SPECTRAL GRAPH TRANSFORMER NETWORKS FOR BRAIN SURFACE PARCELLATION

Ran He[†¶*] *Karthik Gopinath*[†*] *Christian Desrosiers*[†] *Herve Lombaert*[†]

[†] ETS Montréal, Canada
[¶] Beijing Institute of Technology, China

ABSTRACT

The analysis of the brain surface modeled as a graph mesh is a challenging task. Conventional deep learning approaches often rely on data lying in the Euclidean space. As an extension to irregular graphs, convolution operations are defined in the Fourier or spectral domain. This spectral domain is obtained by decomposing the graph Laplacian, which captures relevant shape information. However, the spectral decomposition across different brain graphs causes inconsistencies between the eigenvectors of individual spectral domains, causing the graph learning algorithm to fail. Current spectral graph convolution methods handle this variance by separately aligning the eigenvectors to a reference brain in a slow iterative step. This paper presents a novel approach for learning the transformation matrix required for aligning brain meshes using a direct data-driven approach. Our alignment and graph processing method provides a fast analysis of brain surfaces. The novel Spectral Graph Transformer (SGT) network proposed in this paper uses very few randomly sub-sampled nodes in the spectral domain to learn the alignment matrix for multiple brain surfaces. We validate the use of this SGT network along with a graph convolution network to perform cortical parcellation. Our method on 101 manually-labeled brain surfaces shows improved parcellation performance over a no-alignment strategy, gaining a significant speed (1400 fold) over traditional iterative alignment approaches.

Index Terms— Spectral transformer network, Cortical parcellation, Graph Convolution Network

1. INTRODUCTION

The surface of a human brain is a complex geometrical structure containing multiple convoluted folding patterns. Statistical analysis of the brain surface aids in understanding its anatomy, and machine learning methods are often sought for automating this analysis. Conventional machine learning frameworks exploit spatial information from the Euclidean domain such as image or volumetric coordinates [1, 2]. Similarly, state-of-the-art deep learning approaches [3, 4] operate on data lying in Euclidean spaces, offering a drastic speed advantage over traditional methods. However, the geometry of the brain is highly variable, hindering the direct use of these modern deep learning algorithms over multiple brain surfaces.

Recently, deep learning approaches on irregular graphs [5, 6, 7] have been proposed. These methods formulate a convolution theorem from Fourier space to spectral domains over graphs. One main limitation of these spectral approaches is their lack of expressing surface data in comparable spectral bases across different surface domains [8, 9, 10]. The Laplacian eigenbases are indeed incompatible across multiple geometries, challenging their direct use during training. As a solution, some recent work [11, 12] maps the local information onto geodesic patches and uses conventional template matching in spatial convolutions. For instance, [6] proposed local convolution operation as filtering over small neighborhoods in spatial domain. Their spatial representations of surface data remain, however, defined in a Euclidean space by using polar representations of pixels or mesh vertices.

In the literature, spectral graph matching has been used to transfer surface data across aligned spectral domains [14]. Such strategy [13, 15] enables the learning of spectral graph convolution networks across multiple surface data. These methods, however, involve an explicit computation of a transformation map for each brain towards one reference template. This process of aligning the eigenvectors of graph Laplacians is currently an important computational bottleneck. This expensive step is necessary in such approach to handle the differences across eigenvectors, including sign flips, ordering, and mixing of eigenvectors in higher frequencies. In this work, we propose a framework for learning this transformation function across multiple brain surfaces. In an alternative application for natural image classification, [16] proposes a transformer network for CNNs for learning a transformation matrix to spatially standardize the image data. Similarly, [17] also proposes a transformation network for learning over point clouds of geometric structures. These methods are, however, limited to pointwise information in a Euclidean space. This paper introduces a Spectral Graph Transformer Network (SGT) to learn the parameters for aligning multiple surfaces directly in the spectral domain. We illustrate the learning capabilities of this approach with an application to brain parcellation. We use the aligned coordinates from our SGT network along with a graph convolution network (GCN) for quantifying parcellation. The learnt alignment of 101 manually-labeled brain surfaces [18] reveals that our

[*] Equal contribution of R. He and K. Gopinath.

