SIMPLICIAL COMPLEX REPRESENTATION LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Simplicial complexes form an important class of topological spaces that are frequently used in many application areas such as computer-aided design, computer graphics, and simulation. Representation learning on graphs, which are just 1-d simplicial complexes, has witnessed a great attention and success recently. Due to the additional complexity higher dimensional simplicial hold, there has not been enough effort to extend representation learning to these objects especially when it comes to entire-simplicial complex representation learning. In this work, we propose a method for simplicial complex-level representation learning that embeds a simplicial complex to a universal embedding space in a way that complex-to-complex proximity is preserved. Our method utilizes a simplex-level embedding induced by a pre-trained simplicial autoencoder to learn an entire simplicial complex representation. To the best of our knowledge, this work presents the first method for learning simplicial complex-level representation.

1 INTRODUCTION

Object representation learning aims to learn a mapping that embeds the elementary components of this object into some Euclidean space while preserving the object's structural information. Recently, such methods have gained a great momentum especially with graph representation learning. Specifically, the latter has attracted considerable popularity over the past few years with success in both node-level representation learning (Cui et al. (2018)) and entire graph learning (Narayanan et al. (2017); Tsitsulin et al. (2018)). The applications of such representation on graphs are diverse as they can be used for almost any downstream machine learning task on domains such as graph classification (Hamilton et al. (2017a)) or graph similarity (Heimann et al. (2018)).

Despite the success of graph representation learning in the past few years, there has not been enough efforts to extend representation learning to simplicial complexes. The higher dimensional simplicial complexes often hold additional structure over graphs that might be critical in modeling and must be incorporated to learn the correct representation. For instance, when a simplicial complex is a triangulated manifold and the manifoldsness property is important, then the learned representation must take this property into account. Motivated by the success of graph representation learning, we propose a method for learning simplicial complex representation. Our method utilizes a complex autoencoder proposed in (Hajij et al. (2020a)) and learns an entire simplicial complex representation extracted from simplices embeddings vectors induced by the simplicial complex autoencoder. Our learning function maps every simplicial complex to a universal embedding space in a way that complex-to-complex proximity is preserved. Learning simplicial complex-level representation is essential to perform downstream machine learning tasks on these objects such as simplicial complex classification and similarity ranking. See for instance (Hajij et al. (2018); Fey & Lenssen (2019); Ying et al. (2018); Narayanan et al. (2017)) for related studies on graphs.

The literature of entire graph representation learning is rich and many methods have been proposed recently including Laplacian-based methods (de Lara & Pineau (2018); Tsitsulin et al. (2018)), Implicit factorization techniques (Chen & Koga (2019); Narayanan et al. (2017)), GNN-based methods (Bai et al. (2019)) and pooling based methods (Ying et al. (2018); Bodnar et al. (2020)). We refer the reader to (Cui et al. (2018)) for a recent survey on network embedding. In addition, simplicial complex representation learning, inspired by the success of node2vec Angles & Gutierrez (2008) and Word2Vec Mikolov et al. (2013), started to get attention recently. For example, the works in (Billings et al. (2019); Schaub et al. (2020)) define simplicies emebddings based random walks on simplicial complexes. This was generalized to k-simplex embeddings in Hacker (2020). A general

cell complex autoencoder scheme that describes these random walk-based representations as special cases was suggested in (Hajij et al. (2020a)).

While there are several methods for learning simplex-level representation (Billings et al. (2019); Schaub et al. (2020); Hajij et al. (2020a); Hacker (2020)), the work herein is the first to propose a learning representation of the entire simplicial complex. The rest of the paper is organized as follows. Notation and necessary definitions on simplicial complexes are given in Section 2. Section 3 is devoted to reviewing cell complex neural networks. Our proposed method is given in Section 4.1.

