Simplicial 2-Complex Convolutional Neural Networks

Eric Bunch
Machine Learning Research & Innovation Team
American Family Insurance
Madison, WI 53783
ebunch@amfam.com

Qian You
Machine Learning Research & Innovation Team
American Family Insurance
Madison, WI 53783
qyou@amfam.com

Glenn Fung
Machine Learning Research & Innovation Team
American Family Insurance
Madison, WI 53783
ebunch@amfam.com

Vikas Singh
University of Wisconsin, Madison
Madison, WI 53706
vsingh@biostat.wisc.edu

Abstract

Recently, neural network architectures have been developed to accommodate when the data has the structure of a graph or, more generally, a hypergraph. While useful, graph structures can be potentially limiting. Hypergraph structures in general do not account for higher order relations between their hyperedges. Simplicial complexes offer a middle ground, with a rich theory to draw on. We develop a convolutional neural network layer on simplicial 2-complexes.

1 Introduction and related work

In the last decade, neural networks have gained much popularity in the field of machine learning, in large part due to their modularity, flexibility, and ability to be successfully applied to many different types of problems. Recently there has been extensive work to formulate neural networks in settings where the data comes structured as a graph [8, 7, 1, 9, 3, 11, 12, 7], or a hypergraph [2, 6].

Graph structures in neural networks have proven extremely useful; however they are limited to pairwise connections of nodes. One drawback of the framework of hypergraphs is that they, in general, do not account for higher order relations among the edges of a hypergraph. Simplicial complexes offer a middle ground between graphs and hypergraphs, and have a rich theory to draw from. This document aims to begin to make the case for using the simplicial complex structure in machine learning by introducing a convolution layer on simplicial 2-complexes, akin to the popular convolution layer on graphs defined in [7].

1.1 Simplicial complexes

Definition 1. *A simplicial complex is a collection of subsets S of a finite set X that is closed under taking nonempty subsets.*

The finite sets comprising S are called the *faces* of S, and the dimension of a face s is defined as $\dim(s) = |s| - 1$. The zero-dimensional faces of S will be referred to as the *vertices* of S, and will be denoted v_i. Denote by S_k the collection of all k-dimensional faces of S. We will write an element of S_k as $s = (v_0, v_1, \ldots, v_{k-1})$, where the v_i's are the vertices comprising the face s. We say that the *dimension of S* is the maximum dimension of all its faces. We can see that a simplicial complex of dimension one is equivalent to a graph.

Let S be a simplicial complex, and assume that we have a specific enumeration of the vertices of of S: $v_0, \ldots, v_n$. A *k-chain* of S is defined to be a finite formal sum $\sum_{i=0}^{N} a_i s_i$, where $a_i \in \mathbb{R}$, and $s_i \in S_k$ such that $s_i = (v_{i_0}, v_{i_1}, \ldots, v_{i_{k-1}})$ where $i_0 < i_1 < \cdots < i_{k-1}$. We will denote collection of k-chains of S by C_k. Each C_k forms a finite dimensional vector space over $\mathbb{R}$, where a choice of basis will be taken to be the collection S_k of k-dimensional faces of S. If the collection S_k is empty, we take C_k to be the zero vector space. We also define $C_0 = 0$. For each $k = 1, 2, \ldots$, there is a linear map $\partial_k : C_k \to C_{k-1}$ defined by

$$\partial_k(v_0, \ldots, v_{k-1}) = \sum_{i=0}^{k-1} (-1)^i (v_0, \ldots, \widehat{v_i}, \ldots, v_{k-1}) \tag{1}$$

where $(v_0, \ldots, \widehat{v_i}, \ldots, v_{k-1}) = (v_0, \ldots, v_{i-1}, v_{i+1}, \ldots, v_{k-1}) \in C_{k-1}$. We also define $\partial_0 : C_1 \to C_0$ to be the zero map.

Given the bases described, the linear maps ∂_k can be written as $n_{k-1} \times n_k$ matrices, which we will denote by $\mathbf{B}_k$, following [10]. For each ∂_k, there exists a map $\partial_k^* : C_k \to C_{k+1}$, which is the adjoint of ∂_k. The matrix representation of ∂_k^* is $\mathbf{B}_k^*$. Then we define the kth *Laplacian* of S to be the $n_k \times n_k$ matrix

$$\mathbf{L}_k = \mathbf{B}_k^* \mathbf{B}_k + \mathbf{B}_{k+1} \mathbf{B}_{k+1}^* . \tag{2}$$

We can see that since ∂_0 is the zero map, that $\mathbf{L}_0 = \mathbf{B}_1 \mathbf{B}_1^*$. In the case when S is a one dimensional complex (i.e. a graph), $\mathbf{L}_0$ coincides with the usual definition of the graph Laplacian, which is typically defined as $L = D - A$, where D and A are the diagonal degree matrix and the adjacency matrix respectively of the graph.

