Neural Computing and Applications
https://doi.org/10.1007/s00521-021-06006-6

ORIGINAL ARTICLE

Revisiting convolutional neural network on graphs with polynomial approximations of Laplace–Beltrami spectral filtering

Shih-Gu Huang[1] · Moo K. Chung[2] · Anqi Qiu[3] · Alzheimer's Disease Neuroimaging Initiative

Received: 23 October 2020 / Accepted: 31 March 2021

Abstract
This paper revisits spectral graph convolutional neural networks (graph-CNNs) given in Defferrard (2016) and develops the Laplace–Beltrami CNN (LB-CNN) by replacing the graph Laplacian with the LB operator. We define spectral filters via the LB operator on a graph and explore the feasibility of Chebyshev, Laguerre, and Hermite polynomials to approximate LB-based spectral filters. We then update the LB operator for pooling in the LB-CNN. We employ the brain image data from Alzheimer's Disease Neuroimaging Initiative (ADNI) and Open Access Series of Imaging Studies (OASIS) to demonstrate the use of the proposed LB-CNN. Based on the cortical thickness of two datasets, we showed that the LB-CNN slightly improves classification accuracy compared to the spectral graph-CNN. The three polynomials had a similar computational cost and showed comparable classification accuracy in the LB-CNN or spectral graph-CNN. The LB-CNN trained via the ADNI dataset can achieve reasonable classification accuracy for the OASIS dataset. Our findings suggest that even though the shapes of the three polynomials are different, deep learning architecture allows us to learn spectral filters such that the classification performance is not dependent on the type of the polynomials or the operators (graph Laplacian and LB operator).

Keywords Graph convolutional neural network · Signals on surfaces · Chebyshev polynomial · Hermite polynomial · Laguerre polynomial · Laplace–Beltrami operator.

1 Introduction

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 a 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.

There are two types of approaches for defining convolution on a graph: one through the spatial domain and the other through the spectral domain [8, 57]. Existing spatial approaches, such as diffusion-convolutional neural networks (DCNNs) [2], PATCHY-SAN [18, 44], gated graph sequential neural networks [37], DeepWalk [46], message-passing neural network (MPNN) [21], develop convolution in different ways to process the vertices on a graph whose neighborhood has different sizes and connections. An alternative approach is to take into account of the geometry of a graph and to map individual patches of a graph to a representation that is more amenable to classical convolution, including 2D polar coordinate representation [41], local windowed spectral representation [5], anisotropic variants of heat kernel diffusion filters [6, 7], Gaussian mixture model kernels [43].

✉ Anqi Qiu
bieqa@nus.edu.sg

[1] Department of Biomedical Engineering, National University of Singapore, Singapore 117583, Singapore

[2] Department of Biostatistics and Medical Informatics, University of Wisconsin, Madison, WI 53706, USA

[3] Department of Biomedical Engineering, The N.1 Institute for Health and Institute of Data Science, National University of Singapore, Singapore 117583, Singapore

Published online: 18 September 2021

On the other hand, several graph-CNN methods called "spectral graph-CNN" defines convolution in the spectral domain[9, 14, 23, 32, 35, 51, 56]. The advantage of spectral graph-CNN methods lies in the analytic formulation of the convolution operation. Based on the spectral graph theory, Bruna et al. [9] proposed convolution on graph-structured data in the spectral domain via the graph Fourier transform. However, the eigendecomposition of the graph Laplacian for building the graph Fourier transform is computationally intensive when a graph is large. Moreover, spectral filters in [9] are non-localized in the spatial domain. Defferrard et al. [14] addressed these problems by proposing Chebyshev polynomials to parametrize spectral filters such that the resulting convolution is approximated by the polynomials of the graph Laplacian. Kipf and Welling [32] adopted the first-order polynomial filter and stacked more spectral convolutional layers to replace higher-order polynomial expansions. In [14, 51], it is shown that the k-order Chebyshev polynomial approximation of graph Laplacian filters performs the k-ring filtering operation.

In this study, we revisited the spectral graph-CNN based on the graph Laplacian [14, 51] and developed the Laplace–Beltrami CNN (LB-CNN) , where spectral filters are designed via the Laplace–Beltrami (LB) operator on a graph. We studied the LB operator because it can characterize the underlying geometry of graphs. This may be particularly important for studying human organs since the geometry of human organs reflects their intrinsic and complex anatomy, as well as physiological functions. For instance, the cerebral cortex is composed of ridges (gyri) and valleys (sulci). Due to the way gyri and sulci are curved, the cortex is thicker in gyri but thinner in sulci. Hence, it is preferred to represent brain images in a way that the underlying geometrical information is encoded. One can express the cerebral cortex as a surface embedded in the 3D Euclidean space. Existing literature has demonstrated that such representation incorporates useful geometry information of the brain into machine learning for disease diagnosis [1, 19, 49, 55]. When spectral filters are designed based on the LB operator, we expect that the convolution with these filters incorporates the geometry of the underlying graph. Hence, we call these filters as LB spectral filters.

Next, we investigated whether the proposed LB-CNN is superior to the graph-CNN [14, 51] because the LB operator incorporates the intrinsic geometry of a graph but not the graph Laplacian [47]. We further explored the feasibility of polynomials to approximate LB spectral filters in the LB-CNN as used in the graph-CNN [14, 51]. Beyond Chebyshev polynomials used in the graph-CNN [14, 51], Laguerre and Hermite polynomials were explored in this study since these polynomials have potentials to approximate the heat kernel convolution on a graph as shown in [26, 52].

In this paper, we first reviewed the relevant work of the graph-CNN. In the method section, we introduced the design of LB spectral filters, their parameterization using Chebyshev, Laguerre, Hermite polynomials, and spatial localization. We then discussed the architecture of the LB-CNN while introducing rectified linear unit (ReLU), graph coarsening and pooling, and an update of the LB-operator. In the result section, we first illustrated the spatial localization of the LB spectral filters. Finally, we employed the brain image data from the Alzheimer's Disease Neuroimaging Initiative (ADNI) and Open Access Series of Imaging Studies (OASIS)-3 cohort, and demonstrated the use of the proposed LB-CNN and its classification accuracy and robustness. We compared the computational time and classification performance of the LB-CNN with the spectral graph-CNN [14, 51] when Chebyshev, Laguerre, and Hermite polynomials were used. We employed the trained LB-CNN from the ADNI dataset to the OASIS dataset to examine the robustness of the LB-CNN.

Therefore, the contributions of this study are but not limited to

- providing the approximation of LB spectral filters using Chebyshev, Laguerre, Hermite polynomials and their implementation in the LB-CNN;
- updating the LB operator for pooling in the LB-CNN;
- incorporating larget datasets for experiments;
- demonstrating the feasibility of using the LB operator and different polynomials for graph-CNNs.

2 Related work

The most relevant work to this study is the spectral graph-CNN [9, 14, 23, 32, 35, 51, 56]. Similar to classical CNN, it comprises of three components, convolution on a graph, rectified linear unit (ReLU), and pooling. We now review the convolution operation in the spectral graph-CNN.

Denote a graph as $G = \{V, E\}$ defined by vertex set V and edge set E with the weight of edge connecting vertex i and j being w_{ij}. Then, the graph Laplacian associated with G is defined as

$$\Delta = D - W$$

where $W = (W_{ij})$ is the weighted adjacency matrix and $D = (D_{ij})$ is the diagonal degree matrix with $D_{ii} = \sum_j W_{ij}$.