2 Simplicial Complex Neighborhood Matrices

This section provides the necessary notations to define neighborhood matrices between simplices in a simplicial complex, and hence, we assume the reader has familiarity with basic definitions of simplicial complexes (Hatcher (2005)). Let X be a simplicial complex and n be the dimension of X. Recall that the dimension of X is the dimension of the highest simplex in X. For any $0 \leq k \leq n$, we denote the set of all k-simplices in X by X^k. If X is a simplicial complex of dimension n, then for every $0 < m \leq n$ we denote the set of simplicies in X with dimension less than m by $X^{<m}$. The set $X^{>m}$ is defined similarly. In this work, we assume that the complex X is unoriented. However, the following notion of neighbored on simplicial complexes can be easily extended to oriented simplicial complexes; see for instance (Hajij et al. (2020a); Glaze et al. (2021)) and more recently Bodnar et al. (2021) for various considerations on oriented simplical complexes in the context of neural network computations.

Adjacency relations can be defined on simplicial complexes in a similar fashion as they are defined on graphs. Specifically, let X be a simplicial complex and let c^n denotes a n-simplex in X, and $facets(c^n)$ denotes the set of all $(n-1)$-simplicies X incident to c^n. Two n-simplices a^n and b^n are said to be *adjacent* if there exists an $(n+1)$-simplex c^{n+1} such that $a^n, b^n \in facets(c^{n+1})$. The set of all simplices adjacent to a simplex a in X is denoted by $\mathcal{N}_{adj}(a)$. Dually, a^n and b^n are *coadjacent* in X if there exists an $(n-1)$-simplex c^{n-1} with $a^n, b^n \in cofacets(c^{n-1})$. The set of all cells adjacent to a simplices a in X is denoted by $\mathcal{N}_{adj}(a)$ while the set of all simplices co-adjacent to a simplex a in X is denoted by $\mathcal{N}_{co}(a)$. If a^n, b^n are n-simplices in X, then we define the set $\mathcal{CO}[a^n, b^n]$ to be the intersection of $cofacets(a^n) \cap cofacets(b^n)$. Similarly, the set $\mathcal{C}[a^n, b^n]$ is defined to be the intersection of $facets(a^n) \cap facets(b^n)$. Observe that these notions generalize the analogous notions of adjacency/co-adjacency matrices on graphs. Precisely, let X be a simplicial complex of dimension n, N be the total number of simplices X, and define $\hat{N} := N - |X^n|$. Let $c_1, \cdots, c_{\hat{N}}$ denotes all the simplices in $X^{<n}$. The *adjacency matrix* of X, denoted by A_{adj}, is a matrix of dimension $\hat{N} \times \hat{N}$ and defined by setting $A_{adj}(i,j) = |\mathcal{CO}[c_i, c_j]|$ if the simplex c_i is adjacent to c_j and zero otherwise. We denote the adjacency matrix between k-simplices in X by A^k_{adj}, where $0 \leq k < n$. The co-adjacency matrices A_{co}, A^k_{co} are defined dually by storing $|\mathcal{C}[c_i, c_j]|$ where the simplices c_i and c_j are co-adjacent.

3 Neural Networks on Complexes

This section briefly reviews the basic definitions and notations of cell complex networks (CXN) introduced in (Hajij et al. (2020a)) as it is applicable in our context on simplicial complexes. Specifically, every simplicial complex X is a cell complex where the k-simplexes X in that complex are precisely the k-cells. In what follows, we will use the terms "cell" and "simplex" interchangeably to refer to simplices in a given complex X.

The input for a CXN is specified by cell embeddings $H^{(0)}_m \in \mathbb{R}^{|X^m| \times d_0}$ that define the initial cell features on every m-cell in X. Here, d_0 is the dimension of the input feature embedding dimension of the cells. Given the desired depth $L > 0$ of the CXN net one wants to define on the complex X, the *adjacency message passing scheme* (AMPS) on X consists of $L \times n$ and it is defined by :

$$H^{(k)}_m := M(A_{adj}, H^{(k-1)}_m, H^{(k-1)}_{m+1}; \theta^{(k)}_m) \tag{1}$$

where $0 \leq m \leq n-1$, $1 \leq k \leq L$, $H^{(k)}_m \in \mathbb{R}^{|X^m| \times d_k}$ are the cell embeddings computed after k steps of applying 1, and $\theta^{(k)}_m$ is a trainable weight vector at the layer k, M is the message