As is the case with graph Laplacians, the $\mathbf{L}_k$ describe how to appropriately propagate a signal on S. We will use these $\mathbf{L}_k$ as well as the $\mathbf{B}_k$ and $\mathbf{B}_k^*$ construct a convolutional neural net that generalizes graph neural nets.

2 Simplicial 2-complex convolution layer

Let S be a simplicial complex of dimension 2. For each $k = 0, 1, 2$, we write $n_k = |S_k|$, and we assume we have matrices X_k of dimension $n_k \times F_k$. These will be our *feature matrices*, and F_k is the feature dimension. Intuitively, we think of X_k as describing an F_k-dimensional signal on the dimension k faces of S. One benefit of this approach is that the feature dimension, F_k, can vary with k, allowing us to encode multimodal features.

One crucial step in this setup is properly normalizing the $\mathbf{L}_k$. There are a variety of ways to approach this, either through normalizing the $\mathbf{L}_k$ directly [10], to use a theory of weightings on the simplicial complex itself [5], or to appeal to a more abstract setting [4]. We choose here to follow [10] due in part to ease of implementation, and in part to the impracticality of structuring a typical data set as a high-dimensional simplicial complex.

2.1 Normalization

We follow [10], Def. 3.3 and define the matrices

$$\widetilde{\mathbf{D}}_2 = \max(\operatorname{diag}(|\mathbf{B}_1|\,\mathbb{1}), \mathbf{I}) \qquad \begin{aligned} \mathbf{D}_1 &= 2\operatorname{diag}(|\mathbf{B}_1|\,\mathbf{D}_2\,\mathbb{1}) \\ \mathbf{D}_2 &= \max(\operatorname{diag}(|\mathbf{B}_2|\,\mathbb{1}), \mathbf{I}) \end{aligned} \qquad \begin{aligned} \mathbf{D}_4 &= \mathbf{I} \\ \mathbf{D}_5 &= \operatorname{diag}(|\mathbf{B}_2|\,\mathbb{1}) \end{aligned}$$

$$\widetilde{\mathbf{D}}_3 = \mathbf{I} \qquad\qquad\qquad\qquad \mathbf{D}_3 = \frac{1}{3}\mathbf{I}$$

We also follow [10] and define

$$\begin{aligned} \widetilde{\mathbf{L}}_0 &= \mathbf{B}_1\,\widetilde{\mathbf{D}}_3\,\mathbf{B}_1^*\,\widetilde{\mathbf{D}}_2^{-1} \\ \widetilde{\mathbf{L}}_1 &= \mathbf{D}_2\,\mathbf{B}_1^*\,\mathbf{D}_1^{-1}\,\mathbf{B}_1 + \mathbf{B}_2\,\mathbf{D}_3\,\mathbf{B}_2^*\,\mathbf{D}_2^{-1} \end{aligned} \qquad \widetilde{\mathbf{L}}_2 = \mathbf{D}_4\,\mathbf{B}_2^*\,\mathbf{D}_5^{-1}\,\mathbf{B}_2$$

Define the following

$$\begin{aligned} \mathbf{A}_0^u &= \widetilde{\mathbf{D}}_2 - \mathbf{B}_1\,\widetilde{\mathbf{D}}_3\,\mathbf{B}_1^* \\ \mathbf{A}_1^u &= \mathbf{D}_2 - \mathbf{B}_2\,\mathbf{D}_3\,\mathbf{B}_2^* \end{aligned} \qquad\qquad \begin{aligned} \mathbf{A}_1^d &= \mathbf{D}_2^{-1} - \mathbf{B}_1^*\,\mathbf{D}_1^{-1}\,\mathbf{B}_1 \\ \mathbf{A}_2^d &= \mathbf{D}_4^{-1} - \mathbf{B}_2^*\,\mathbf{D}_5^{-1}\,\mathbf{B}_2\,. \end{aligned}$$

Finally, we define

$$\begin{aligned} \widetilde{\mathbf{A}}_0^u &= (\mathbf{A}_0^u + \mathbf{I})(\widetilde{\mathbf{D}}_2 + \mathbf{I})^{-1} \\ \widetilde{\mathbf{A}}_1^u &= (\mathbf{A}_1^u + \mathbf{I})(\mathbf{D}_2 + \mathbf{I})^{-1} \end{aligned} \qquad\qquad \begin{aligned} \widetilde{\mathbf{A}}_1^d &= (\mathbf{D}_2 + \mathbf{I})(\mathbf{A}_1^d + \mathbf{I}) \\ \widetilde{\mathbf{A}}_2^d &= (\mathbf{D}_4 + \mathbf{I})(\mathbf{A}_2^d + \mathbf{I}). \end{aligned}$$

The $\mathbf{A}_i^\alpha$ are meant to generalize the adjacency matrix of a graph, and the $\widetilde{\mathbf{A}}_i^\alpha$ are meant to play the role of normalized adjacency matrices with added self-loops, as used in [7].

