mesh.

$$\text{Mean pooling}: \quad f[i]^{(j+1)} \longleftarrow \frac{1}{7}\sum_{r\in\Omega_i^{(j)}} f[r]^{(j)},$$

or

$$\text{Max pooling}: \quad f[i]^{(j+1)} \longleftarrow \max_{r\in\Omega_i^{(j)}} f[r]^{(j)},$$

where $f[i]^{(j+1)}$ denotes the value at vertex x_i in the $j+1$th level mesh $T^{(j+1)}$. Similarly, we can define the pooling operator with a stride of 4 (and more) by running the same procedure on the vertices and its 1-ring and 2-ring neighbors (and more) in the next finer level mesh.

3.6. Architecture and implementation of the vertex-CNN

We are now well equipped with all the components for a vertex-CNN network. Fig. 3 illustrates one of vertex-CNN architectures. The vertex-CNN network is composed of $L+1$ connected stages, $M^{(1)},\ldots,M^{(L)},M^{(L+1)}$. The first L stages are the stages for feature extraction. Each stage contains three layers: (1) a convolutional layer with multiple 1-ring filters; (2) a ReLU layer; (3) a pooling layer with a stride of 2 that uses mean pooling:

$$M^{(\ell)}: \text{input} \rightarrow \text{Convolution} \rightarrow \text{ReLU} \rightarrow \text{Mean Pooling} \rightarrow \text{Output}$$

The last stage $M^{(L+1)}$ is the fully connected layer whose nodes are assigned with the values of the vertices on the mesh at stage $M^{(L)}$. Finally, the classification layer uses the features extracted in the previous stages as input and outputs classification labels based on a softmax function.

In practice, the input mesh, $T^{(L)}$, is extracted from image data. We then prepare a series of hierarchical triangulated meshes, $T^{(L-1)},T^{(L-2)},\ldots,T^{(0)}$, according to the subdivision scheme introduced in Section 3.4. These meshes correspond to the mesh in each stage. The 1-ring convolution operation is implemented in the form of matrix multiplication. The ReLU and pooling layers are implemented using the standard procedure of classical CNN. The vertex-CNN is trained using the Adam optimization algorithm. We implement this vertex-CNN on a semi-regular triangulated mesh in Tensorflow.

4. Results

In this section, we employed MRI scans from ADNI and compared the vertex-CNN with existing graph methods in terms of computational time and classification performance. We chose spectral graph CNN in [9] because its convolutional filters were defined in the spectral domain while the convolutional filters of the vertex-CNN were defined in the spatial domain. Hence, the vertex-CNN and spectral graph CNN [9] were comparable. We then explored the classification accuracy and robustness of a single-channel vertex-CNN and a multiple-channel vertex-CNN as well as the prediction of AD as a function of the AD onset.

4.1. MRI data and analysis

4.1.1. ADNI cohort

This study employed data from the ADNI cohort. The ADNI-1 cohort included 1013 subjects (243 cognitive normal (CON), 415 mild cognitive impairment (MCI), and 355 Alzheimer's disease (AD)). The ADNI-2 cohort included 1149 subjects (400 CON, 488 MCI, and 261 AD). For each ADNI cohort, the number of visits of each subject varied from 1 to 7 (i.e., baseline, 3-, 6-, 12-, 24-, 36-, and 48-month). At each visit, subjects were diagnosed as one of the three clinical statuses based on the criteria described in the ADNI protocol (http://adni.loni.usc.edu). The demographic information of the subjects from ADNI-1 and ADNI-2 is provided in Table 1.

4.1.2. MRI acquisition and analysis

Structural T1-weighted MRI scans were acquired using either 1.5T or 3T scanners. For the 1.5T scanners, the imaging protocol followed: repetition time (TR) = 2400 ms, minimum full echo time (TE), inversion time (TI) = 1000 ms, flip angle = 8°, field-of-view (FOV) = 240 × 240 mm^2, acquisition matrix = 256 × 256 × 170 in the x-, y-, and z-dimensions, yielding a voxel size of 1.25 × 1.25 × 1.2 mm^3. For the 3T scanners, the imaging protocol was set to be: TR = 2300 ms, minimum full TE, TI = 900 ms, flip

Table 1
Demographic information of the ADNI cohort with MRI scans.

	CON	MCI	AD
ADNI-1			
Number of subjects*	243	415	355
Number of MRI scans	1067	1515	1016
Female/Male	493/574	525/990	443/573
Age (Mean ± SD)	76.8 ± 5.3	75.9 ± 7.3	76.3 ± 7.2
ADNI-2			
Number of subjects*	400	488	261
Number of MRI scans	1122	1460	587
Female/Male	607/515	663/797	254/333
Age (Mean ± SD)	75.3 ± 6.8	73 ± 7.7	75.3 ± 7.7

Abbreviations. CON: Control normal; MCI: Mild cognitive impairment; AD: Alzheimer's disease.
The number of subjects for each group was based on the clinical status during the MRI acquisition visit. There were subjects who fall into 2 or more groups due to the conversion from one clinical status to another.

angle = 8°, FOV = 260 × 260 mm^2, acquisition matrix = 256 × 256 × 170, yielding a voxel size of 1.0 × 1.0 × 1.2 mm^3.