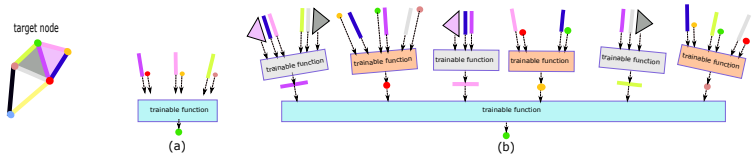

Figure 1: Two layers cell complex neural network (Hajij et al. (2020a)) illustrated on a simplicial complex. The computations are illustrated with respect to the green target vertex.

propagation function that depends on the weights $\theta_m^{(k)}$, the cell embeddings $H_m^{(k)}$ and the adjacency matrix of X. The propagation function M can be implemented in many ways. For instance, in (Hajij et al. (2020a)) a generalization for graph convolutional neural networks (Kipf & Welling (2016)) to convolutional cell complex networks was provided. See also 6 for other variants of the above message passing scheme. Finally, see (Bunch et al. (2020); Ebli et al. (2020)) for related implementations on simplicial complexes.

Note that the information flow using equation (1) on the complex from the lower dimensional cells to the higher ones. Further, note that the message passing scheme given in 1 does not update the feature vectors associated with the final $n-$ cells on the complex. If such a property is desirable, then equation 1 must be adjusted using co-adjacency information of the simplicial complex. See (Hajij et al. (2020a)) for details. Figure 1 demonstrates how the cell embeddings are updated.

4 Entire Simplicial Complex Learning

Our proposed method for learning entire simplicial complex representation relies on collecting node-level simplicial representation and combining them together in order to obtain simplicial complex-level representation. In this section, we review the AMPS-simplicial autoencoder introduced in (Hajij et al. (2020a)). We then show how it can be utilized to obtain an entire-level simplicial representation by using metric learning.

4.1 Simplicial Complex AutoEncoders

Let X be a simplicial complex of dimension n. A simplicial complex autoencoder on X consists of the following three components (Hamilton et al. (2017b); Hajij et al. (2020a)):

- A encoder-decoder system. Specifically, An encoder is a function of the form : $enc : X^{<n} \to \mathbb{R}^d$ and it associates to every $k-$ simplex a^k in X an embedding $\mathbf{z}_{a^k}$ in $\mathbb{R}^d$. On the other hand, a decoder is a function of the form : $dec : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^+$ and it associates to every pair of simplex embeddings $(\mathbf{z}_{a^k}, \mathbf{z}_{c^l})$ a measure of similarity $dec(\mathbf{z}_{a^k}\mathbf{z}_{c^l})$ that quantifies some notion of relationship between a^k and c^l. The functions enc and dec are trainable functions. In particular, the encoder can be chosen to be a cell complex network as illustrated in Section 3.
- A user-defined similarity measure. We seek to train the encoder-decoder functions such that the trained similarity is as close as possible to the user-defined similarity: $dec(enc(a^k), enc(c^l)) = dec(\mathbf{z}_{a^k}, \mathbf{z}_{c^l}) \approx sim_X(a^k, c^l)$, where $sim_X : X^{<n} \times X^{<n} \to \mathbb{R}^+$ is a user-defined function such that $sim_X(a^k, c^l)$ reflects a user-defined similarity between the two simplicies a^k and c^l in $X^{<n}$. For instance, the similarity measure function on X can be simply chosen the adjacency matrix A_{adj} defined in Section 3.
- A user-defined loss function. Training the encoder-decoder system is done by specifying a loss function $l : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ and defining:

$$\mathcal{L}_k = \sum_{\textit{all possible } \mathcal{CO}[a^k, c^k] \subset X^{k+1}} l(dec(enc(\mathbf{z}_{a^k}), enc(\mathbf{z}_{c^k})), sim(a^k, c^k)), \tag{2}$$

  and then final set: $\mathcal{L} := \sum_{k=0}^{n-1} \mathcal{L}_k$.