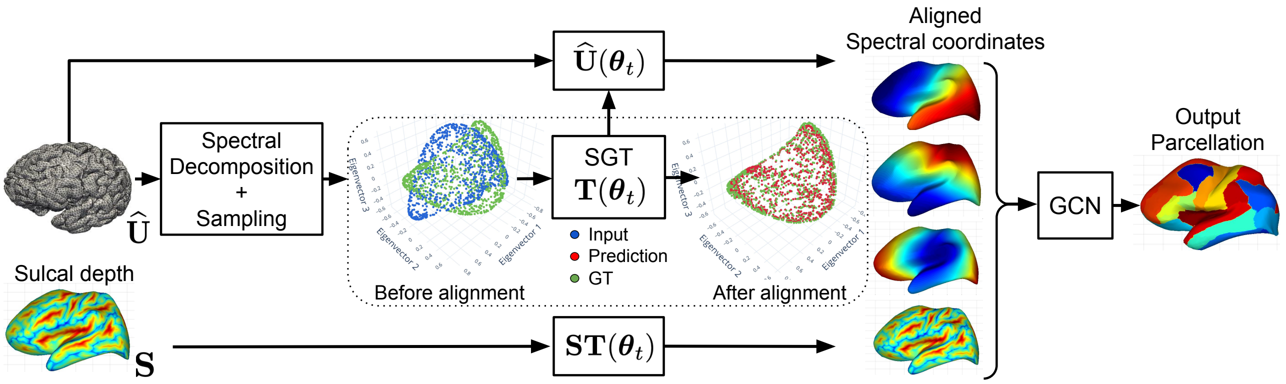

Fig. 1. **Overview of the our architecture:** The spectral decomposition of a brain graph is randomly sub-sampled as an input point cloud to a SGT network. The SGT learns the transformation parameters aligning the eigenvectors of multiple brains. The learnt transformation matrix is multiplied to the original spectral coordinates and fed to the GCN for parcellation. The point cloud is illustrated before and after alignment with our SGT. The GCN architecture follows recommendations from [13].

approach improves brain parcellation by 4.4%, from an average Dice overlap of 78.8% to 83.2%. The performance of our method is shown to be at par with traditional alignment strategies, performing at 84.4%, but gains a significant speed improvement. The learning of an end-to-end SGT and GCN model enables a direct, automatic learning of surface data across multiple brains. Our SGT part learns a transformation matrix that handles the eigenvector differences, while the GCN part focuses on the brain parcellation. The next section details the fundamentals of our SGT and GCN model, followed by an evaluation of our alignment strategy for graph convolutions.

2. METHOD

An overview of the method is shown in Fig. 1. Firstly, the cortical surfaces modeled as brain graphs are embedded in a spectral manifold using the graph Laplacian operator. Secondly, graph nodes are randomly sampled in the spectral embeddings and fed to the SGT network to align the brain embeddings. Finally, a GCN provides a labeled graph as output, taking spectral coordinates and cortical sulcal depth as input.

2.1. Spectral embedding of brain graphs

Let $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ be a brain graph defined with node set $\mathcal{V}$, such that $|\mathcal{V}| = N$, and edge set $\mathcal{E}$. Each node i has a feature vector $\mathbf{x}_i \in \mathbb{R}^3$ representing its 3D coordinates. We map $\mathcal{G}$ to a low-dimension manifold using the normalized graph Laplacian operator $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$, where $\mathbf{A}$ is the weighted adjacency matrix and $\mathbf{D}$ the diagonal degree matrix. In this work, we define the weight between two adjacent nodes as the inverse of their Euclidean distance. Let $\mathbf{L} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^\top$ be the eigendecomposition of $\mathbf{L}$, the normalized spectral coordinates of nodes are given by $\widehat{\mathbf{U}} = \boldsymbol{\Lambda}^{-\frac{1}{2}}\mathbf{U}$.

2.2. Spectral transformer network

The normalized spectral coordinates $\widehat{\mathbf{U}}$ from the spectral embedding of $\mathbf{L}$ is only defined up to an orthogonal transformation. We thus need to align the spectral representations of different brain graphs to a common representation. As a base reference, we align the normalized spectral embedding of all brain surfaces to a template $\widehat{\mathbf{U}}_{\text{ref}}$ in the dataset. This traditional alignment process involves computing an expensive optimal orthogonal transform based on iterative Procustes algorithm [14]. Let $\hat{\mathbf{u}}_i$, one row of $\widehat{\mathbf{U}}$, be the normalized spectral coordinates of node i of a brain model. The overall alignment to a reference model $\widehat{\mathbf{U}}^{(0)}$ can thus be formulated as:

$$\arg\min_{\pi, \mathbf{T}} \sum_{i=1}^{N} \left\| \hat{\mathbf{u}}_i \, \mathbf{T} \; - \; \hat{\mathbf{u}}^{(0)}_{\pi(i)} \right\|_2^2, \qquad (1)$$