All T1-weighted images were segmented using FreeSurfer (version 5.3.0) [17]. The processed images were quality checked based on the criteria listed on https://surfer.nmr.mgh.harvard.edu/fswiki/FsTutorial. We represented cortical thickness, surface area, volume, curvature, Jacobian, and sulcal depth on the cortical surface generated by FreeSurfer. We employed large deformation diffeomorphic metric mapping (LDDMM) algorithm [50,12] to align individual cortical surfaces to the atlas and transferred these cortical measures of each subject to the atlas.

As each subject may have multiple MRI scans, one at each visit, this study included all available T1-weighted images with good quality after processing. We used the clinical status at the MRI acquisition as the classification ground truth. For instance, a subject with multiple scans may have different clinical labels if he/she was from one clinical status to another over time. From 3783 scans from ADNI-1, we discarded 185 scans that missed demographic information or diagnosis labels, resulting in 3598 scans used below. From 3365 scans from ADNI-2, we discarded 196 scans that missed demographic information or diagnosis labels, resulting in 3169 scans used in the following CNN analysis.

4.2. Comparison of computational cost with spectral graph CNN

We compared the computational cost between the vertex-CNN and spectral graph CNN [9] by employing cortical thickness data from the ADNI cohort. The architecture of the vertex-CNN and spectral graph CNN [9] was similar to that illustrated in Fig. 3. Both the vertex-CNN and spectral graph CNN [9] were designed with 3 CNN layers, a fully connected layer, and a finally connected layer with 128 nodes, and 1 classification stage. The convolution in the spectral graph CNN was approximated using Chebyshev polynomial with an order of 3. The network was trained based on 1024 scans from ADNI-2. The network parameters were trained with a mini-batch size of 64, an initial learning rate of $1e^{-3}$, and 10 epochs. The experiments were run on Tesla V100 GPU (32 GB memory). We compared the two networks when the number of filters in each layer was 10, 20, 30, 40, 50, 60. As shown in Fig. 4, both vertex-CNN and spectral graph CNN increased the computational cost when the number of filters in each layer increased. The vertex-CNN was approximately 3 times faster than the spectral graph CNN.

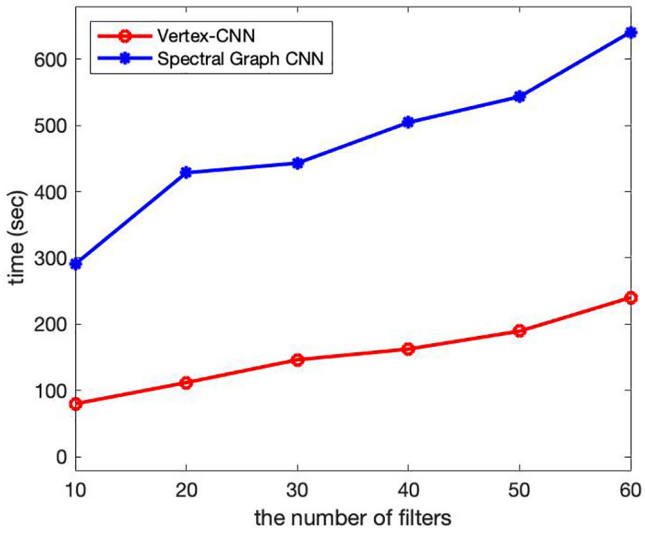

Fig. 4. The computation cost of the vertex-CNN and spectral graph CNN as a function of the number of filters in each layer.

4.3. Comparison of classification accuracy with spectral graph CNN

We compared the classification accuracy between the vertex-CNN and spectral graph CNN [9] based on the ADNI-2 data. Again, the cortical thickness data from ADNI-2 was used in this experiment. The architecture of the vertex-CNN and spectral graph CNN was similar to that in Fig. 3. We tuned their network parameters, including the number of layers and the number of filters, and a learning rate, based on geometric mean (GMean = $\sqrt{SEN \times SPE}$, where SEN and SPE respectively represent sensitivity (SEN) and specificity (SPE). We chose this measure because it not only maximizes the accuracy of each of the two classes but also minimizes the difference between the sensitivity and specificity, i.e., the balanced performance for both the positive and negative classes. Hence, the network parameters were determined so that each network achieved its best balanced sensitivity and specificity. Based on GMean, the vertex-CNN was designed with 4 CNN layers and 1 classification stage. The numbers of the filters in these convolutional layers were $[8, 16, 32, 64]$, respectively. The classification layer was implemented using a fully connected layer with 512 nodes and with a softmax output. The spectral graph CNN [9] incorporated 3 CNN layers with the number of filters of [8,16,32] respectively and a final fully connected layer with 128 nodes. The convolution in the spectral graph CNN was approximated using Chebyshev polynomial with an order of 3. The network parameters were trained with a mini-batch size of 64, an initial learning rate of $1e^{-3}$, a weight decay of 0.05, and a momentum of 0.9. During the training process, a l_2-norm regularization function of $5e^{-4}$ was applied as the weight of the final fully connected layer to prevent overfitting to the training data.