Assume that δ has the eigendecomposition $\Delta = U \Lambda U^{\mathrm{T}}$, where $\Lambda = (\lambda_j)$ is a diagonal matrix consisting of the eigenvalues and $U = [\psi_0, \psi_1, \ldots]$ is the Fourier basis consisting of the eigenvectors [14]. Then, the convolution

of a graph signal f and a spectral filter g with spectrum $g(\lambda)$ on the graph can be written as

$$h = g(\Delta)f = Ug(\Lambda)U^{\mathrm{T}}f.$$

This formulation inspires the spectral convolutional layer in the spectral graph-CNN introduced by Bruna et al. [9]. However, the computation of the eigendecomposition of Δ is costly. The forward and inverse graph Fourier transforms (U and U^{T}) in each spectral filter are a lack of fast computation and cause a computational bottleneck in the graph-CNN, especially when graphs are large scale. Moreover, the spectral filters, where $g(\lambda)$ are designed as linear combinations of cubic B-spline basis, may not spatially localized [15, 23].

3 Methods

In this section, we will introduce the LB-CNN and its three major components, including convolution, rectified linear unit (ReLU), and pooling. We will first describe LB spectral filters in the convolutional layer. In particular, we will introduce the polynomial approximation of the LB spectral filters to overcome challenges on (1) computational time; (2) spatial localization. We then define a pooling operation via coarsening a graph and updating the LB operator.

3.1 Laplace–Beltrami spectral filters

3.1.1 Polynomial approximation of LB spectral filters

Consider the Laplace–Beltrami (LB) operator Δ on surface $\mathscr{M}$. Let ψ_j be the jth eigenfunction of the LB-operator with eigenvalue λ_j

$$\Delta\psi_j = \lambda_j\psi_j \ , \tag{1}$$

where $0 = \lambda_0 \leq \lambda_1 \leq \lambda_2 \leq \cdots$. A signal $f(x)$ on the surface $\mathscr{M}$ can be represented as a linear combination of the LB eigenfunctions

$$f(x) = \sum_{j=0}^{\infty} c_j\psi_j(x) \ , \tag{2}$$

where c_j is the jth coefficient associated with the eigenfunction $\psi_j(x)$.

We now consider an LB spectral filter g on $\mathscr{M}$ with spectrum $g(\lambda)$ as:

$$g(x,y) = \sum_{j=0}^{\infty} g(\lambda_j)\psi_j(x)\psi_j(y). \tag{3}$$

Based on Eq. (2), the convolution of a signal f with the filter g can be written as:

$$h(x) = g * f(x) = \sum_{j=0}^{\infty} g(\lambda_j)c_j\psi_j(x). \tag{4}$$

As suggested in [12, 14, 22, 31, 52, 53], the filter spectrum $g(\lambda)$ in Eq. (4) can be approximated as the expansion of Chebyshev polynomials, T_k, $k = 0, 1, 2, \ldots, K-1$, such that

$$g(\lambda) = \sum_{k=0}^{K-1} \theta_k T_k(\lambda) \ . \tag{5}$$

θ_k is the kth expansion coefficient associated with the kth Chebyshev polynomial. T_k is the Chebyshev polynomial of the form $T_k(\lambda) = \cos(k\cos^{-1}\lambda)$. The left panel on Fig. 1 shows the shape of the kth Chebyshev polynomial up to order 6. We can rewrite the convolution in Eq. (4) as

$$h(x) = g * f(x) = \sum_{k=0}^{K-1} \theta_k T_k(\Delta)f(x). \tag{6}$$

Likewise, $g(\lambda)$ in Eq. (4) can also be approximated using other polynomials, such as Laguerre or Hermite polynomials [45]. T_k in Eq. (6) can be replaced by Laguerre, L_k, or Hermite, H_k, polynomials, where

$$\begin{aligned} L_k(\lambda) &= \sum_{l=0}^{k} \binom{k}{l} \frac{(-\lambda)^l}{l!}, \\ H_k(\lambda) &= k! \sum_{l=0}^{\lfloor k/2 \rfloor} \frac{(-1)^l (2\lambda)^{k-2l}}{l!(k-2l)!} \ , \end{aligned} \tag{7}$$

In this paper, we adopt the following normalized definition of Hermite polynomials:

$$\bar{H}_k(\lambda) = \frac{1}{\sqrt{2^k k!}} H_k(\lambda) \tag{8}$$

where the inner product of $\bar{H}_k$ with itself is independent of k. The last two panels of Fig. 1 show the shapes of Laguerre and Hermite polynomials up to order 6, respectively.

3.1.2 Numerical implementation of LB spectral filters via polynomial approximations

We now discretize the surface $\mathscr{M}$ as a triangulated mesh, $\mathscr{G} = \{V, E\}$, with a set of triangles and vertices v_i. For the implementation of the LB spectral filters, we adopt the discretization scheme of the LB operator in [52]. The ijth element of the LB-operator on $\mathscr{G}$ can be computed as:

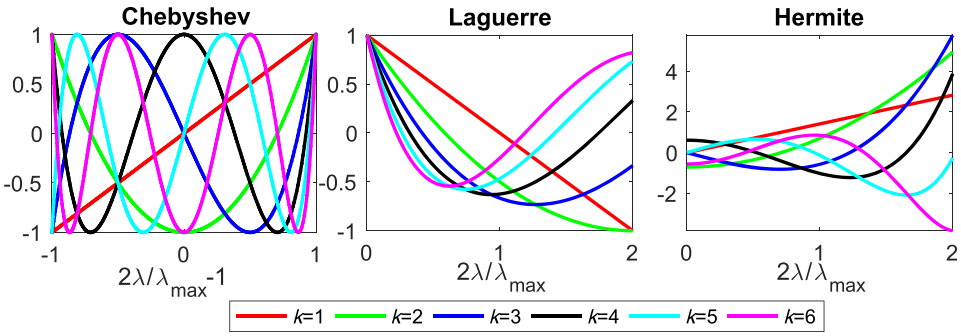

Fig. 1 Chebyshev, Laguerre and Hermite polynomials of order from 1 to 6. The eigenvalues were scaled and shifted to $[-1, 1]$ for Chebyshev polynomials and to [0, 2] for Laguerre and Hermite polynomials, where $\lambda_{\max}$ is the maximum eigenvalue of the LB-operator

$$\Delta_{ij} = C_{ij}/A_i, \tag{9}$$

where A_i is estimated by the Voronoi area of nonobtuse triangles [42] and the Heron's area of obtuse triangles containing v_i [42, 52]. The off-diagonal entries are defined as $C_{ij} = -(\cot\theta_{ij} + \cot\phi_{ij})/2$ if v_i and v_j form an edge, otherwise $C_{ij} = 0$. The diagonal entries C_{ii} are computed as $C_{ii} = -\sum_j C_{ij}$. Other cotan discretizations of the LB operator are discussed in [10, 11, 48]. When the number of vertices on $\mathscr{M}$ is large, the computation of the LB eigenfunctions can be costly [25].