2.2 Convolution layer

We define a convolutional neural network layer using the simplicial complex structure in the following way. Assume we have hidden features at level h: $\{X_k^{(h)}\}_{k=0}^2$, where $X_k^{(h)}$ is a matrix of dimension $n_k \times F_k^{(h)}$, and $X_k^{(0)} = X_k$. Then $X_k^{(h+1)}$ can be obtained by

$$X_0^{(h+1)} = \sigma\left(\mathbf{D}_1^{-1}\,\mathbf{B}_1\,X_1^{(h)}W_{0,1}^{(h)} + \widetilde{\mathbf{A}}_0^u X_0^{(h)}W_{0,0}^{(h)}\right) \tag{3}$$

$$X_1^{(h+1)} = \sigma\left(\mathbf{B}_2\,\mathbf{D}_3\,X_2^{(h)}W_{1,2}^{(h)} + (\widetilde{\mathbf{A}}_1^d + \widetilde{\mathbf{A}}_1^u)X_1^{(h)}W_{1,1}^{(h)} + \mathbf{D}_2\,\mathbf{B}_1^*\,\mathbf{D}_1^{-1}\,X_0^{(h)}W_{1,0}^{(h)}\right) \tag{4}$$

$$X_2^{(h+1)} = \sigma\left(\widetilde{\mathbf{A}}_2^d X_2^{(h)}W_{2,2}^{(h)} + \mathbf{D}_4\,\mathbf{B}_2^*\,\mathbf{D}_5^{-1}\,X_1^{(h)}W_{2,1}^{(h)}\right) \tag{5}$$

where σ is an activation function, the $W_{i,j}^{(h)}$'s are matrices of learnable weights of dimensions $F_i^{(h)} \times F_j^{(h)}$. One can see that if we have a simplicial complex of dimension one (i.e. a graph), and the features X_1 are all zeros, then the hidden layer output of the simplicial complex convolutional layer in Eq. 3 becomes

$$X_0^{(h+1)} = \sigma\left(\widetilde{\mathbf{A}}_0^u X_0^{(h)}W_{0,0}^{(h)}\right). \tag{6}$$

This coincides with the definition of a convolution layer in a graph convolutional network [7].

3 Experiments

Once we have all the $X_k^{(h)}$, for $k = 0, 1, 2$, and $h = 0, \ldots, H$, we horizontally concatenate the features to form $X_k^{[0:H]} = [X_k^{(0)} : X_k^{(1)} : \cdots : X_k^{(H)}]$ to form an $n_k \times F_H$ matrix, where

$F_H = \sum_{h=0}^{H} F_k^{(h)}$. Then we have 3 feature matrices $\{X_k^{[0:H]}\}_{k=0}^2$. These can then be used for further downstream learning tasks, e.g. classifying faces of a certain dimension (akin to node or edge classification in graph learning), joint tasks, or classifying a simplicial complex itself when it is among a collection of simplicial complexes. The latter setting in general requires ways of dealing with simplicial complexes with differing structures.

3.1 MNIST classification

We give a brief explanation of how an image can be encoded as a simplicial 2-complex with feature matrices, and employ simplicial complex convolution layers in a network to classify handwritten digits in the MNIST data set. We detail a proof of concept for how these simplicial complex convolution layers can be used to augment traditional convolution layers through an ablation study.

Given an image of size $h \times w$ pixels, and a choice of *kernel size* k and *step size* s (the same idea from traditional CNNs), we form a grid of zero faces that is $\left(\lfloor \frac{h-k}{s} \rfloor + 1\right) \times \left(\lfloor \frac{w-k}{s} \rfloor + 1\right)$. We then add 1-faces to connect any zero faces that are horizontally, vertically, or diagonally adjacent. We then add two faces wherever possible; that is, a triplet of zero faces holds a two face whenever they are all pairwise connected by a one face. Figure 1 gives an example depiction of such a simplicial complex. The feature vector attached to a zero face is the $k^2 \times 1$ vector resulting from flattening the $k \times k$ square of pixel values corresponding to that zero face. These are collected into an $n_0 \times k^2$ matrix X_0, with rows indexed by zero faces, and columns indexed by feature values. To both one faces and two faces, the features attached are 1×1 vectors with the value 1.