This alignment step is computationally expensive, taking a few seconds to converge. It typically alternatives between finding a transformation $\mathbf{T}$, and a node matching $\pi(\cdot)$. Also, the alignment process is independent of the final target task. Our SGT learns instead the transformation matrix $\mathbf{T}$ for every brain graph in a data-driven manner. As input to the network, we provide $\widehat{\mathbf{U}}_{\text{sub}}$, a set of N randomly sub-sampled $\widehat{\mathbf{U}}$ coordinates, chosen similarly to [17], with enough samples to recover $\mathbf{T}$. Most information on graph connectivity is encoded in the first eigencomponents of $\mathbf{L}$, which enable savings in processing times. For instance, we could only keep the first 3 components for the learning step. In our experiments, $\widehat{\mathbf{U}}_{\text{sub}}$ is a matrix of size $N \times 3$. Fig. 1 describes the architecture of our spatial transformer network.

The model first applies a sequence of two point-wise linear transformation layers on $\widehat{\mathbf{U}}_{\text{sub}}$, each one followed by a non-linear rectifier (ReLU) function. Such layer takes a $N \times M_{l-1}$ matrix $\mathbf{X}$ as input and post-multiplies it by a $M_{l-1} \times M_l$ parameter matrix $\mathbf{W}_l$ to give an output matrix of size $N \times M_l$. This transformation, which is similar to 1×1 convolutions in CNNs, expresses each embedded node with respect to a shared set of M_l hyper-planes in the spectral space, and is used to capture the global shape of the embedding. In our model, we use $M_1 = 256$ for the first layer and $M_2 = 128$ for the second one (note that $M_0 = 3$). Next, the output of the second point-wise transformation layer is converted to a fixed-size representation of size 128×1 by applying average pooling. Last, to get the final spectral transformation matrix, we apply three Multilayer Perceptron (MLP) layers of size [128, 64, 9], also with ReLU activations, and reshape the output of the last layer into a 3×3 matrix. This transformation matrix is multiplied to the normalized spectral coordinates $\widehat{\mathbf{U}}$ to obtain the aligned spectral coordinates.

The parameters of the spectral transformer network are optimized by computing the mean square error between the predicted coordinates and spectral coordinates $\widetilde{\mathbf{U}}$ obtained with the iterative alignment method. To enforce regularization during training, and match the possible rotation and flip ambiguity of eigendecomposition, we also add a second loss term imposing the transformation matrix to be orthogonal. The final loss function is given by:

$$E_{\mathrm{spt}}(\boldsymbol{\theta}_t) \,=\, \|\widetilde{\mathbf{U}} - \widehat{\mathbf{U}}\, \mathbf{T}(\boldsymbol{\theta}_t)\|_F^2 + \|\mathbf{T}(\boldsymbol{\theta}_t)\mathbf{T}^\top(\boldsymbol{\theta}_t) - \mathbf{I}\|_F^2. \quad (2)$$

2.3. Graph convolution on surfaces

The second part of our end-to-end model is based on a geometric convolutional neural network that maps the now-aligned spectral coordinates to a common comparable graph embedding. A generalized convolution operation on a graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, with $\mathcal{N}_i = \{j \,|\, (i,j) \in \mathcal{E}\}$, as the neighbors of node $i \in \mathcal{V}$, is defined as

$$z_{ip}^{(l)} \;=\; \sum_{j \in \mathcal{N}_i} \sum_{q=1}^{M_l} \sum_{k=1}^{K_l} w_{pqk}^{(l)} \cdot y_{jq}^{(l)} \cdot \varphi(\widehat{\mathbf{u}}_i, \widehat{\mathbf{u}}_j;\, \boldsymbol{\theta}_k^{(l)}) \,+\, b_p^{(l)}, \quad (3)$$

where $\varphi(\widehat{\mathbf{u}}_i, \widehat{\mathbf{u}}_j; \boldsymbol{\theta}_k)$ is a symmetric kernel in the embedding space with parameter $\boldsymbol{\theta}_k$. In this work, we follow [13] and use a Gaussian kernel: $\varphi(\widehat{\mathbf{u}}_i, \widehat{\mathbf{u}}_j; \boldsymbol{\mu}_k, \sigma_k) \;=\; \exp\big(-\sigma_k \,\|(\widehat{\mathbf{u}}_j - \widehat{\mathbf{u}}_i) - \boldsymbol{\mu}_k\|^2\big)$.