For the sake of simplicity, we denote the kth order polynomial as P_k, where P_k can represent Chebyshev, Laguerre, or Hermite polynomial. We take the advantage of the recurrence relation of these polynomials (Table 1) and compute LB spectral filters recursively as follows.

1. compute Δ based on Eq. (9) for the triangulated mesh $\mathscr{G}$;
2. compute the maximum eigenvalue $\lambda_{\max}$ of Δ. For the standardization across surface meshes, we normalize Δ as $\widetilde{\Delta} = \frac{2\Delta}{\lambda_{\max}} - I$ such that the eigenvalues are mapped from $[0, \lambda_{\max}]$ to $[-1, 1]$ for Chebyshev polynomials [14, 26]. I is an identity matrix. For Laguerre and Hermite polynomials, we normalize Δ as $\widetilde{\Delta} = \frac{2\Delta}{\lambda_{\max}}$, which maps the eigenvalues from $[0, \lambda_{\max}]$ to [0, 2];
3. for a signal $f(x)$, compute $P_k(\widetilde{\Delta})f(x)$ recursively by

$$P_{k+1}(\widetilde{\Delta})f = A_k\widetilde{\Delta}\ P_k(\widetilde{\Delta})f + B_kP_k(\widetilde{\Delta})f + C_kP_{k-1}(\widetilde{\Delta})f, \tag{10}$$

with the initial conditions $P_{-1}(\widetilde{\Delta})f(x) = 0$ and $P_0(\widetilde{\Delta})f(x) = f(x)$. The recurrence relations of different polynomials are given in Table 1.

Step 3 is repeated from $k = 0$ till $k = K - 2$.

3.1.3 Localization of spectral filters based on polynomial approximations

Analogue to the spatial localization property of Chebyshev polynomial approximation of graph Laplacian spectral filters [14], we can show that Chebyshev, Laguerre, or Hermite polynomial approximation of LB spectral filters also have this localization property. We consider the discretization of Δ given in Eq. (9). Consider two vertices v_i and v_j on $\mathscr{G}$. We can define the shortest distance between v_i and v_j, denoted by $d_{\mathscr{G}}(i,j)$, as the minimum number of edges on the path connecting v_i and v_j. Hence,

$$(\Delta^K)_{i,j} = 0 \quad if \quad d_{\mathscr{G}}(i,j) > K, \tag{11}$$

where Δ^K denotes the K-th power of the LB operator Δ [52]. In other words, the coverage of $(\Delta^K)_{i,j}$ is localized in the ball with radius k from the central vertex.

$P_k(\Delta)$ can be represented in terms of $\Delta^0, \Delta, \ldots, \Delta^k$ and is k-localized $(P_k(\Delta))_{i,j} = 0$ if $d_{\mathscr{G}}(i,j) > k$ according to Eq. (11). The spectral filter g composed of $P_0(\Delta)$, $P_1(\Delta)$, ..., $P_{K-1}(\Delta)$ is a spatially localized filter with localization property given by

Table 1 The recurrence relation of Chebyshev, Laguerre and Hermite polynomials in spectral filtering

Method	Recurrence relations
Chebyshev[a]	$T_{k+1}(\widetilde{\Delta})f = (2-\delta_{k0})\widetilde{\Delta}\ T_k(\widetilde{\Delta})f - T_{k-1}(\widetilde{\Delta})f$
Laguerre	$L_{k+1}(\widetilde{\Delta})f = \frac{-\widetilde{\Delta}\ L_k(\widetilde{\Delta})f + (2k+1)L_k(\widetilde{\Delta})f - kL_{k-1}(\widetilde{\Delta})f}{k+1}$
Hermite	$\bar{H}_{k+1}(\widetilde{\Delta})f = \sqrt{\frac{2}{k+1}}\widetilde{\Delta}\ \bar{H}_k(\widetilde{\Delta})f - \sqrt{\frac{k}{k+1}}\bar{H}_{k-1}(\widetilde{\Delta})f$

[a] δ_{k0} is Kronecker delta

$$(g(\Delta))_{i,j} = 0 \quad if \quad d_{\mathscr{G}}(i,j) > K-1. \tag{12}$$

In practice, we can also show the spatial localization of filter g composed of $P_0(\Delta), P_1(\Delta), ..., P_{K-1}(\Delta)$ by applying g to an impulse signal f_j with 1 at vertex v_j and 0 at the others. Then, the filter output is given by $g(x, v_j) = \sum_{k=0}^{K-1} \theta_k P_k(\Delta) f_j(x)$. When $x = v_i$ satisfying $d_{\mathscr{G}}(i,j) > K-1$, since $(P_k(\Delta))_{i,j} = 0$, we have

$$g(v_i, v_j) = \sum_{k=0}^{K-1} \theta_k \big(P_k(\Delta) f_j(x)\big)_i = \sum_{k=0}^{K-1} \theta_k (P_k(\Delta))_{ij} = 0. \tag{13}$$

3.2 Rectified linear unit

Similar to classical CNN, a *rectified linear unit* (ReLU) in the LB-CNN can be represented by many nonlinear 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 LB-CNN on a mesh, we adopt the well-known ReLU:

$$\sigma(z) = \max\{0, z\}, \quad z \in \mathbb{R}.$$

3.3 Mesh coarsening and pooling

For the LB-CNN, the pooling layer involves mesh coarsening, pooling of signals, and an update of the LB operator. First, we adopt the Graclus multilevel clustering algorithm [16] to coarsen a graph based on the graph Laplacian. This algorithm is built on the METIS [30] to cluster similar vertices together from a given graph by a greedy algorithm. At each coarsening level, two neighboring vertices with maximum local normalized cut are matched until all vertices are explored [50].

In our case, the discrete LB-operator Δ in Eq. (9) is used. The local normalized cut on a mesh is computed by $-\Delta_{ij}(1/\Delta_{ii} + 1/\Delta_{jj})$. The coarsening process is repeated until the coarsest level is achieved. After coarsening, a balanced binary tree is generated where each node has either one (i.e., singleton) or two child nodes. Fake nodes are added to pair with those singleton. The weights of the edges involving fake nodes are set as 0. Then, the pooling on this binary tree can be efficiently implemented as a simple 1-dimensional pooling of size 2. For the update of the LB operator for a coarsen mesh, when two matched vertices are merged as a new vertex together at a coarser level, the weight of the new vertex is defined as the sum of the weights of the edges involving the two vertices. By doing so, each coarsened mesh has its updated Δ.

3.4 LB-CNN architecture

We are now well equipped with all the components for a LB-CNN network. The LB-CNN network is composed of total $L+1$ connected stages. The first L stages are the stages for feature extraction. Each stage contains three sequentially concatenated layers: (1) a convolutional layer with multiple LB spectral filters; (2) a ReLU layer; (3) a pooling layer with stride 2 or higher that uses average pooling. In the last stage, a fully connected layer followed by a softmax function is employed to make a decision, and the output layer contains classification labels.

Figure 2 illustrates one of LB-CNN architectures that are analogous to classical CNN for image data defined on equi-spaced grids. In this example, the i-th convolution layer is composed of 2^{i+2} LB spectral filters that can be approximated using Chebyshev, Laguerre, and Hermite polynomials, an ReLU, and an average pooling with pooling size $2^{\max\{5-i,1\}}$ and stride being the same as the pooling size. In the fully connected layer, there are 128 hidden nodes, and an l_2-norm regularization with weight of 5×10^{-4} is applied to prevent overfitting.