Table 1 shows several concert methods to define the autoencoders on simplicial complexes. See also Section 6 for other variants of simplicial complex autoencoders with other message passing schemes.

Table 1: Various definitions of simplicial somplex autoEncoders.

Method	Decoder	similarity	Loss
Laplacian eigenmaps Belkin & Niyogi (2001)	$\|\|\mathbf{z}_a - \mathbf{z}_c\|\|_2^2$	general	$dec(\mathbf{z}_a, \mathbf{z}_c).sim(a, c)$
Inner product methods Ahmed et al. (2013)	$\mathbf{z}_a^T \mathbf{z}_c$	$A_{adj}(a, c)$	$\|\|dec(\mathbf{z}_a, \mathbf{z}_c) - sim(a, c)\|\|_2^2$
Random walk methods Grover & Leskovec (2016); Perozzi et al. (2014)	$\frac{e^{\mathbf{z}_a^T \mathbf{z}_c}}{\sum_{b \in X^k} e^{\mathbf{z}_a^T \mathbf{z}_b}}$	$p_X(a\|c)$	$-log(dec(\mathbf{z}_a, \mathbf{z}_c))$

4.2 Learning Entire Simplicial Complex Embedding

Let $enc : X^{<n} \to \mathbb{R}^d$ be simplicial complex encoder $enc : X^{<n} \to \mathbb{R}^d$. Denote by $\mathcal{U}_X \in \mathbb{R}^{\hat{N} \times d}$ to the simplices embeddings of $X^{<n}$ that are induced by the function enc. Our proposed method relies on learning a weighted sum of the simplex-level representations encoded in $\mathcal{U}_X$. Specifically, we seek to learn a simplicial complex-level embedding of the form:

$$\mathbf{h}_X = \sum_{m=1}^{\hat{N}} w_m(\mathcal{U}_X; W^{(k)}) \mathbf{z}_m \tag{3}$$

where $w_m(\mathcal{U}_X; W^{(k)}) \in \mathbb{R}$ is a weight of the simplex embedding $\mathbf{z}_m$ that depends on $\mathcal{U}_X$ and parametrized by $W^{(k)} \in \mathbb{R}^{d \times d}$, a trainable weight matrix [1]. The weight w_m can be chosen in many different ways, here we simply follow (Bai et al. (2019)) and define the weight as:

$$w_m(\mathcal{U}_X; W^{(k)}) = \sigma((\mathbf{z}_m)^T \mathrm{RELU}(W^{(k)}(\sum_{n=1}^{\hat{N}} \mathbf{z}_n)), \tag{4}$$

where $\sigma(x) = \frac{1}{1+exp(-x)}$. Finally, the embedding $\mathbf{h}_X$ can be learned in multiple ways. For instance, given a collection of simplicial complex $\{X_i\}_{i=1}^m$ one may learn a complex-to-complex proximity embeddings by minimizing the objective: $\mathcal{L} = \sum_{i=1}^m \sum_{j=1}^m (||h_{X_i} - h_{X_j}|| - d_{ij})^2$, where $D = [d_{ij}]$ is an appropriately chosen distance matrix on the simplicial complexes $\{X_i\}_{i=1}^m$. For example, the Haussdorf distance on simplicial complexes (Marin (2020)) can be employed to compute the distance matrix D. Alternatively, in special case when the simplicial complex is a triangulated mesh, more efficient methods to compute the metrics can be utilized such as persistence homology-based metrics (Hajij et al. (2020b); Zhang et al. (2019)) or Laplacian-based methods (Crane et al. (2013)).

