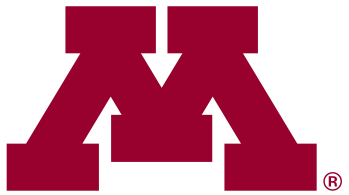
Sheaves for AI:

Graph Representation Learning through Sheaf Theory

Thomas Gebhart

University of Minnesota, Department of Computer Science

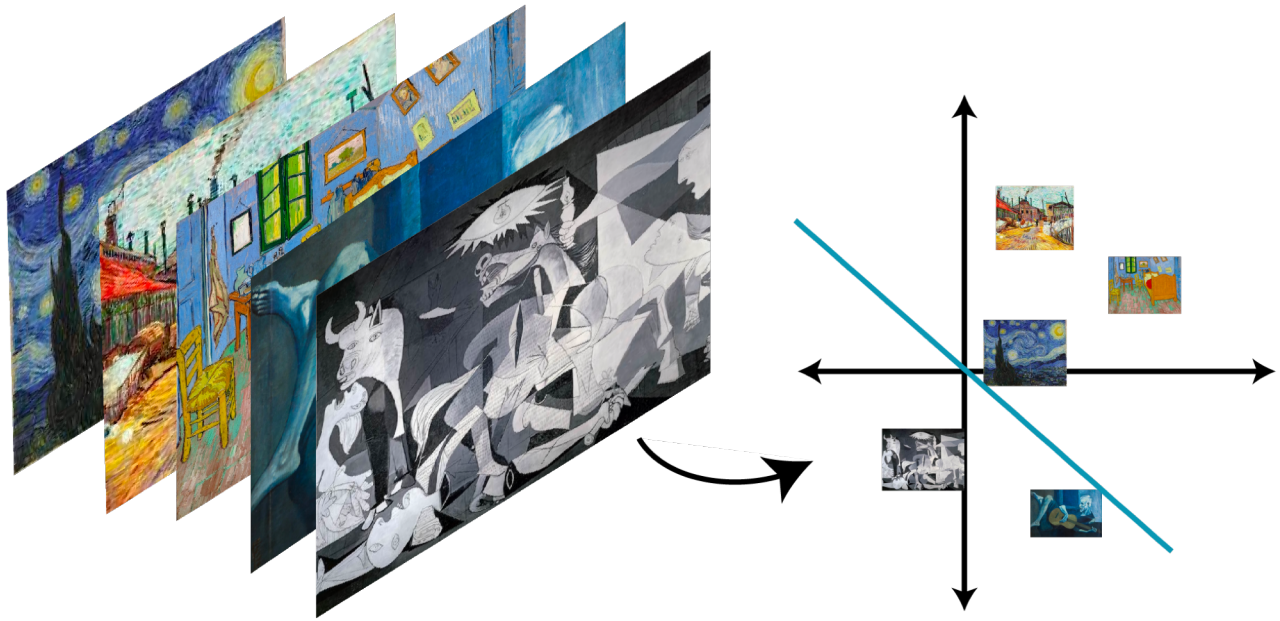


Outline

- Motivation: Graph Representation Learning
- Sheaves
- Cellular Sheaves
- Applications

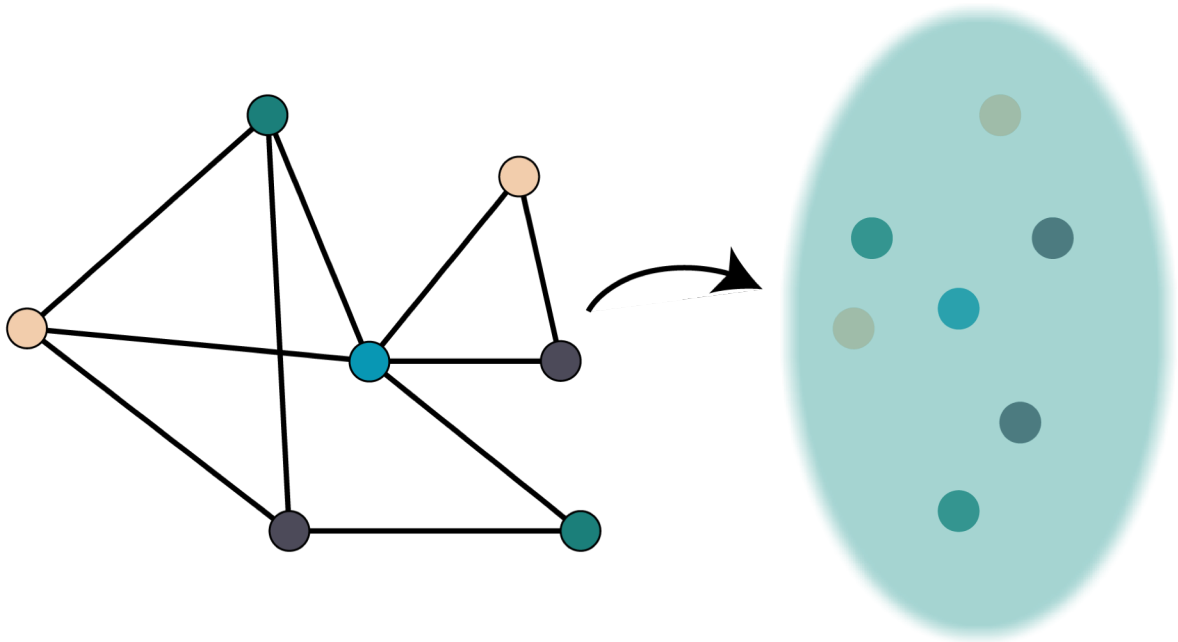
Representation Learning

Representation learning, also called feature learning, is ubiquitous in ML.



Broadly, we seek a tractable representation of data sampled from a complex space.

This representation should preserve structure within the data space, informing a down-stream task (regression, classification, reconstruction, etc.).



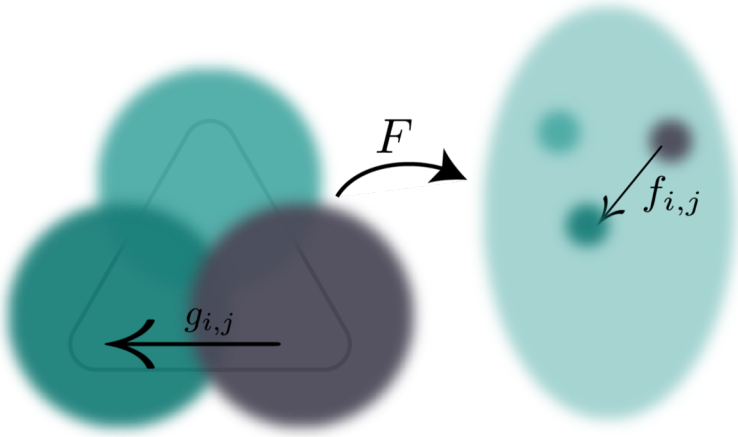
Representation Learning

Let $X = \bigcup X_i$ be a space, viewed as the union of open sets $X_i \in X$.

Define a *representation* as a map $F : X \to V$ such that for any relationship $g_{i,j} : X_i \to X_j$ in X, there exists an associated $f_{i,j} : \boldsymbol{x}_i \to \boldsymbol{x}_j$ in V such that $f_{i,j}(F(X_i)) = F(g_{i,j}(X_i))$.

In machine learning, V is almost always taken to be $\mathbb{R}^n$.

In practice, the most useful g maps are usually unknown and the representation won't precisely commute.



Inductive biases like local smoothness or the distributional hypothesis often fill this gap.

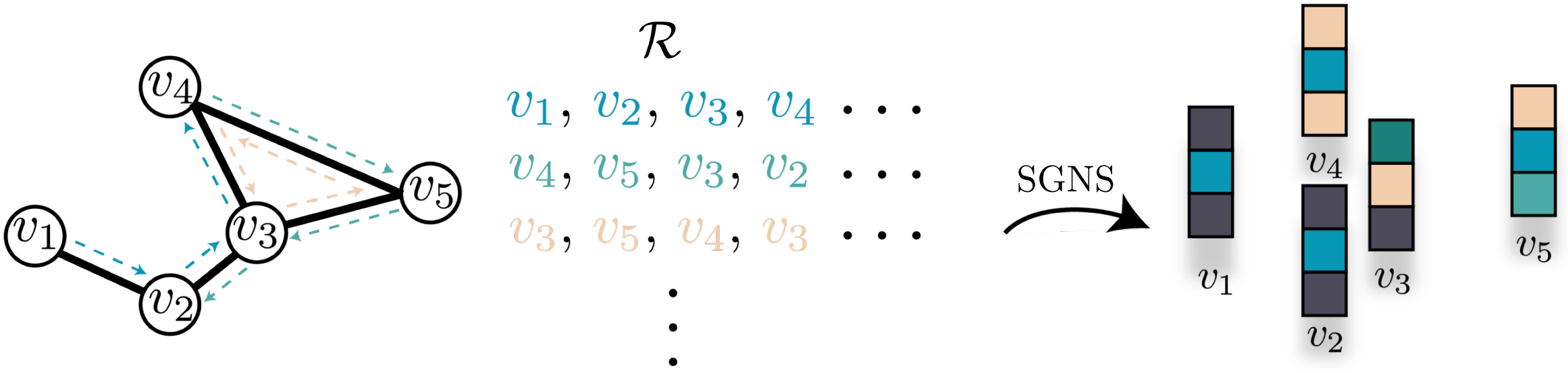
Let's look at two examples: node embedding and graph signal processing.

Node Embedding with DeepWalk

Let $G = (V, E)$ be a graph. We seek $F : V \to \mathbb{R}^d$ which encodes the graph structure.

1. Generate a "corpus" $\mathcal{R}$ of node visits by aggregating random walks from each $v \in V$.
2. Learn shallow embeddings via skip-gram with negative sampling (SGNS).

Result: nearby nodes in G are assigned similar vectors in $\mathbb{R}^d$.



Node Embedding with DeepWalk

Qiu et al. show that this node embedding procedure is implicitly matrix factorization.

Let $\boldsymbol{P}$ be the transition matrix for a random walk on $G = (V, E)$.