All the networks can be trained by the back propagation algorithm with 30 epochs, mini-batch size of 32, initial learning rate of 10^{-3}, learning rate decay of 0.05 for every 20 epochs, momentum of 0.9 and no dropout.

All the network models were implemented and trained using Python 3.7 (www.python.org) and TensorFlow 1.13.1 (www.tensorflow.org) library on NIVIDIA Tesla V100-SXM2 GPU with 32GB RAM and Intel Xeon Gold 5118 CPU with 2.30GHz.

3.5 Evaluation metrics

In this study, we quatify the classification performance using four metrics: accuracy (ACC), sensitivity (SEN) specificity (SPE), and geometric mean (GMean). These metrics are defined as:

$$\begin{aligned}
\mathrm{ACC} &= \frac{\mathrm{TP}+\mathrm{TN}}{\mathrm{TP}+\mathrm{TN}+\mathrm{FN}+\mathrm{FP}},\\
\mathrm{SEN} &= \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}},\\
\mathrm{SPE} &= \frac{\mathrm{TN}}{\mathrm{TN}+\mathrm{FP}},\\
\mathrm{GMean} &= \sqrt{\mathrm{SEN}\times\mathrm{SPE}},
\end{aligned}$$

where TP, TN, FN, and FP are the true positive, true negative, false negative and false positive, respectively.

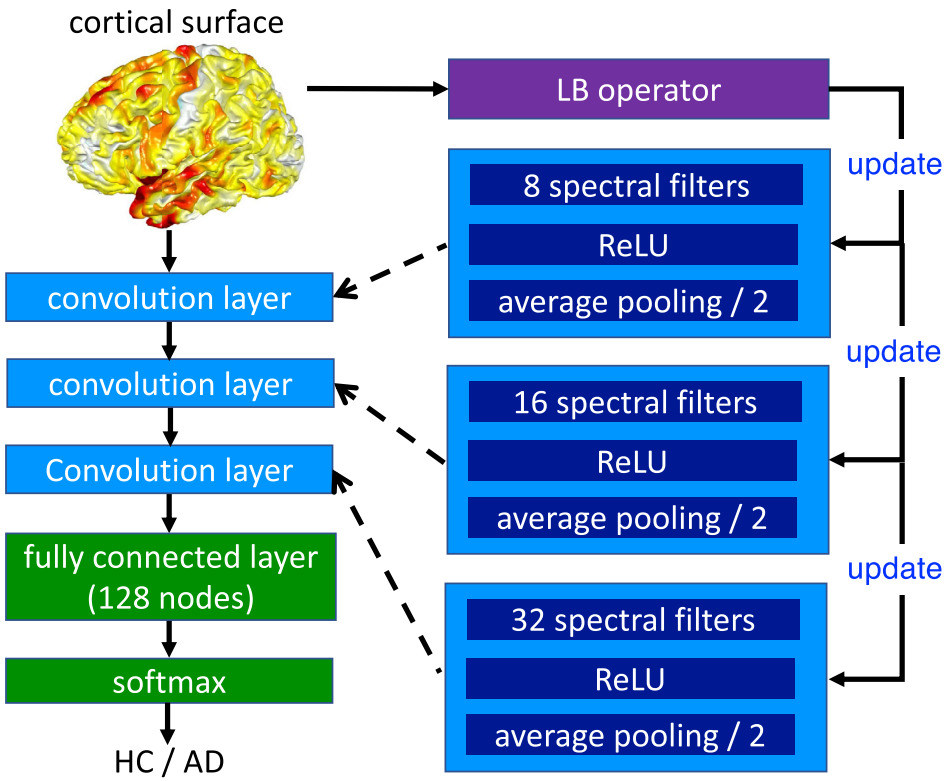

Fig. 2 An example of LB-CNN architectures for the classification of health controls (HC) and Alzheimer's Disease (AD) patients. The cortical surface is represented as a triangulated mesh with 655,360 triangles and 327,684 vertices

3.6 Datasets and MRI processing

We utilized the structural T1-weighted MRI from the Alzheimer's Disease Neuroimaging Initiative (ADNI)-2 cohort (http://adni.loni.ucla.edu) and the Open Access Series of Imaging Studies (OASIS)-3 cohort (www.oasis-brains.org). This study aimed to illustrate the use of the LB-CNN and spectral graph-CNN via the HC/AD classification since it has been well studied using T1-weighted image data (e.g., [3, 13, 24, 27, 34, 38, 39, 53]). Hence, this study only involved subjects with HC or AD scans. Each subject may have multiple MRI scans due to multiple visits. We included all T1-weighted images with good quality after processing but excluded scans without clinical label, age, or gender information.

ADNI-2 cohort We included 653 subjects aged from 55 to 95 years from the ADNI-2 cohort (400 HC subjects ; 261 AD subjects). There were 8 subjects who fell into both diagnostic groups due to the conversion from HC to AD. There were total 1122 scans for HC and 587 scans for AD.

OASIS-3 cohort The OASIS-3 dataset included 1014 subjects aged from 42 to 97 years (776 HC subjects; 267 AD subjects). There were 29 subjects who fell into both diagnostic groups due to the conversion from HC to AD. There was a total of 1925 scans (1603 for HC; 322 for AD).

Structural MRI Preprocessing The structural T1-weighted images from both cohorts were segmented using FreeSurfer (version 5.3.0) [20]. The white and pial cortical surfaces were generated at the boundary between white and gray matter and the boundary of gray matter and CSF, respectively. Cortical thickness was computed as the distance between the white and pial cortical surfaces. It represents the depth of the cortical ribbon. We represented cortical thickness on the mean surface, the average between the white and pial cortical surfaces. We employed large deformation diffeomorphic metric mapping (LDDMM) [17, 58] to align individual cortical surfaces to the atlas and transferred the cortical thickness of each subject to the atlas. The cortical atlas surface was represented as a triangulated mesh with 655,360 triangles and 327,684 vertices. At each surface vertex, a spline regression implemented by piecewise step functions [29] was performed to regress out the effects of age and gender. The residuals from the regression were used in the below spectral graph-CNN and LB-CNN.

4 Results

4.1 Spatial localization of the LB spectral filters via polynomial approximations

Figure 3 shows the localization property of spectral filters using the Chebyshev, Laguerre and Hermite polynomials.

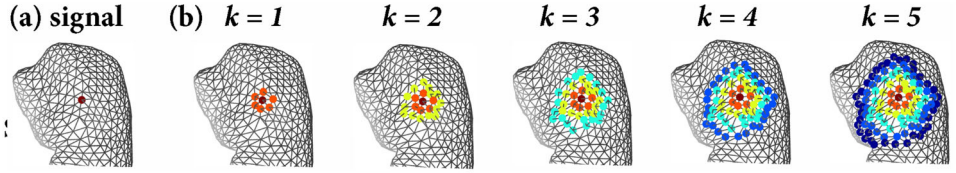

Fig. 3 The first panel shows the signal with 1 at one vertex and 0 at the other vertices of the hippocampus. The rest of panels show the spatial localization of spectral filters using Chebyshev T_k, Laguerre L_k and Hermite $\bar{H}_k$ polynomials for $k = 1, 2, \ldots, 5$

The input signal is 1 at only one vertex and 0 at all other vertices of the hippocampus. The $P_k(\Delta)$ is strictly localized in a ball of radius k, i.e., k rings from the central vertex. Figure 4

The first panel on Fig. 4 considers a signal having 1 on a small patch and 0 on the rest of a hippocampus surface mesh with 1184 vertices and 2364 triangles. Figure 4 shows the convolutions of this signal with spectral filters $g = T_k$, L_k or $\bar{H}_k$ for $k = 1, 4, 7, 10$. The spectral filters designed by different polynomials show different impacts on the signal in the spatial domain. These findings suggested that the spectral LB filters generated by different polynomials covers a larger spatial area when the order of the polynomials, k, increases.

