A Simple Hypergraph Kernel Convolution based on Discounted Markov Diffusion Process

Anonymous Author(s)
Affiliation
Address
email

Abstract

Kernels on discrete structures evaluate pairwise similarities between objects which capture semantics and inherent topology information. Existing kernels on discrete structures are only developed by topology information(such as adjacency matrix of graphs), without considering original attributes of objects. This paper proposes a two-phase paradigm to aggregate comprehensive information on discrete structures leading to a Discount Markov Diffusion Learnable Kernel (DMDLK). Specifically, based on the underlying projection of DMDLK, we design a Simple Hypergraph Kernel Convolution (SHKC) for hidden representation of vertices. SHKC can adjust diffusion steps rather than stacking convolution layers to aggregate information from long-range neighborhoods which prevents oversmoothing issues of existing hypergraph convolutions. Moreover, we utilize the uniform stability bound theorem in transductive learning to analyze critical factors for the effectiveness and generalization ability of SHKC from a theoretical perspective. The experimental results on several benchmark datasets for node classification tasks verified the superior performance of SHKC over state-of-the-art methods.

1 Introduction

In real-world applications, original data with discrete structures is relatively prevalent. For instance, graphs are commonly used in citation networks, social networks, and protein interaction networks to represent pairwise topology relationships between discrete objects. In the community of Graph Neural Networks (GNNs) (Kipf & Welling, 2017a; Chen et al., 2021; Zhang et al., 2022c) and Hypergraph Neural Networks (Feng et al., 2019; Yadati et al., 2019), researchers focused on designing convolution operators to obtain effective hidden representations for downstream tasks. It is worth addressing that the spectral view and message passing view play important roles in inspiring the design of those convolutions. In another perspective, we start from abstracting a convolution to a projection ϕ. Most of the existing convolution networks for the discrete structure like graphs and hypergraphs are essentially derived from some heuristic intuitions and rational principles aiming to obtain an effective representation extraction projection ϕ. This hidden projection maps objects from original space $\mathbf{\Omega}$ to hidden representations in Euclidean space.

In some sense, choosing a kernel on $\mathbf{\Omega} \times \mathbf{\Omega}$ is the same as choosing a feature extraction projection for objects in $\mathbf{\Omega}$ (Haussler, 1999). From this perspective, we need to generate a symmetrical and positive semi-definite kernel matrix on $\mathbf{\Omega} \times \mathbf{\Omega}$ (Kondor & Lafferty, 2002) to obtain a rational projection. However, earlier researches pay attention to generating kernels on graphs directly from the prior topology(i.e. adjacency matrix) without considering the inherent original attributes of vertices. We propose a two-phase paradigm to conduct Discount Markov Diffusion Learnable Kernels(DMDLK) for discrete structures. The respective roles of the two phases are as follows: (i) aggregate topology structure information with original attributes; (ii) aggregate information among channels by learnable parameters.

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

If the topology of the original data can be determined a priori, for instance, given the adjacency matrix for a graph, the two-phases paradigm can be naturally adapted to conduct convolution operators for any discrete structure. However, there are flaws to model discrete data by graphs when original objects inherently meet complex topology structures. Studies have continued to concentrate on utilizing hypergraphs to represent complicated topology structures (Feng et al., 2019; Zhou et al., 2006; Agarwal et al., 2006; Chitra & Raphael, 2019; Zhang et al., 2022b,a). In this paper, we pay attention to hypergraphs which serve as more general discrete structures than simple graphs. Specifically, we modify a generalized transition matrix from Zhang et al. (2022b) to develop DMDLK for hypergraphs leading to a novel convolution operator for hypergraphs named Simple Hypergraph Kernel Convolution (SHKC).It can aggregate long-range information and obtain high-level performance by enlarging the diffusion step rather than stacking multi convolution layers. The latter always leads to the well-known oversmoothing issue. To illustrate the effectiveness of SHKC from a theoretical view, we use the tool of uniform stability theorem in transductive learning to analyze the rationality and effect of detailed factors in SHKC.

Overall, the main contributions highlight as follows: (i) We propose a two-phase paradigm to conduct DMDLK for discrete structures; (ii) By introducing a modified generalized transition matrix of hypergraph into this paradigm, we obtain Simple Hypergraph Kernel Convolution(SHKC) which can avoid stacking multi-layers to aggregated long-range information by simply adjusting the diffusion steps; (iii) We utilize uniform stability theorem to explain the promising effectiveness and generalization ability of SHKC. (iv) Empirical results on downstream node classification tasks and object recognition tasks show the advanced performance of SHKC which corresponds well to the theoretical analysis.

2 Related Works

2.1 Kernels for Discrete Structures

Kernel functions (Schölkopf et al., 2002; Shawe-Taylor et al., 2004; Fouss et al., 2012) $k : \mathbf{\Omega} \times \mathbf{\Omega} \to \mathbb{R}$ can directly compute pairwise similarities $k(\mathbf{x}, \mathbf{x}')$ by implicitly constructing a projection $\phi : \mathbf{\Omega} \to \mathcal{H}_k$ from the original space $\mathbf{\Omega}$ into a high-dimensional and more well-separated Hilbert space $\mathcal{H}_k$. Discrete structures, such as strings,graphs,and trees, are dominant in real-world applications like classification and collaborative recommendation (Fouss et al., 2012). Kernels on discrete structures focus on mining pairwise similarity between objects which captures the semantics inherent in discrete structures (Fouss et al., 2012), such as length of the shortest path and the total number of paths between vertices in a graph. Kernels on discrete structures were first proposed by Kondor & Lafferty (2002) and expanded on graphs by Smola & Kondor (2003). Fouss et al. (2006) gives a thorough overview of several kernels on graph and proposes a kernel named regularized commute-time kernel. However, those research focus on capturing similarity between vertices directly from the topology(i.e. adjacency matrix) without considering inherent attributes of vertices.

2.2 Hypergraphs Convolution Networks

Hypergraphs whose edges contain more than two vertices can be seen as more generalized discrete structures than graphs. Hypergraph have shown its promising ability to model more complex topology information than graphs (Zhou et al., 2006; Feng et al., 2019). Feng et al. (2019) firstly introduces hypergraphs into deep learning community motivated by capturing multi-modal topology information through vertex-hyperedge incident matrix. Yadati et al. (2019) introduces a non-linear Laplacian Matrix for hypergraphs deducing a more expressive convolution. Dong et al. (2020) proposes a message -passing based model with a two steps message propagation between vertex and hyperedge. Ding et al. (2020) involves attention mechanisms in hypergraphs for the task of text classification. Moreover, thanks to the intensive research of Markov process on hypergraphs in (Zhou et al., 2006; Chitra & Raphael, 2019; Carletti et al., 2021; Zhang et al., 2022b), we immediately stitch the generalized hypergraph transition matrix (Zhang et al., 2022b) with the two-phases paradigm to conduct our effective SHKC.

3 Preliminaries

Notions. In this paper, we use a boldface capital letter $\mathbf{A} \in \mathbb{R}^{N \times M}$ to denote an $N \times M$ matrix and use A_{ij} or $A(i,j)$ to denote its ij-th entry. We use the boldface letter $\mathbf{x}$ or $\vec{\theta}$ to indicate the column vector, where x_i or $\vec{\theta}_i$ is the i-th entry of $\mathbf{x}$ or $\vec{\theta}$. The word "vector" always denotes a column vector in this paper. Thus, we use the transposition of the column vector $\mathbf{x}^\top$ or $(\vec{\theta})^\top$ to denote the row vector. For vectors with subscript like $\mathbf{x}_k$ and $\vec{\theta}_k$ mean k-th column of the matrix where x_{ki} and $\vec{\theta}_{ki}$ also mean their i-th component. Let $\mathbf{I} \in \mathbb{R}^{N \times N}$ denote the identity matrix, and $\mathbf{e}_i$ represents the vector where the i-th component is 1 and otherwise 0.

3.1 Kernels on Discrete Structures

Given an original space $\mathbf{\Omega}$ with $|\mathbf{\Omega}| = N$ discrete objects , the kernel function k can be uniquely represented by a $N \times N$ matrix called Kernel Matrix. The Matrix should generally be symmetrical and positive semi-definite (Haussler, 1999). A finite vertex set $\mathcal{V} = \{\mathbf{v}_1, \cdots, \mathbf{v}_N\}$ denotes the original space $\mathbf{\Omega}$ and an implicit $\phi(\mathbf{v}_i)$ maps the vertex $\mathbf{v}_i$ to a Hilbert space $\mathcal{H}_k$. In fact, $\phi(\mathbf{v}_i)$ denotes a hidden representation for $\mathbf{v}_i$ in $\mathcal{H}_k$. Thus, We call $\mathcal{H}_k$ "hidden representation space" according to its specific meaning. Give original attributes $\mathbf{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_N\}^\top \in \mathbb{R}^{N \times d}$ whose i-th row $\mathbf{x}_i^\top \in \mathbb{R}^{1 \times d}$ denotes a d-dimension attribute on the i-th vertex $\mathbf{v}_i$. Define $\mathbf{K}$ as the kernel matrix where $K_{ij} = k(\mathbf{v}_i, \mathbf{v}_j)$ is the pairwise similarity through a kernel function k.

An Instinctive Kernel on Discrete Structures. An effective kernel should capture semantics information inside the discrete structure induced by ϕ, which cannot be trivially represented by the original attributes as we discussed below. A most instinctive idea for the design of the kernel is that define $\mathbf{K}_{ori} = \mathbf{X}\mathbf{X}^\top$ directly from the original attributes where $K_{ori}(i,j) = k(\mathbf{v}_i, \mathbf{v}_j) = \mathbf{x}_i^\top \mathbf{x}_j$. Here the underlying ϕ from $\mathbf{v}_i$ to its hidden representation denotes as $\phi(\mathbf{v}_i) = \mathbf{x}_i = \mathbf{X}^\top \mathbf{e}_i$. Obviously, ϕ only captures original attributes without neither structure information nor information aggregated among different channels of original attributes. It is significant to address that there is a distance in the hidden representation space associated with the kernel: $(d(\mathbf{v}_i, \mathbf{v}_j))^2 = \|\phi(\mathbf{v}_i) - \phi(\mathbf{v}_j)\|_2^2 = (\mathbf{e}_i - \mathbf{e}_j)^\top \mathbf{K}_{ori}(\mathbf{e}_i - \mathbf{e}_j)$. This indicates meaningful distances defined on hidden representation space can be associated with effective kernels. There has been various kernels on graphs aiming at extracting semantics information inside the graphs (Fouss et al., 2012). However, these kernels are developed merely from adjacency matrix of graphs without considering information from original attributes of vertices which means those kernels can not be trivially used in GNNs. From this perspective, we propose a two-phase paradigm for effective kernels and its induced convolutional operators.