As the walk length becomes infinite, DeepWalk embeddings approach a factorization of:

$$\boldsymbol{M}^{\mathrm{DW}}(T) = \log\left(\frac{\nu(G)}{T}\left(\sum_{r=1}^{T}\boldsymbol{P}^r\right)\boldsymbol{D}^{-1}\right)$$

flat representations

$$\boldsymbol{M}^{\mathrm{DW}}_{u,v} = \frac{\#(u,v)|\mathcal{R}|}{\#(u)\#(v)} \xrightarrow{p} \frac{\nu(G)}{2T}\left(\frac{1}{d_v}\sum_{r=1}^{T}(\boldsymbol{P}^r)_{u,v} + \frac{1}{d_u}\sum_{r=1}^{T}(\boldsymbol{P}^r)_{v,u}\right)$$

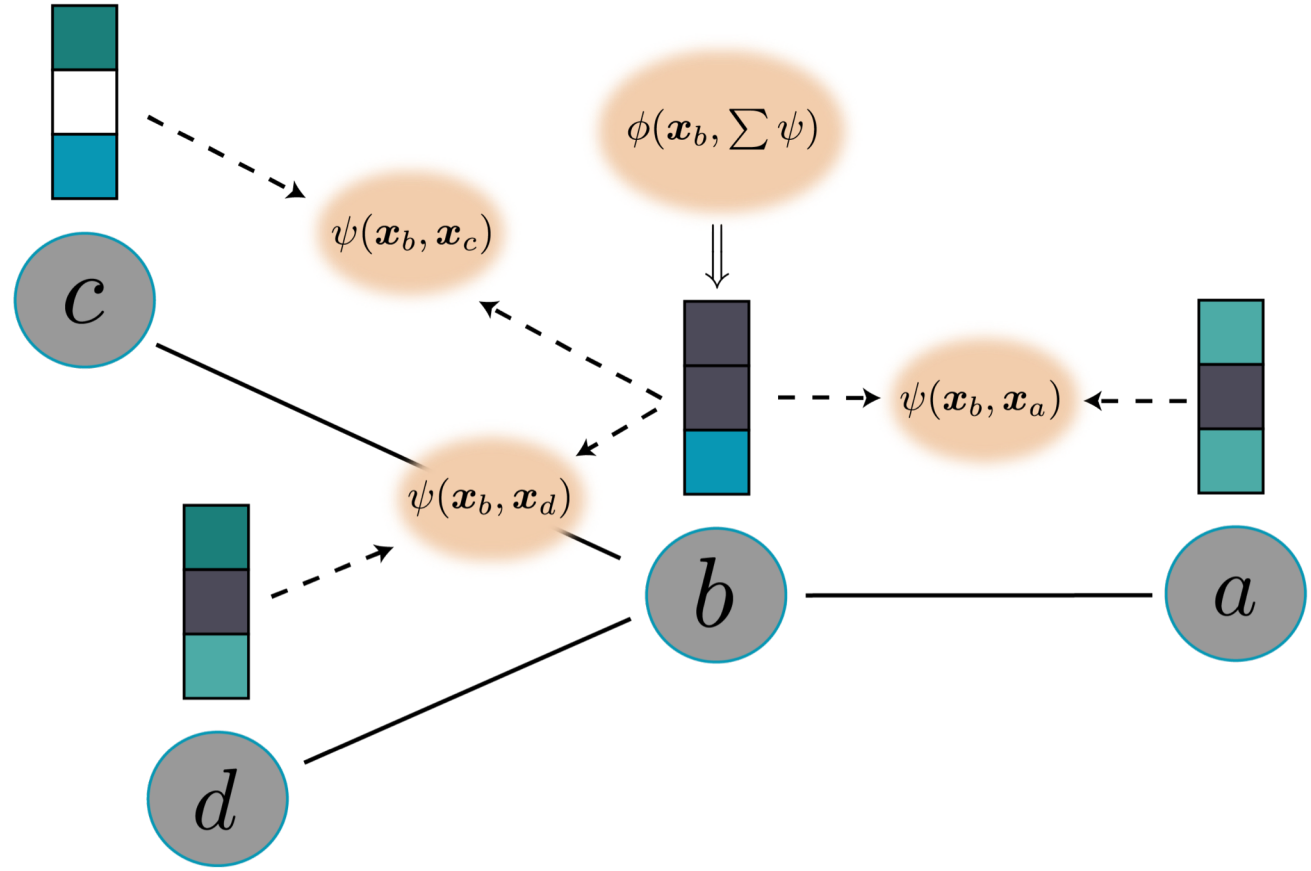
If edges were instead typed, could we learn representations which respect this heterogeneity?

Qiu, Jiezhong, et al. "Network embedding as matrix factorization: Unifying deepwalk, line, pte, and node2vec." *Proceedings of the eleventh ACM international conference on web search and data mining*. 2018.

Graph Neural Networks

Given a graph $G = (V, E, \boldsymbol{A})$ with node features $\boldsymbol{x}_v = \boldsymbol{x}_v^{(0)} \in \mathbb{R}^{d_0}$ with d_0 channels:

$$\boldsymbol{x}_v^{(l+1)} = \underbrace{\phi\left(\boldsymbol{x}_v^{(l)}, \sum_{u \in \mathcal{N}(v)} \psi(\boldsymbol{x}_v^{(l)}, \boldsymbol{x}_u^{(l)})\right)}_{\text{“MPNN”}}$$



$\psi : \mathbb{R}^{d_l} \times \mathbb{R}^{d_l} \to \mathbb{R}^m$ is a message function.

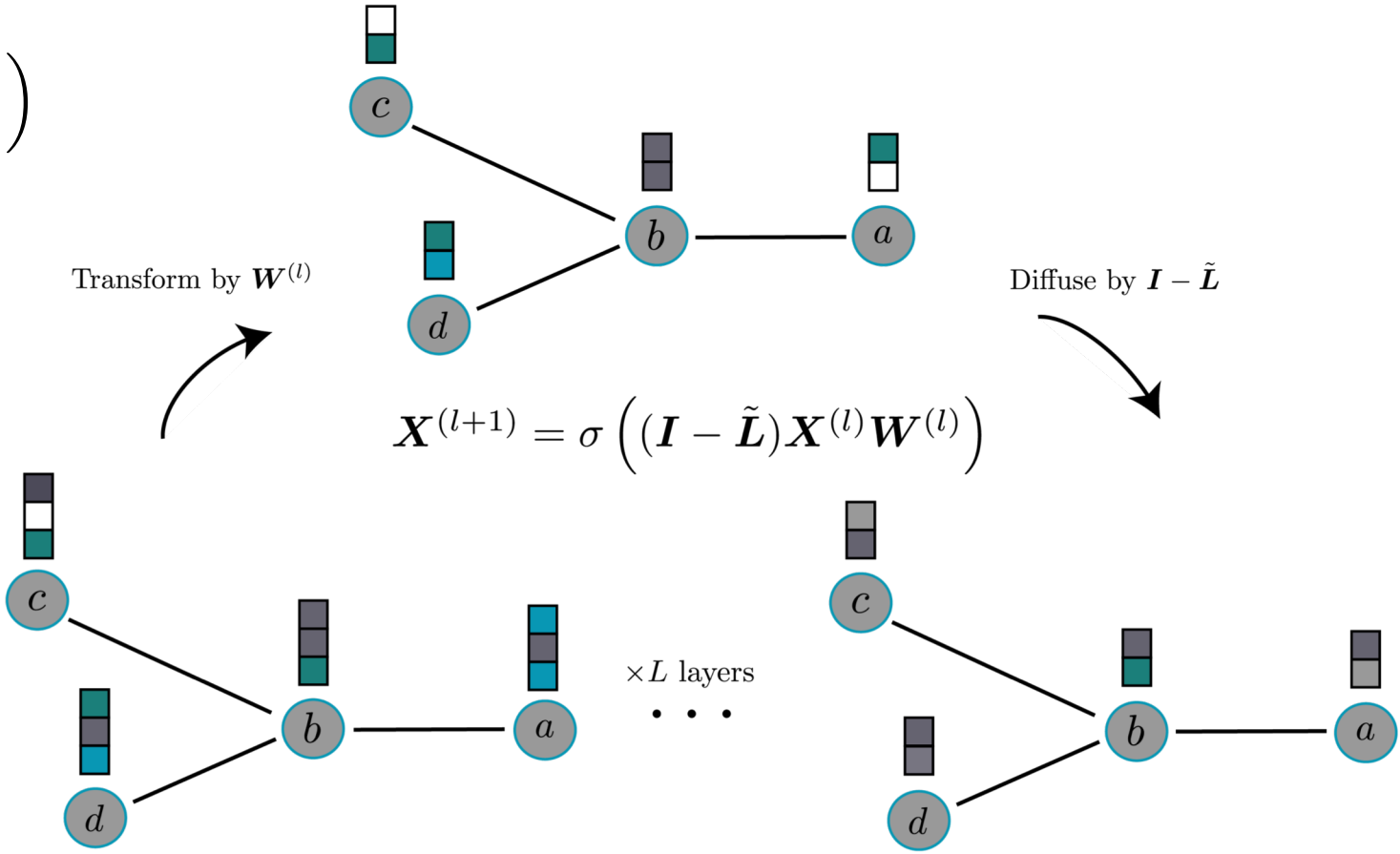
$\phi : \mathbb{R}^{d_l} \times \mathbb{R}^m \to \mathbb{R}^{d_{l+1}}$ is a readout/update function.

Graph Convolutional Networks (GCN)

$$\begin{aligned}\boldsymbol{X}^{(l+1)} &= \sigma\left(\boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2}\boldsymbol{X}^{(l)}\boldsymbol{W}^{(l)}\right)\\ &= \sigma\left(\tilde{\boldsymbol{A}}\boldsymbol{X}^{(l)}\boldsymbol{W}^{(l)}\right)\\ &= \sigma\left((\boldsymbol{I}-\tilde{\boldsymbol{L}})\boldsymbol{X}^{(l)}\boldsymbol{W}^{(l)}\right)\end{aligned}$$

$\boldsymbol{D} = \boldsymbol{A}\mathbf{1}$ is diagonal degree matrix.

$\tilde{\boldsymbol{L}}$ is the graph Laplacian.



Repeated application of $(\boldsymbol{I}-\tilde{\boldsymbol{L}})$ minimizes:

$$\begin{aligned}\epsilon(\mathbf{x}_i, G) &= \mathbf{x}_i^\top \tilde{\boldsymbol{L}}\mathbf{x}_i\\ &= \sum_{(u,v)=e\in E} \tilde{\boldsymbol{A}}_{u,v}(\boldsymbol{X}_{u,i}-\boldsymbol{X}_{v,i})^2\end{aligned}$$

} *Dirichlet energy*

Graph Convolutional Networks

Fixing $\boldsymbol{W}^{(l)} = \boldsymbol{I}$ and stacking GCN layers performs gradient descent on $\epsilon(G, \mathbf{x}_i)$.

A deep GCN of this form will "learn" smooth node features: a degree-weighted local averaging dictated by the graph topology.

The features will approach $\ker \boldsymbol{L}$ (constant).