4.2 Comparison of spectral graph-CNN and LB-CNN

We aimed to compare the computational cost and classification accuracy of the spectral graph-CNN [14, 53] and LB-CNN on the cortical thickness of the HC and the AD patients, while Chebyshev, Laguerre and Hermite polynomials were used to approximate spectral filters (Table 2).

In our experiments, the architecture of the spectral graph-CNN and LB-CNN was the same as shown in Fig. 2 except the number of layers. Ten-fold cross-validation was applied to the dataset (HC: $n = 1122$; AD: $n = 587$). One fold of real data was left out for testing. The remaining nine folds were further separated into training (75%) and validation (25%) sets randomly. To prevent potential data leakage in the ten-fold cross-validation, we constructed

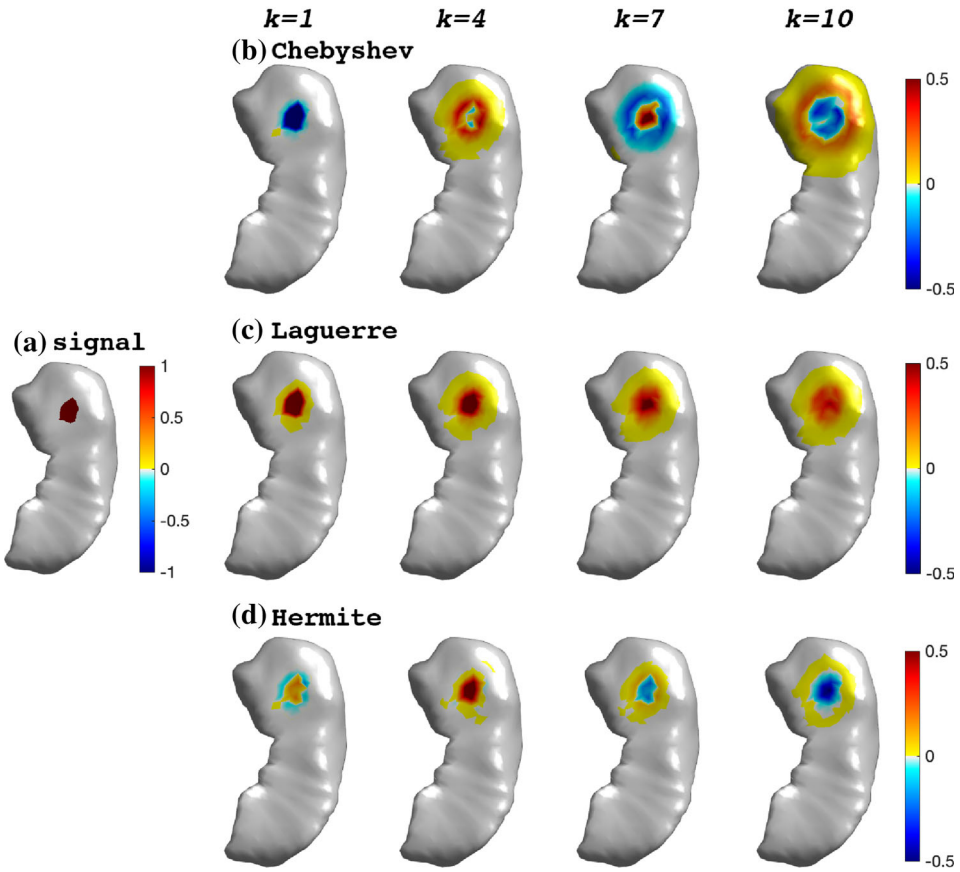

Fig. 4 The first panel shows the input signal, where 1 is on a red region and 0 on the rest of the hippocampus. The rest of panels show the signals after filtering via LB spectral filters, $g = T_k$, L_k and $\bar{H}_k$ for $k = 1, 4, 7, 10$, where T_k, L_k and $\bar{H}_k$ are the Chebyshev, Laguerre, and Hermite polynomials, respectively

Table 2 Classification performance of the spectral graph-CNN and LB-CNN with Chebyshev, Laguerre, and Hermite polynomial approximations

Spectral CNN	Polynomial	Layer	K	ACC (%)	SEN (%)	SPE (%)	GMean (%)
Graph	Chebyshev	4	6	89.8 ± 0.4	90.1 ± 0.9	89.6 ± 0.6	89.9 ± 0.4
	Laguerre	5	7	90.0 ± 0.5	91.7 ± 1.1	89.2 ± 0.6	90.4 ± 0.6
	Hermite	3	7	87.1 ± 0.5	86.6 ± 1.6	87.4 ± 0.9	87.0 ± 0.7
LB	Chebyshev	5	7	90.9 ± 0.6	91.3 ± 0.1	90.7 ± 0.5	91.0 ± 0.7
	Laguerre	5	7	91.0 ± 0.4	91.2 ± 0.9	90.9 ± 0.8	91.1 ± 0.4
	Hermite	4	7	88.2 ± 0.6	87.5 ± 0.6	88.4 ± 1.1	88.0 ± 0.4

ACC, accuracy; SEN, sensitivity; SPE, specificity; GMean, geometric mean

non-overlap training, validation, and testing sets with respect to subjects instead of MRI scans. This ensured that the scans from the same subjects were in the same set. The above data splitting was done for the HC and AD groups separately so that the ratio of the number of subjects in the two groups was similar in all sets.

4.2.1 Computational cost

The computation cost of the LB-CNN was similar to that of the spectral graph-CNN since they only differed in the edge weights between vertices. The computation of the LB-operator or the graph operator only took 1.2 seconds for the brain surface mesh with 655,360 triangles and 327,684 vertices. Table 3 shows the mean and standard deviation of the training time over the ten-fold cross-validation for each network with 3 convolutional layers and six-order polynomial approximation. Two-sample t-tests showed no significant differences in the computation cost between the LB-CNN and spectral graph-CNN ($p = 0.79$ for Chebyshev, $p = 0.46$ for Laguerre, and $p = 0.75$ for Hermite polynomials).