4 Two Phases for Effective Kernels on Discrete Structure

4.1 Phase I: Topology Information Aggregation with Original Attributes

In the first phase, we mainly consider effective approaches to excavate structure information from the underlying topology of discrete structures. We start from the perspective of defining a specific Markov Diffusion Process on discrete structures. Note that this idea is directly suitable for hypergraphs as the random walk on hypergraphs has been intensively researched in (Chitra & Raphael, 2019; Carletti et al., 2021; Zhang et al., 2022b; Chen & Zhang, 2022).

Discounted Average Visiting Rate. For a Markov process on a finite state space $\{s_i, \cdots, s_N\}$, we denote $\mathbf{T}$ as the probability transition matrix where $T_{ik} = \Pr\{s(t+1) = s_k | s(t) = s_i\}$. Note that a τ-step transition from state i to k can be formulated as $\Pr\{s(t+\tau) = s_k | s(t) = s_i\} = (\mathbf{T}^\tau)_{ik}$. We define the *discounted average visiting rate* as:

$$\bar{v}_{ik}(t) = \frac{1}{t} \sum_{\tau=1}^{t} \alpha^\tau \Pr\{s(\tau) = s_k | s(0) = s_i\}, t \in \mathbb{Z}^{++} \tag{1}$$

The discounted average visiting rate is considered as the probability of a random walker starting from s_i standing in s_k during t steps diffusion. It is modified by introducing a scalar discounted factor from the average visiting rate (Fouss et al., 2006) to alleviate long-range information aggregation

which is motivated by a thought that more global information aggregation makes undistinguished among vertices.

Discounted Markov Diffusion Process. Recall that we have the original attributes $\mathbf{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_N\}^\top$ where $\mathbf{x}_i \in \mathbb{R}^d$ is a d-dimension vector which can be seen as d channels signal on $\mathbf{v}_i$. We define a t-step process of vertex signal passing among the discrete structure named the *discounted Markov diffusion process with original attributes* which follows the formulation as:

$$\mathbf{y}_k^{(t)} = \beta \sum_{i=1}^{N} \bar{v}_{ik}(t)\mathbf{x}_i + (1-\beta)\mathbf{x}_k \tag{2}$$

This equation depicts a process where original attributes are aggregated to the vertex $\mathbf{v}_k$ from all other vertices during a t-step diffusion. The diffusion process captures the topology information underlying the discrete structure through the associated transition matrix. We call this phase as *topology information aggregation with original attributes*. Then we define the *t-step Discounted Markov Diffusion distance with original attributes* $d_M^{(t)}$ between $\mathbf{v}_i, \mathbf{v}_j$ as:

$$\begin{aligned} d_M^{(t)}(\mathbf{v}_i, \mathbf{v}_j) &= \|\mathbf{y}_i^{(t)} - \mathbf{y}_j^{(t)}\|_2 = \|\mathbf{X}^\top(\beta\mathbf{Z}(t) + (1-\beta)\mathbf{I})(\mathbf{e}_i - \mathbf{e}_j)\|_2 \\ &= [(\mathbf{e}_i - \mathbf{e}_j)^\top \mathbf{K}_M(t)(\mathbf{e}_i - \mathbf{e}_j)]^{\frac{1}{2}} \end{aligned}$$

where $\mathbf{K}_M(t) = (\beta\mathbf{Z}(t) + (1-\beta)\mathbf{I})^\top \mathbf{X}\mathbf{X}^\top(\beta\mathbf{Z}(t) + (1-\beta)\mathbf{I})$ denotes the *Discounted Markov Diffusion Kernel with original attributes* and $\mathbf{Z}(t) = \frac{1}{t}\sum_{\tau=1}^{t} \alpha^\tau \mathbf{T}^\tau$. The underlying projection behind the kernel denotes $\phi^{(t)}(\mathbf{v}_i) = \mathbf{e}_i^\top(\beta\mathbf{Z}(t) + (1-\beta)\mathbf{I})^\top\mathbf{X}$. It is easy to see that $\mathbf{K}_M(t)$ is symmetrical and positive semi-definite which means $\mathbf{K}_M(t)$ is indeed a kernel. However, the underlying projection still has flaws to represent comprehensive vertex representation. The projection failed to involve information interactions between channels of original attributes which leads to the second phase to remedy for the flaws.

4.2 Phase II: Channels Aggregation

From Eq. (2), it is obvious that the vertex representation of $\mathbf{v}_k$ derived from $\mathbf{K}_M$ only captures information within the same channel. Specifically, for channel c: $y_{kc}^{(t)} = \beta\sum_{i=1}^{N} \bar{v}_{ik}(t)x_{ic} + (1-\beta)x_{kc}$. Thus, in the second phase, we consider aggregating among channels to find a more comprehensive feature space $\mathcal{H}_k$.

Discounted Markov Diffusion Learnable Kernel (DMDLK). Define $\mathbf{\Theta} = \{\vec{\theta}_1, \cdots, \vec{\theta}_M\} \in \mathbb{R}^{d\times M}$ where $\vec{\theta}_m = \{\theta_{1m}, \cdots, \theta_{dm}\}^\top \in \mathbb{R}^{d\times 1}$ denotes the m-th set of weights concerning all channels and θ_{cm} specifically denotes the weight of the c-th channel in the aggregation. Thus we can aggregate information among channels by:

$$\tilde{y}_k^{(t)}(\vec{\theta}_m) = \sum_{c=1}^{d}\left(\sum_{i=1}^{N} \beta\bar{v}_{ik}(t)x_{ic}\right)\theta_{cm} + (1-\beta)x_{kc}\theta_{cm}$$

Here $\tilde{y}_k^{(t)}(\vec{\theta}_m)$ captures both topology information and information aggregated among channels by $\vec{\theta}_m$ for $\mathbf{v}_k$. Let $\tilde{\mathbf{y}}_k^{(t)}(\mathbf{\Theta}) = \{\tilde{y}_k^{(t)}(\vec{\theta}_1), \cdots, \tilde{y}_k^{(t)}(\vec{\theta}_M)\}^\top$. Naturally, we can define a new distance between two vertices:

$$(\tilde{d}^{(t)}(\mathbf{v}_i, \mathbf{v}_j))^2 = \|\tilde{\mathbf{y}}_i^{(t)}(\mathbf{\Theta}) - \tilde{\mathbf{y}}_j^{(t)}(\mathbf{\Theta})\|_2^2 = (\mathbf{e}_i - \mathbf{e}_j)^\top \mathbf{K}_M^{\mathbf{\Theta}}(t)(\mathbf{e}_i - \mathbf{e}_j)$$

We define the *Discounted Markov Diffusion Learnable Kernel* as:

$$\mathbf{K}_M^{\mathbf{\Theta}}(t) = (\beta\mathbf{Z}(t) + (1-\beta)\mathbf{I})^\top \mathbf{X}\mathbf{\Theta}\mathbf{\Theta}^\top\mathbf{X}^\top(\beta\mathbf{Z}(t) + (1-\beta)\mathbf{I})$$

It is also easy to see that $\mathbf{K}_M^{\mathbf{\Theta}}(t)$ is symmetrical and positive semi-definite. The underlying projection from the original space $\mathbf{\Omega}$ to the new hidden representation space denotes as

$$\tilde{\phi}_{\mathbf{\Theta}}^{(t)}(\mathbf{v}_i) = \mathbf{e}_i^\top(\beta\mathbf{Z}(t) + (1-\beta)\mathbf{I})^\top\mathbf{X}\mathbf{\Theta} \tag{3}$$

Totally, the new hidden representation $\tilde{\phi}_{\mathbf{\Theta}}^{(t)}(\mathbf{v}_i)$ of $\mathbf{v}_k$ can be described as the two phases: (i) original signals on all vertices propagate to $\mathbf{v}_k$ concerning the discounted Markov diffusion process which aggregates the structure information underlying the hypergraph. (ii) $\mathbf{\Theta}$ is introduced to aggregate information among different channels of the representation after the first phase. In the next part, we would directly expand $\mathbf{K}_M^{\mathbf{\Theta}}(t)$ to an effective convolutional operator for hypergraphs.

4.3 Simple Hypergraph Kernel Convolution Networks

To give a specific formulation for a DMDLK, we should identify a specific topology for the discrete structure. In this paper, we make effort for hypergraphs as it has a wider concept and is capable of engaging more sophisticated topology information than graphs.

Generalized Hypergraph (Zhang et al., 2022b). Assume a hypergraph defined as $\mathcal{H}(\mathcal{V}, \mathcal{E}, \mathbf{W}, \mathbf{Q}_1, \mathbf{Q}_2)$. Here the $\mathcal{V}$ is a finite vertex set with N vertices and $\mathcal{E}$ denotes a hyperedge set where each hyperedge $e \in \mathcal{E}$ can contain more than 2 vertices. $\mathbf{W} \in \mathbb{R}^{|\mathcal{E}|\times|\mathcal{E}|}$ is a diagonal matrix whose diagonal denotes the prior weights of hyperedges. $\mathbf{Q}_1$ and $\mathbf{Q}_2 \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{E}|}$ denote two different edge-dependent vertex weights matrices. $Q_i(v, e)$ denotes the weight of vertex v depending on an incident hyperedge e which means the vertex v contributes different weights to different hyperedges. Note if e is not linked to v, $Q_i(v, e) = 0$. The edge-dependent vertex weighs represent more fine-grained relations between vertices and hyperedges, which indicates hypergraphs are capable of modeling more sophisticated topology information than simple graphs. This motivates us to concentrate on hypergraphs to research more effective convolutional operators for discrete structures.