The graph Laplacian has many interpretations.

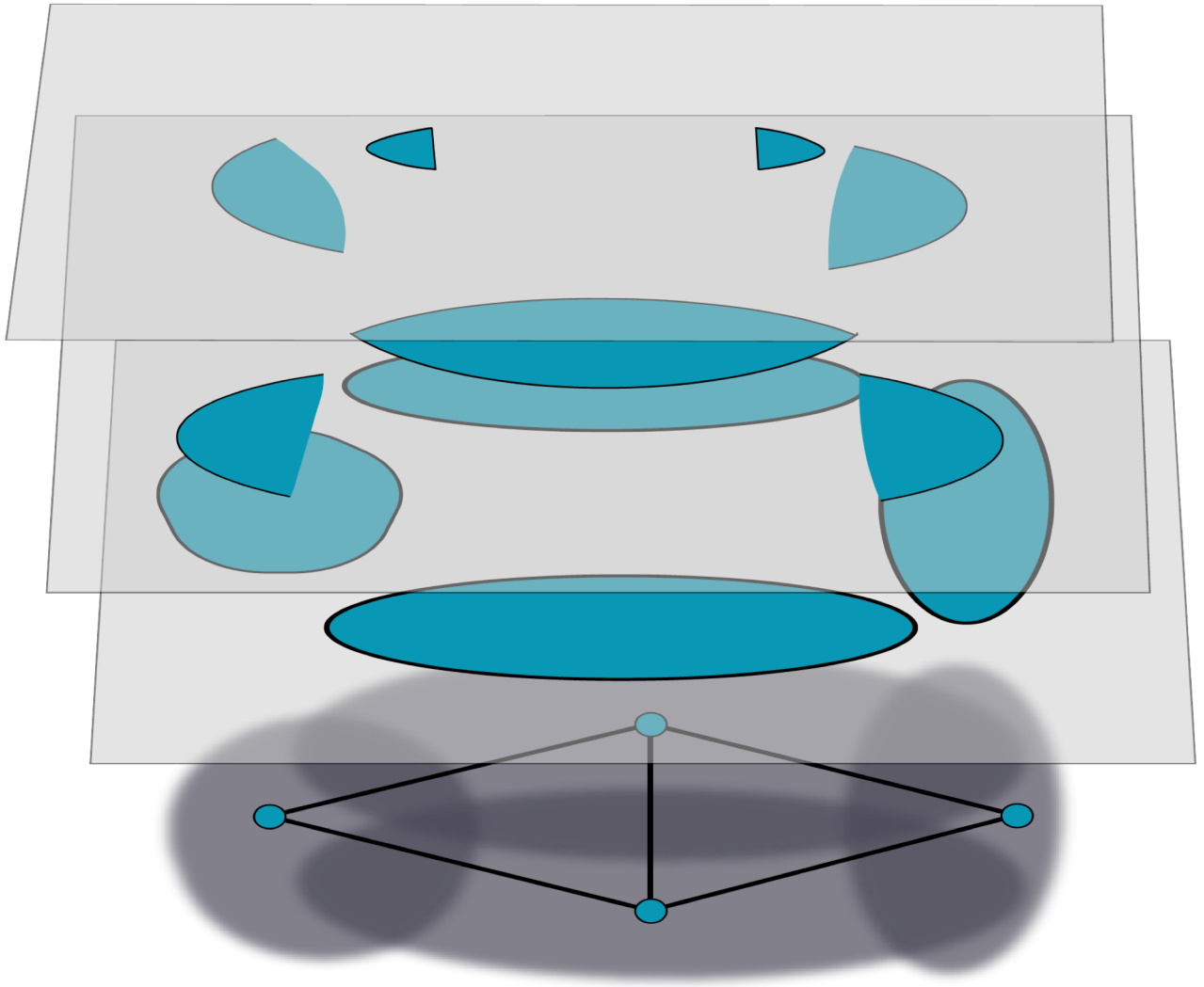
$$\tilde{\boldsymbol{L}} = \boldsymbol{I} - \boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2}$$

$$\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{A} = \boldsymbol{\delta}^\top \boxed{\boldsymbol{\delta}}$$ coboundary / incidence matrix

$$\boldsymbol{L}\boldsymbol{x} = \sum_{(u,v)=e\in E} \boldsymbol{A}_{u,v}(\boldsymbol{x}_u - \boldsymbol{x}_v)$$

Can we derive a deep architecture whose message passing avoids constant/flat functions?

Sheaves



Primordial ooze

Level 1

A (cellular) sheaf is a mathematical specification for associating data to a graph.

Level 2

A sheaf is a system of coefficients for computing cohomology.

Level 3

A sheaf is an equalizer of a set of constraints imposed by an open cover.

Level 4

A sheaf is a functor that satisfies some gluing axioms expressed by limits.

Level 4: Categorical Definition

SHEAVES, COSHEAVES AND APPLICATIONS

Justin Michael Curry

A DISSERTATION
in
Mathematics

Presented to the Faculties of
The University of Pennsylvania
in Partial Fulfillment of the Requirements for
the Degree of

Doctor of Philosophy

2014

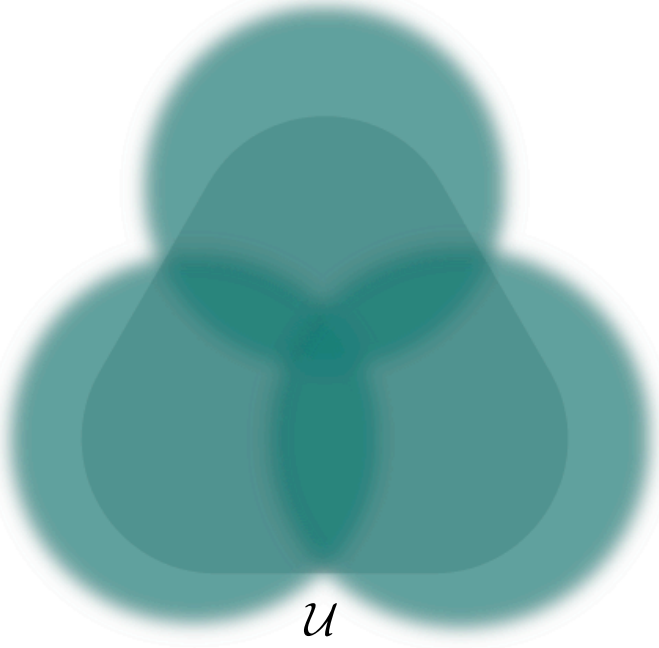
Robert W. Ghrist, Andrea Mitchell University Professor

Categorical Definition

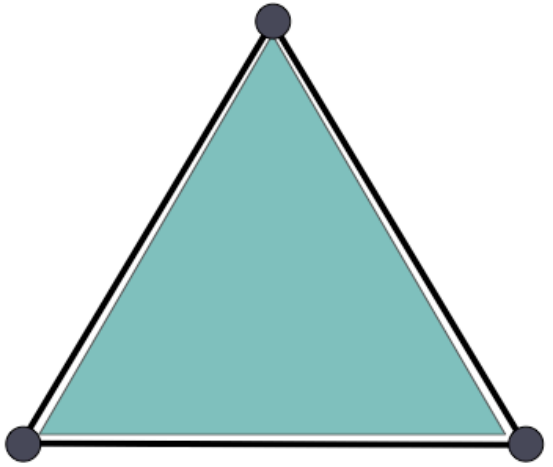
Let $\mathcal{U} = \{U_i\}$ be an open cover of U in topological space X.

The *nerve* $\mathcal{N}(\mathcal{U})$ is a simplicial complex whose elements are subsets $I = \{i_0, \ldots, i_n\}$ where $U_I = U_{i_0} \cap \cdots \cap U_{i_n} \neq \emptyset$.

Define the functor $\iota_{\mathcal{U}}^{\mathrm{op}} : \mathcal{N}(\mathcal{U})^{\mathrm{op}} \to \mathbf{Open}(X)^{\mathrm{op}}$.

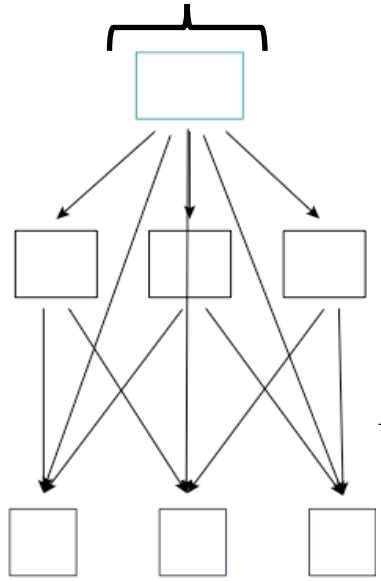


$\mathcal{U}$



$\mathcal{N}(\mathcal{U})$

The nerve organizes into a category



$I \to J$ if $J \subseteq I$

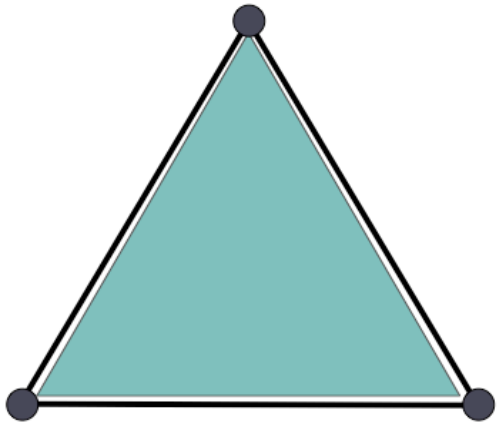
Categorical Definition

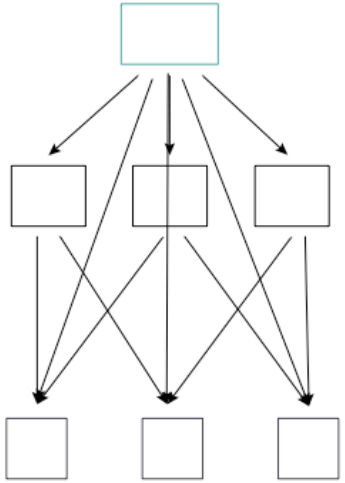
A *pre-sheaf valued in* $\mathbf{D}$ is a functor $\mathcal{F} : \mathbf{Open}(X)^{\mathrm{op}} \to \mathbf{D}$ with *restriction maps* $\mathcal{F}_{V,U} : \mathcal{F}(V) \to \mathcal{F}(U)$ between open sets $U \subset V$ as morphisms.