We define a fully-convolutional network comprising of 4 graph convolution layers with sizes 256, 128, 64, and 32. Each layer has $K_l = 6$ Gaussian kernels similar to [13, 19]. The total target parcels are 32, hence, our last layer is of size 32. Leaky ReLU is applied after each layer to obtain our filter responses. A softmax operation is used after the last graph convolution layer in order to obtain the probabilities of the mutually-exclusive parcels at each node. Our output loss function employs a cross-entropy with Dice loss for all parcellations. Our final end-to-end model comprising of a spectral transformer and a graph convolution network for brain parcellation is trained using the loss function given by

$$E_{\mathrm{final}}(\boldsymbol{\theta}_t, \boldsymbol{\theta}_g) \;=\; \lambda\, E_{\mathrm{spt}}(\boldsymbol{\theta}_t) \;+\; E_{\mathrm{gcn}}(\boldsymbol{\theta}_g). \quad (4)$$

This final loss E_{final} is minimized by back-propagating the error using standard gradient descent optimization.

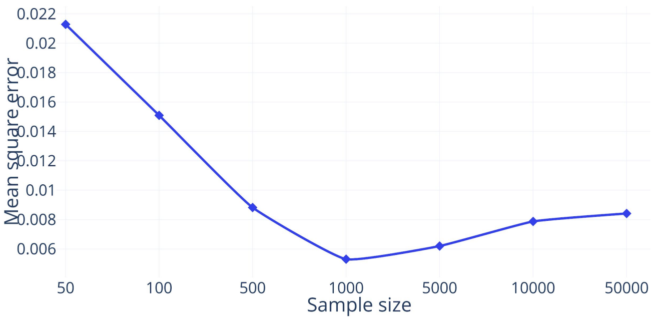

Fig. 2. **SGT network data sampling:** Each point indicates the performance of our SGT model in terms of mean square error. The best mean square error is achieved for models with 1000 nodes as input to our SGT.

3. EXPERIMENTS AND RESULTS

In this section, we evaluate how inputs affect our SGT network. The optimal SGT parameters are thereafter used to train our end-to-end model for brain parcellation. We validate our approach on the Mindboggle [18] dataset containing manually-labeled brain surfaces. The dataset contains 101 cortical meshes, each with 102K to 185K vertices and 32 manually-labeled parcels. We randomly split the dataset into training, validation and testing in a 70-10-20$\%$ ratio for our experiments. Here, we induce random sign flips on the eigenvectors of the training dataset to balance flipping and rotation variance. The performance of the methods are measured in terms of average Dice overlap and Hausdorff distances. The experiments are carried out on an i7 desktop computer with 16GB of RAM and a Nvidia Titan Xp GPU.

3.1. Spectral transform data sampling

Our spectral transformer network takes as input a set of points in the spectral domain. The number of eigenvectors is fixed to three, as suggested in [13]. To evaluate the effect of input size N, we sample spectral points randomly from 50 to $50,000$. We study the performance of spectral alignment using our SGT model in terms of mean square error.

The results shown in Fig. 2 illustrate that the best alignment performance is achieved with a sub-sampling size of $N = 1000$. The input data with $N = 50, 100, 500$ is inadequate to capture the complete geometric information of the brain, as seen in Fig. 2. In addition to lower performance, a higher number of nodes also increases memory consumption and computation time. The gain in mean square error for input size over $N = 1000$ can also be seen in Fig. 2.