Probability Transition Matrix on Hypergraph. Firstly, we introduce a generalized random walk defined on hypergraphs from (Zhang et al., 2022b) where the probability of the random walk from vertex u to vertex v with a two-step manner is:

$$\Pr(u, v) = \sum_e \left(\frac{w(e)Q_1(u, e)\delta(e)\rho(\delta(e))}{d(u)} \right) \left(\frac{Q_2(v, e)}{\delta(e)} \right) \tag{4}$$

Here, $\delta(e) = \sum_v Q_2(v, e)$ is defined as the degree of hyperedge e and $d(u) = \sum_e w(e)Q_1(u, e)\delta(e)\rho(\delta(e))$ is defined as the degree of vertex u. $\rho(\cdot)$ denotes a selectable function that determines the effect of $\delta(e)$ to $d(u)$. $\mathbf{Q}_1$ and $\mathbf{Q}_2$ play their roles of representing fine-grained topology information in the generalized random walk. In practice, we choose $\mathbf{Q}_1 = \mathbf{Q}_2 = \mathbf{Q}$ with a rational assumption that the vertex weights depending on hyperedge keep constant in the two-step random walk. Then we choose to use a modified symmetrical form of $\mathbf{P}$ to define the probability transition matrix $\mathbf{T}$ on hypergraph:

$$\mathbf{T} = \mathbf{D}_{\mathcal{V}}^{-1/2}\mathbf{Q}\mathbf{W}\rho(\mathbf{D}_{\mathcal{E}})\mathbf{Q}^{\top}\mathbf{D}_{\mathcal{V}}^{-1/2} \tag{5}$$

Hyperedges Containing Isolated Vertex. However, as we analyze the effectiveness of $\tilde{\phi}_{\mathbf{\Theta}}^{(t)}(\mathbf{v}_i)$ from a transductive learning perspective in Section 5, we find it is important to bound the l_1-norm of $\mathbf{T}$ to derive a bounded generalization gap between training error and testing error. We find it derive from a flaw of the defined random walk in Eq. (4) where those hyperedges containing only one vertex(isolated vertex) have not been taken into account. Unfortunately, when there are isolated vertices in a hypergraph, the l_1-norm of $\mathbf{T}$ defined above can not be bounded, leading to an inadequate performance in downstream classification tasks. This also provides a new view to explain the experiments of Huang & Yang (2021) and Zhang et al. (2022b) w.r.t self-loops. We find it derive from a flaw of the defined random walk in Eq. (4) where those hyperedges containing only one vertex(isolated vertex) have not been taken into account. This can be amended by modifying $\mathbf{Q}$: $\tilde{\mathbf{Q}} = \text{concate}(\mathbf{Q}, \{\mathbf{e}_{i_1}, \cdots, \mathbf{e}_{i_k}\})$ if there is an isolated vertex subset $\{\mathbf{v}_{i_1}, \cdots, \mathbf{v}_{i_k}\} \subset \mathcal{V}$. Here $\mathbf{e}_{i_k} \in \mathbb{R}^{|\mathcal{V}|}$ denotes a vector whose i_k-th component is 1 otherwise 0. However, if the number of isolated vertices is large, this modification leads to extra expensive computation costs due to the added dimension of $\mathbf{Q}$. In fact, this modification is equal to set $T_{ii} = 1$ if the original $T_{ii} = 0$. Thus, we take an approximate method by modifying $\mathbf{T}$ directly to define *the generalized transition matrix on hypergraphs* :

$$\tilde{\mathbf{T}} = \tilde{\mathbf{D}}_{\mathcal{V}}^{-1/2}(\mathbf{Q}\mathbf{W}\rho(\mathbf{D}_{\mathcal{E}})\mathbf{Q}^{\top} + \mathbf{I})\tilde{\mathbf{D}}_{\mathcal{V}}^{-1/2} \tag{6}$$

where $\tilde{D}_{\mathcal{V}}(i, i) = \tilde{d}(v_i) = 1 + \sum_e w(e)Q(u, e)\delta(e)\rho(\delta(e))$. This modification is well-known as *the renormalization trick* proposed in GCNs (Kipf & Welling, 2017b). We hope to describe why this trick works from this perspective. Finally, we give a proposition below to bound l_1-norm of $\tilde{\mathbf{T}}$.

Proposition 4.1 (Bound l_1 norm of $\tilde{\mathbf{T}}$)**.** *For any hyperedge $e \in \mathcal{E}$, let n_e be the number of vertices incident to e;For any vertex $v \in \mathcal{V}$, let n_v be the number of hyperedges incident to v. Assume that $\mathbf{Q}$ is a normalized edge-dependent vertex weight matrix ($Q(v, e) \in [0, 1]$), $\mathbf{W}$ is a normalized prior hyperedge weights diagonal matrix, $n_e \leq E$ for any hyperedge e,$n_v \leq D$ for any vertex v ,and ρ_{max} is the maximum value of $\rho(x)$ when $x \in [0, E]$. Let $\tilde{\mathbf{T}}$ be the generalized transition matrix for a hypergraph defined above. Then, $\|\tilde{\mathbf{T}}\|_1 \leq \sqrt{1 + \rho_{max}ED}$.*

The proof is referred to in Appendix D.5. During the analysis of the effectiveness of $\tilde{\mathbf{y}}_k^{(t)}(\mathbf{\Theta})$ in Section 5, this proposition plays a critical role to derive a bounded generalization gap between training error and testing error.

Simple Hypergraph Kernel Convolution (SHKC). Take the transition matrix $\tilde{\mathbf{T}}$ into the projection $\tilde{\phi}_{\mathbf{\Theta}}^{(t)}(\mathbf{v}_i)$ underlying $\mathbf{K}_M^{\mathbf{\Theta}}(t)$ in Eq. (3). We have the hidden representation on all vertices as: $\mathbf{H}^{(t)} = (\beta \sum_{\tau=1}^{t} \frac{\alpha^\tau}{t} \tilde{\mathbf{T}}^\tau + (1-\beta)\mathbf{I})\mathbf{X}\mathbf{\Theta}$.

Recall that t is the diffusion step. α is the introduced discounted factor to weaken long-range global information through the diffusion process and β plays a role in balancing information aggregated to the vertex through the diffusion process and original signals in the vertex itself. By adding an activation function $\psi(\cdot)$ to make up nonlinearity of the underlying projection, we define the *Simple Hypergraph kernel Convolution* as follows:

$$\mathbf{H}^{(t)} = \psi\left(\left(\beta \sum_{\tau=1}^{t} \frac{\alpha^\tau}{t} \tilde{\mathbf{T}}^\tau + (1-\beta)\mathbf{I}\right)\mathbf{X}\mathbf{\Theta}\right) \tag{7}$$

where $\mathbf{\Theta}$ is the parameter of the filter to be learned during training. SHKC could gain stronger local and weaker global information, thereby improving the expressive power in a long diffusion step. Thus, SHKC can choose a long step t to replace stacking multi convolution layers for long-range information aggregation. The latter always leads to an oversmoothing issue in GNNs. Actually, SHKC is a spatial-based model while we can analyze it (detailed in Appendix B) from a spectral-based view which may lead to some intrinsic connections between the kernel-based convolution and spectral-based convolution.

5 A Transductive Learning Perspective for Generalization Analysis for SHKC

For simplicity, we give the theoretical analysis of SHKC in the setting of binary classification. Let $\mathbf{H}^{(L)}$ be the hidden representation of all vertices with the L-steps discounted Markov diffusion process. The i-th row $\mathbf{h}_i^{(L)}$ in $\mathbf{H}^{(L)}$ denotes the hidden representation of $\mathbf{v}_i$: $\mathbf{h}_i^{(L)} = \psi(\tilde{\phi}_{\mathbf{\Theta}}^{(L)}(\mathbf{v}_i))$. Let $\vec{\omega}$ be the weight of a downstream classifier f. Thus, the prediction label of vertex $\mathbf{v}_i$ denotes as $f(\mathbf{h}_i^{(L)}) = \sigma(\mathbf{h}_i^{(L)}\vec{\omega})$ where σ is the sigmoid function.

We start from introducing the transductive uniform stability bound(USB) theorem from El-Yaniv & Pechyony (2006)(detailed in Appendix E). The USB theorem shows that the generalization gap depends on the uniform stability μ of $\mathcal{M}$. A most recent work (Cong et al., 2021) decompose μ into three parts: Lipschitz constant $L_{\mathcal{M}}$, upper bound on gradient $G_{\mathcal{M}}$, and smoothness constant $S_{\mathcal{M}}$ of the learning model $\mathcal{M}$, which is formulated as $\mu = \frac{2\eta L_{\mathcal{M}} G_{\mathcal{M}}}{m} \sum_{t=1}^{T} (1+\eta S_{\mathcal{M}})^{t-1}$ where T is training steps and η is the learning rate. We follow this approach to evaluate the effectiveness of $\tilde{\phi}_{\mathbf{\Theta}}^{(L)}$ which is described by the generalization gap of SHKC. Then the main effort to analyze the generalization gap boils down to the bound of $L_{\mathcal{M}}, G_{\mathcal{M}}, S_{\mathcal{M}}$ for SHKC. We firstly start from three Lemmas shown in Appendix C. Then, we get our main results:

Theorem 5.1 (Main Results For SHKC)**.** *Assume that (i) for any vertex i, its norm of the feature vector is bounded by a constant $\|\mathbf{x}_i\|_2 \leq C_x$; (ii) the norm of learnable parameters of SHKC is bounded as $\|\mathbf{\Theta}\|_2 \leq C_{\mathbf{\Theta}}$; (iii) the norm of the weight of the classifier is bounded as: $\|\vec{\omega}\|_2 \leq 1$. Let α, L, β be the hyper-parameters defined in SHKC during training and η be the learning rate. Then the model SHKC is μ_{SHKC} uniform stable with $\mu_{SHKC} = \frac{2\eta L_{\mathcal{M}} G_{\mathcal{M}}}{m} \sum_{t=1}^{T} (1+\eta S_{\mathcal{M}})^{t-1}$ where:*

$$L_{\mathcal{M}} = C_x C_{\alpha\beta L} \max\{1, C_\Theta\}; \quad G_{\mathcal{M}} = C_x C_{\alpha\beta L}(1 + C_\Theta);$$
$$S_{\mathcal{M}} = C_x^2 C_{\alpha\beta L}^2 \max\{1, C_\Theta\}^2 + C_x^2 C_{\alpha\beta L}^2 C_\Theta + C_x C_{\alpha\beta L}$$

Here $C_{\alpha\beta L} = \frac{\beta}{L} \sum_{l=1}^{L} (\alpha d_T)^l + (1-\beta)$ and $d_T = \sqrt{1 + \rho_{max} ED}$ is the l_1 norm of the generalized transition matrix for a hypergraph in Proposition 4.1.