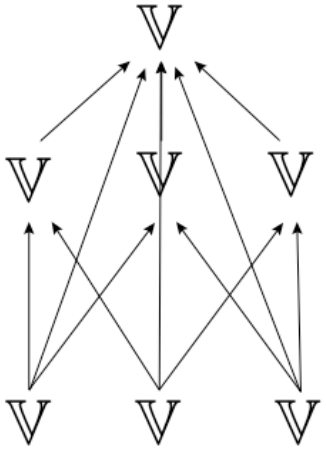
The pre-sheaf $\mathcal{F}$ is a *sheaf on* $\mathcal{U}$ if the following unique map from $\mathcal{F}(U)$ to the limit of $\mathcal{F} \circ \iota_{\mathcal{U}}^{\mathrm{op}}$ is an isomorphism:

$$\mathcal{F}(U) \to \varprojlim_{I \in \mathcal{N}(\mathcal{U})} \mathcal{F}(U_I)$$

$\mathbf{D}$ must be complete.







Level 3: Sheaves as Equalizers

Note that $\varprojlim \mathcal{F}(U) \hookrightarrow \prod \mathcal{F}(U_i)$: any cone must factor through each vertex.

Thus, a pre-sheaf is a sheaf if the following diagram is an equalizer:

$$\mathcal{F}(U) \xrightarrow{e} \prod \mathcal{F}(U_i) \underset{f^+}{\overset{f^-}{\rightrightarrows}} \prod_{i \sim j} \mathcal{F}(U_i \cap U_j)$$

$$e_i = \mathcal{F}_{U,U_i} \qquad f^-_{ij} = \mathcal{F}_{i,ij} \circ \pi_i \qquad f^+_{ij} = \mathcal{F}_{j,ij} \circ \pi_j$$

set function equalizer	**fiber**	**kernel**
Letting $\prod \mathcal{F}(U_i) = A$, $\prod_{i\sim j} \mathcal{F}(U_i \cap U_j) = B$	Letting $\prod \mathcal{F}(U_i) = A$, $\prod_{i\sim j} \mathcal{F}(U_i \cap U_j) = B$, $f^+(a) = b, \forall a \in A$	Letting $\prod \mathcal{F}(U_i) = \mathbb{V}$, $\prod_{i\sim j} \mathcal{F}(U_i \cap U_j) = \mathbb{W}$
$\mathrm{eq}(f^+, f^-) =$ $\{a \in A \mid f^+(a) = f^-(a)\}$	$\mathrm{eq}(f^+, f^-) =$ $(f^-)^{-1}(b) \subseteq A$	$\mathrm{eq}(f^+, f^-) =$ $\ker(f^+ - f^-) = \ker \delta$

Recap

Sheaves assign data to a space in a way which agrees with its topology.

$$\mathcal{F} : \mathbf{Open}(X)^{\mathrm{op}} \to \mathbf{D}$$

This consistent data assignment sits within the product of the nerve vertices.

$$\varprojlim \mathcal{F}(U) \hookrightarrow \prod \mathcal{F}(U_i)$$

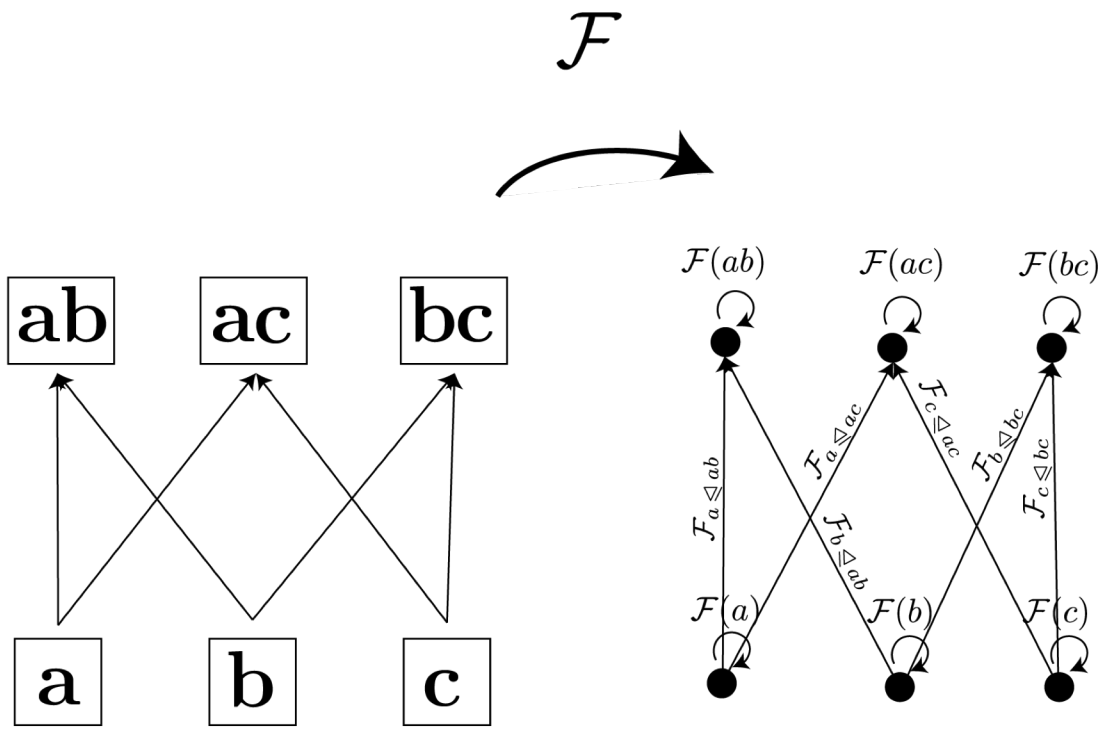
Sheaves are equalizers of the diagram taking the product of vertices to the product of their pairwise intersections under individual restriction onto their intersections.

$$\mathcal{F}(U) \xrightarrow{e} \prod \mathcal{F}(U_i) \underset{f^+}{\overset{f^-}{\rightrightarrows}} \prod_{i \sim j} \mathcal{F}(U_i \cap U_j)$$

When **D** is **Vect**, we can compute sheaves as kernels.

$$\mathcal{F}(U) \cong \ker(f^+ - f^-) = \ker \delta$$

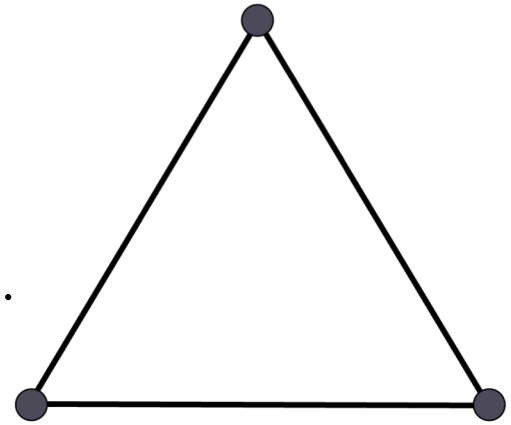
Cellular Sheaves

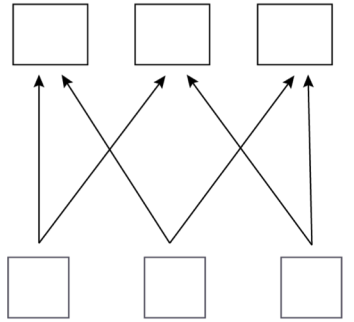


Graph Topology

We are already familiar with how a graph $G = (V, E)$ can be encoded as a cell complex.

Vertices $v \in V$ are 0-cells, edges $e \in E$ are 1-cells.



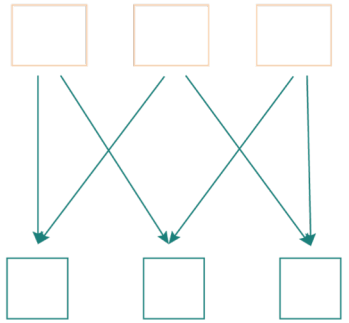


Generate an open cover from the star $\mathrm{st}(\sigma)$ of each cell σ.

Product space derived from the product of vertex representations $\prod_{v \in V} \mathcal{F}(v)$.

Alexandrov topology

Restriction to edges $\prod_{e \in E} \mathcal{F}(e)$ is cover intersection.



f^+, f^-, δ map vertices to edges according to incidence.

Towards Applications

To define a sheaf, we require:

	Within machine learning, these are often:
A topological space X	Given
An open set $U \subseteq X$ and an open cover $\mathcal{U}$ on U.	Ignored
A data category $\mathbf{D}$	Assumed **FVect**
Restriction maps $\mathcal{F}_{V,U} : V \to U$	Ignored/Trivial

By treating graphs as cell complexes endowed with the Alexandrov topology, cellular sheaves can provide a model for representation learning on graphs.

Sheaves on Graphs

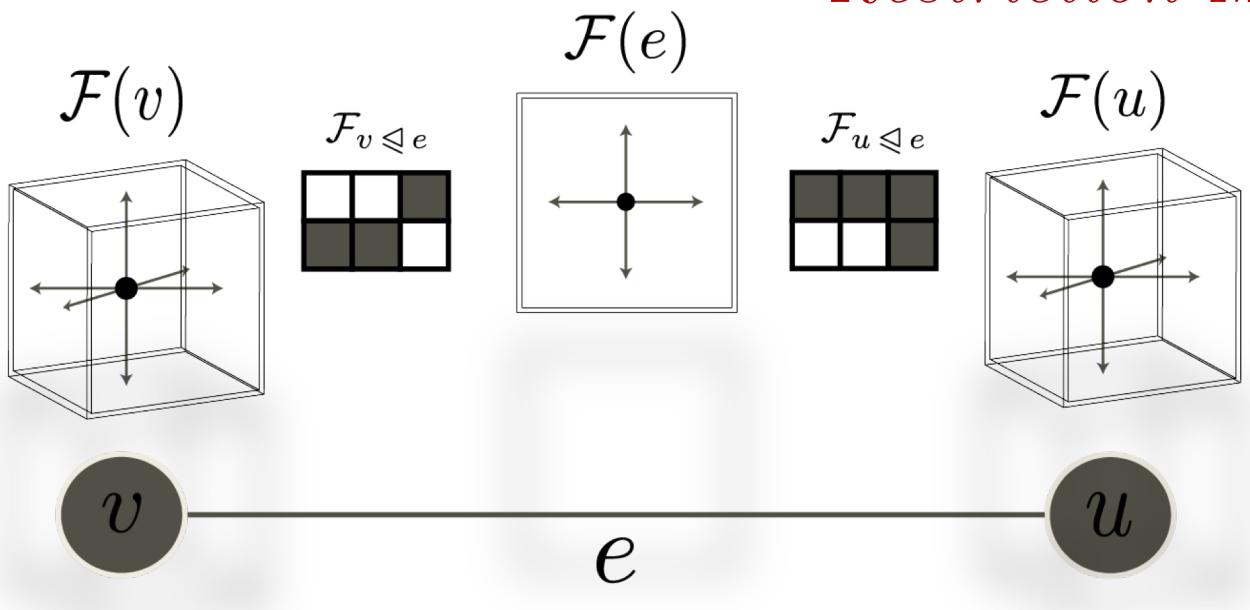
A cellular sheaf $\mathcal{F}$ on a graph $G = (V, E)$ consists of the following data:

- a vector space $\mathcal{F}(v)$ for each vertex $v \in V$,
- a vector space $\mathcal{F}(e)$ for each edge $e \in E$,

} *Stalks*

- a linear map $\boldsymbol{\mathcal{F}}_{v \trianglelefteq e} : \mathcal{F}(v) \to \mathcal{F}(e)$ for each incident vertex-edge pair $v \trianglelefteq e$ of G.

