HIGH SKIP NETWORKS: A HIGHER ORDER GENERALIZATION OF SKIP CONNECTIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

We present High Skip Networks (HSNs), a higher order generalization of skip connection neural networks to simplicial complexes. HSNs exploit higher order structure encoded in a simplicial domain by creating multiple feed-forward paths of signals computed over the input complex. Some feed-forward paths may propagate the signal through various higher order structures. For example, if we want to propagate signals over edges, some feed-forward paths may go from edges to triangles and then back to edges. Similar to Euclidean skip connection networks, all paths are combined together at the end by addition or concatenation. We demonstrate the effectiveness of HSNs on synthetic and real datasets. Our preliminary results show that HSNs lead to a statistically significant improvement in the generalization error when compared to base models without high skip components.

1 INTRODUCTION

The past few years have witnessed a growing interest in learning data on topological spaces modeled as manifolds (Shuman et al., 2016), simplicial complex (Ebli et al., 2020), cell complexes (Hajij et al., 2020) and hypergraphs (Feng et al., 2019). This interest is driven by a simultaneous success of old and emerging tools including topological data analysis (Edelsbrunner & Harer, 2010) and topological signal processing (Roddenberry et al., 2022). Much of the interest, however, is motivated by the success of graph neural networks (GNN)-based models (Zhou et al., 2020) at tackling a wide range of problems including molecular design (Künzle et al., 2019), predicting protein structure (Jing & Xu, 2021), relational reasoning (Zhang et al., 2020; Schlichtkrull et al., 2018) and physics (Battiston et al., 2021). Although GNN-based models achieved promising performance, these models can only model pairwise relationships, which limits its use in applications that require *multiway relationships* or higher order relationship abstractions (Bick et al., 2021). Examples of these applications include higher-order interactions in physical complex systems (Battiston et al., 2021) and higher-order protein interactions (Xiao & Deng, 2020).

The earlier works for learning on topological spaces focused on generalizing the graph convolutional neural networks to process signals on graph edges (Roddenberry & Segarra, 2019; Schaub et al., 2020). Around the same time, hypergraph convolutional operators were introduced in Jiang et al. (2019); Feng et al. (2019). Learning on higher order complexes was introduced in Ebli et al. (2020); Hajij et al. (2020), extending graph neural networks Fey & Lenssen (2019); Battaglia et al. (2018); Morris et al. (2019); Kipf & Welling (2016a). Recently, several works (e.g., Roddenberry et al. (2021); Bodnar et al. (2021); Bick et al. (2021); Schaub et al. (2021; 2020)) are proposed for nonlinear higher order signal processing.

This work is the first that proposes a higher order generalization of skip connection neural networks (He et al., 2016; Ronneberger et al., 2015) with simplicial complexes, and introduces *High Skip Networks* (HSNs). Skip connection layers are arguably one of the most successful layers that allowed the training of very deep neural networks. The proposed HSNs generalize skip connection networks because of their ability to model higher order relationships. The immediate advantage of HSNs over regular networks defined on these domains (e.g., Roddenberry et al. (2022); Ebli et al. (2020)) is the same that of its Euclidean analogues: training deeper neural networks. HSNs also hold other unique characteristics. For example, HSNs exploit higher order structure information encoded in the domain to construct a final output. In particular, HSNs construct multiple feed-forward paths of signals computed over the input complex. Some of these paths may propagate the signal via higher

order structures thereby exploiting this structural information. For example, if we want to propagate signals over edges, some feed-forward paths may go from edges to triangles and then back to edges. These feed-forward paths are combined together at the end by addition or concatenation similar to their Euclidean analogues (He et al., 2016).

2 BACKGROUND: ABSTRACT SIMPLICIAL COMPLEXES

This section provides the basic notations and definitions used in the paper. Further details on these complexes can be found in any classical textbook on algebraic topology (e.g., Hatcher (2005)).