Proof sees in Appendix D.4. The key of the three components uniquely corresponded to SHKC is $C_{\alpha\beta L}$. We conclude that: (i) In $C_{\alpha\beta L}$, it is easy to see that α discounts d_T^l when l is large which corresponds to our intuition to weaken the effect of global information on vertex representation. (ii) The defined discounted average visiting rate in Eq. (1) leads to $\frac{1}{L}$ in $C_{\alpha\beta L}$ which relatively limits the

enlargement of $C_{\alpha\beta L}$ as L increases. This keeps relative balanced weights to information aggregated from different diffusion steps. (iii) β is introduced to balance the $C_{\alpha\beta L}$ between $\frac{1}{L}\sum_{l=1}^{L}(\alpha d_T)^l$ and 1. This corresponds to the intuition of balancing the effect between aggregated information and original feature to the hidden representation of vertices. Furthermore, this theorem reveals that SHKC can tighten the generalization bound by adjusting α,β and L.

6 Experiments

6.1 Citation Network Classification

This is a semi-supervised node classification task. The datasets we use are hypergraph benchmarks constructed by Yadati et al. (2019)(See Appendix Table 5). We adopt the same public datasets[1] and train-test splits in Yadati et al. (2019). Note these datasets satisfy $\mathbf{Q}_1 = \mathbf{Q}_2 = \mathbf{H}$. For baselines, we involve MLP with explicit Hypergraph Laplacian Regularization (MLP+HLR), HNHN, HyperSAGE, HGAT, UniGNN, HGNN and HyperGCN.

Table 1: Summary of classification accuracy(%) results. We report the average test accuracy and its standard deviation over 10 train-test splits. The number in parentheses corresponds to the number of diffusion step of SHKC. (OOM: out of memory)

Dataset	Architecture	Cora (co-authorship)	DBLP (co-authorship)	Cora (co-citation)	Pubmed (co-citation)	Citeseer (co-citation)
MLP+HLR	-	59.8±4.7	63.6±4.7	61.0±4.1	64.7±3.1	56.1±2.6
FastHyperGCN Yadati et al. (2019)	spectral-based	61.1±8.2	68.1±9.6	61.3±10.3	65.7±11.1	56.2±8.1
HyperGCN Yadati et al. (2019)	spectral-based	63.9±7.3	70.9±8.3	62.5±9.7	68.3±9.5	57.3±7.3
HGNN (Feng et al., 2019)	spectral-based	63.2±3.1	68.1±9.6	70.9±2.9	66.8±3.7	56.7±3.8
HNHN (Dong et al., 2020)	message-passing	64.0±2.4	84.4±0.3	41.6±3.1	41.9±4.7	33.6±2.1
HGAT (Ding et al., 2020)	message-passing	65.4±1.5	OOM	52.2±3.5	46.3±0.5	38.3±1.5
HyperSAGE Arya et al. (2020)	message-passing	72.4±1.6	77.4±3.8	69.3±2.7	72.9±1.3	61.8±2.3
UniGNN (Huang & Yang, 2021)	message-passing	75.3±1.2	88.8±0.2	70.1±1.4	74.4±1.0	63.6±1.3
SHKC (ours)	kernel-based	**76.05±0.7(6)**	**89.17±0.2(16)**	**70.64±1.8 (32)**	**75.08±1.1(4)**	**65.14±1.0(32)**

Comparison with SOTAs. As shown in Table 1, the results successfully verify the effectiveness of SHKC which achieves a new SOTA performance across all five datasets. We have the observations: (i) SHKC consistently outperforms the baselines, indicating that it can utilize the elaborately designed diffusion process. (ii) HGNN, HNHN and HGAT show poor performance on disconnected datasets(e.g. Citeseer), mainly due to the values of the row corresponding to an isolated vertex leading to information loss which corresponds to the theoretical explanation in Section 4.3 that l_1-norm of the transition matrix in Eq.(5) can not be bounded. By modifying Eq.(5) to Eq.(6), SHKC makes up the flaws of performance degradation when meets isolated vertex.(iii) SHKC has a lower bias and standard deviation than others, showing better generalization. Furthermore, comparison of running time and computational complexity with existing hypergraph neural networks can be found in Appendix A.1.

6.2 Visual Object Classification

Table 2: Test accuracy on visual object classification. GVCNN+MVCNN represents combining the features or structures to generate multi-modal data.

Datasets	Feature	Structure	HGNN	UniGNN	HGAT	SHKC(ours)
NTU	MVCNN	MVCNN	80.11±0.38	75.25±0.17	80.40±0.47	**82.56±0.39**
	GVCNN	GVCNN	84.26±0.30	84.63± 0.21	84.45±0.12	83.35±0.30
	BOTH	BOTH	83.54±0.50	84.45±0.40	84.05±0.36	**85.12±0.25**
Model-Net40	MVCNN	MVCNN	91.28±0.11	90.36±0.10	91.29±0.15	**92.01±0.08**
	GVCNN	GVCNN	92.53±0.06	**92.88±0.10**	92.44±0.11	92.69±0.06
	BOTH	BOTH	97.15±0.14	96.69±0.07	96.44±0.15	**97.78±0.03**

This experiment is about semi-supervised learning. We employ two public benchmarks: Princeton ModelNet40 dataset (Wu et al., 2015) and the National Taiwan University (NTU) 3D model dataset (Chen et al., 2003) to evaluate our method. We follow HGNN (Feng et al., 2019) to

[1] https://github.com/malllabiisc/HyperGCN

preprocess the data by MVCNN (Su et al., 2015) and GVCNN (Feng et al., 2018). Finally, we use the datasets provided by the public Code [2]. Details can be found in Appendix A.2.

Results. Table 3 depicts that SHKC significantly outperform the image-input or point-input methods. These results demonstrate that SHKC can capture the similarity between objects in the hidden representation space to improve the performance of the classification task. Table 2 compares our methods with HGNN on NTU and ModelNet40. From those results, we can see that our methods outperform HGNN on both single modality and multi-modality (BOTH) datasets and our SHKC achieves much better performance on multi-modality compared with others. These results reveal that our SHKC has the advantage of combining such multi-modal information through concatenating the weighted incidence matrices ($\mathbf{Q}$) of hypergraphs, which means merging the multi-level hyperedges.

Table 3: Classification accuracy (%) on ModelNet40. The *embedding* means the output representations of MVCNN+GVCNN Extractor.

Methods	input	Accuracy
MVCNN (Feng et al., 2018)	image	90.1
PointNet (Qi et al., 2017a)	point	89.2
PointNet++ (Qi et al., 2017b)	point	90.1
DGCNN (Wang et al., 2019)	point	92.2
InterpCNN (Mao et al., 2019)	point	93.0
SimpleView (Uy et al., 2019)	image	93.6
pAConv (Xu et al., 2021)	point	93.9
HGAT (Ding et al., 2020)	embedding	96.4
UniGNN (Huang & Yang, 2021)	embedding	96.7
HGNN Feng et al. (2019)	embedding	97.2
SHKC(ours)	embedding	**97.7**

6.3 Over-Smoothing Analysis

It is worth to note that a stacked K-layers convolution could capture information from k-hop neighborhood. In order to capture long-range information, HGNN and HyperGCN are required to stack multi convolution layers leading to performance descending, as shown in Table 4, which is well-known as over-smoothing issues. However, our proposed SHKC significantly avoids the performance descending by enlarge step of diffusion to capture long-range information rather than stacking multi convolution layers. Given the same number of layers with diffusion steps , it can be observed that SHKC outperforms the other methods in almost all datasets, especially when number of layers is large.

Table 4: Summary of classification accuracy (%) results with various depths. In our SHKC, the number of layers is equivalent to t in Eq. (7). We report mean test accuracy over 10 train-test splits.

Dataset	Method	Layers/diffusion steps					
		2	4	8	16	32	64
Cora (co-authorship)	HyperGCN	60.66	57.50	31.09	31.10	30.09	31.09
	HGNN	69.23	67.23	60.17	29.28	27.15	26.62
	SHKC (ours)	74.60	**75.78**	**75.70**	**75.04**	**75.26**	**74.79**
DBLP (co-authorship)	HyperGCN	84.82	54.65	22.37	23.96	23.04	24.13
	HGNN	88.55	88.28	85.38	27.64	27.62	27.56
	SHKC (ours)	86.63	88.26	**89.00**	**89.17**	**89.05**	**88.60**
Cora (co-citation)	HyperGCN	62.35	58.29	31.09	31.17	31.09	29.68
	HGNN	55.60	55.72	42.10	26.16	24.40	24.43
	SHKC (ours)	62.21	64.57	**67.59**	**68.96**	**69.37**	**68.15**
Pubmed (co-citation)	HyperGCN	68.12	63.59	39.99	39.97	40.01	40.02
	HGNN	46.41	47.16	40.93	40.24	40.30	40.29
	SHKC (ours)	74.39	**74.91**	**74.41**	**73.90**	**72.79**	**71.49**
Citeseer (co-citation)	HyperGCN	56.94	36.75	20.72	20.41	20.16	18.95
	HGNN	39.93	38.98	36.67	19.91	19.86	19.79
	SHKC (ours)	61.63	**62.75**	**63.86**	**64.62**	**65.14**	**65.10**

7 Conclusion

In this paper, we review the design of convolution for discrete structure from a kernel perspective. We propose a two-phase paradigm that play roles in topology information aggregation and channel aggregation respectively to conduct convolutions. Specifically, we concentrate on hypergraph which is considered as a more general discrete structure to capture complex topology information. The proposed SHKC could adjust diffusion step to aggregate long-range information which avoids stacking multi existing convolution layers which leads to oversmoothing issues. Analysis based on uniform stability theorem corresponds to the outperforming empirical results on downstream tasks.

References

Agarwal, Sameer, Branson, Kristin, and Belongie, Serge. Higher order learning with graphs. In Proceedings of the 23rd international conference on Machine learning, pp. 17–24, 2006.

[2] https://github.com/iMoonLab/HGNN

Arya, Devanshu, Gupta, Deepak K, Rudinac, Stevan, and Worring, Marcel. Hypersage: Generalizing inductive representation learning on hypergraphs. arXiv preprint arXiv:2010.04558, 2020.

Carletti, Timoteo, Fanelli, Duccio, and Lambiotte, Renaud. Random walks and community detection in hypergraphs. Journal of Physics: Complexity, 2021.

Chen, Ding-Yun, Tian, Xiao-Pei, Shen, Yu-Te, and Ouhyoung, Ming. On visual similarity based 3d model retrieval. In Computer graphics forum, volume 22, pp. 223–232. Wiley Online Library, 2003.

Chen, Guanzi and Zhang, Jiying. Preventing over-smoothing for hypergraph neural networks. arXiv preprint arXiv:2203.17159, 2022.

Chen, Yuzhao, Bian, Yatao, Zhang, Jiying, Xiao, Xi, Xu, Tingyang, Rong, Yu, and Huang, Junzhou. Diversified multiscale graph learning with graph self-correction. arXiv preprint arXiv:2103.09754, 2021.