Next, we compared the computational cost of the LB-CNN with 3 convolutional layers using different polynomial approximations. Figure 5 shows the training time of the LB-CNNs using the Chebyshev, Laguerre and Hermite approximation of order $K = 2$, 4, 6 and 8. Regardless of which polynomial was used, the training time increased as K increased since more trainable parameters were needed to characterize the spectral filters. Given K, the three

Table 3 Computational cost of the LB-CNN and spectral graph-CNN with 3 convolutional layers and six-order polynomial approximation

CNN	Chebyshev	Laguerre	Hermite
Graph (s)	34406 ± 1941	34961 ± 1217	33625 ± 1310
LB (s)	34169 ± 2051	34305 ± 2494	33908 ± 2385

The average and standard deviation of the training time are listed over the tenfold cross-validation

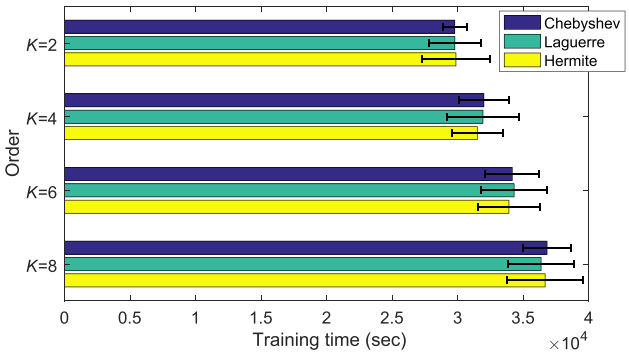

Fig. 5 Computational cost of the LB-CNN using Chebyshev (blue), Laguerre (green) and Hermite (yellow) approximations of different order K. The ten-fold cross-validation was repeated for 10 times. The average and standard deviation of the training time are shown (color figure online)

polynomial approximation methods had similar computation cost ($p > 0.56$).

4.2.2 Classification performance

To compare classification performance of the spectral graph-CNN and LB-CNN on HC and AD, a number of convolutional layers and polynomial approximation order were tuned for each CNN independently to achieve the best classification accuracy and geometric mean (GMean) on the validation set. The spectral graph-CNNs with Chebyshev, Laguerre and Hermite approximations, respectively, required 4 convolutional layers with polynomial order of $K = 6$, 5 layers with $K = 7$, and 3 layers with $K = 7$. The LB-CNNs with Chebyshev and Laguerre approximations needed 5 layers with $K = 7$, while the LB-CNN with Hermite approximation required 4 layers with $K = 7$. Table 2 lists the accuracy, sensitivity, specificity and Gmean of all these CNNs in classifying AD and HC.

Two-sample t-test found no significant difference in classification accuracy between the spectral graph-CNN and LB-CNN. For instance, when Chebyshev polynomials were used to approximate the spectral filters, the spectral graph-CNN classification accuracy was 89.9%, while the

LB-CNN classification accuracy was 90.9% ($p = 5.4 \times 10^{-5}$). Likewise, there were no group differences in classification accuracy between the spectral graph-CNN and LB-CNN when Laguerre and Hermite polynomial approximations were used (Laguerre: $p = 3.5 \times 10^{-4}$; Hermite:$p = 9.5 \times 10^{-4}$). Hence, the LB-CNN slightly improved the classification performance compared to the spectral graph-CNN.

As for the comparisons among the three different polynomials, the classification performance of the Laguerre approximation was comparable to the Chebyshev approximation (graph-CNN: $p = 0.27$; LB-CNN: $p = 0.81$). However, the classification performance of both Chebyshev and Laguerre polynomial approximations was greater than that of the Hermite polynomial approximation (all $p < 5.4 \times 10^{-10}$). In [26], Hermite polynomial approximation shows slower convergence to heat kernel, compared to Chebyshev and Laguerre polynomial approximations.

4.3 LB-CNN robustness

We employed transfer learning technique and applied the LB-CNN and graph-CNN models trained on the ADNI-2 dataset to the OASIS-3 dataset to demonstrate the robustness of the proposed models. In details, the network model setting and hyperparameters were exactly the same as used in the experiments of the ADNI-2 data classification as stated in Table 2. We applied the same data splitting strategy to the OASIS-3 dataset for ten-fold cross-validation in this experiement. The last layer of the network models that performed the best (the highest Gmean and accuracy) on the ADNI-2 validation dataset was fine-tuned on the training set of the OASIS-3 dataset. Then, the performance of the fine-tuned models was evaluated on the testing set of the OASIS-3 dataset. Table 4 lists the classification performance evaluated via ten-fold cross-validation. These results suggested the same conclusion, that is, the LB-CNN had the classification accuracy comparable with that of the spectral graph-CNN (Chebyshev: $p = 0.058$; Laguerre: $p = 0.073$; Hermite:$p = 0.058$). Moreover, the Chebyshev and Laguerre approximation showed comparable accuracy (graph-CNN: $p = 0.87$; LB-CNN: $p = 0.81$) and were slightly better than the Hermite approximation (all $p < 4.8 \times 10^{-4}$).

Compared to the ADNI-2 dataset, the OASIS-3 dataset showed a lower classification accuracy rate. This may partly because the AD patients in the OASIS-3 dataset had lower clinical dementia rating scale sum of boxes (CDR-SB) scores than those in the ADNI-2 dataset. Figure 6 showed the cumulative distribution fuction of the CDR-SB score of the AD patients in the ADNI-2 dataset (CDR-SB $= 5.7 \pm 2.8$) and OASIS-3 dataset (CDR-SB $= 4.6 \pm 3.0$). The Kolmogorov-Smirnov test showed a significant difference in the cumulative distribution functions (CDFs) between the two datasets ($p = 2.3 \times 10^{-11}$), suggesting that the AD patients in the ADNI-2 dataset may be more demented than those in the OASIS-3 dataset.

5 Conclusions

In this study, we revisited the spectral graph-CNN [14, 51] and developed the LB-CNN by replacing the graph Laplacian by the LB operator. We also employed Chebyshev, Laguerre, and Hermite polynomials to approximate the LB spectral filters in the LB-CNN and spectral graph-CNN. Based on cortical thickness of the ADNI and OASIS datasets, the classification accuracy of the LB-CNN and spectral graph-CNN [14, 51] was comparable. The three polynomials had the similar computational cost and showed comparable classification accuracy in the LB-CNN or spectral graph-CNN [14, 51]. Our findings suggest that even though the shapes of the three polynomials are different, deep learning architecture allows to learn spectral filters such that the classification performance is not dependent on the type of the polynomials or the operators (graph Laplacian and LB operator).

Table 4 Classification performance of the spectral graph-CNN and LB-CNN on the OASIS-3 dataset

Spectral CNN	Polynomial	Layer	K	ACC (%)	SEN (%)	SPE (%)	Gmean (%)
Graph	Chebyshev	4	6	80.2 ± 0.9	75.1 ± 2.5	81.3 ± 1.4	78.1 ± 1.0
	Laguerre	5	7	80.3 ± 0.7	75.3 ± 2.1	81.4 ± 0.9	78.3 ± 1.0
	Hermite	3	7	78.5 ± 0.9	70.9 ± 1.9	80.0 ± 1.1	75.3 ± 1.1
LB	Chebyshev	5	7	81.0 ± 0.6	76.9 ± 1.8	81.8 ± 0.8	79.3 ± 0.8
	Laguerre	5	7	80.9 ± 0.7	76.9 ± 1.3	81.7 ± 0.8	79.3 ± 0.7
	Hermite	4	7	79.3 ± 0.8	70.7 ± 2.2	81.0 ± 1.0	75.6 ± 1.1

Note that the spectral graph-CNN and LB-CNN were trained on the ADNI-2 dataset and fine-tuned based on the OASIS-3 dataset