5 Conclusion and Further Directions

In Section 4.2, we describe one way to learn the embeddings $\mathbf{h}_X$ in an unsupervised fashion. The method learns the proximity between simplical complexes that is encoded in a pre-computed matrix D on a dataset of simplicial complexes. The problem with this method is that it requires the computation of the entire distance matrix which might be computationally inefficient. There are many other potentially good methods to learn such embeddings in an end-to-end fashion. For instance, a potential method to learn the metric of simplicial complex embeddings can be done by utilizing triplet loss method as proposed in (Hoffer & Ailon (2015)). Metric learning with the triplet loss method construct a triplet net which consists of three shared parameter feedforward networks. The network is fed three embeddings $\mathbf{h}_X$, $\mathbf{h}_{X^+}$ and $\mathbf{h}_{X^-}$ where $\mathbf{h}_X$ and $\mathbf{h}_{X^+}$ are of the same class, and $\mathbf{h}_X$ and $\mathbf{h}_{X^-}$ are of different class. We stress here the fact that while a binary label is utilized when choosing the triplet (X, X^+, X^-), triplet network can effectively learn a metric and determine which simplicial complexes are closer to a given complex X. In other words, the interpretation of sharing the same class is correlated with embedding closeness in the embedding space (Hoffer & Ailon (2015)). The advantages of the triplet loss method is that we do not need to pre-compute a distance matrix on the simplicial complexes training set. However, the triplet loss method requires a labeled training dataset which might not be always available.

[1] Note that in equation (3), we did not include the dimension of simplex embedding in the training. This restriction is not needed and we are only making this assumption for notational convenience.

REFERENCES

Amr Ahmed, Nino Shervashidze, Shravan Narayanamurthy, Vanja Josifovski, and Alexander J Smola. Distributed large-scale natural graph factorization. In *Proceedings of the 22nd international conference on World Wide Web*, pp. 37–48, 2013.

Renzo Angles and Claudio Gutierrez. Survey of graph database models. *ACM Computing Surveys (CSUR)*, 40(1):1–39, 2008.

Yunsheng Bai, Hao Ding, Yang Qiao, Agustin Marinovic, Ken Gu, Ting Chen, Yizhou Sun, and Wei Wang. Unsupervised inductive whole-graph embedding by preserving graph proximity. 2019.

Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. *Advances in neural information processing systems*, 14:585–591, 2001.

Jacob Charles Wright Billings, Mirko Hu, Giulia Lerda, Alexey N Medvedev, Francesco Mottes, Adrian Onicas, Andrea Santoro, and Giovanni Petri. Simplex2vec embeddings for community detection in simplicial complexes. *arXiv preprint arXiv:1906.09068*, 2019.

Cristian Bodnar, Cătălina Cangea, and Pietro Liò. Deep graph mapper: Seeing graphs through the neural lens. *arXiv preprint arXiv:2002.03864*, 2020.

Cristian Bodnar, Frasca Fabrizio, Yu Wang, Nina Otter, Guido Montúfar, Pietro Liò, and Bronstein Michael. Weisfeiler and lehman go topological: Message passing simplicial networks. *arXiv preprint arXiv:2103.03212*, 2021.

Eric Bunch, Qian You, Glenn Fung, and Vikas Singh. Simplicial 2-complex convolutional neural nets. *NeurIPS workshop on Topological Data Analysis and Beyond*, 2020.

Hong Chen and Hisashi Koga. Gl2vec: Graph embedding enriched by line graphs with edge features. In *International Conference on Neural Information Processing*, pp. 3–14. Springer, 2019.

Keenan Crane, Clarisse Weischedel, and Max Wardetzky. Geodesics in heat: A new approach to computing distance based on heat flow. *ACM Transactions on Graphics (TOG)*, 32(5):1–11, 2013.

Peng Cui, Xiao Wang, Jian Pei, and Wenwu Zhu. A survey on network embedding. *IEEE Transactions on Knowledge and Data Engineering*, 31(5):833–852, 2018.

Nathan de Lara and Edouard Pineau. A simple baseline algorithm for graph classification. *arXiv preprint arXiv:1810.09155*, 2018.

Stefania Ebli, Michaël Defferrard, and Gard Spreemann. Simplicial neural networks. *NeurIPS 2020Workshop TDA and Beyond*, 2020.

Matthias Fey and Jan Eric Lenssen. Fast graph representation learning with pytorch geometric. *arXiv preprint arXiv:1903.02428*, 2019.