Chitra, Uthsav and Raphael, Benjamin. Random walks on hypergraphs with edge-dependent vertex weights. In International Conference on Machine Learning, pp. 1172–1181. PMLR, 2019.

Cong, Weilin, Ramezani, Morteza, and Mahdavi, Mehrdad. On provable benefits of depth in training graph convolutional networks. Advances in Neural Information Processing Systems, 34, 2021.

Ding, Kaize, Wang, Jianling, Li, Jundong, Li, Dingcheng, and Liu, Huan. Be more with less: Hypergraph attention networks for inductive text classification. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 4927–4936, 2020.

Dong, Yihe, Sawin, Will, and Bengio, Yoshua. Hnhn: Hypergraph networks with hyperedge neurons. arXiv preprint arXiv:2006.12278, 2020.

El-Yaniv, Ran and Pechyony, Dmitry. Stable transductive learning. In International Conference on Computational Learning Theory, pp. 35–49. Springer, 2006.

Feng, Yifan, Zhang, Zizhao, Zhao, Xibin, Ji, Rongrong, and Gao, Yue. Gvcnn: Group-view convolutional neural networks for 3d shape recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 264–272, 2018.

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

Fouss, Francois, Yen, Luh, Pirotte, Alain, and Saerens, Marco. An experimental investigation of graph kernels on a collaborative recommendation task. In Sixth International Conference on Data Mining (ICDM'06), pp. 863–868. IEEE, 2006.

Fouss, François, Francoisse, Kevin, Yen, Luh, Pirotte, Alain, and Saerens, Marco. An experimental investigation of kernels on graphs for collaborative recommendation and semisupervised classification. Neural networks, 31:53–72, 2012.

Haussler, David. Convolution kernels on discrete structures. Technical report, Technical report, Department of Computer Science, University of California . . . , 1999.

Huang, Jing and Yang, Jie. Unignn: a unified framework for graph and hypergraph neural networks. In Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21, 2021.

Kipf, Thomas N and Welling, Max. Semi-supervised classification with graph convolutional networks. In Proceedings of the International Conference on Learning Representations, 2017a.

Kipf, Thomas N and Welling, Max. Semi-supervised classification with graph convolutional networks. In Proceedings of the International Conference on Learning Representations, 2017b.

Kondor, Risi Imre and Lafferty, John. Diffusion kernels on graphs and other discrete structures. In Proceedings of the 19th international conference on machine learning, volume 2002, pp. 315–322, 2002.

Mao, Jiageng, Wang, Xiaogang, and Li, Hongsheng. Interpolated convolutional networks for 3d point cloud understanding. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 1578–1587, 2019.

Qi, Charles R, Su, Hao, Mo, Kaichun, and Guibas, Leonidas J. Pointnet: Deep learning on point sets for 3d classification and segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 652–660, 2017a.

Qi, Charles Ruizhongtai, Yi, Li, Su, Hao, and Guibas, Leonidas J. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In NIPS, 2017b.

Rossi, Ryan and Ahmed, Nesreen. The network data repository with interactive graph analytics and visualization. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 29, 2015.

Schölkopf, Bernhard, Smola, Alexander J, Bach, Francis, et al. Learning with kernels: support vector machines, regularization, optimization, and beyond. MIT press, 2002.

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

Shawe-Taylor, John, Cristianini, Nello, et al. Kernel methods for pattern analysis. Cambridge university press, 2004.

Smola, Alexander J and Kondor, Risi. Kernels and regularization on graphs. In Learning theory and kernel machines, pp. 144–158. Springer, 2003.

Su, Hang, Maji, Subhransu, Kalogerakis, Evangelos, and Learned-Miller, Erik. Multi-view convolutional neural networks for 3d shape recognition. In Proceedings of the IEEE international conference on computer vision, pp. 945–953, 2015.

Uy, Mikaela Angelina, Pham, Quang-Hieu, Hua, Binh-Son, Nguyen, Thanh, and Yeung, Sai-Kit. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 1588–1597. IEEE Computer Society, 2019.

Wang, Yue, Sun, Yongbin, Liu, Ziwei, Sarma, Sanjay E, Bronstein, Michael M, and Solomon, Justin M. Dynamic graph cnn for learning on point clouds. Acm Transactions On Graphics (tog), 38(5):1–12, 2019.

Wu, Zhirong, Song, Shuran, Khosla, Aditya, Yu, Fisher, Zhang, Linguang, Tang, Xiaoou, and Xiao, Jianxiong. 3d shapenets: A deep representation for volumetric shapes. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1912–1920, 2015.

Xu, Mutian, Ding, Runyu, Zhao, Hengshuang, and Qi, Xiaojuan. Paconv: Position adaptive convolution with dynamic kernel assembling on point clouds. arXiv preprint arXiv:2103.14635, 2021.

Yadati, Naganand, Nimishakavi, Madhav, Yadav, Prateek, Nitin, Vikram, Louis, Anand, and Talukdar, Partha. Hypergcn: A new method for training graph convolutional networks on hypergraphs. In Advances in Neural Information Processing Systems, pp. 1511–1522, 2019.

Zhang, Jiying, Chen, Yuzhao, Xiao, Xi, Lu, Runiu, and Xia, Shu-Tao. Learnable hypergraph laplacian for hypergraph learning. In ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 4503–4507. IEEE, 2022a.

Zhang, Jiying, Li, Fuyang, Xiao, Xi, Xu, Tingyang, Rong, Yu, Huang, Junzhou, and Bian, Yatao. Hypergraph convolutional networks via equivalency between hypergraphs and undirected graphs, 2022b. URL https://arxiv.org/abs/2203.16939.

Zhang, Jiying, Xiao, Xi, Huang, Long-Kai, Rong, Yu, and Bian, Yatao. Fine-tuning graph neural networks via graph topology induced optimal transport. In Raedt, Lud De (ed.), Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22, pp. 3730–3736. International Joint Conferences on Artificial Intelligence Organization, 7 2022c. doi: 10.24963/ijcai.2022/518. URL https://doi.org/10.24963/ijcai.2022/518. Main Track.

Zhang, Zizhao, Lin, Haojie, Gao, Yue, and BNRist, KLISS. Dynamic hypergraph structure learning. In IJCAI, pp. 3162–3169, 2018.

Zhou, Dengyong, Huang, Jiayuan, and Schölkopf, Bernhard. Learning with hypergraphs: Clustering, classification, and embedding. Advances in neural information processing systems, 19:1601–1608, 2006.

A Details of experiments

All the settings of our experiments follow Zhang et al. (2022b). The details show below.

A.1 Citation Network Classification

Datasets. The datasets we use for citation network classification include co-authorship and co-citation datasets: PubMed, Citeseer, Cora (Sen et al., 2008) and DBLP (Rossi & Ahmed, 2015). We adopt the hypergraph version of those datasets directly from Yadati et al. (2019), where hypergraphs are created on these datasets by assigning each document as a node and each hyperedge represents (a) all documents co-authored by an author in the co-authorship dataset and (b) all documents cited together by a document in co-citation dataset. The initial features of each document (vertex) are represented by bag-of-words features. The details about vertices, hyperedges and features are shown in Table 5.

Table 5: Real-world hypergraph datasets used in our citation network classification task.

Dataset	# vertices	# Hyperedges	# Features	# Classes	# isolated vertices
Cora (co-authorship)	2708	1072	1433	7	320(11.8%)
DBLP (co-authorship)	43413	22535	1425	6	0 (0.0%)
Pubmed (co-citation)	19717	7963	500	3	15877 (80.5%)
Cora (co-citation)	2708	1579	1433	7	1274 (47.0%)
Citeseer (co-citation)	3312	1079	3703	6	1854 (55.9%)

Settings and baselines. We adopt the same dataset and train-test splits (10 splits) as provided in their publically available implementation[3]. Note that this dataset just has the edge-independent vertex weights $\mathbf{H}$, which is also called the incidence matrix. So this experiment can be regarded as a special case of the specific application of our model (i.e. $\mathbf{Q} = \mathbf{H}$).

For baselines MLP+HLR, HNHN (Dong et al., 2020) , HyperSAGE (Arya et al., 2020), UniGNN (Huang & Yang, 2021), HGNN (Feng et al., 2019) and FastHyperGCN (Yadati et al., 2019), HyperGCN (Yadati et al., 2019), UniGNN (Huang & Yang, 2021) we reuse the results reported by Huang & Yang (2021). For HNHN (Dong et al., 2020) and HGAT (Ding et al., 2020), we implement them according to their public code.

We use cross-entropy loss and Adam SGD optimizer with early stopping with the patience of 100 epochs to train SHSC. For hyper-parameters, we use the grid search strategy. More details of hyper-parameters can be found in Table 8.

Running Time and Computational Complexity Firstly, we analyze the theoretical computational complexity of SHKC: For SHKC, the computational cost is the $\mathcal{O}(K|E|d + K|\mathcal{V}|d)$, which includes K sparse matrix multiplication and K summation over filters($|\mathcal{V}|d$ is the cost of adding features X). Then, we compare the running time with existing models in Table 6. The results illustrate that our method is of the same order of magnitude as SOTA's approach UniGNN and outperforms the HyperGCN and HGAT.

[3]https://github.com/malllabiisc/HyperGCN, Apache License

Table 6: The average training time per epoch with different methods on citation network classification task is shown below and timings are measured in seconds.(OOM: Out of Memory)

Methods	cora coauthorship	dblp coauthorship	cora cocitation	pubmed cocitation	citeseer cocitation
HyperGCN	0.150±0.058	1.181±0.071	0.151±0.029	1.203±0.104	0.130±0.029
HGNN	0.005±0.002	0.081±0.006	0.005±0.040	0.008±0.002	0.005±0.002
UniGNN	0.014±0.044	0.042±0.040	0.014±0.042	0.023±0.043	0.0168±0.043
HNHN	0.001±0.0026	0.007±0.014	0.0010±0.004	0.009±0.006	0.001±0.003
HGAT	0.381±0.080	OOM	0.279±0.083	1.329±0.016	0.286±0.087
SHKC (ours)	0.055±0.001	0.291±0.001	0.205±0.003	0.135±0.056	0.193±0.057

A.2 Visual Object Classification

Table 7: summary of the ModelNet40 and NTU datasets

Dataset	ModelNet40	NTU
Objects	12311	2012
MVCNN Feature	4096	4096
GVCNN Feature	2048	2048
Training node	9843	1639
Testing node	2468	373
Classes	40	67

Datasets and Settings. We employ two public benchmarks: Princeton ModelNet40 dataset (Wu et al., 2015) and the National Taiwan University (NTU) 3D model dataset (Chen et al., 2003), as shown in Table 7.