This study employed the 10-fold cross-validation, where the scans from the same subject were assigned to the validation (or testing) to avoid the data leakage issue in the predictive model. We performed the same procedure as mentioned above for three classifiers, including CON vs. AD, CON vs. MCI, and MCI vs. AD. Table 2 lists the classification accuracy, sensitivity, specificity, and GMean for each classifier. Our proposed vertex-CNN performed better than the spectral graph CNN in all the three classifiers in terms of classification accuracy, sensitivity, and GMean. Moreover, our vertex-CNN approach provided a relatively lower variability across all the four evaluation measures.

4.4. Multi-channel vertex-CNN for MCI and AD classification

In this study, we investigated the robustness of the vertex-CNN for the classification of MCI and AD. For this, we designed one-channel vertex-CNN (see Fig. 3) and six-channel vertex-CNN (see Fig. 5). The ADNI-2 data (n = 3169) were used to train the network. The ADNI-1 data (n = 3598) were used to examine the robustness of the vertex-CNN.

First, we applied the one-channel vertex-CNN to six individual cortical measures, including cortical thickness, surface area, volume, Jacobian, curvature, and sulcal depth. The one-channel vertex-CNN was optimized based on the abovementioned procedure. Based on the ADNI-2 data and the 10-fold cross-validation, the one-channel vertex-CNN on cortical thickness, in general, outperformed the one-channel vertex-CNNs on the other five cortical measures in terms of the classification accuracies of CON vs AD, and MCI vs AD (Table 3).

We directly applied these trained one-channel vertex-CNNs to the ADNI-1 data and showed a similar trend that the one-channel vertex-CNN on thickness classified MCI and AD better than the other five one-channel vertex-CNNs (Table 4).

Second, we designed a six-channel vertex-CNN based on the cortical measures, including cortical thickness, surface area, vol-

Table 2
Comparison between vertex-CNN and spectral graph CNN in terms of accuracy (ACC), sensitivity (SEN), specificity (SPE), and geometric mean (GMean). These results were based on cortical thickness from the ADNI-2 data.

Model	Task	ACC (%)	SEN (%)	SPE (%)	GMean (%)
vertex-CNN	CON vs. AD	**89.2** ± **0.5**	**85.2** ± **1.0**	**91.3** ± **0.6**	**88.2** ± **0.5**
	CON vs. MCI	**73.3** ± **1.1**	**67.5** ± **2.5**	**76.4** ± **1.7**	**71.8** ± **1.2**
	MCI vs. AD	**65.4** ± **1.3**	**66.5** ± **2.2**	64.4 ± 2.7	**65.4** ± **1.4**
spectral graph CNN	CON vs. AD	85.8 ± 0.8	83.5 ± 3.2	87.5 ± 2.8	85.4 ± 0.8
	CON vs. MCI	69.3 ± 2.2	65.6 ± 7.6	72.0 ± 5.4	68.5 ± 3.0
	MCI vs. AD	65.2 ± 1.6	62.6 ± 5.2	**68.0** ± **6.6**	65.3 ± 1.4

Abbreviations. CON: control normal; AD: Alzheimer's disease; MCI: mild cognitive impairment; ACC: accuracy; SEN: sensitivity; SPE: specificity; GMean: geometric mean.