Nicholas Glaze, T Mitchell Roddenberry, and Santiago Segarra. Principled simplicial neural networks for trajectory prediction. *arXiv preprint arXiv:2102.10058*, 2021.

Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 855–864, 2016.

Celia Hacker. k-simplex2vec: a simplicial extension of node2vec. *NeurIPS workshop on Topological Data Analysis and Beyond*, 2020.

Mustafa Hajij, Bei Wang, Carlos Scheidegger, and Paul Rosen. Visual detection of structural changes in time-varying graphs using persistent homology. In *2018 ieee pacific visualization symposium (pacificvis)*, pp. 125–134. IEEE, 2018.

Mustafa Hajij, Kyle Istvan, and Ghada Zamzmi. Cell complex neural networks. *NeurIPS 2020 Workshop TDA and Beyond*, 2020a.

Mustafa Hajij, Elizabeth Munch, and Paul Rosen. Fast and scalable complex network descriptor using pagerank and persistent homology. In *2020 International Conference on Intelligent Data Science Technologies and Applications (IDSTA)*, pp. 110–114. IEEE, 2020b.

William L Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. *arXiv preprint arXiv:1706.02216*, 2017a.

William L Hamilton, Rex Ying, and Jure Leskovec. Representation learning on graphs: Methods and applications. *arXiv preprint arXiv:1709.05584*, 2017b.

Allen Hatcher. *Algebraic topology*. 2005.

Mark Heimann, Haoming Shen, Tara Safavi, and Danai Koutra. Regal: Representation learning-based graph alignment. In *Proceedings of the 27th ACM international conference on information and knowledge management*, pp. 117–126, 2018.

Elad Hoffer and Nir Ailon. Deep metric learning using triplet network. In *International workshop on similarity-based pattern recognition*, pp. 84–92. Springer, 2015.

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

Ivan Marin. Hausdorff metric between simplicial complexes. *arXiv preprint arXiv:2001.11707*, 2020.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

Annamalai Narayanan, Mahinthan Chandramohan, Rajasekar Venkatesan, Lihui Chen, Yang Liu, and Shantanu Jaiswal. graph2vec: Learning distributed representations of graphs. *arXiv preprint arXiv:1707.05005*, 2017.

Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 701–710, 2014.

Michael T Schaub, Austin R Benson, Paul Horn, Gabor Lippner, and Ali Jadbabaie. Random walks on simplicial complexes and the normalized hodge 1-laplacian. *SIAM Review*, 62(2):353–391, 2020.

Nino Shervashidze, Pascal Schweitzer, Erik Jan Van Leeuwen, Kurt Mehlhorn, and Karsten M Borgwardt. Weisfeiler-lehman graph kernels. *Journal of Machine Learning Research*, 12(9), 2011.

Anton Tsitsulin, Davide Mottin, Panagiotis Karras, Alexander Bronstein, and Emmanuel Müller. Netlsd: hearing the shape of a graph. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 2347–2356, 2018.

Rex Ying, Jiaxuan You, Christopher Morris, Xiang Ren, William L Hamilton, and Jure Leskovec. Hierarchical graph representation learning with differentiable pooling. *arXiv preprint arXiv:1806.08804*, 2018.

Yunhao Zhang, Haowen Liu, Paul Rosen, and Mustafa Hajij. Mesh learning using persistent homology on the laplacian eigenfunctions. *arXiv preprint arXiv:1904.09639*, 2019.

6 Appendix

In this section we briefly review various geometric message passing schemes on a general oriented and non-oriented cell complexes suggested in Hajij et al. (2020a).