3.2. Brain surface parcellation

We now evaluate the performance of our end-to-end SGT and GCN model on brain surface parcellation. The predicted transformation matrix from SGT aligns all brain surfaces. The number of embedded node coordinates used during

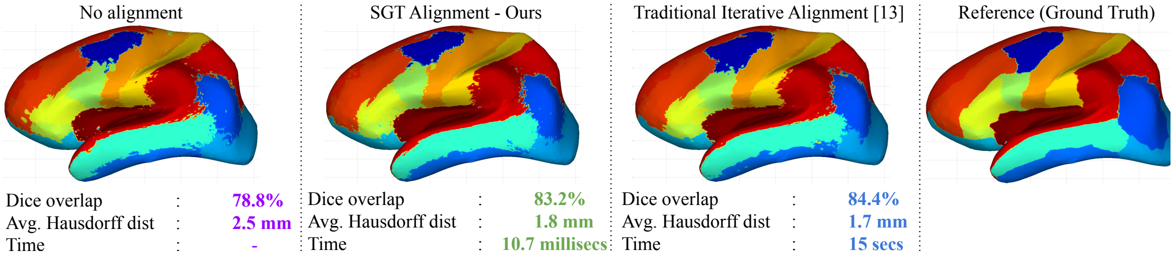

Fig. 3. **Brain parcellation:** Performance comparison of different alignment strategies using GCN measured with average Dice overlaps and Hausdorff distances. Model trained with no SGT yields a low Dice of 78.6% with irregular segmentation boundaries. Training an end-to-end SGT and GCN model achieves a Dice overlap of 83.2%, similar to the performance of a traditional alignment model, which scores 84.4%. The Hausdorff distances and qualitative results show equivalent results between the two methods. However, a significant speed gain, from 10.7 *seconds* to 10.7 *milliseconds* is achieved with our SGT.

Table 1. **Different alignment strategies with GCN approaches** – Average Dice overlaps (in %) over 32 parcels on test set are shown along with classification accuracy (in %), and average Hausdorff distances (in millimeters).

Method	**Dice overlap (%)**	**Accuracy (%)**	**Avg. Hausdorff (mm)**
No Alignment	78.82 ± 4.02	81.68 ± 3.88	2.54 ± 2.86
Pretrained + Orthogonal	81.97 ± 3.20	84.14 ± 2.88	1.99 ± 2.19
Pretrained + MSE	82.29 ± 4.46	84.38 ± 4.09	1.94 ± 2.23
End-to-end (Ours - 10.7 *milliseconds*)	83.26 ± 3.66	85.17 ± 3.48	1.85 ± 2.04
Traditional Alignment [13] (15 seconds)	84.42 ± 2.59	85.99 ± 2.53	1.76 ± 1.75

training SGT is set to $N = 1000$. These nodes are randomly sub-sampled for each subject during the training of our end-to-end model.

Our method is compared with different alignment strategies for graph parcellation. We show the limitations of ignoring the spectral alignment. The GCN trained with non-aligned spectral coordinates achieves a Dice overlap of 78.8%. This low performance is due to the incompatibility of eigenbases across brain surfaces. Training our end-to-end SGT with GCN provides a performance improvement of 4.4% over no alignment. Next, our transformer network is trained independently from the parcellation task in order to learn the SGT weights. The rationale of this experiment is to evaluate the use of a fixed alignment strategy for learning the GCN model. We evaluate the use of both SGT loss and orthogonal regularization independently. The model trained only with orthogonal regularization improves by 3.1%, from 78.8% to 81.9%. This increase indicates the benefit of learning regularized rotation and flipping. We see a further performance boost by training our SGT model with a mean square error. Table 1 shows a similar improvement of 3.4% compared to not using alignment. Note that updating the weights of both SGT and GCN in an end-to-end framework further guides the learning of the transformation matrix. This experiment setup trains the SGT model to learn a transformation most suitable for the parcellation task. Our end-to-end model indeed yields an improvement in average Dice overlap to 83.4% from 82.2% when trained separately. The results of the experiments are reported in Table 1.

4. CONCLUSION

This paper presents a novel end-to-end framework for learning a spectral transformation required for graph convolution networks. The proposed SGT network learns a transformation in the spectral domain that maps input spectral coordinates to a reference set. We first evaluate the optimal size of the coordinate set necessary for training the SGT network. Next, our experiments on brain surface parcellation validate the benefits of our alignment strategy. Training a GCN model without any alignment results in a low Dice overlap and irregular parcel boundaries as shown in Fig. 3. The conventional procedure of aligning different brain surfaces to a reference is an expensive computational step. Our method learns this alignment step automatically by capturing the geometry of the brain, yielding a Dice overlap of 83.2%. Qualitatively, as illustrated in Fig. 3, the performance of our method is similar to a GCN trained with traditional alignment, however computation times are reduced by a 1400-fold, from 15 seconds to 10.7 *milliseconds*. The use of SGT is evaluated in this paper with brain surface parcellation as an application. Nevertheless, our method can potentially be used for other surface analysis problems such as disease classification or identifying new geometry-related biomarkers.