The ablation study takes as the base a neural net that has one traditional convolution layer, followed by a fully connected network with one hidden layer. Next, a single graph convolution layer is prepended to this network to form the first comparison. The graph is taken to be the one formed by taking only the zero and one faces of the simplicial complex described above. For the final comparison, a single simplicial complex convolution layer is prepended to the initial architecture described.

We describe first a particular 1D convolution layer. For a feature matrix X of dimension $n \times F$, we flatten X to a matrix of dimension $1 \times nF$. We denote by $\mathbf{CONV1D}_F$ a 1D convolution layer with kernel size of F and step size F. We write $\mathbf{CONV1D}$ when the input feature size is understood. We can see that in the setting described above, $\mathbf{CONV1D}_{k^2}$ applied to a flattened X_0 (of size $1 \times n_0 k^2$) is equivalent to a 2D convolution layer with kernel size k and step size s.
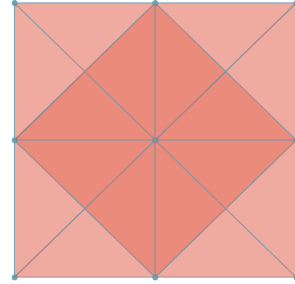

Figure 1: Depiction of the type of 2-simplicial complex structure constructed for the experiments described.

We denote by $\mathbf{FC}$ a fully connected network with one hidden layer of size 32, and output layer of size 10 (for the 10 classes of MNIST). We denote by $\mathbf{GCONV}$ a graph convolutional layer with embedding dimension 32. We denote by $\mathbf{SCCONV}$ a simplicial complex convolutional layer with embedding dimensions of 32 for each face dimension. In the ablation study, the combination $\mathbf{SCCONV} - \mathbf{CONV1D}$ applies a 1D convolution layer to each of the three feature matrices; one for each face dimension.

Each model was trained (tested) on a stratified sample of 1000 images from the designated training (testing) portion of the MNIST data set. Each model was trained for 300 epochs with a batch size of 8, learning rate of 10^{-4}, dropout rate of 10% on the fully connected layers, and batch normalization. The kernel size and step size are both taken to be 4 for all models. The metrics reported in Table 1 are the average accuracy, with the standard deviation denoted in the usual fashion, for five iterations of the experiment setup described above. On each experiment, the test and train set are sampled anew.

We note that these metrics are not state of the art, and there is still a great deal to still investigate in this direction. The results of this ablation study are suggestive enough to warrant further study of the simplicial complex convolution layer in general.

4

Model	acc. $\pm$ std.
CONV1D – **FC**	87.40 ± 0.70
GCONV – **CONV1D** – **FC**	87.42 ± 0.59
SCCONV – **CONV1D** – **FC**	$\mathbf{91.10} \pm 0.40$

Table 1: Ablation study results. Statistics are calculated over five experiments.

References

[1] J. Atwood and D. Towsley. Diffusion-convolutional neural networks. In *NIPS*, 2016.

[2] S. Bai, F. Zhang, and P. H. S. Torr. Hypergraph convolution and hypergraph attention. *ArXiv*, abs/1901.08150, 2019.

[3] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun. Spectral networks and locally connected networks on graphs. *CoRR*, abs/1312.6203, 2014.

[4] J. Hansen and R. Ghrist. Toward a spectral theory of cellular sheaves. *Journal of Applied and Computational Topology*, 3(4):315–358, Aug 2019.

[5] D. Horak and J. Jost. Spectra of combinatorial laplace operators on simplicial complexes. *Advances in Mathematics*, 244:303–336, 2011.

[6] E. S. Kim, W. Y. Kang, K. W. On, Y. J. Heo, and B. T. Zhang. Hypergraph attention networks for multimodal learning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14569–14578, 2020.

[7] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017.

[8] M. Niepert, M. Ahmed, and K. Kutzkov. Learning convolutional neural networks for graphs. In M. F. Balcan and K. Q. Weinberger, editors, *PMLR*, volume 48 of *Proceedings of Machine Learning Research*, pages 2014–2023, New York, New York, USA, 20–22 Jun 2016. PMLR.

[9] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2009.

[10] M. T. Schaub, A. R. Benson, P. Horn, G. Lippner, and A. Jadbabaie. Random walks on simplicial complexes and the normalized hodge 1-laplacian. *SIAM Review*, 62(2):353–391, Jan 2020.

[11] S. Verma and Z.-L. Zhang. Graph capsule convolutional neural networks, 2018.

[12] M. Zhang, Z. Cui, M. Neumann, and Y. Chen. An end-to-end deep learning architecture for graph classification. In *AAAI*, pages 4438–4445, 2018.