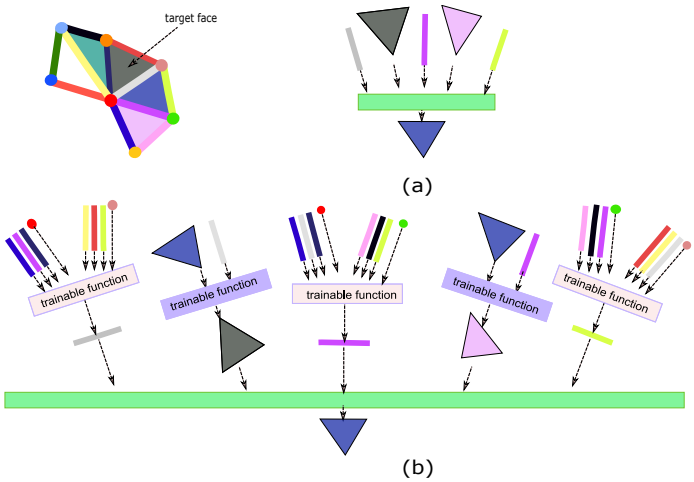

Figure 2: Coadjacency message passing scheme. The above illustrates the computation of a two layer cell complex network with respect the specified target face.

6.1 Geometric Message Passing Schemes (GMPS)

In Hajij et al. (2020a) three general message passing schemes on cell complexes were suggested. These schemes are trivially applicable in our case here on simplicial complexes. In order to implement a specific neural network of a complex, here a simplicial complex autoencoder, one must select a particular message passing scheme or a combination of them. This consequently affects the final representation of the simplicial complex. In Section 3 we mentioned one possible geometric message passing scheme which is AMPS. In this section we review briefly these *geometric message passing schemes* (GMPS) on a general cell complex net (See Figure 4 for an illustration of the flow of data computations with these schemes). Note that the simplicial complex autoencoder given in 4.1 assumes AMPS. Other simplicial complex autoencoders can be defined similarly using the message passing schemes that we shall present.

6.1.1 Adjacency Message Passing Scheme (AMPS)

These are the same message passing schemes we review in Section 3. The reader is referred to that section for more details.

6.1.2 Co-adjacency Message Passing Scheme (CMPS)

Co-adjacency message passing scheme is very similar to the AMPS we mentioned in Section 3. The only difference is that it utilises the co-adjacency relations of a given face instead of the adjacency matrix. Specifically, let $H_m^{(0)} \in \mathbb{R}^{|X^m| \times d_0}$ be initial cell feature vector on every m-cell in X. Let $L > 0$ be the desired depth of the CXN one wants to define on a complex X, the *Coadjacency Message Passing Scheme* (CMPS) on X consists of $L \times n$ and it is defined by :

$$H_{n-m}^{(k)} := M(A_{co}, , H_{n-m}^{(k-1)}, H_{n-m-1}^{(k-1)}; \theta_{n-m}^{(k)}) \tag{5}$$

where $0 \leq m \leq n-1$, $1 \leq k \leq L$, $H_{n-m}^{(k)} \in \mathbb{R}^{|X^{n-m}| \times d_k}$ are the embeddings computed after k steps of applying 1, and $\theta_{n-m}^{(k)}$ is a trainable weight vector at the layer k, M is the message propagation function that depends on the weights $\theta_{n-m}^{(k)}$, the cell embeddings $H_{n-m}^{(k)}$ and the adjacency matrix of X. See Figure 2 for an illustration of such message passing scheme on a simplicial complex.

Note that with CMPS, the flow of information goes from higher cells to lower ones. This explains the strange index choice in equation (5). Moreover, note that the feature vectors associated with the zero-cells are never updated.

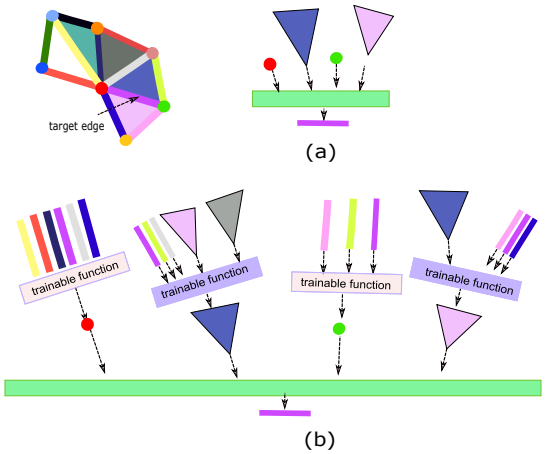

Figure 3: Homology and cohomology message passing scheme. The figure illustrates the computation of a two layers cell complex network with respect the specified edge.