In this experiment, each 3D object is represented by the feature vectors which are extracted by Multi-view Convolutional Neural Network (MVCNN) (Su et al., 2015) or Group-View Convolutional Neural Network (GVCNN) (Feng et al., 2018). The features generated by different methods can be considered as multi-modality features. The hypergraph structure we designed is similar to Zhang et al. (2018)(but they did not give spectral guarantees for supporting the rationality of their practices). We represent the hypergraph structure as an edge-dependent vertex weight $\mathbf{Q}$ (i.e. $\mathbf{Q}_1 = \mathbf{Q}_2 = \mathbf{Q}$). Specifically, we firstly generate hyperedges by k-*NN* approach, i.e. each time one object can be selected as a centroid and its k nearest neighbors are used to generate one hyperedge including the centroid itself (in our experiment, we set $k = 6$). Then, given the features of the data, the vertex-weight matrix $\mathbf{Q}$ is defined as

$$\mathbf{Q}(v, e) = \begin{cases} \exp(\frac{-d(v,v_c)}{\gamma \hat{d}^2}), & \text{if } v \in e \\ 0, & \text{otherwise,} \end{cases} \tag{8}$$

where $d(v, v_c)$ is the euclidean distance of features between an object v and the centroid object v_c in the hyperedge and $\hat{d}$ is the average distance between objects. γ is a hyper-parameter to control the flatness. As we have two-modality features generated by MVCNN and GVCNN, we can obtain the matrix $\mathbf{Q}_{\{i\}}$ which corresponds to the data of the i-th modality ($i \in \{1, 2\}$). After all the hypergraphs from different features have been generated, these matrices $\mathbf{Q}_{\{i\}}$ can be concatenated to build the multi-modality hypergraph matrix $\mathbf{Q} = [\mathbf{Q}_{\{1\}}, \mathbf{Q}_{\{2\}}]$. The features generated by GVCNN or MVCNN can be singly used, or concatenated to a multi-modal feature for constructing the hypergraphs.

We use cross-entropy loss and Adam SGD optimizer with early stopping with the patience of 100 epochs to train SHSC. More details of hyper-parameters can be found in Table 8.

A.3 Sensitivity Analysis

Performance of SHKC on co-authorship cora, co-citation cora and co-citation pubmed with different ρ, β and α is reported in Figure 1 where $\rho(\cdot) = (\cdot)^\sigma$. For σ, we can see that the best choice will vary depending on the dataset but mainly concentrate around -0.5, which verifies that the effect of hyperedges degree to the transition probability is various and it has negative effect in most cases. For β, with the growth of β, the performance of SHKC stably increases, which means that the topology information is successfully aggregated through the defined diffusion process. Moreover, the fact that the performance remains stable on cora (coauthorship) and pubmed (cocitation) when β is at 0.8-1.0

Table 8: Hyper-parameter search range for citation network classification and visual object classification.

Methods	Hyper-parameter	Range
SHKC	σ	{-2,-1,-0.5,0,0.5,1,2}
	γ(visual object classification)	{0.1, 0.2, 0.4, 0.5, 0.8,1.0}
	Learning rate	{0.001, 0.005, 0.01}
	Hidden dimension	{128}
	Layers	{2,4,6,8,16,32,64}
	Weight decay	{1e-3,1e-4,5e-4, 1e-5}
	α	{1,0.97,0.95,0.9,0.85,0.8,0.75,0.7,0.65,0.6}
	β	{ 1,0.95,0.90,0.85,0.8 }
	Optimizer	Adam
	Epoch	1000
	Early stopping patience	100
	GPU	Tesla V100

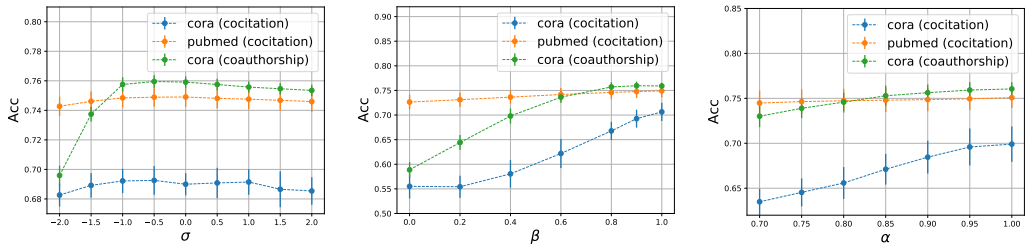

Figure 1: Test accuracy by varying the hyper-parameters σ (left), β (middle) and α (right).

suggests we can only adjust this hyper-parameter at a range of value close to 1. For α, the tendency of it is similar to β, so we can also adjust this hyper-parameter at a range of value close to 1 to obtain a satisfying performance.

B Spectral view of SHKC

Spectral View of SHKC. Actually, SHKC is a spatial-based model while we can analyze it from a spectral-based view. Let $\mathbf{L} = \mathbf{I} - \tilde{\mathbf{T}}$ denote the normalized Laplacian matrix. Then the SHKC can be viewed as a special spectral-based polynomial filter through $\sum_{i=0}^{I} \xi_i \mathbf{L}^i = \sum_{k=0}^{K} \theta_k \tilde{\mathbf{T}}^k$ where $\theta_k = \frac{\alpha^k}{K}$. Finally, we can deduce the coefficients of the special polynomial filter as $\xi_i = (-1)^i \sum_{k=i}^{K} \binom{k}{i} \frac{\alpha^k}{K}$ showing the strong relationships between SHKC and spectral-based model. The explicit form of ξ_i can be derived as:

$$\sum_{k=0}^{K} \theta_k \tilde{\mathbf{T}}^k = \sum_{k=0}^{K} \theta_k (\mathbf{I} - \mathbf{L})^k = \sum_{k=0}^{K} \theta_k \sum_{i=0}^{k} \binom{k}{i} (-1)^i \mathbf{L}^i$$
$$= \sum_{i=0}^{K} \left(\sum_{k=i}^{K} (-1)^i \binom{k}{i} \frac{\alpha^k}{K} \right) \mathbf{L}^i$$

The Spectrum Analysis of SHKC We calculate the eigenvalues of $\left(\beta \sum_{k=1}^{K} \frac{\alpha^k}{K} \tilde{\mathbf{T}}^k + (1-\beta)\mathbf{I}\right)$ with various (α, β) on NTU2012 dataset ($|\mathcal{V}| = 2012$) and count the number of eigenvalues in different size ranges, which is shown in Table 9. The table suggests that our SHKC can capture both low and high-frequency information of the graph signal, depending on the selection of the appropriate hyper-parameters (α, β).

Table 9: The number of eigenvalues of SHKC in different size ranges on NTU2012 dataset.

α, β	$\lambda \geq 0.0$	$\lambda \geq 0.1$	$\lambda \geq 0.2$	$\lambda \geq 0.3$	$\lambda \geq 0.4$	$\lambda \geq 0.5$	$\lambda \geq 0.6$	$\lambda \geq 0.7$	$\lambda \geq 0.8$	$\lambda \geq 0.9$
(1,1)	2012	364	163	108	82	66	52	38	26	13
(1,0.8)	2012	327	140	90	71	51	33	19	1	0
(0.8,1)	2012	161	40	0	0	0	0	0	0	0
(0.8,0.8)	2012	139	8	0	0	0	0	0	0	0

C Main Lemma.

Lemma C.1. *Let $h_{max}^{(l)} = max_i \|[\mathbf{H}^{(l)}]_{i,:}\|_2$ be the maximum norm of the vertex hidden representation with an l-step diffusion. Then for L, $h_{max}^{(L)}$ is bounded by:*

$$h_{max}^{(L)} \leq (\frac{\beta}{L}\sum_{l=1}^{L}(\alpha\sqrt{1+\rho_{max}ED})^l + 1 - \beta)C_x C_\Theta$$

The proof is referred to in Appendix D.1. Lemma C.1 derives the bound of the maximum norm of the hidden representation among all vertices which is required in the proofs of Lemma C.3 and Theorem 5.1.

Lemma C.2. *Define $\Delta h_{max}^{(l)} = \max_i \|[\mathbf{H}^{(l)}(\mathbf{\Theta}) - \mathbf{H}^{(l)}(\tilde{\mathbf{\Theta}})]_{i,:}\|_2$ as the maximum distance of hidden vertex representation with an l-step diffusion between different parameters $\mathbf{\Theta}$ and $\tilde{\mathbf{\Theta}}$. Then, this distance with an l-step diffusion can be bounded by:*

$$\Delta h_{max}^{(L)} \leq (\frac{\beta}{L}\sum_{l=1}^{L}(\alpha d_T)^l + 1 - \beta)C_x\|\Delta\mathbf{\Theta}\|_2$$

where $\Delta\mathbf{\Theta} = \mathbf{\Theta} - \tilde{\mathbf{\Theta}}$.

The proof is referred to Appendix D.2. This Lemma bounds the maximum norm of changes of hidden vertex representation with L-step diffusion between two different sets of parameters which is significant to compute the Lipschitz $L_{\mathcal{M}}$ and smoothness constant $S_{\mathcal{M}}$ of SHKC.

Lemma C.3. *Define $\mathbf{G}_{\mathbf{\Theta}} = \frac{\partial f}{\partial \mathbf{\Theta}}$ as the gradient of the prediction concerning learnable weights $\mathbf{\Theta}$ and $\mathbf{G}_{\vec{\omega}} = \frac{\partial f}{\partial \vec{\omega}}$ as the gradient of the prediction concerning the weights of the classifier. Let $\Delta_{\mathbf{\Theta}}\mathbf{G} = \mathbf{G}_{\mathbf{\Theta}} - \mathbf{G}_{\tilde{\mathbf{\Theta}}}$ and $\Delta_{\vec{\omega}}\mathbf{G} = \mathbf{G}_{\vec{\omega}} - \mathbf{G}_{\tilde{\vec{\omega}}}$ be the difference between gradient computed on two different learnable and classifier weights. Then we have:*

$$\begin{aligned}
&\|\mathbf{G}_{\mathbf{\Theta}}\|_2 \leq C_x C_{\alpha\beta L}; \quad \|\mathbf{G}_{\vec{\omega}}\|_2 \leq C_x C_{\alpha\beta L} C_\Theta; \\
&\|\Delta_{\mathbf{\Theta}}\mathbf{G}\|_2 \leq C_x^2 C_{\alpha\beta L}^2 \|\Delta\mathbf{\Theta}\|_2 + C_x^2 C_{\alpha\beta L}^2 C_\Theta \|\Delta\vec{\omega}\|_2 + C_x C_{\alpha\beta L}\|\Delta\vec{\omega}\|_2; \\
&\|\Delta_{\vec{\omega}}\mathbf{G}\|_2 \leq C_x^2 C_{\alpha\beta L}^2 C_\Theta^2 \|\Delta\vec{\omega}\|_2 + C_x^2 C_{\alpha\beta L}^2 C_\Theta \|\Delta\mathbf{\Theta}\|_2 + C_x C_{\alpha\beta L}\|\Delta\mathbf{\Theta}\|_2
\end{aligned}$$

where $C_{\alpha\beta L} = \frac{\beta}{L}\sum_{l=1}^{L}(\alpha d_T)^l + 1 - \beta$ and $d_T = \sqrt{1+\rho_{max}ED}$ in Proposition 4.1.