Restriction Maps



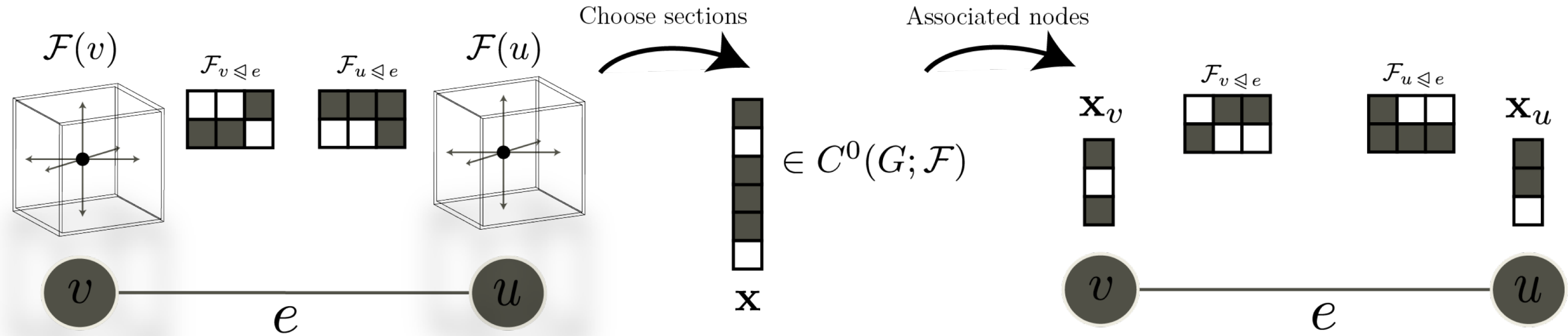
Sheaves on Graphs

Define the $\mathcal{F}$-valued space of signals on vertices and edges, respectively by:

$$C^0(G;\mathcal{F}) = \bigoplus_{v \in V} \mathcal{F}(v) \qquad \text{Space of 0-cochains}$$

$$C^1(G;\mathcal{F}) = \bigoplus_{e \in E} \mathcal{F}(e) \qquad \text{Space of 1-cochains}$$

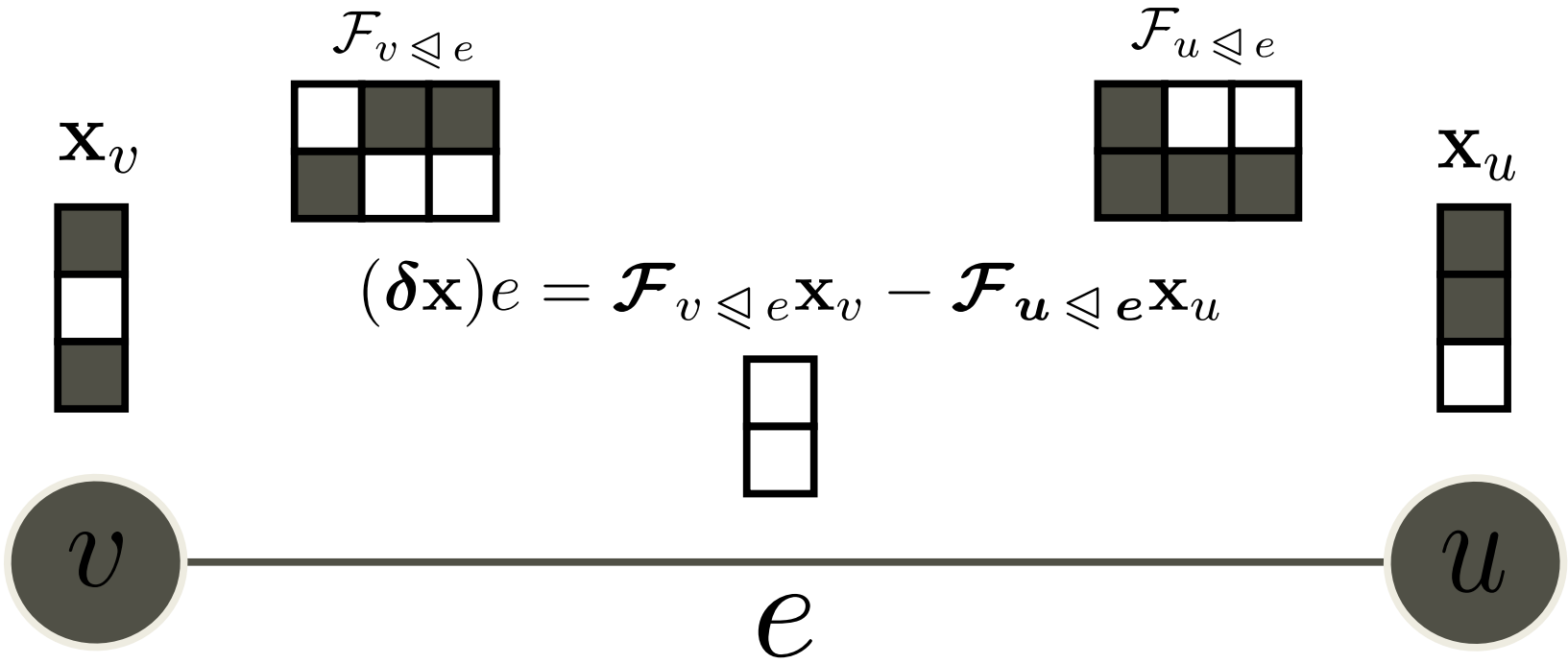
For $\mathcal{F}$ vector-valued, we may view $\mathbf{x} \in C^0(G;\mathcal{F})$ as a concatenation of vectors.



Sheaves on Graphs

The coboundary map $\delta : C^0(G;\mathcal{F}) \to C^1(G;\mathcal{F})$ computes along edge $e = (u, v)$:

$$(\delta \mathbf{x})_e = \underbrace{\boldsymbol{\mathcal{F}}_{v \trianglelefteq e}}_{f^+} \mathbf{x}_v - \underbrace{\boldsymbol{\mathcal{F}}_{u \trianglelefteq e}}_{f^-} \mathbf{x}_u$$



Sheaves on Graphs

The space of globally-consistent assignments of data to the vertices of G is isomorophic to:

$$H^0(G;\mathcal{F}) = \ker \delta \subset C^0(G;\mathcal{F})$$

Thus, given restriction maps $\mathcal{F}_{v \trianglelefteq e}$ for each vertex-edge pair, data representations consistent with the topology of G may be computed by computing $\ker \delta$.

$\delta : C^0(G;\mathcal{F}) \to C^1(G;\mathcal{F})$ takes signals on nodes to (oriented) signals on edges.

If we mapped these signals which have been transformed by restriction maps back down to the nodes, would this define a gradient operator which performs diffusion/message passing?

Sheaves on Graphs

Given coboundary $\boldsymbol{\delta}$ we may define the *sheaf Laplacian* $\boldsymbol{L}_{\mathcal{F}}$ as:

$$\boldsymbol{L}_{\mathcal{F}} = \boldsymbol{\delta}^{\top}\boldsymbol{\delta}$$

$$(\boldsymbol{L}_{\mathcal{F}})_{u,v} = \begin{pmatrix} \boldsymbol{\mathcal{F}}_{v \trianglelefteq e}^{\top}\boldsymbol{\mathcal{F}}_{v \trianglelefteq e} & -\boldsymbol{\mathcal{F}}_{v \trianglelefteq e}^{\top}\boldsymbol{\mathcal{F}}_{u \trianglelefteq e} \\ -\boldsymbol{\mathcal{F}}_{u \trianglelefteq e}^{\top}\boldsymbol{\mathcal{F}}_{v \trianglelefteq e} & \boldsymbol{\mathcal{F}}_{u \trianglelefteq e}^{\top}\boldsymbol{\mathcal{F}}_{u \trianglelefteq e} \end{pmatrix}$$

$$(\boldsymbol{L}_{\mathcal{F}}\mathbf{x})_{v} = \sum_{u,v \trianglelefteq e} \boldsymbol{\mathcal{F}}_{v \trianglelefteq e}^{\top}(\boldsymbol{\mathcal{F}}_{v \trianglelefteq e}\mathbf{x}_{v} - \boldsymbol{\mathcal{F}}_{u \trianglelefteq e}\mathbf{x}_{u})$$

$$\mathbf{x}^{\top}\boldsymbol{L}_{\mathcal{F}}\mathbf{x} = \sum_{u,v \trianglelefteq e} \left\|\boldsymbol{\mathcal{F}}_{v \trianglelefteq e}\mathbf{x}_{v} - \boldsymbol{\mathcal{F}}_{u \trianglelefteq e}\mathbf{x}_{u}\right\|^{2} = E(\mathbf{x}, \mathcal{F})$$

$E(\mathbf{x}, \mathcal{F}) = \mathbf{x}^{\top}\boldsymbol{L}_{\mathcal{F}}\mathbf{x} = 0$ implies $\mathbf{x} \in H^{0}(G; \mathcal{F})$.