Acknowledgment – This work was supported financially by the MITACS Globalink Internship Program, the Fonds de Recherche du Quebec (FQRNT), the Research Council of Canada (NSERC) and NVIDIA with the donation of a GPU.

5. REFERENCES

[1] Tianhao Zhang and Christos Davatzikos, "ODVBA: Optimally-discriminative voxel-based analysis," *TMI*, 2011.

[2] Xue Hua, Derrek P. Hibar, Christopher R. Ching, Christina P. Boyle, Priya Rajagopalan, et al., "Unbiased tensor-based morphometry: Improved robustness and sample size estimates for Alzheimer's disease clinical trials," *NeuroImage*, 2013.

[3] Jose Dolz, Karthik Gopinath, Jing Yuan, Herve Lombaert, Christian Desrosiers, and Ismail Ben Ayed, "Hyperdense-net: A hyper-densely connected CNN for multi-modal image segmentation," *TMI*, 2019.

[4] Konstantinos Kamnitsas, Christian Ledig, Virginia F. J. Newcombe, Joanna P. Simpson, Andrew D. Kane, David K. Menon, Daniel Rueckert, and Ben Glocker, "Efficient multi-scale 3D CNN with fully connected CRF for accurate brain lesion segmentation," *MedIA*, 2017.

[5] Michael M. Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst, "Geometric deep learning: Going beyond euclidean data," *IEEE Signal Processing*, 2017.

[6] Federico Monti, Davide Boscaini, Jonathan Masci, Emanuele Rodolà, Jan Svoboda, and Michael M. Bronstein, "Geometric deep learning on graphs and manifolds using mixture model CNNs," in *CVPR*, 2017.

[7] Ron Levie, Federico Monti, Xavier Bresson, and Michael M. Bronstein, "CayleyNets: Graph convolutional neural networks with complex rational spectral filters," in *ICLR*, 2018.

[8] Michael M. Bronstein, Klaus Glashoff, and Terry A. Loring, "Making laplacians commute," *CoRR*, 2013.

[9] Artiom Kovnatsky, Michael M. Bronstein, Alex M. Bronstein, Klaus Glashoff, and Ron Kimmel, "Coupled quasi-harmonic bases," in *Computer Graphics Forum*, 2013.

[10] Davide Eynard, Artiom Kovnatsky, Michael M. Bronstein, Klaus Glashoff, and Alexander M. Bronstein, "Multimodal manifold analysis by simultaneous diagonalization of Laplacians.," *IEEE PAMI*, 2015.

[11] Jonathan Masci, Davide Boscaini, Michael M. Bronstein, and Pierre Vandergheynst, "Geodesic convolutional neural networks on Riemannian manifolds," in *ICCV-3dRR*, 2015.

[12] Davide Boscaini, Jonathan Masci, Emanuele Rodolà, and Michael M. Bronstein, "Learning shape correspondence with anisotropic convolutional neural networks," in *NIPS*, 2016.

[13] Karthik Gopinath, Christian Desrosiers, and Herve Lombaert, "Graph convolutions on spectral embeddings for cortical surface parcellation," *MedIA*, 2019.

[14] Herve Lombaert, Michael Arcaro, and Nicholas Ayache, "Brain transfer: Spectral analysis of cortical surfaces and functional maps.," in *IPMI*, 2015.

[15] Karthik Gopinath, Christian Desrosiers, and Herve Lombaert, "Adaptive graph convolution pooling for brain surface analysis," in *IPMI*, 2019.

[16] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al., "Spatial transformer networks," in *NIPS*, 2015.

[17] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *CVPR*, 2017.

[18] Arno Klein, Satrajit S. Ghosh, Forrest S. Bao, Joachim Giard, Yrjö Häme, Eliezer Stavsky, et al., "Mindboggling morphometry of human brains," *PLOS Bio*, 2017.

[19] Karthik Gopinath, Christian Desrosiers, and Herve Lombaert, "Graph convolutions on spectral embeddings: Learning of cortical surface data," in *NeurIPS Workshop on Medical Imaging*, 2018.