Let $\mathcal{V} = \{v_1, ..., v_n\}$ be a set of vertices. A set of $k+1$ distinct elements of $\mathcal{V}$ is called a $k-$ *simplex* on $\mathcal{V}$. In particular, the 0-simplices are the singletons of the set $\mathcal{V}$ and are called vertices, 1-simplices are called edges, and 2-simplices are called faces or triangles. An *abstract simplicial complex* on $\mathcal{V}$ is a finite collection of simplices X from $\mathcal{V}$ such that $\sigma \in X$ and $\beta \subset \sigma$ implies $\beta \in X$. The dimension of X is the dimension of the highest simplex in X. For any $0 \leq k \leq n$, we denote by X^k the set of all k-simplices. By setting $|X^k| = N^k$, we give each simplex in X an orientation, which is the choice of ordering on the vertices of the simplex. Two orientations of the same simplex are equivalent if they are related via an even permutation.

We use $\mathcal{C}_k(X)$ to denote the $\mathbf{R}$-vector space of all linear combinations of all oriented k-simplices in X. Element of $\mathcal{C}_k(X)$ are typically called *k-chains*, in particular, k-simplices in X are called *primitive k-chains*. We make the convention that an orientation change of a primitive k-chain to correspond to a change in its sign; for instance, $[v_0, v_1] = -[v_1, v_0]$. Finally, we give the simplices in X^k an ordering and identify the jth element in this ordered set with j^{th} canonical basis element in $\mathbf{R}^{N_k}$. This identification yields a vector space isomophism between $\mathcal{C}_k(X)$ and $\mathbf{R}^{N_k}$, which we will assume for the rest of the paper. The space of *k-cochains* is the dual space of k-chains' space and will be denoted by $C^k(X) := C_k^*(X)$. Being finite dimensional, these two spaces are isomorphic. A k-cochain element can be interpreted as a real signal defined on the k-simplices of X (Grady & Polimeni, 2010). In particular, $0-$cochains correspond to the graph signals (Ortega et al., 2018). The *boundary operator* $\partial_k : \mathcal{C}_k(X) \to \mathcal{C}_{k-1}(X)$ is defined on the primitive k-chains via: $\partial_k[v_0, v_1, \cdots, v_k] = \sum_{i=1}^{k}(-1)^i[v_0, \cdots, \hat{v}_i, \cdots, v_k]$, where $\hat{v}_i$ indicates that v_i is missing from the sequence that determines the simplex. For an abstract simplicial complex of dimension n, the boundary operators $\{\partial_k\}_{k=1}^{n}$ completely determine the structure of X. Additionally, these operators satisfy $\partial_{k-1} \circ \partial_k = 0$. Since the complex X is finite, the boundary operator ∂_k can be represented by a matrix $\mathbf{B}_k$ and the typical bases used to construct this matrix are the primitive chains of X^k and X^{k-1} with a fixed order on these chains. Intuitively, this matrix describes the incidence between the primitive chains of X^k and X^{k-1} and is referred to as the kth *incidence matrix* of the complex X. Dually, the coboundary maps are defined as the transpose ∂_k^T of the boundary maps. The (co)boundary can be used to define *the combinatorial k-Hodge Laplacian* $L_k = \partial_k^\top \partial_k + \partial_{k+1}\partial_{k+1}^\top$. Similarly, we can define the *higher-order adjacency matrix* $A_k : C^k(X) \to C^k(X)$ that describes the way k-primitive chains are adjacent to each other. In particular A_0 is the regular graph adjacency matrix.