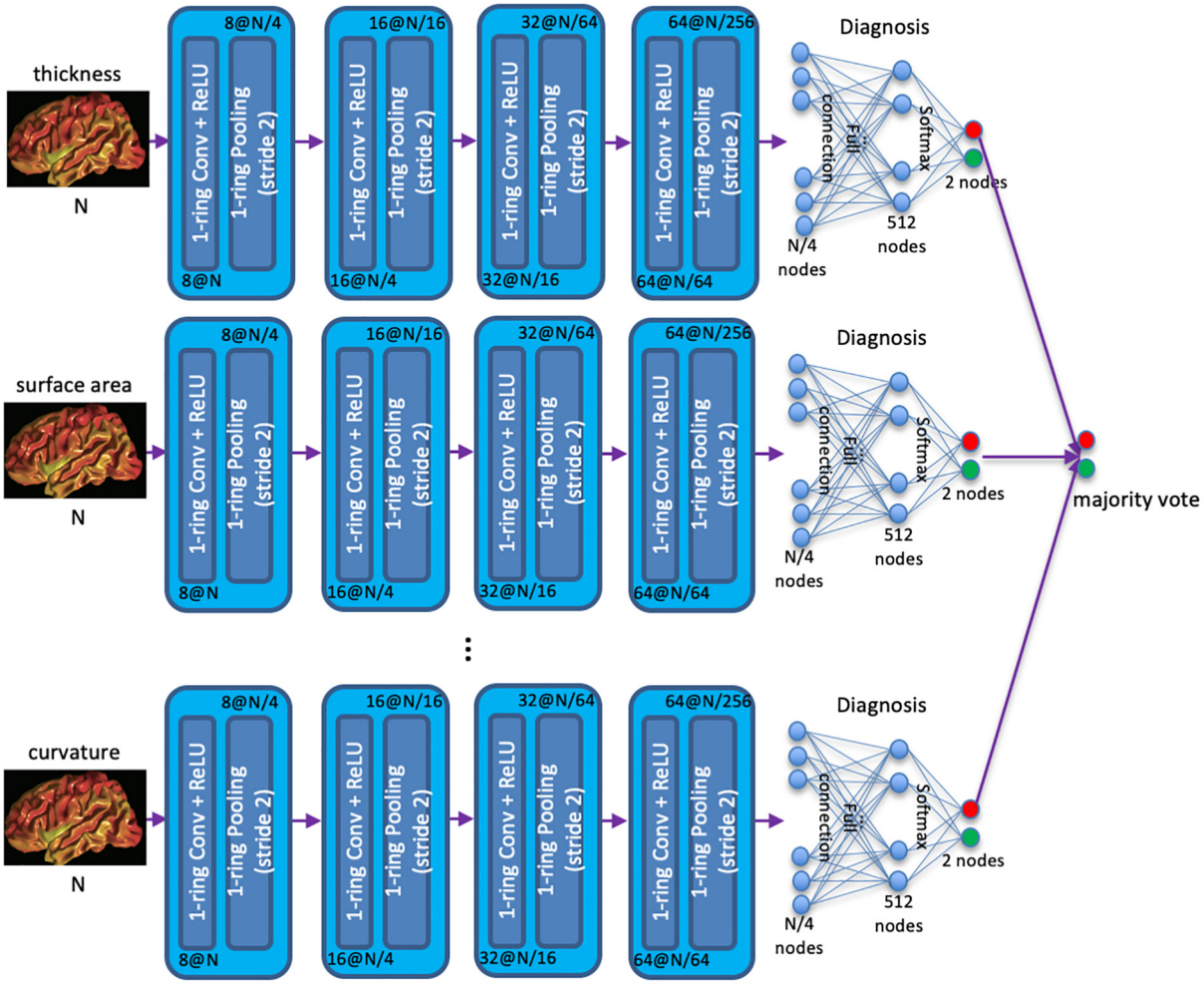

Fig. 5. The architecture of the vertex-CNN with multiple channels on a semi-regular triangulated mesh. Each channel of this network contains four layers that respectively include 8, 16, 32, and 64 filters. $N = 163842$ represents the number of vertices on the input mesh.

ume, Jacobian, curvature, and sulcal depth. Individual channel vertex-CNN was optimized based on the above procedure. We assembled the classification results from the six channels via a majority vote. While based on the above findings (Table 3), the channel on cortical thickness took four votes and the rest channels only took one vote each. Table 5 lists the classification accuracy for CON and AD, CON and MCI, and MCI and AD of the ADNI-2 data based on the six-channel vertex-CNN and the 10-fold cross-validation.

Table 6 shows the robustness of the six-channel vertex-CNN directly applying to the ADNI-1 data. Note that the classification accuracy for CON vs MCI in the ADNI-1 data was higher than that in the ADNI-2, and it was the opposite for the classification of MCI vs AD. This may partly because the MCI subjects in the ADNI-1 dataset were relatively more severe than those in the ADNI-2 dataset.

4.5. Prediction of MCI cconversion to AD

We examined whether the CON vs AD six-channel vertex-CNN classifier built based on the ADNI-2 dataset can be used to predict the conversion of MCI to AD. As mentioned above, the training of this classifier was done via the 10-fold cross-validation. We chose the CON vs AD classifier that best performed over the 10-fold validations based on GMean. We then applied it to the cortical measures of 23 early MCI (55 scans) and 48 late MCI (137 scans)

Table 3
The classification accuracy of one channel vertex-CNN based on cortical measures of the ADNI-2 dataset (n = 3169), including cortical thickness, surface area, curvature, Jacobian, sulcal depth, and volume.

Task	Channel	ACC (%)	SEN (%)	SPE (%)	GMean (%)
CON vs. AD	Thickness	**89.2 ± 0.5**	**85.2 ± 1.0**	**91.3 ± 0.6**	**88.2 ± 0.5**
	surface area	79.9 ± 0.4	68.4 ± 1.7	85.9 ± 0.8	76.7 ± 0.8
	curvature	80.0 ± 1.0	77.2 ± 1.6	81.4 ± 1.2	79.3 ± 1.1
	Jacobian	81.8 ± 0.9	79.1 ± 1.8	83.2 ± 0.4	81.1 ± 1.1
	sulcal depth	81.1 ± 0.3	79.5 ± 1.5	81.9 ± 0.8	80.7 ± 0.5
	volume	88.2 ± 0.4	84.3 ± 1.3	90.2 ± 0.9	87.2 ± 0.4
CON vs. MCI	Thickness	71.8 ± 0.8	72.8 ± 1.1	70.5 ± 1.7	71.7 ± 0.8
	surface area	69.7 ± 2.1	69.8 ± 2.4	69.5 ± 2.3	69.6 ± 2.0
	curvature	**80.6 ± 1.0**	**80.0 ± 1.4**	**81.4 ± 1.6**	**80.7 ± 1.0**
	Jacobian	72.6 ± 1.2	72.2 ± 1.2	73.0 ± 3.0	72.6 ± 2.4
	sulcal depth	74.3 ± 1.0	76.2 ± 1.7	71.7 ± 1.8	73.9 ± 1.0
	volume	72.9 ± 1.1	74.3 ± 1.8	71.1 ± 2.6	72.7 ± 1.2
MCI vs. AD	thickness	**75.7 ± 0.7**	67.8 ± 1.7	**78.8 ± 1.5**	**73.1 ± 0.6**
	surface area	65.2 ± 0.9	55.8 ± 2.7	68.9 ± 2.1	62.0 ± 0.8
	curvature	60.4 ± 1.4	55.4 ± 4.2	62.4 ± 3.6	58.7 ± 0.7
	Jacobian	65.1 ± 1.1	59.9 ± 2.1	67.2 ± 1.4	63.4 ± 1.2
	sulcal depth	64.1 ± 0.5	60.0 ± 1.3	65.8 ± 0.7	62.8 ± 0.6
	volume	73.0 ± 1.0	**68.8 ± 2.3**	74.6 ± 2.0	71.6 ± 0.6