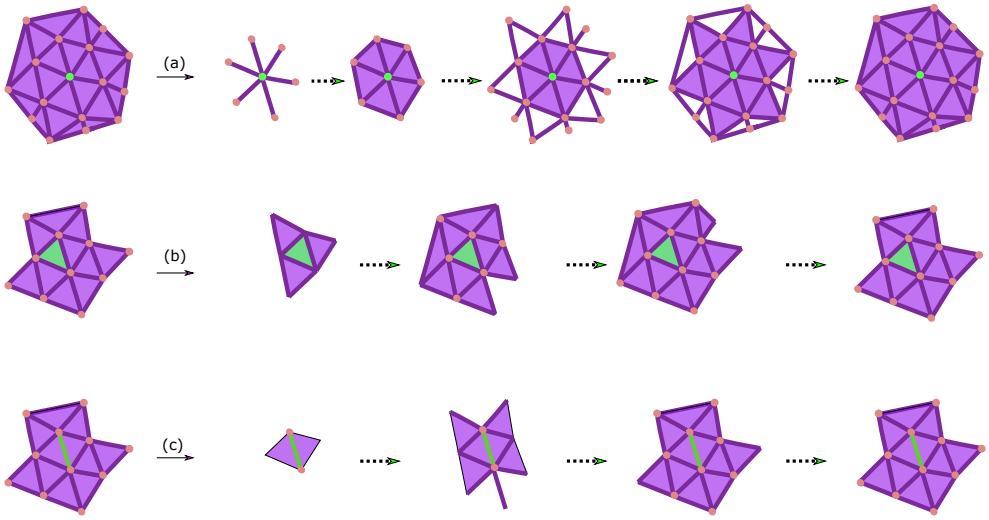

Figure 4: Illustration of the flow of information with various geometric message passing schemes (GMPS). (a) Flow of information with adjacency message passing schemes (AMPS). (b) Flow of information with co-adjacency message passing scheme (CMPS). (c) Flow of information with homology and cohomology message passing scheme (HCMPS). In all examples, the computations flow is given from the green simplex.

6.1.3 Homology and Cohomology Message Passing Scheme (HCMPS)

For this final message passing scheme it is more convenient to adapt a non-matrix notation.

Let c_m be a cell in a, possible oriented, cell complex X. Denote by $Bd(x)$ to the set of cells y of dimension $k-1$ such that $y \in \partial(x)$, the boundary of x, such that x and y have compatible orientations. In the same manner, $CoBd(x)$ denotes all cells of $y \in X$ with $h \in \partial(y)$ such that x and y have compatible orientations. Let $\mathcal{I}(x)$ be $Bd(x) \cup CoBd(x)$, the *Homology and Cohomolgy Message Passing Scheme* (HCMPS) is given by :

$$h_{c^m}^{(k)} := \alpha_m^{(k)} \left(h_{c^m}^{(k-1)}, E_{a \in \mathcal{I}(x)} \left(\phi_{m,d(a)}^{(k)} (h_{c^m}^{(k-1)}, h_a^{(k-1)}) \right) \right) \in \mathbb{R}^{l_m^k} \tag{6}$$

where $h_{c^m}^{(k)} \in \mathbb{R}^{l_m^k}$, E is permutation invariant differentiable functions, and $\alpha_m^{(k)}, \phi_m^{(k)}$ are trainable differentiable functions. An example of applying the HCMPS is given in Figure 3.

In is worths mentioning that with AMPS, and CMPS and a variant of HCMPS [2] were utilized recently by (Bodnar et al. (2021)) to study the expressive power of graph neural networks. In particular, it was shown that AMPS, the variant of HCMPS when combined with a simplicial complex-based Weisfeiler-Lehman colouring procedure (Bodnar et al. (2021)) one obtains a test that is strictly more powerful than the Weisfeiler-Lehman test (Shervashidze et al. (2011)).

[2]The variant separates the boundary and the coboundary in our HCMPS to two separate message passing schemes.