Toward a spectral theory of cellular sheaves

Jakob Hansen[1] · **Robert Ghrist**[1,2]

Received: 4 August 2018 / Accepted: 16 August 2019 / Published online: 30 August 2019

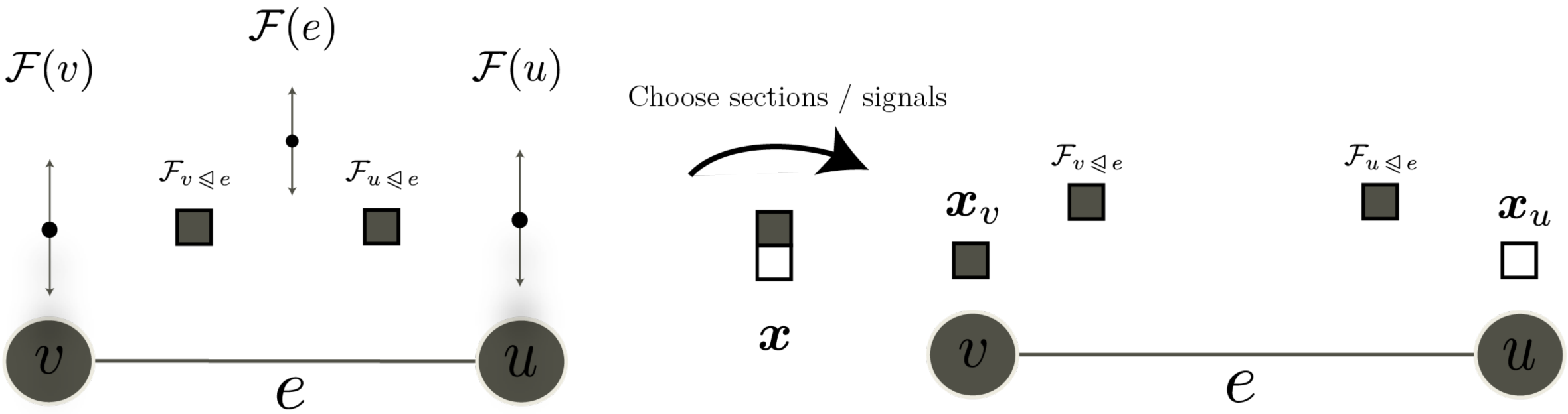
Abstract
This paper outlines a program in what one might call *spectral sheaf theory*-

Hansen, Jakob, and Robert Ghrist. "Toward a spectral theory of cellular sheaves." *Journal of Applied and Computational Topology* 3.4 (2019).

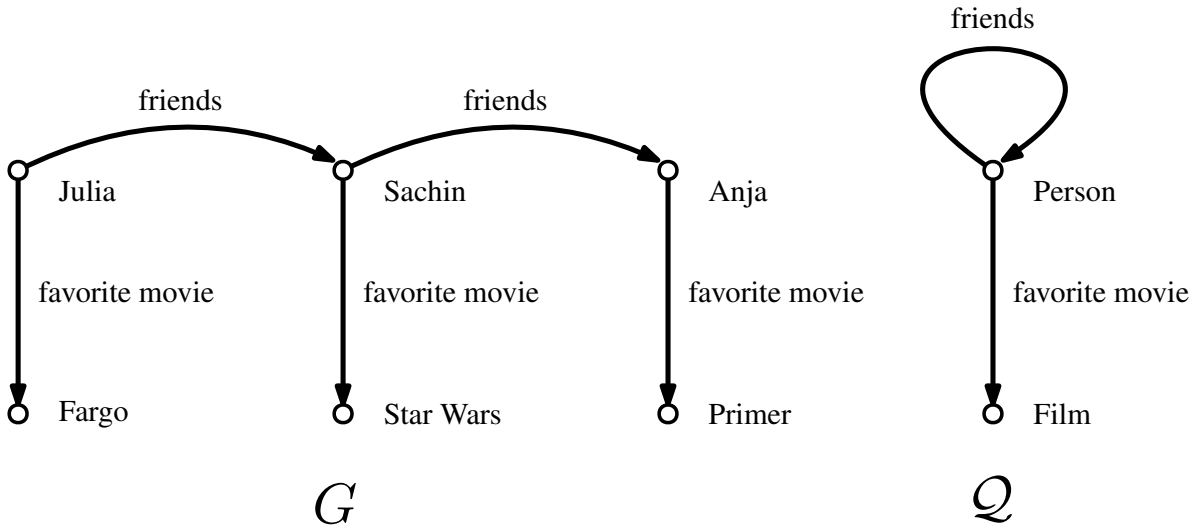
Graph Laplacian

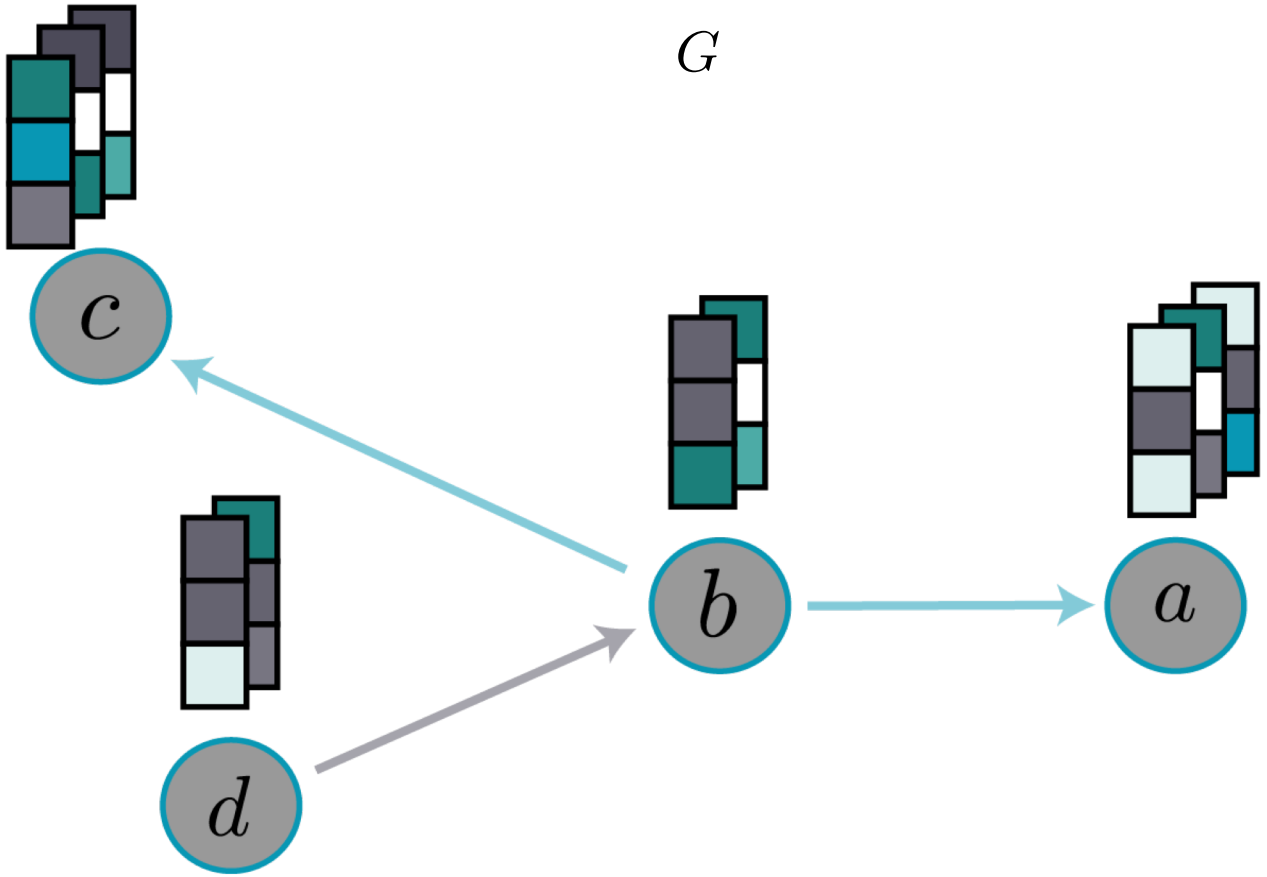
We can recover the graph Laplacian $\boldsymbol{L}_G$ in this sheaf-theoretic language.

Set $\mathcal{F}(v) = \mathbb{R}$ for all vertices $v \in V$ and choose restriction maps such that $\boldsymbol{\mathcal{F}}_{v \trianglelefteq e}^{\top} \boldsymbol{\mathcal{F}}_{u \trianglelefteq e} = \boldsymbol{A}_{u,v}$ for all edges $e = (u, v)$.



Applications





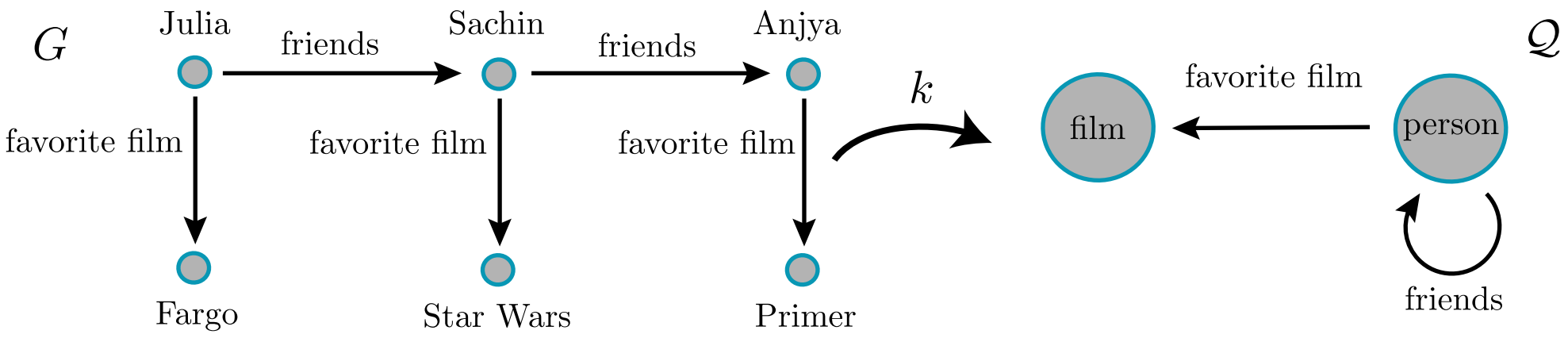
Knowledge Graph Embedding

A *knowledge graph* G is a typed graph composed of entities (vertices) and relations (edges).

Let $\mathcal{S}$ be a set of entity types and $\mathcal{R}$ a set of relations. Suppose that each $r \in \mathcal{R}$ may hold between an entity of type $\mathfrak{h}(r) \in \mathcal{S}$ and an entity of type $\mathfrak{t}(r) \in \mathcal{S}$. The tuple $\mathcal{Q} = (\mathcal{S}, \mathcal{R}, \mathfrak{h}, \mathfrak{t})$ is a *knowledge schema* which may be viewed as a directed multigraph.