Table 4
The robustness of one channel vertex-CNN trained based on cortical measures of the ADNI-2 data (n = 3169) and applied to the ADNI-1 data (n = 3598).

Task	Channel	ACC (%)	SEN (%)	SPE (%)	GMean(%)
CON vs. AD	Thickness	**88.6**	82.6	94.4	**88.3**
	surface area	77.0	61.5	91.7	75.1
	curvature	81.9	76.6	87.0	81.6
	Jacobian	84.0	**85.1**	82.9	84.0
	Sulcal depth	77.1	66.0	87.7	76.1
	Volume	87.7	77.5	**97.4**	86.8
CON vs. MCI	Thickness	**80.3**	**87.5**	70.0	78.3
	Surface area	74.6	69.5	81.7	75.4
	Curvature	77.9	67.2	**93.2**	79.1
	Jacobian	73.7	73.3	74.3	73.8
	Sulcal depth	76.6	76.8	76.3	76.6
	Volume	73.4	61.7	90.1	**85.0**
MCI vs. AD	Thickness	**66.8**	72.6	62.8	**67.6**
	Surface area	53.1	**73.3**	39.6	53.9
	Curvature	56.0	55.3	56.5	55.9
	Jacobian	63.0	58.0	66.4	62.0
	Sulcal depth	62.1	65.3	60.0	62.6
	Volume	67.5	65.0	**69.2**	67.1

Table 5
The classification accuracy of the six-channel vertex-CNN based on the ADNI-2 data and 10-fold cross-validation (n = 3169). The six-channel vertex-CNN incorporated cortical thickness, surface area, volume, Jacobian, curvature, and sulcal depth data.

Task	ACC (%)	SEN (%)	SPE (%)	GMean (%)
CON vs. AD	90.2 ± 0.3	86.1 ± 0.8	92.3 ± 0.5	89.1 ± 0.4
CON vs. MCI	75.1 ± 0.8	75.5 ± 1.1	74.7 ± 1.4	75.1 ± 0.8
MCI vs. AD	76.0 ± 0.6	67.1 ± 1.4	79.5 ± 1.3	73.0 ± 0.4

Table 6
The robustness of the six-channel vertex-CNN directly applying to the ADNI-1 data.

Task	ACC (%)	SEN (%)	SPE (%)	GMean (%)
CON vs. AD	89.5	83.1	95.7	89.2
CON vs. MCI	82.2	85.9	77.0	81.3
MCI vs. AD	67.1	72.7	63.4	67.9

subjects who were subsequently diagnosed as AD in the follow-up visits. The six-channel vertex-CNN, trained for the CN vs. AD task, correctly predicted the conversion of 47/55 early MCI scans (85.5%) and 129/137 (94.2%) late MCI scans to AD. Fig. 6 illustrates the prediction accuracy as a function of the time interval for the MCI conversion to AD, suggesting that as the time interval increased, the prediction accuracy decreased. Note that the prediction accuracy is high partly because a subset of the AD subjects were used in the training of the CON vs AD multi-channel vertex CNN classifier due to the limited sample of AD subjects.

We further evaluated the robustness of the CON vs AD six-channel vertex-CNN classifier built based on the ADNI-2 dataset by applying it to the ADNI-1 dataset. In the ADNI-1 dataset, 166 MCI subjects were converted to AD at the follow-up visits and

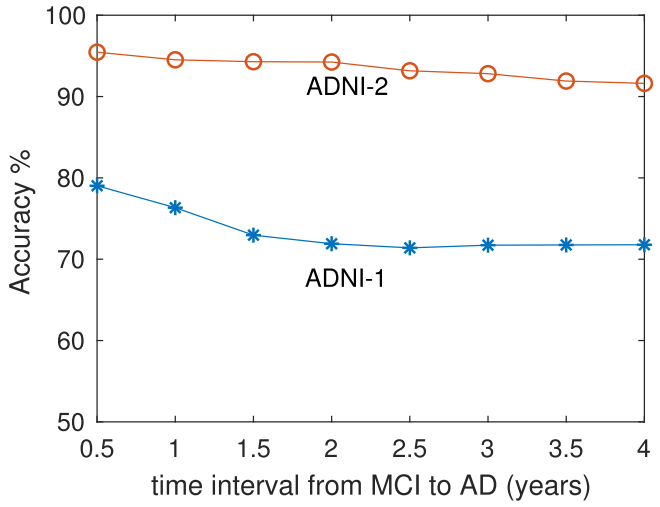

Fig. 6. The relationship of the prediction accuracy and time interval of the MCI conversion to AD in the ADNI-1 (blue line) and ADNI-2 (red line) datasets. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

there were total 498 MRI scans. The CON vs AD six-channel vertex-CNN classifier predicted the conversion of MCI to AD at the accuracy rate of 359/498 (72.1%), comparable to existing results (see review in [34,8]). Furthermore, Fig. 6 illustrates the prediction accuracy as a function of the time interval from MCI to AD. Again, this suggests that as the time interval increased, the prediction accuracy decreased. This trend was similar to that shown in the ADNI-2 dataset.