The proof is referred to Appendix D.3.

D Missing Proofs.

D.1 The Lemma 1

Lemma D.1. *Let $h_{max}^{(l)} = max_i \|[\mathbf{H}^{(l)}]_{i,:}\|_2$ be the maximum norm of the vertex hidden representation with an l-step diffusion. Then for L, $h_{max}^{(L)}$ is bounded by:*

$$h_{max}^{(L)} \leq (\frac{\beta}{L}\sum_{l=1}^{L}(\alpha\sqrt{1+\rho_{max}ED})^l + 1 - \beta)C_x C_\Theta$$

Proof. For deduction simplicity, we use $\mathbf{T}$ to represent the generalized transition matrix in the following deduction and define $d_T = \sqrt{1+\rho_{max}ED}$ from Proposition 4.1. We first deduce the

reclusive formulation of SHKC between layers as:

$$\mathbf{H}^{(0)} = \mathbf{X}\mathbf{\Theta};$$

$$\mathbf{H}^{(l+1)} = \frac{\alpha l}{l+1}\mathbf{T}\mathbf{H}^{(l)} + \frac{\alpha}{l+1}\mathbf{T}\mathbf{H}^{(0)}, \quad l = 0, \cdots, L-2; \tag{9}$$

$$\mathbf{H}^{(L)} = \psi(\beta(\frac{\alpha(L-1)}{L}\mathbf{T}\mathbf{H}^{(L-1)} + \frac{\alpha}{L}\mathbf{T}\mathbf{H}^{(0)}) + (1-\beta)\mathbf{X}\mathbf{\Theta}) = \psi(\mathbf{Z}^{(L)}) \tag{10}$$

where ψ is the ReLu activation function.

From $h_{max}^{(L)}$, we deduce the bound as:

$$\begin{aligned}
h_{max}^{(L)} &= \max_i \|[\psi(\beta(\frac{\alpha(L-1)}{L}\mathbf{T}\mathbf{H}^{(L-1)} + \frac{\alpha}{L}\mathbf{T}\mathbf{H}^{(0)}) + (1-\beta)\mathbf{X}\mathbf{\Theta})]_{i,:}\|_2 \\
&\leq \max_i \|\frac{[\beta\alpha(L-1)}{L}\mathbf{T}\mathbf{H}^{(L-1)}]_{i,:}\|_2 + \|[\frac{\beta\alpha}{L}\mathbf{T}\mathbf{X}\mathbf{\Theta}]_{i,:}\|_2 + \|(1-\beta)[\mathbf{X}\mathbf{\Theta}]_{i,:}\|_2 \\
&\leq_{(a)} \max_i \frac{\beta\alpha(L-1)}{L}\|\sum_j T_{ij}\mathbf{h}_j^{(L-1)}\|_2 + \frac{\beta\alpha}{L}\|\sum_j T_{ij}\mathbf{h}_j^{(0)}\|_2 + (1-\beta)C_x C_\Theta \\
&\leq \frac{\beta\alpha(L-1)}{L} d_T h_{max}^{(L-1)} + \frac{\beta\alpha}{L} d_T C_x C_\Theta + (1-\beta)C_x C_\Theta
\end{aligned} \tag{11}$$

where inequality (a) is due to Proposition 4.1 and assumptions in Theorem E.1. From the reclusive formulation of SHKC in Eq. (9), we have:

$$\begin{aligned}
h_{max}^{(L-1)} &\leq \frac{\alpha(L-2)}{L-1} d_T h_{max}^{(L-2)} + \frac{\alpha}{L-1} d_T C_x C_\Theta \\
&\cdots \\
&\leq \frac{\sum_{l=1}^{L-1}(\alpha d_T)^l}{L-1} C_x C_\Theta
\end{aligned} \tag{12}$$

Add the Eq. (12) to Eq. (11) we get:

$$h_{max}^{(L)} \leq (\frac{\beta}{L}\sum_{l=1}^{L}(\alpha d_T)^l + 1 - \beta)C_x C_\Theta$$

□

D.2 The Lemma 2

Lemma D.2. *Define* $\Delta h_{max}^{(l)} = \max_i \|[\mathbf{H}^{(l)}(\mathbf{\Theta}) - \mathbf{H}^{(l)}(\tilde{\mathbf{\Theta}})]_{i,:}\|_2$ *as the maximum distance of hidden vertex representation with an* l*-step diffusion between different parameters* $\mathbf{\Theta}$ *and* $\tilde{\mathbf{\Theta}}$*. Then, this distance with an* l*-step diffusion can be bounded by:*

$$\Delta h_{max}^{(L)} \leq (\frac{\beta}{L}\sum_{l=1}^{L}(\alpha d_T)^l + 1 - \beta)C_x\|\Delta\mathbf{\Theta}\|_2$$

where $\Delta\mathbf{\Theta} = \mathbf{\Theta} - \tilde{\mathbf{\Theta}}$.

Proof. We first denote $\mathbf{H}^{(l)}(\tilde{\mathbf{\Theta}})$ as $\tilde{\mathbf{H}}^{(l)}$ for similarity in deduction below.

$$\begin{aligned}
\Delta h_{max}^{(L)} &= \max_i \|[\mathbf{H}^{(L)}(\mathbf{\Theta}) - \mathbf{H}^{(L)}(\tilde{\mathbf{\Theta}})]_{i,:}\|_2 \\
&\leq \max_i \|\frac{[\beta\alpha(L-1)}{L}\mathbf{T}(\mathbf{H}^{(L-1)} - \tilde{\mathbf{H}}^{(L-1)}) \\
&+ \frac{\beta\alpha}{L}\mathbf{T}\mathbf{X}(\mathbf{\Theta} - \tilde{\mathbf{\Theta}}) + (1-\beta)\mathbf{X}(\mathbf{\Theta} - \tilde{\mathbf{\Theta}})]_{i,:}\|_2 \\
&\cdots \\
&\leq (\frac{\beta}{L}\sum_{l=1}^{L}(\alpha d_T)^l + 1 - \beta)C_x\|\Delta\mathbf{\Theta}\|_2
\end{aligned}$$

□

D.3 The Lemma 3

Lemma D.3. *Define $\mathbf{G}_{\mathbf{\Theta}} = \frac{\partial f}{\partial \mathbf{\Theta}}$ as the gradient of the prediction concerning learnable weights $\mathbf{\Theta}$ and $\mathbf{G}_{\vec{\omega}} = \frac{\partial f}{\partial \vec{\omega}}$ as the gradient of the prediction concerning the weights of the classifier. Let $\Delta_{\mathbf{\Theta}}\mathbf{G} = \mathbf{G}_{\mathbf{\Theta}} - \mathbf{G}_{\tilde{\mathbf{\Theta}}}$ and $\Delta_{\vec{\omega}}\mathbf{G} = \mathbf{G}_{\vec{\omega}} - \mathbf{G}_{\tilde{\vec{\omega}}}$ be the difference between gradient computed on two different learnable and classifier weights. Then we have:*

$$
\begin{aligned}
&\|\mathbf{G}_{\mathbf{\Theta}}\|_2 \leq C_x C_{\alpha\beta L}; \quad \|\mathbf{G}_{\vec{\omega}}\|_2 \leq C_x C_{\alpha\beta L} C_{\Theta};\\
&\|\Delta_{\mathbf{\Theta}}\mathbf{G}\|_2 \leq C_x^2 C_{\alpha\beta L}^2 \|\Delta\mathbf{\Theta}\|_2 + C_x^2 C_{\alpha\beta L}^2 C_{\Theta}\|\Delta\vec{\omega}\|_2 + C_x C_{\alpha\beta L}\|\Delta\vec{\omega}\|_2;\\
&\|\Delta_{\vec{\omega}}\mathbf{G}\|_2 \leq C_x^2 C_{\alpha\beta L}^2 C_{\Theta}^2 \|\Delta\vec{\omega}\|_2 + C_x^2 C_{\alpha\beta L}^2 C_{\Theta}\|\Delta\mathbf{\Theta}\|_2 + C_x C_{\alpha\beta L}\|\Delta\mathbf{\Theta}\|_2
\end{aligned}
$$

where $C_{\alpha\beta L} = \frac{\beta}{L}\sum_{l=1}^{L}(\alpha d_T)^l + 1 - \beta$ and $d_T = \sqrt{1 + \rho_{max} ED}$ in Proposition 4.1.