ACC, accuracy; SEN, sensitivity; SPE, specificity; GMean, geometric mean

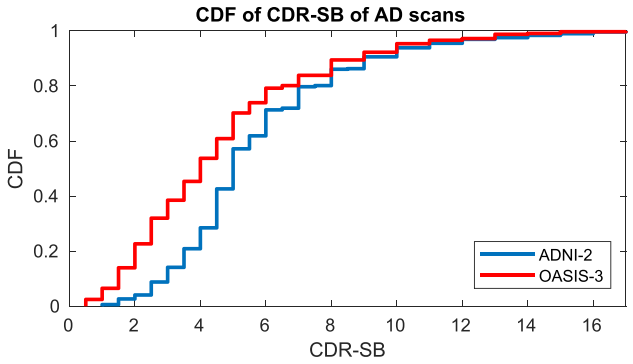

Fig. 6 The cumulative distribution functions (CDFs) of clinical dementia rating scale sum of boxes (CDR-SB) scores among the AD patients in the ADNI-2 (blue) and OASIS-3 (red) datasets (color figure online)

Our study showed the feasibility of employing the LB operator in the graph-CNN. Our findings did not strongly suggest that the LB operator might improve the classification accuracy of the graph-CNN compared to the graph Laplacian operator. The graph Laplacian is characterized by graph connectivity, while the LB operator is characterized not only by graph connectivity, but also by local angles and areas related to the geometry of a graph. Even though the LB operator may have an advantage in smoothing a signal on a graph [48], signals processed after many layers of spectral filters generated using the graph Laplacian and LB operators may not be different. This is partly because of many layers of linear and nonlinear operations. Hence, the performance of the LB-CNN and spectral graph-CNN is relatively comparable.

Our study suggested the comparable computational time and classification performance among Chebyshev, Laguerre, and Hermite polynomial approximations of the LB spectral filters in the graph-CNN. The spectral filters represented by these polynomials in this study are categorized as finite impulse response (FIR) graph filters [28]. Recently, infinite impulse response (IIR) graph filters [28] with rational frequency response have received much attention. Compared to a FIR filter, an IIR filter combines a FIR filter with feedback from previous filter outputs. Autoregressive moving average (ARMA) filters [4, 40], Cayley filter [36], personalized PageRank [33], and feedback-looped filters [54] have been used in graph-CNNs. Such kind of rational filters requires a matrix inversion to compute the denominator, which is computationally expensive for large graphs and inefficient for neural networks. Our research provides a possibility to parameterize IIR filters as shown in this paper and to improve computational cost.

Funding This research/project is supported by the National Science Foundation MDS-2010778, National Institute of Health R01 EB022856, EB02875, and 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. This research was also supported by the A*STAR Computational Resource Centre through the use of its high-performance computing facilities.

Availability of data and material Data used in preparation of this article were obtained from the Alzheimer's Disease Neuroimaging Initiative (ADNI) database (http://adni.loni.ucla.edu).

Code availability https://bieqa.github.io/deeplearning.html.

Declaration

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Apostolova LG, Dinov ID, Dutton RA, Hayashi KM, Toga AW, Cummings JL, Thompson PM (2006) 3D comparison of hippocampal atrophy in amnestic mild cognitive impairment and alzheimer's disease. Brain 129:2867–2873
2. Atwood J, Towsley D (2015) Diffusion-convolutional neural networks. arXiv preprint arXiv:151102136
3. Basaia S, Agosta F, Wagner L, Canu E, Magnani G, Santangelo R, Filippi M, Initiative ADN, et al. (2019) Automated classification of alzheimer's disease and mild cognitive impairment using a single MRI and deep neural networks. NeuroImage Clin 21:101645
4. Bianchi FM, Grattarola D, Alippi C, Livi L (2019) Graph neural networks with convolutional arma filters. arXiv preprint arXiv:190101343
5. Boscaini D, Masci J, Melzi S, Bronstein MM, Castellani U, Vandergheynst P (2015) Learning class-specific descriptors for deformable shapes using localized spectral convolutional networks. Comput Graph Forum 34(5):13–23
6. Boscaini D, Masci J, Rodoia E, Bronstein M (2016a) Learning shape correspondence with anisotropic convolutional neural networks. In: NIPS'16 Proceedings of the 30th international conference on neural information processing systems, ACM, pp 3197–3205
7. Boscaini D, Masci J, Rodola E, Bronstein MM, Cremers D (2016b) Anisotropic diffusion descriptors. Comput Graph Forum 35(2):431–441
8. Bronstein M, Bruna J, LeCun Y, Szlam A, Vandergheynst P (2017) Geometric deep learning: going beyond Euclidean data. IEEE Signal Process Mag 34(4):18–42
9. Bruna J, Zaremba W, Szlam A, LeCun Y (2013) Spectral networks and locally connected networks on graphs. arXiv preprint arXiv:13126203
10. Chung M, Taylor J (2004) Diffusion smoothing on brain surface via finite element method. Proc IEEE Int Symp Biomed Imaging (ISBI) 1:432–435
11. Chung M, Qiu A, Seo S, Vorperian H (2015) Unified heat kernel regression for diffusion, kernel smoothing and wavelets on manifolds and its application to mandible growth modeling in CT images. Med Image Anal 22:63–76