5. Discussion

This paper presents a vertex-CNN for structured data on semi-regular triangulated meshes. The proposed network exploits the geometric property of semi-regular triangulated meshes (i.e., each vertex has 6 connected neighbors) to define the operations of convolution and pooling. The implementation of the vertex-CNN is simple and its computation is elegant and efficient. We compared it with an existing graph CNN [9] that is in parallel with the vertex-CNN. We provided a wide series of carefully designed experiments. Our study included the full set of the ADNI data and demonstrated the classification performance and robustness of the vertex-CNN for the application of brain images.

Our results suggest that our vertex-CNN algorithm is faster than the spectral graph CNN [9]. This is partly because the convolution operation in vertex-CNN can be simply computed via matrix multiplication while the convolution in spectral graph CNN [9] is approximated via Chebyshev polynomials. Also, the mesh coarsening procedure used in the pooling of the spectral graph CNN [9] requires the addition of fake vertices with the singletons such that each vertex has two children. This procedure increases the dimension of the graph used in the spectral graph CNN. In contrast, the pooling operation in our proposed vertex-CNN is with a stride of 2, similar to that of dyadic down-sampling on equi-spaced Euclidean grids. Compared to the spectral graph CNN [9], our vertex-CNN improves the classification accuracies of MCI and AD.

We applied this vertex-CNN to the evaluation of its accuracy and robustness for the diagnosis of MCI and AD and the prediction of the MCI conversion to AD. In the past decades, substantial studies reported the classification among CON, MCI, and AD based on the ADNI dataset (e.g., [26,33,14,51,46]. Some of them were based on multi-modal brain images and reported the classification accuracy better than that in Table 2 (e.g., [33,14,51,46]). But the sample size was relatively small hence the robustness of the classification results is unclear. Also, most of the existing studies based on the T1-weighted MRI and deep neural networks reported the classification accuracy derived from one subset of the study sample, which may be biased to the subsample chosen (see review in [3]). Our study overcame this via evaluating our results based on 10-fold cross-validation and independent training (ADNI-2) and testing samples (ADNI-1). This study, to our best knowledge, is the first one that incorporated the data from ADNI-1 and ADNI-2 and hence produced the reliable and robust classification and prediction results.

One limitation of our proposed approach is that it requires meshes to be semi-regular. In general, the construction of semi-regular meshes for 3D image data is not an issue. However, our approach cannot be easily generalized to analyze data on general graphs, such as social and citation networks [42]. Similarly, our approach may not be able to apply to point clouds [49], graph search [20,48], deep feature search [47], etc.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

We like to thank the National Supercomputing Centre Singapore for providing the computing resource for this study. This research/project is supported by the National Research Foundation, Singapore under its AI Singapore Programme (AISG Award No: AISG-GC-2019–002). Additional funding is provided by the Singapore Ministry of Education (Academic research fund Tier 1; NUHSRO/2017/052/T1-SRP-Partnership/01), NUS Institute of Data Science at the National University of Singapore.

References

[1] L.G. Apostolova, I.D. Dinov, R.A. Dutton, K.M. Hayashi, A.W. Toga, J.L. Cummings, P.M. Thompson, 3D comparison of hippocampal atrophy in amnestic mild cognitive impairment and alzheimer's disease, Brain 129 (2006) 2867–2873.
[2] J. Atwood, D. Towsley, Diffusion-convolutional neural networks. arXiv preprint arXiv:1511.02136, 2015..
[3] S. Basaia, F. Agosta, L. Wagner, E. Canu, G. Magnani, R. Santangelo, M. Filippi, A. D.N. Initiative, Automated classification of alzheimer's disease and mild cognitive impairment using a single mri and deep neural networks, NeuroImage: Clin. 21 (2019) 101645..
[4] D. Boscaini, J. Masci, S. Melzi, M.M. Bronstein, U. Castellani, P. Vandergheynst, Learning class-specific descriptors for deformable shapes using localized spectral convolutional networks, Comput. Graph. Forum 34 (2015) 13–23.
[5] D. Boscaini, J. Masci, E. Rodoia, M. Bronstein, Learning shape correspondence with anisotropic convolutional neural networks, in: NIPS'16 Proceedings of the 30th International Conference on Neural Information Processing Systems, ACM, 2016, pp. 3197–3205.
[6] D. Boscaini, J. Masci, E. Rodola, M.M. Bronstein, D. Cremers, Anisotropic diffusion descriptors, Comput. Graph. Forum 35 (2016) 431–441.
[7] J. Bruna, W. Zaremba, A. Szlam, Y. LeCun, Spectral networks and locally connected networks on graphs. arXiv preprint arXiv:1312.6203, 2013..
[8] R. Cuingnet, E. Gerardin, J. Tessieras, G. Auzias, S. Lehéricy, M.O. Habert, M. Chupin, H. Benali, O. Colliot, T.A.D.N. Initiative, Automatic classification of patients with alzheimer's disease from structural mri: a comparison of ten methods using the adni database, Neuroimage 56 (2011) 766–781.
[9] M. Defferrard, X. Bresson, P. Vandergheynst, Convolutional neural networks on graphs with fast localized spectral filtering, in: Proceedings of the 30th International Conference on Neural Information Processing Systems, 2016, pp. 3844–3852.
[10] I.S. Dhillon, Y. Guan, B. Kulis, Weighted graph cuts without eigenvectors a multilevel approach, IEEE Trans. Pattern Anal. Mach. Intell. 29 (2007) 1944–1957.
[11] B. Dong, Q. Jiang, C. Liu, Z. Shen, Multiscale representation of surfaces by tight wavelet frames with applications to denoising, Appl. Comput. Harmonic Anal. 41 (2016) 561–589.