Proof. Note that the classifier f has the formulation as $f(\mathbf{h}_i^{(L)}) = \sigma(\mathbf{h}_i^{(L)}\vec{\omega})$ where σ is the sigmoid function and $\mathbf{z}_i^{(L)}$ is the ith node representation in the last layer before the activation function ψ.

$$
\begin{aligned}
\|\mathbf{G}_{\mathbf{\Theta}}\|_2 &= \frac{1}{m}\sum_i \|\frac{\partial Loss(f(\mathbf{h}_i^{(L)}), y_i)}{\partial \mathbf{\Theta}}\|_2\\
&\leq \max_i \kappa\sigma'(z_i)\|\frac{\partial \vec{\omega}^\top \mathbf{h}_i^{(L)}}{\partial \mathbf{\Theta}}\|_2\\
&\leq \max_i \kappa\|([\beta\sum_{l=1}^{L}\frac{(\alpha\mathbf{T})^l}{L}\mathbf{X}+(1-\beta)\mathbf{X}]_{i,:})^\top \mathbf{1} Diag(\psi'(\mathbf{z}_i^{(L)})\odot\vec{\omega})\|_2\\
&\leq \max_i \kappa\|([\beta\sum_{l=1}^{L}\frac{(\alpha\mathbf{T})^l}{L}\mathbf{X}+(1-\beta)\mathbf{X}]_{i,:})^\top \mathbf{1}\|_2\|\vec{\omega}\|_2\\
&\leq \max_i \kappa\|[\beta\sum_{l=1}^{L}\frac{(\alpha\mathbf{T})^l}{L}\mathbf{X}+(1-\beta)\mathbf{X}]_{i,:}\|_2\\
&\leq \kappa C_x(\frac{\beta\sum_{l=1}^{L}(\alpha d_T)^l}{L} + 1 - \beta)
\end{aligned}
$$

For the norm of $\mathbf{G}_{\vec{\omega}}$:

$$
\begin{aligned}
\|\mathbf{G}_{\vec{\omega}}\|_2 &= \frac{1}{m}\sum_i \|\frac{\partial Loss(f(\mathbf{h}_i^{(L)}), y_i)}{\partial \vec{\omega}}\|_2\\
&\leq \max_i \kappa\sigma'(z_i)\mathbf{h}_i^{(L)}\\
&\leq \kappa(\frac{\beta}{L}\sum_{l=1}^{L}(\alpha d_T)^l + 1 - \beta)C_x C_{\Theta}
\end{aligned}
$$

For the norm of $\Delta_{\mathbf{\Theta}}\mathbf{G}$:

$$
\begin{aligned}
\|\Delta_{\mathbf{\Theta}}\mathbf{G}\|_2 &\leq \max_i \kappa\|\frac{\partial \mathbf{h}_i^{(L)}}{\partial \mathbf{\Theta}}(\frac{\partial f(\mathbf{h}_i^{(L)})}{\partial \mathbf{h}_i^{(L)}} - \frac{\partial f(\tilde{\mathbf{h}}_i^{(L)})}{\partial \tilde{\mathbf{h}}_i^{(L)}})\|_2\\
&\leq \max_i \kappa\|[\beta\sum_{l=1}^{L}\frac{(\alpha\mathbf{T})^l}{L}\mathbf{X}+(1-\beta)\mathbf{X}]_{i,:})^\top \mathbf{1} Diag(\mathbf{r}-\tilde{\mathbf{r}})\|_2\\
&\leq \kappa C_x C_{\alpha\beta L}(\|\Delta\vec{\omega}\|_2(1 + h_{max}^{(L)}) + \Delta h_{max}^{(L)})\\
&\leq \kappa C_x^2 C_{\alpha\beta L}^2\|\Delta\mathbf{\Theta}\|_2 + \kappa C_x^2 C_{\alpha\beta L}^2 C_{\Theta}\|\Delta\vec{\omega}\|_2 + \kappa C_x C_{\alpha\beta L}\|\Delta\vec{\omega}\|_2
\end{aligned}
$$

where $\mathbf{r} = \sigma'(\vec{\omega}^\top\mathbf{h}_i^{(L)})\psi'(\mathbf{z}_i^{(L)})\odot\vec{\omega}$ and $C_{\alpha\beta L} = \frac{\beta}{L}\sum_{l=1}^{L}(\alpha d_T)^l + 1 - \beta$.

Finally, for $\Delta_{\vec{\omega}}\mathbf{G}$:

$$\begin{aligned}\|\Delta_{\vec{\omega}}\mathbf{G}\|_2 &\leq \kappa(\Delta h_{max}^{(L)} + h_{max}^{(L)}(\Delta h_{max}^{(L)} + h_{max}^{(L)}\|\vec{\omega}\|_2)) \\ &\leq \kappa C_x^2 C_{\alpha\beta L}^2 C_\Theta^2 \|\Delta\vec{\omega}\|_2 + \kappa C_x^2 C_{\alpha\beta L}^2 C_\Theta \|\Delta\mathbf{\Theta}\|_2 + \kappa C_x C_{\alpha\beta L}\|\Delta\mathbf{\Theta}\|_2\end{aligned}$$

□

D.4 The Theorem 5.1

Theorem 5.1 (Main Results For SHKC)**.** *Assume that (i) for any vertex i, its norm of the feature vector is bounded by a constant $\|\mathbf{x}_i\|_2 \leq C_x$; (ii) the norm of learnable parameters of SHKC is bounded as $\|\mathbf{\Theta}\|_2 \leq C_{\mathbf{\Theta}}$; (iii) the norm of the weight of the classifier is bounded as: $\|\vec{\omega}\|_2 \leq 1$. Let α, L, β be the hyper-parameters defined in SHKC during training and η be the learning rate. Then the model SHKC is μ_{SHKC} uniform stable with $\mu_{SHKC} = \frac{2\eta L_{\mathcal{M}} G_{\mathcal{M}}}{m}\sum_{t=1}^{T}(1+\eta S_{\mathcal{M}})^{t-1}$ where:*

$$\begin{aligned}&L_{\mathcal{M}} = C_x C_{\alpha\beta L}\max\{1, C_\Theta\}; \quad G_{\mathcal{M}} = C_x C_{\alpha\beta L}(1 + C_\Theta); \\ &S_{\mathcal{M}} = C_x^2 C_{\alpha\beta L}^2 \max\{1, C_\Theta\}^2 + C_x^2 C_{\alpha\beta L}^2 C_\Theta + C_x C_{\alpha\beta L}\end{aligned}$$

Here $C_{\alpha\beta L} = \frac{\beta}{L}\sum_{l=1}^{L}(\alpha d_T)^l + (1-\beta)$ and $d_T = \sqrt{1+\rho_{max}ED}$ is the l_1 norm of the generalized transition matrix for a hypergraph in Proposition 4.1.

Proof. We use $\{\mathbf{\Theta}, \vec{\omega}\}$ and $\{\tilde{\mathbf{\Theta}}, \tilde{\vec{\omega}}\}$ to denote two different sets of parameters throughout the model SHKC. We first give the deduction of $L_{\mathcal{M}} = C_x C_{\alpha\beta L}\max\{1, C_\Theta\}$ from Lemma C.1 and Lemma C.2:

$$\begin{aligned}\max_i |f(\mathbf{\Theta}, \vec{\omega}|\mathbf{x}_i) - f(\tilde{\mathbf{\Theta}}, \tilde{\vec{\omega}}|\mathbf{x}_i)| &\leq \Delta h_{max}^{(L)} + h_{max}^{(L)}\|\Delta\vec{\omega}\|_2 \\ &\leq C_x C_{\alpha\beta L}\|\Delta\mathbf{\Theta}\|_2 + C_x C_{\alpha\beta L} C_\Theta \|\Delta\vec{\omega}\|_2 \\ &\leq C_x C_{\alpha\beta L}\max\{1, C_\Theta\}(\|\Delta\mathbf{\Theta}\|_2 + \|\Delta\vec{\omega}\|_2)\end{aligned}$$

Then, from Lemma C.3, we get the bound of the gradient $G_{\mathcal{M}} = C_x C_{\alpha\beta L}(1 + C_\Theta)$ Finally, from Lemma C.3, we have the smoothness of $\mathcal{M}$ that $S_{\mathcal{M}}$:

$$\begin{aligned}&\max_i \|\nabla_{\mathbf{\Theta},\mathbf{v}} f(\mathbf{\Theta}, \vec{\omega}|\mathbf{x}_i) - \nabla_{\tilde{\mathbf{\Theta}},\tilde{\mathbf{v}}} f(\tilde{\mathbf{\Theta}}, \tilde{\vec{\omega}}|\mathbf{x}_i)\|_2 \leq \\ &(\|\Delta\mathbf{\Theta}\|_2 + \|\Delta\mathbf{v}\|_2)(C_x^2 C_{\alpha\beta L}^2 \max\{1, C_\Theta\}^2 + C_x^2 C_{\alpha\beta L}^2 C_\Theta + C_x C_{\alpha\beta L})\end{aligned}$$

□

D.5 Proof of Proposition 4.1

Proof. We have $\tilde{\mathbf{T}}$ to be formulated as:

$$\tilde{\mathbf{T}} = \tilde{\mathbf{D}}_v^{-1/2}(\mathbf{I} + \mathbf{Q}\mathbf{W}\rho(\mathbf{D}_{\mathcal{E}})\mathbf{Q}^\top)\tilde{\mathbf{D}}_v^{-1/2}$$

Then the l_1 norm of $\tilde{\mathbf{T}}$ is bounded as:

$$\begin{aligned}\|\tilde{\mathbf{T}}\|_1 &= \max_i \sum_j |\tilde{T}_{ij}| = \max_i \sum_j \frac{\mathbf{1}\{i=j\} + \sum_e Q(i,e)W(e)\rho(\delta(e))Q(j,e)}{\sqrt{\tilde{d}_v(i)}\sqrt{\tilde{d}_v(j)}} \\ &\leq_{(a)} \max_i \frac{\tilde{d}_v(i)}{\sqrt{\tilde{d}_v(i)}} = \max_i \sqrt{\tilde{d}_v(i)} \\ &\leq_{(b)} \sqrt{1 + E\rho_{max}}\end{aligned}$$

where $\tilde{d}_v(i) = 1 + \sum_e Q(i,e)w(e)\rho(\delta(e))\delta(e)$ and $\delta(e) = \sum_v Q(v,e)$. The inequality (a) is due to $\tilde{d}_v(i) \geq 1$. The inequality (b) is due to $\delta(e) = \sum_v Q(v,e) \leq n_e \leq E$ and

$$\begin{aligned}\tilde{d}_v(i) &= 1 + \sum_e Q(i,e)w(e)\rho(\delta(e))\delta(e) \\ &\leq 1 + \sum_e \rho(\delta(e))\delta(e) \leq 1 + \rho_{max}ED\end{aligned}$$

□

E Uniform Stability Bound

We slightly modify the Uniform Stability Bound in El-Yaniv & Pechyony (2006) for more distinct analysis of generalization error of SHKC.

Theorem E.1 (Uniform Stability Bound (El-Yaniv & Pechyony, 2006))**.** *Denote $\mathcal{M}$ to be a transductive learning model with uniform stability μ. Let $K(m,n) = \sum_{i=1}^{m} \frac{n^2}{(n+i)^2}$. For a κ-Lipschitz loss function bounded in $[0,1]$ and $\delta > 0$, the gap between the training error and testing error is bounded as:*

$$gap(\mathcal{M}, \gamma, \delta) \leq \mu\kappa(1 + \mathcal{O}(2\sqrt{2K(m,n)\ln\delta^{-1}})) + \mathcal{O}(\frac{m+n}{mn}\sqrt{2K(m,n)\ln\delta^{-1}})$$

where m,n denote the number of samples in the training and testing sets respectively.