3 PROPOSED HIGH SKIP NETWORKS (HSNS)

HSNs are inspired by categorical notions. In category theory, the term *factors through* refers to a composition of morphisms. If we are given three objects A, B, and C and a map $f : A \to B$, which can be written as a composition $f = h \circ g$ with $g : A \to B$ and $h : B \to C$, then we say that f factors through B. The notion can be intuitively described via the following commutative diagram:

$$\begin{tikzcd} & B \arrow[dr, "h"] & \\ A \arrow[ur, "g"] \arrow[rr, "f"] & & C \end{tikzcd}$$

Now let $\mathbf{s}^i$ be in i-cochain and suppose that act on $\mathbf{s}^i$ via two neural networks $F, K : \mathcal{C}^i(X) \to \mathcal{C}^j(X)$ such that $K = h \circ g$ with $g : \mathcal{C}^i(X) \to \mathcal{C}^k(X)$, and $h : \mathcal{C}^k(X) \to \mathcal{C}^j(X)$.

$$\begin{tikzcd} & \mathcal{C}^k(X) \arrow[dr, "(0,h)"] & & \\ \mathcal{C}^i(X) \arrow[ur, "g"] \arrow[rr, "(F,0)"] & & \mathcal{C}^j(X) \bigoplus \mathcal{C}^j(X) \arrow[r, "+"] & C^j(X) \end{tikzcd}$$

This diagram represents a flow of a cochain $\mathbf{s}^i$ through two different feed-forward paths, one of which "factors through" $\mathcal{C}^k(X)$ in the sense that it gets mapped to a cochain supported on the k-simplices via the map g before landing in $\mathcal{C}^j(X)$[1]. We call the above setup where some feed-forward paths go via higher order simplices as the proposed high skip network. Typically, the final j- cochains $K(\mathbf{s}^i)$, $F(\mathbf{s}^i)$ are combined together to form a single j-cochain $K(\mathbf{s}^i) + F(\mathbf{s}^i)$. The networks F, g, and h can be chosen as vanilla convolutional networks (Bunch et al., 2020; Ebli et al., 2020) or more general higher order message passing networks (Hajij et al., 2020). The following diagram illustrates how this setting generalizes the classical skip connection setup:

$$\begin{tikzcd} & \mathbf{R}^n \arrow[dr, "(0,\mathrm{Id})"] & & \\ \mathbf{R}^n \arrow[ur, "\mathrm{Id}"] \arrow[rr, "(F,0)"] & & \mathbf{R}^n \bigoplus \mathbf{R}^n \arrow[r, "+"] & \mathbf{R}^n \end{tikzcd}$$

In fact, HSNs can also be viewed as a non-identity skip connection (Orhan & Pitkow, 2017) with the difference of having that skip connection across multiple dimensions in the complex. The work by Orhan & Pitkow (2017) demonstrated that non-identity skip connections lead to training improvements. This is also demonstrated in our results in the next section.

4 Experiments & Results

We evaluated HSNs using synthetic and real dataset. We performed three experiments for three tasks: link prediction, node classification, and vector field decomposition. Note that our aim is not to outperform the state-of-the-art methods. Instead, we aim to demonstrate that our method leads to a statistically significant improvement in the generalization error when compared to a base-model with a similar number of weights and network structure, except for the high skip part.

Link Prediction. This experiment evaluates the performance of HSN models for link prediction using two citation network datasets: Cora and Citeseer (Sen et al., 2008). For data setup, we removed a portion of the citation edges while keeping the features of the nodes. The validation and tests sets are obtained from the removed links and the same number of randomly sampled non-edges.