[12] J. Du, L. Younes, A. Qiu, Whole brain diffeomorphic metric mapping via integration of sulcal and gyral curves, cortical surfaces, and images, NeuroImage 56 (2011) 162–173.
[13] D.K. Duvenaud, D. Maclaurin, J. Aguilera-Iparraguirre, R. Gomez-Bombarelli, T. Hirzel, A. Aspuru-Guzik, R.P. Adams, Convolutional networks on graphs for learning molecular fingerprints. arXiv preprint arXiv:1509.09292, 2015..
[14] M. Dyrba, F. Barkhof, A. Fellgiebel, M. Filippi, L. Hausner, K. Hauenstein, et al., Predicting prodromal alzheimer's disease in subjects with mild cognitive impairment using machine learning classification of multimodal multicenter diffusion-tensor and magnetic resonance imaging data, J. Neuroimag. 25 (2015) 738–747.
[15] A. Esteva, B. Kuprel, R.A. Novoa, J. Ko, S.M. Swetter, H.M. Blau, S. Thrun, Dermatologist-level classification of skin cancer with deep neural networks, Nature 542 (2017) 115.
[16] Y. Fan, R. Gur, R. Gur, X. Wu, D. Shen, M. Calkins, C. Davatzikos, Unaffected family members and schizophrenia patients share brain structure patterns: a high-dimensional pattern classification study, Biol. Psychiatry 63 (2008) 118–124.
[17] B. Fischl, D.H. Salat, E. Busa, M. Albert, M. Dieterich, C. Haselgrove, A. van der Kouwe, R. Killiany, D. Kennedy, S. Klaveness, A. Montillo, N. Makris, B. Rosen, A. M. Dale, Whole brain segmentation: automated labeling of neuroanatomical structures in the human brain, Neuron 33 (2002) 341–355.
[18] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proc. IEEE Conf. Comput. Vision Pattern Recognition, 2016, pp. 770–778..
[19] M. Henaff, J. Bruna, Y. LeCun, Deep convolutional networks on graph-structured data. arXiv preprint arXiv:1506.05163, 2015..
[20] K. Hu, B. Shen, Y. Zhang, C. Cao, F. Xiao, X. Gao, Automatic segmentation of retinal layer boundaries in oct images using multiscale convolutional neural network and graph search, Neurocomputing 365 (2019) 302–313.
[21] T.N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907, 2016..
[22] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, in: Proc. Advances in Neural Inform. Process. Syst., 2012, pp. 1097–1105..
[23] S.I. Ktena, S. Parisot, E. Ferrante, M. Rajchl, M. Lee, B. Glocker, D. Rueckert, Distance metric learning using graph convolutional networks: application to functional brain networks. arXiv preprint arXiv:1703.02161, 2017..
[24] Y. Li, D. Tarlow, M. Brockschmidt, R. Zemel, Gated graph sequence neural networks. arXiv preprint arXiv:1511.05493, 2015..
[25] G. Litjens, T. Kooi, B.E. Bejnordi, A.A.A. Setio, F. Ciompi, M. Ghafoorian, J.A. Van Der Laak, B. Van Ginneken, C.I. Sánchez, A survey on deep learning in medical image analysis, Med. Image Anal. 42 (2017) 60–88.
[26] X. Liu, D. Tosun, M. Weiner, N. Schuff, for the Alzheimer's Disease Neuroimaging Initiative, Locally linear embedding (lle) for mri based alzheimer's disease classification, Neuroimage 83 (2013) 148–157..
[27] M. Lounsbery, T.D. DeRose, J. Warren, Multiresolution analysis for surfaces of arbitrary topological type, ACM Trans. Graph. (TOG) 16 (1997) 34–73.
[28] J. Masci, D. Boscaini, M.M. Bronstein, P. Vandergheynst, Geodesic convolutional neural networks on riemannian manifolds, in: 2015 IEEE International Conference on Computer Vision (ICCV), IEEE, pp. 832–840.
[29] F. Milletari, N. Navab, S.A. Ahmadi, V-net: fully convolutional neural networks for volumetric medical image segmentation, in: 2016 Fourth International Conference on 3D Vision (3DV), 2016, IEEE, pp. 565–571.
[30] F. Monti, D. Boscaini, J. Masci, E. ROdola, J. Svoboda, M.M. Bronstein, Geometric deep learning on graphs and manifolds using mixture model cnns. arXiv preprint arXiv:1611.08402, 2016..
[31] F. Monti, D. Boscaini, J. Masci, E. Rodolá, J. Svoboda, M.M. Bronstein, Geometric deep learning on graphs and manifolds using mixture model cnns. arXiv preprint arXiv:1611.08402, 2016..