12. Coifman RR, Maggioni M (2006) Diffusion wavelets. Appl Comput Harmonic Anal 21(1):53–94
13. Cuingnet R, Gerardin E, Tessieras J, Auzias G, Lehéricy S, Habert MO, Chupin M, Benali H, Colliot O (2011) Automatic classification of patients with Alzheimer's disease from structural MRI: a comparison of ten methods using the ADNI database. Neuroimage 56(2):766–781
14. Defferrard M, Bresson X, Vandergheynst P (2016a) Convolutional neural networks on graphs with fast localized spectral filtering. In: Proceedings of the 30th international conference on neural information processing systems, NIPS, pp 3844–3852
15. Defferrard M, Bresson X, Vandergheynst P (2016b) Convolutional neural networks on graphs with fast localized spectral filtering. In: NIPS'16 Proceedings of the 30th international conference on neural information processing systems, ACM, pp 3844–3852
16. Dhillon IS, Guan Y, Kulis B (2007) Weighted graph cuts without eigenvectors a multilevel approach. IEEE Trans Pattern Anal Mach Intell 29(11):1944–1957
17. Du J, Younes L, Qiu A (2011) Whole brain diffeomorphic metric mapping via integration of sulcal and gyral curves, cortical surfaces, and images. NeuroImage 56(1):162–173
18. Duvenaud DK, Maclaurin D, Aguilera-Iparraguirre J, Gomez-Bombarelli R, Hirzel T, Aspuru-Guzik A, Adams RP (2015) Convolutional networks on graphs for learning molecular fingerprints. arXiv preprint arXiv:150909292
19. Fan Y, Gur R, Gur R, Wu X, Shen D, Calkins M, Davatzikos C (2008) Unaffected family members and schizophrenia patients share brain structure patterns: a high-dimensional pattern classification study. Biol Psychiatry 63(1):118–124
20. Fischl B, Salat DH, Busa E, Albert M, Dieterich M, Haselgrove C, Van Der Kouwe A, Killiany R, Kennedy D, Klaveness S et al (2002) Whole brain segmentation: automated labeling of neuroanatomical structures in the human brain. Neuron 33(3):341–355
21. Gilmer J, Schoenholz SS, Riley PF, Vinyals O, Dahl GE (2017) Neural message passing for quantum chemistry. arXiv preprint arXiv:170401212
22. Hammond DK, Vandergheynst P, Gribonval R (2011) Wavelets on graphs via spectral graph theory. Appl Comput Harmonic Anal 30(2):129–150
23. Henaff M, Bruna J, LeCun Y (2015) Deep convolutional networks on graph-structured data. arXiv preprint arXiv:150605163
24. Hosseini-Asl E, Keynton R, El-Baz A (2016) Alzheimer's disease diagnostics by adaptation of 3d convolutional network. In: 2016 IEEE international conference on image processing (ICIP). IEEE, pp 126–130
25. Huang SG, Lyu I, Qiu A, Chung M (2019) Fast polynomial approximation of heat diffusion on manifolds and its application to brain sulcal and gyral graph pattern analysis. IEEE Transactions on Medical Imaging, pp under 2nd review. arXiv:1911.02721
26. Huang SG, Lyu I, Qiu A, Chung MK (2020) Fast polynomial approximation of heat kernel convolution on manifolds and its application to brain sulcal and gyral graph pattern analysis. IEEE Trans Med Imaging 39(6):2201–2212
27. Islam J, Zhang Y (2018) Brain mri analysis for alzheimer's disease diagnosis using an ensemble system of deep convolutional neural networks. Brain Inform 5(2):2
28. Isufi E, Loukas A, Simonetto A, Leus G (2017) Filtering random graph processes over random time-varying graphs. IEEE Trans Signal Process 65(16):4406–4421
29. James G, Witten D, Hastie T, Tibshirani R (2013) An introduction to statistical learning, vol 112. Springer, Berlin
30. Karypis G, Kumar V (1998) A fast and high quality multilevel scheme for partitioning irregular graphs. SIAM J Sci Comput 20(1):359–392
31. Kim WH, Pachauri D, Hatt C, Chung MK, Johnson S, Singh V (2012) Wavelet based multi-scale shape features on arbitrary surfaces for cortical thickness discrimination. In: Advances in neural information processing systems, pp 1241–1249
32. Kipf TN, Welling M (2016) Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:160902907
33. Klicpera J, Bojchevski A, Günnemann S (2018) Predict then propagate: graph neural networks meet personalized pagerank. arXiv preprint arXiv:181005997
34. Korolev S, Safiullin A, Belyaev M, Dodonova Y (2017) Residual and plain convolutional neural networks for 3d brain MRI classification. In: 2017 IEEE 14th international symposium on biomedical imaging (ISBI 2017). IEEE, pp 835–838
35. Ktena SI, Parisot S, Ferrante E, Rajchl M, Lee M, Glocker B, Rueckert D (2017) Distance metric learning using graph convolutional networks: application to functional brain networks. arXiv preprint arXiv:170302161
36. Levie R, Monti F, Bresson X, Bronstein MM (2018) Cayleynets: graph convolutional neural networks with complex rational spectral filters. IEEE Trans Signal Process 67(1):97–109
37. Li Y, Tarlow D, Brockschmidt M, Zemel R (2015) Gated graph sequence neural networks. arXiv preprint arXiv:151105493
38. Liu M, Zhang J, Adeli E, Shen D (2018) Landmark-based deep multi-instance learning for brain disease diagnosis. Med Image Anal 43:157–168
39. Liu X, Tosun D, Weiner MW, Schuff N, Initiative ADN et al (2013) Locally linear embedding (lle) for MRI based alzheimer's disease classification. Neuroimage 83:148–157
40. Loukas A, Simonetto A, Leus G (2015) Distributed autoregressive moving average graph filters. IEEE Signal Process Lett 22(11):1931–1935
41. Masci J, Boscaini D, Bronstein MM, Vandergheynst P (2015) Geodesic convolutional neural networks on riemannian manifolds. In: 2015 IEEE international conference on computer vision (ICCV). IEEE, pp 832–840
42. Meyer M, Desbrun M, Schröder P, Barr AH (2003) Discrete differential-geometry operators for triangulated 2-manifolds. In: Visualization and mathematics III, Springer, pp 35–57
43. Monti F, Boscaini D, Masci J, Rodolà E, Svoboda J, Bronstein MM (2016) Geometric deep learning on graphs and manifolds using mixture model cnns. arXiv preprint arXiv:161108402
44. Niepert M, Ahmed M, Kutzkov K (2016) Learning convolutional neural networks for graphs. In: Proceeding of the 33rd international conference on machine learning. ACM, p 2014–2023
45. Olver FWJ, Lozier DW, Boisvert RF, Clark CW (2010) NIST handbook of mathematical functions. Cambridge University Press, Cambridge
46. Perozzi B, Al-Rfou R, Skiena S (2014) Deepwalk: online learning of social representations. In: Proceedings of the 20th ACM SIGKDD, ACM, pp 701–710
47. Perrault-Joncas DC, Meilă M, McQueen J (2017) Improved graph Laplacian via geometric consistency. In: Proceedings of the 31st international conference on neural information processing systems, pp 4460–4469
48. Qiu A, Bitouk D, Miller M (2006) Smooth functional and structural maps on the neocortex via orthonormal bases of the Laplace–Beltrami operator. IEEE Trans Med Imaging 25:1296–1396
49. Qiu A, Fennema-Notestine C, Dale A, Miller M, the Alzheimer's Disease Neuroimaging Initiative (2009) Regional shape abnormalities in mild cognitive impairment and Alzheimer's disease. Neuroimage 45:656–661

50. Shi J, Malik J (2000) Normalized cuts and image segmentation. IEEE Trans Pattern Anal Mach Intell 22(8):888–905
51. Shuman DI, Ricaud B, Vandergheynst P (2016) Vertex-frequency analysis on graphs. Appl Comput Harmonic Anal 40(2):260–291
52. Tan M, Qiu A (2015) Spectral Laplace–Beltrami wavelets with applications in medical images. IEEE Trans Med Imaging 34:1005–1017
53. Wee CY, Liu C, Lee A, Poh JS, Ji H, Qiu A, Initiative ADN (2019) Cortical graph neural network for AD and MCI diagnosis and transfer learning across populations. NeuroImage Clin 23:101929
54. Wijesinghe WAS, Wang Q (2019) Dfnets: Spectral cnns for graphs with feedback-looped filters. In: Advances in neural information processing systems, pp 6009–6020
55. Yang X, Goh A, Chen S, Qiu A (2013) Evolution of hippocampal shapes across the human lifespan. Hum Brain Mapp 34:3075–3085
56. Yi L, Su H, Guo X, Guibas L (2017) Syncspeccnn: Synchronized spectral cnn for 3d shape segmentation. In: Computer vision and pattern recognition (CVPR), conference on. IEEE, pp 6584–6592
57. Zhang Z, Cui P, Zhu W (2020) Deep learning on graphs: a survey. IEEE Trans Knowl Data Eng. Preprint. https://doi.org/10.1109/TKDE.2020.2981333
58. Zhong J, Phua DYL, Qiu A (2010) Quantitative evaluation of lddmm, freesurfer, and caret for cortical surface mapping. Neuroimage 52(1):131–141