To test the proposed method, we created two graph variational autoencoder (GVAE) models (Kipf & Welling, 2016b). The first one is essentially the same model proposed in Kipf & Welling (2016b), which consists of encoders for the mean and the standard deviation of the latent space. Specifically, the encoder consists of two graph neural network : $\mu = \mathrm{GCN}_\mu(s_0, A_0)$ and $\log \sigma = \mathrm{GCN}_\sigma(X, A_0)$. Here s_0 is the input 0-cochain[2], $\mathrm{GCN}(s_0, A_0) = \tilde{A}_0 \phi(\tilde{A}_0 s_0 \mathbf{W}_0)\mathbf{W}_1$ with $\tilde{A}_0 = D_0^{-1/2} A_0 D_0^{-1/2}$ - $\mathbf{W}_i$ are the trainable parameters, and ϕ is the Relu activation function. The decoder has the form $P(A_0|Z) = \prod_{i=1}^{\mathbf{N}_0} \prod_{j=1}^{\mathbf{N}_0} P(A_{ij}|z_i, z_j)$ with $P(A_{ij} = 1|z_i, z_j) = \sigma(z_i^T z_j)$. On the other hand, the HSN model is simply defined as $\mathrm{HSN}(s_0, A_0, \partial_1) = \tilde{A}_0 \phi(\tilde{A}_0 s_0 \mathbf{W}_0)\mathbf{W}_1 + \partial_1 \phi(\partial_1^T s_0 \mathbf{W}_0)\mathbf{W}_1$, and is used to replace the encoder models of the mean and standard deviations above. The decoder in this second model is defined as before. Note that our HSN model factors through the edges, and it has the same number of parameters as the GVAE base model. Further, note that the difference between the base model and the proposed HSN model is the high skip part $\partial_1 \phi(\partial_1^T s_0 \mathbf{W}_0)\mathbf{W}_1$, which might appear as redundant information, but actually contributes to performance improvement. The performance of link prediction using the proposed HSN is reported in Table 1 for the Cora and Citeseer datasets. We reported the results using the area under the ROC curve (AUC) and the average

[1] Note that this diagram is not commutative.

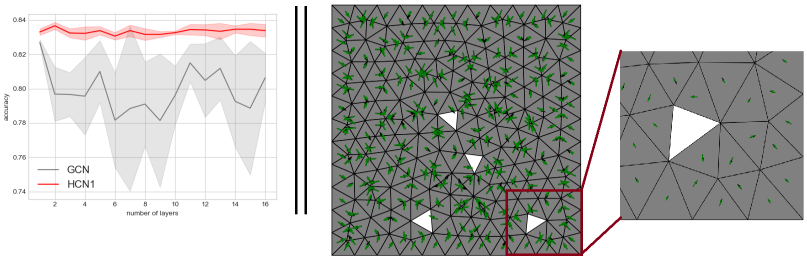
[2] Recall that that 0- cochains on graphs correspond to signals or features defined on the nodes of that graph as we discussed in Section 2.

Table 1: Link Prediction on the Corda and the Citeseer datasets. Bold values indicate superiority.

Method	AUC (Cora)	AP (Cora)	AUC (Citeseer)	AP (Citeseer)
GVAE	0.91183 ± 0.0558	0.91538 ± 00481	0.89558 ± 0.01316	0.91241 ± 0.01092
HSN (ours)	**0.93073** $\pm$ **0.00273**	**0.93650** $\pm$ **0.00489**	**0.91307** $\pm$ **0.00984**	**0.92334** $\pm$ **0.00612**

precision (AP) on the test dataset. The reported results are the mean and standard deviation for 10 independent runs with random initializations on the same dataset splits.

Node Classification. We performed semi-supervised node classification to categorize the nodes in a graph given the class labels of a few nodes; i.e., given a small number of labeled data from the Cora dataset. We compared the performance of node classification obtained using a deep HSN model against a base skip connection GCNs. The base skip connection is given by $\mathrm{GCN}(\mathbf{s}_0^0, A_0) = \tilde{A}_0(\tilde{A}_0 s_0^0 \mathbf{W}_0 + \tilde{A}_0\, \mathrm{GCN}^d(\mathbf{s}_0^d, A_0)\mathbf{W})\mathbf{W}$ where $\mathrm{GCN}_i(\mathbf{s}_0^i, A_0) = \phi(\tilde{A}_0(\phi(\tilde{A}_0 \mathbf{s}_0^{i-1}\mathbf{W}_i)\mathbf{W}_i') + \mathbf{s}_0^{i-1})$, where d is the depth of the network. The HSN model is given via: $\mathrm{HSN}_1(\mathbf{s}_0^0, A_0, \partial_1) = \tilde{A}_0(\tilde{A}_0 s_0^0 \mathbf{W}_0 + \partial_1(\mathrm{HSN}_e^d(\mathbf{s}_1^d, \partial_1) + \mathbf{s}_1^d)\mathbf{W})\mathbf{W}$ where $\mathrm{HSN}_e^0(\mathbf{s}_1^i, A_1) = \phi(\tilde{A}_1(\phi(\tilde{A}_1 \mathbf{s}_1^{i-1}\mathbf{W}_i)\mathbf{W}_i') + \mathbf{s}_1^{i-1})$, and $\mathbf{s}_1^0 := \partial_1^T \mathbf{s}_0^0 \mathbf{W}_0$. Note that the networks are essentially similar, and they only differ in the structural matrices: A_0, A_1 and ∂_1, which determine the way the higher skip connection feed-forward paths are defined. Figure 1.a presents the results of 10 independent runs, for each depth, with random initializations on the same dataset splits. We observe that the performance of HSN does not decay as the number of layers increases whereas the base model performs worse than the proposed HSN as we increase the number of layers.