[32] M. Niepert, M. Ahmed, K. Kutzkov, Learning convolutional neural networks for graphs, in: Proceeding of the 33rd International Conference on Machine Learning, ACM, 2016, pp. 2014–2023..
[33] T. Nir, J. Villalon-Reina, G. Prasad, N. Jahanshad, S. Joshi, A.E.A. Toga, Dti-based maximum density path analysis and classification of alzheimer's disease, Neurobiol. Aging 36 (2017) S132–S140..
[34] E. Pellegrini, L. Ballerini, M. del C Valdes Hernandez, F.M. Chappell, V. Gonzalez-Castro, D. Anblagan, S. Danso, S.M. noz Maniega, D. Job, C. Pernet, G. Mair, T.J. MacGillivray, E. Trucco, J.M. Wardlaw, Machine learning of neuroimaging for assisted diagnosis of cognitive impairment and dementia: a systematic review. Alzheimer's Dementia Diagn., Assess., Disease Monitor. 10 (2018) 519–535..
[35] B. Perozzi, R. Al-Rfou, S. Skiena, Deepwalk: Online learning of social representations, in: Proceedings of the 20th ACM SIGKDD, 2014, ACM, pp. 701–710.
[36] A. Qiu, C. Fennema-Notestine, A. Dale, M. Miller, the Alzheimer's Disease Neuroimaging Initiative, Regional shape abnormalities in mild cognitive impairment and Alzheimer's disease, Neuroimage 45 (2009) 656–661..
[37] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only look once: unified, real-time object detection, in: Proc. IEEE Conf. Comput. Vision Pattern Recognition, 2016, pp. 779–788..
[38] H.C. Shin, H.R. Roth, M. Gao, L. Lu, Z. Xu, I. Nogues, J. Yao, D. Mollura, R.M. Summers, Deep convolutional neural networks for computer-aided detection: Cnn architectures, dataset characteristics and transfer learning, IEEE Trans. Med. Imag. 35 (2016) 1285–1298.
[39] D.I. Shuman, B. Ricaud, P. Vandergheynst, Vertex-frequency analysis on graphs, Appl. Comput. Harmonic Anal. 40 (2016) 260–291.
[40] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556, 2014..
[41] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, in: Proc. IEEE Conf. Comput. Vision Pattern Recognition, 2015, pp. 1–9..
[42] Q. Tan, N. Liu, X. Hu, Deep representation learning for social network analysis, Frontiers in Big Data (2019).
[43] J. Xie, L. Xu, E. Chen, Image denoising and inpainting with deep neural networks, Adv. Neural Inf. Process. Syst. (2012) 341–349.
[44] X. Yang, A. Goh, S. Chen, A. Qiu, Evolution of hippocampal shapes across the human lifespan, Hum Brain Mapp. 34 (2013) 3075–3085.
[45] L. Yi, H. Su, X. Guo, L. Guibas, Syncspeccnn: synchronized spectral cnn for 3d shape segmentation, in: Conference on Computer Vision and Pattern Recognition (CVPR), 2017, IEEE, pp. 6584–6592..
[46] G. Yu, Y. Liu, D. Shen, Graph-guided joint prediction of class label and clinical scores for the alzheimer's disease, Brain Struct. Funct. 221 (2016) 3787–3801.
[47] J. Yu, M. Tan, H. Zhang, D. Tao, Y. Rui, Hierarchical deep click feature prediction for fine-grained image recognition, IEEE Trans. Pattern Anal. Mach. (2019), Intelligence.
[48] J. Yu, J. Yao, J. Zhang, Z. Yu, D. Tao, Sprnet: single-pixel reconstruction for one-stage instance segmentation, IEEE TCYB (2020).
[49] W. Zhang, S. Su, B. Wang, L. Sun, Local k-nns pattern in omni-direction graph convolution neural network for 3d point clouds, Neurocomputing (IF 4.438) 413 (2020) 487–498.
[50] J. Zhong, D.Y.L. Phua, A. Qiu, Quantitative evaluation of lddmm, freesurfer, and caret for cortical surface mapping, NeuroImage 52 (2010) 131–141.
[51] X. Zhu, H.I. Suk, D. Shen, A novel matrix-similarity based loss function for joint regression and classification in ad diagnosis, Neuroimage 100 (2014) 91–105.

Chaoqiang Liu, postodoctoral fellow at the Department of Biomedical Engineering, National University of Singapore.

Hui Ji, Associate Professor at the Department of Mathematics, National University of Singapore.

Anqi Qiu, Associate Professor at the Department of Biomedical Engineering, National University of Singapore.