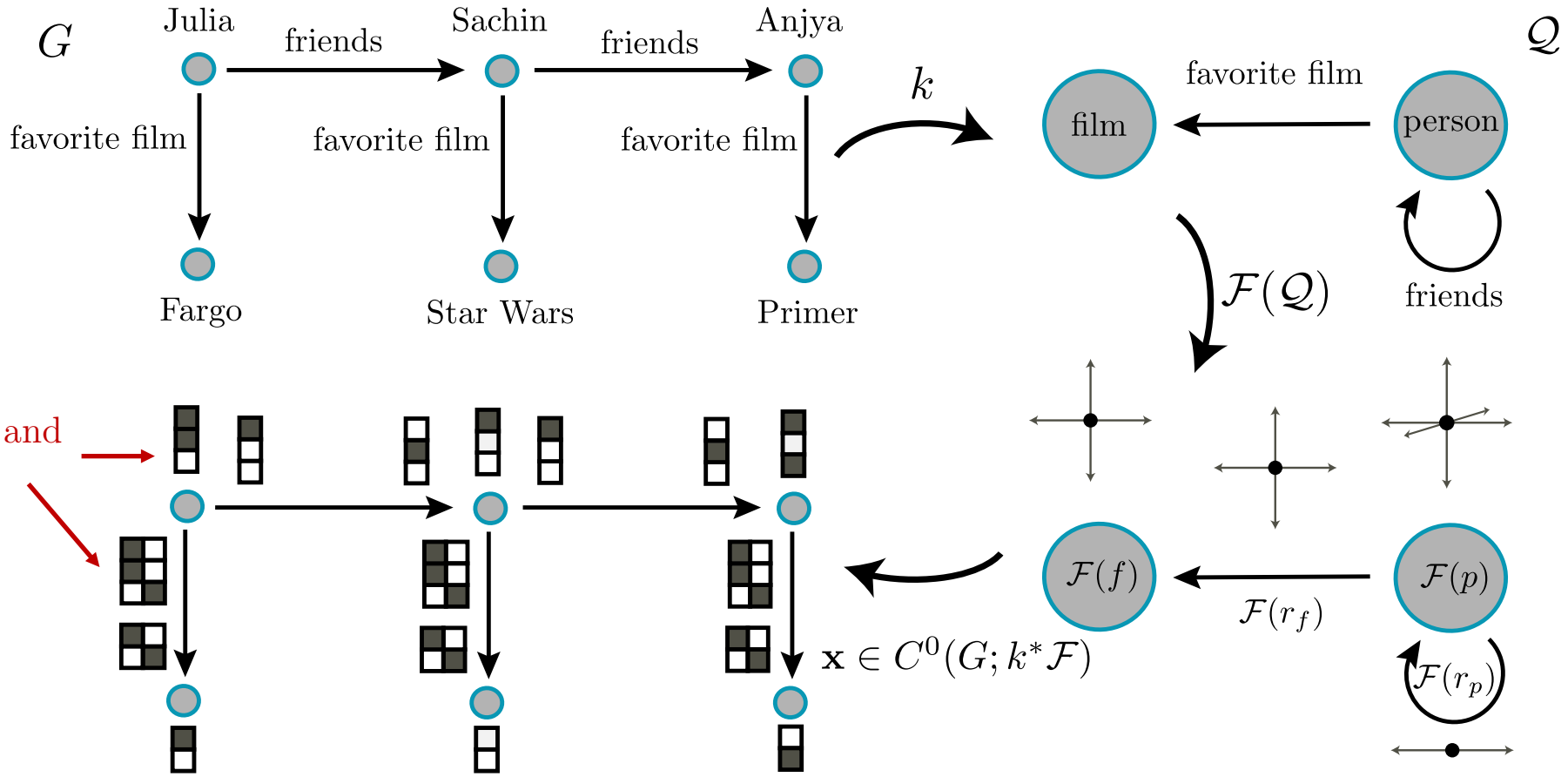
We seek embeddings of G, instantiating $\mathcal{Q}$, which can be used for e.g. link prediction.

Let $k : G \to \mathcal{Q}$ be a graph morphism which instantiates G with typing.



Knowledge Graph Embedding

Given a graph morphism $k : G \to \mathcal{Q}$ instantiating a knowledge graph G from a schema $\mathcal{Q}$, a *knowledge sheaf embedding* of G is a sheaf $\mathcal{F}$ on $\mathcal{Q}$ together with a 0-cochain $\mathbf{x} \in C^0(G; k^*\mathcal{F})$.

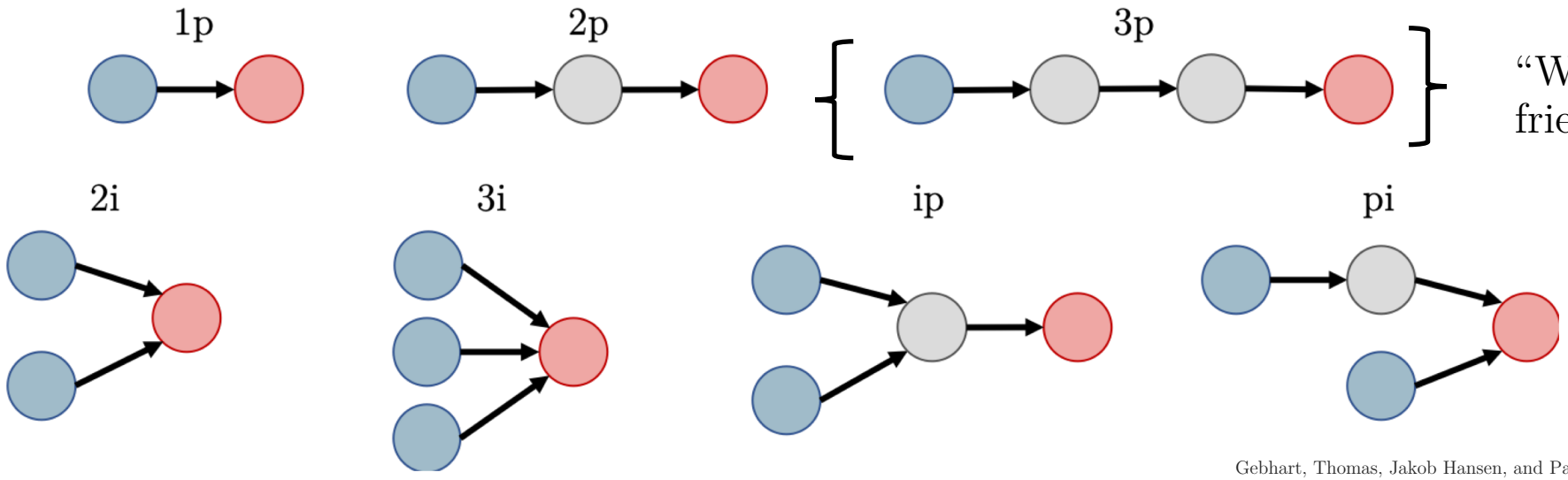


Knowledge Graph Embedding

Many of the most popular knowledge graph embedding methods (TransE, SE, RotatE) are implicitly learning sheaf representations.

These methods use contrastive learning to find entity and relation representations which are as consistent as possible *with respect to the typing schema.*

The sheaf Laplacian allows one to solve boundary value problems on the knowledge graph via harmonic extension, providing a method for reasoning over conjunctive queries.



"What is Julia's friend's friend's favorite movie?"

Gebhart, Thomas, Jakob Hansen, and Paul Schrater. "Knowledge Sheaves: A Sheaf-Theoretic Framework for Knowledge Graph Embedding." *International Conference on Artificial Intelligence and Statistics*. PMLR, 2023.

Sheaf Neural Networks

Can we learn representations of sheaf-valued signals to e.g. classify 0-cells (vertices) of a cell complex (graph)? We seek:

$$f^{\mathrm{SCN}} : C^0(G;\mathcal{F}) \to C^0(G;\mathcal{F}')$$

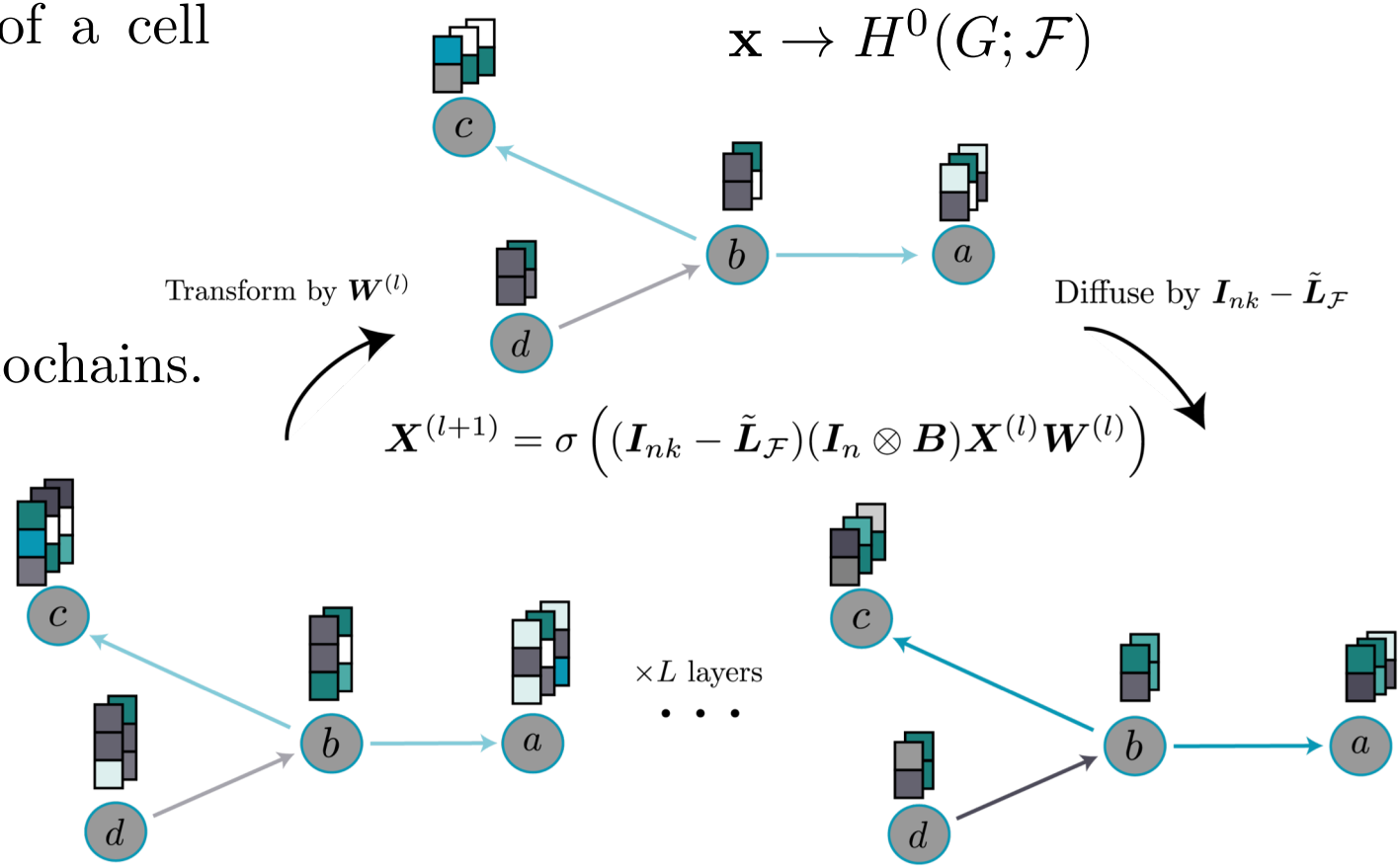
Let $\boldsymbol{X} \in \mathbb{R}^{nk \times d}$ be d, k-dimensional 0-cochains.