(a) (b) (c)

Figure 1: (a) A comparison of two networks, baseline skip connection GNN and proposed HCN. (b) and (c) represent the plot of the 1-cochain $\partial_1^T \alpha$ both the original edges signal (black arrows) on the simplicial complex and the reconstructed ones inferred from the network (green arrows). The black and green arrows overlap demonstrates good performance; 1-cochain is visualized as a vector field by interpolating it from the edge values using Whitney interpolation (Bell & Hirani, 2012).

Vector Field Decomposition. Any k-cochain can be uniquely decomposed into gradient, curl, and harmonic parts. This decomposition is called the Hodge decomposition. Namely, for any k-cochain s, there are unique cochains $\alpha \in C^{k-1}(X)$ and $\beta \in C^{k+1}(X)$ such that $s = \partial_k^T \alpha + \partial_{k+1}\beta + h$. This idea has been exploited in a neural network architecture proposed by Roddenberry et al. (2021). Here, we used a similar HSN architecture to factor an edge signal through the nodes and the faces of a 2d-simplicial complex. Specifically, we generated a random 1-cochain on a 2d-simplicial complex with 332 faces and 131 nodes (See Figure 1.b). We pre-computed the Hodge decomposition of this cochain. Then, we used this decomposition to create the dataset pairs with the input being the random 1-cochain s and the output being the projections $\partial_1^T \alpha$ and $\partial_2 \beta$. We tested the ability of HSN models on the regression problem of predicting the projections $\partial_1^T \alpha$ and $\partial_2 \beta$ given the input. To this end, we treated this problem as the link prediction problem, and masked certain edges of the random 1-cochain. We then trained the following HSN network on the remaining parts of the signal:

$$\mathrm{HSN}(\mathbf{s}_1, A_1, \partial_1, \partial_2) = \partial_1^T(\phi \partial_1 \mathbf{s}_1 \mathbf{W}_1)\mathbf{W}_2 + \tilde{A}_1(\phi \tilde{A}_1 \mathbf{s}_1^l \mathbf{W}_3)\mathbf{W}_4 + \partial_2(\phi \partial_2^T \mathbf{s}_1 \mathbf{W}_5)\mathbf{W}_6 \quad (1)$$

where $\mathbf{W}_i$ represents parameter matrices. Recall that this network factors through the nodes and edges. Figure 1.b and Figure 1.c show the inference of the trained network on the entire 1-cochain. The results in the figure show that the original (black arrows) and the reconstructed (green arrows) signals follow each other closely.

These preliminary results for the three tasks are promising. They demonstrate the ability of the proposed method to improve performance and generalization error as compared to the base models.

REFERENCES

Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018.

Federico Battiston, Enrico Amico, Alain Barrat, Ginestra Bianconi, Guilherme Ferraz de Arruda, Benedetta Franceschiello, Iacopo Iacopini, Sonia Kéfi, Vito Latora, Yamir Moreno, et al. The physics of higher-order interactions in complex systems. *Nature Physics*, 17(10):1093–1098, 2021.

Nathan Bell and Anil N Hirani. Pydec: software and algorithms for discretization of exterior calculus. *ACM Transactions on Mathematical Software (TOMS)*, 39(1):1–41, 2012.

Christian Bick, Elizabeth Gross, Heather A Harrington, and Michael T Schaub. What are higher-order networks? *arXiv preprint arXiv:2104.11329*, 2021.

Cristian Bodnar, Fabrizio Frasca, Yuguang Wang, Nina Otter, Guido F Montufar, Pietro Lio, and Michael Bronstein. Weisfeiler and lehman go topological: Message passing simplicial networks. In *International Conference on Machine Learning*, pp. 1026–1037. PMLR, 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.

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

Herbert Edelsbrunner and John Harer. *Computational topology: an introduction*. American Mathematical Soc., 2010.

Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. Hypergraph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 3558–3565, 2019.

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

Leo J Grady and Jonathan R Polimeni. *Discrete calculus: Applied analysis on graphs for computational science*, volume 3. Springer, 2010.

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

Allen Hatcher. *Algebraic topology*. 2005.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Jianwen Jiang, Yuxuan Wei, Yifan Feng, Jingxuan Cao, and Yue Gao. Dynamic hypergraph neural networks. In *IJCAI*, pp. 2635–2641, 2019.

Xiaoyang Jing and Jinbo Xu. Fast and effective protein model refinement using deep graph neural networks. *Nature Computational Science*, 1(7):462–469, 2021.

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

Thomas N Kipf and Max Welling. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*, 2016b.

Matthias Künzle, Marcel Lach, Made Budiarta, and Tobias Beck. Higher-order structures based on molecular interactions for the formation of natural and artificial biomaterials. *ChemBioChem*, 20 (13):1637–1641, 2019.

Christopher Morris, Martin Ritzert, Matthias Fey, William L Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. Weisfeiler and leman go neural: Higher-order graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 4602–4609, 2019.

A Emin Orhan and Xaq Pitkow. Skip connections eliminate singularities. *arXiv preprint arXiv:1701.09175*, 2017.

Antonio Ortega, Pascal Frossard, Jelena Kovačević, José MF Moura, and Pierre Vandergheynst. Graph signal processing: Overview, challenges, and applications. *Proceedings of the IEEE*, 106 (5):808–828, 2018.

T Mitchell Roddenberry and Santiago Segarra. Hodgenet: Graph neural networks for edge data. In *2019 53rd Asilomar Conference on Signals, Systems, and Computers*, pp. 220–224. IEEE, 2019.

T Mitchell Roddenberry, Nicholas Glaze, and Santiago Segarra. Principled simplicial neural networks for trajectory prediction. In *International Conference on Machine Learning*, pp. 9020–9029. PMLR, 2021.

T Mitchell Roddenberry, Michael T Schaub, and Mustafa Hajij. Signal processing on cell complexes. *IEE ICASSP*, 2022.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241. Springer, 2015.

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.

Michael T Schaub, Yu Zhu, Jean-Baptiste Seby, T Mitchell Roddenberry, and Santiago Segarra. Signal processing on higher-order networks: Livin'on the edge... and beyond. *Signal Processing*, 187:108149, 2021.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *European Semantic Web Conference*, pp. 593–607. Springer, 2018.

Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI magazine*, 29(3):93–93, 2008.

David I Shuman, Benjamin Ricaud, and Pierre Vandergheynst. Vertex-frequency analysis on graphs. *Applied and Computational Harmonic Analysis*, 40(2):260–291, 2016.

Ze Xiao and Yue Deng. Graph embedding-based novel protein interaction prediction via higher-order graph convolutional network. *PloS one*, 15(9):e0238915, 2020.

Shi-Xue Zhang, Xiaobin Zhu, Jie-Bo Hou, Chang Liu, Chun Yang, Hongfa Wang, and Xu-Cheng Yin. Deep relational reasoning graph network for arbitrary shape text detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9699–9708, 2020.

Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. Graph neural networks: A review of methods and applications. *AI Open*, 1:57–81, 2020.