A sheaf convolutional network layer:

$$f^{\mathrm{SCN}}(\boldsymbol{X}) = \sigma\left((\boldsymbol{I}_{nk} - \tilde{\boldsymbol{L}}_{\mathcal{F}})(\boldsymbol{I}_n \otimes \boldsymbol{B})\boldsymbol{X}\boldsymbol{W}\right)$$

$\tilde{\boldsymbol{L}}_{\mathcal{F}}$: normalized sheaf Laplacian.

$\boldsymbol{B}, \boldsymbol{W}$: learnable parameters.



Repeated application of $\boldsymbol{I}_{nk} - \tilde{\boldsymbol{L}}_{\mathcal{F}}$ smoothes each co-chain with respect to the *sheaf structure*: more expressive, non-flat representations are learnable.

Sheaf Neural Networks

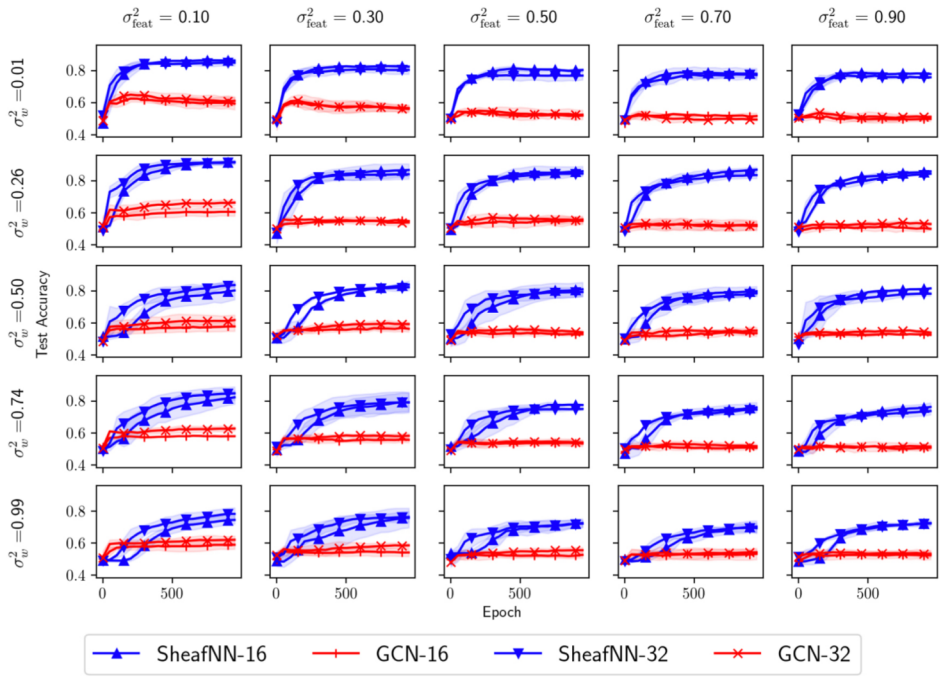
GCN	SCN
$f^{\mathrm{GCN}} : \mathcal{X}(G, \mathbb{R}^{d_{l-1}}) \to \mathcal{X}(G, \mathbb{R}^{d_l})$	$f^{\mathrm{SCN}} : C^0(G; \mathcal{F}) \to C^0(G; \mathcal{F}')$
$f^{\mathrm{GCN}}(\boldsymbol{X}) = \sigma\left((\boldsymbol{I} - \tilde{\boldsymbol{L}}_G)\boldsymbol{X}\boldsymbol{W}\right)$	$f^{\mathrm{SCN}}(\boldsymbol{X}) = \sigma\left((\boldsymbol{I}_{nk} - \tilde{\boldsymbol{L}}_{\mathcal{F}})(\boldsymbol{I}_n \otimes \boldsymbol{B})\boldsymbol{X}\boldsymbol{W}\right)$
$\dot{\boldsymbol{X}}(t+1) = -\tilde{\boldsymbol{L}}_G \boldsymbol{X}(t)$	$\dot{\boldsymbol{X}}(t+1) = -\tilde{\boldsymbol{L}}_{\mathcal{F}} \boldsymbol{X}(t)$
$\epsilon(\mathbf{x}, \boldsymbol{A}) = \mathbf{x}^\top \tilde{\boldsymbol{L}}_G \mathbf{x} = \sum_{(u,v)=e} \boldsymbol{A}_{u,v}(\mathbf{x}_v - \mathbf{x}_u)^2$	$E(\mathbf{x}, \mathcal{F}) = \mathbf{x}^\top \tilde{\boldsymbol{L}}_{\mathcal{F}} \mathbf{x} = \sum_{u,v \trianglelefteq e} \lVert \boldsymbol{\mathcal{F}}_{v \trianglelefteq e} \boldsymbol{D}_v^{-1/2} \mathbf{x}_v - \boldsymbol{\mathcal{F}}_{u \trianglelefteq e} \boldsymbol{D}_u^{-1/2} \mathbf{x}_u \rVert^2$

Sheaf Neural Networks

SCNs learn representations of sheaf-valued signals, and can also be used to improve expressivity of graph representations.

Bodnar et al. recently showed that by varying the subspace from which restriction maps are chosen/learned, one can increase the separation power of sheaf diffusion.

SCNs learn on signed graphs



and can learn heterogeneous representations:

Table 1: Results on node classification datasets sorted by their homophily level. Top three models are coloured by **First**, **Second**, **Third**. Our models are marked **NSD**.

	Texas	Wisconsin	Film	Squirrel	Chameleon	Cornell	Citeseer	Pubmed	Cora
Hom level	**0.11**	**0.21**	**0.22**	**0.22**	**0.23**	**0.30**	**0.74**	**0.80**	**0.81**
#Nodes	183	251	7,600	5,201	2,277	183	3,327	18,717	2,708
#Edges	295	466	26,752	198,493	31,421	280	4,676	44,327	5,278
#Classes	5	5	5	5	5	5	7	3	6
Diag-NSD	**85.67**±6.95	**88.63**±2.75	**37.79**±1.01	**54.78**±1.81	**68.68**±1.73	**86.49**±7.35	**77.14**±1.85	89.42±0.43	87.14±1.06
O(d)-NSD	**85.95**±5.51	**89.41**±4.74	**37.81**±1.15	**56.34**±1.32	**68.04**±1.58	**84.86**±4.71	76.70±1.57	**89.49**±0.40	86.90±1.13
Gen-NSD	82.97±5.13	**89.21**±3.84	**37.80**±1.22	53.17±1.31	67.93±1.58	**85.68**±6.51	76.32±1.65	89.33±0.35	87.30±1.15
GGCN	**84.86**±4.55	86.86±3.29	37.54±1.56	**55.17**±1.58	**71.14**±1.84	**85.68**±6.63	**77.14**±1.45	89.15±0.37	**87.95**±1.05
H2GCN	**84.86**±7.23	87.65±4.98	35.70±1.00	36.48±1.86	60.11±2.15	82.70±5.28	77.11±1.57	**89.49**±0.38	**87.87**±1.20
GPRGNN	78.38±4.36	82.94±4.21	34.63±1.22	31.61±1.24	46.58±1.71	80.27±8.11	77.13±1.67	87.54±0.38	**87.95**±1.18
FAGCN	82.43±6.89	82.94±7.95	34.87±1.25	42.59±0.79	55.22±3.19	79.19±9.79	N/A	N/A	N/A
MixHop	77.84±7.73	75.88±4.90	32.22±2.34	43.80±1.48	60.50±2.53	73.51±6.34	76.26±1.33	85.31±0.61	87.61±0.85
GCNII	77.57±3.83	80.39±3.40	37.44±1.30	38.47±1.58	63.86±3.04	77.86±3.79	**77.33**±1.48	**90.15**±0.43	**88.37**±1.25
Geom-GCN	66.76±2.72	64.51±3.66	31.59±1.15	38.15±0.92	60.00±2.81	60.54±3.67	**78.02**±1.15	**89.95**±0.47	85.35±1.57
PairNorm	60.27±4.34	48.43±6.14	27.40±1.24	50.44±2.04	62.74±2.82	58.92±3.15	73.59±1.47	87.53±0.44	85.79±1.01
GraphSAGE	82.43±6.14	81.18±5.56	34.23±0.99	41.61±0.74	58.73±1.68	75.95±5.01	76.04±1.30	88.45±0.50	86.90±1.04
GCN	55.14±5.16	51.76±3.06	27.32±1.10	53.43±2.01	64.82±2.24	60.54±5.30	76.50±1.36	88.42±0.50	86.98±1.27
GAT	52.16±6.63	49.41±4.09	27.44±0.89	40.72±1.55	60.26±2.50	61.89±5.05	76.55±1.23	87.30±1.10	86.33±0.48
MLP	80.81±4.75	85.29±3.31	36.53±0.70	28.77±1.56	46.21±2.99	81.89±6.40	74.02±1.90	87.16±0.37	75.69±2.00

Hansen, Jakob, and Thomas Gebhart. "Sheaf neural networks." *NeurIPS 2020 Workshop on Topological Data Analysis and Beyond* (2020).

Bodnar, Cristian, et al. "Neural sheaf diffusion: A topological perspective on heterophily and oversmoothing in gnns." *Advances in Neural Information Processing Systems* 35 (2022).

Questions?

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

Chanpuriya, Sudhanshu, and Cameron Musco. "Infinitewalk: Deep network embeddings as Laplacian embeddings with a nonlinearity." *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2020.

Curry, Justin Michael. *Sheaves, cosheaves and applications*. University of Pennsylvania, (2014).

Hansen, Jakob, and Robert Ghrist. "Toward a spectral theory of cellular sheaves." *Journal of Applied and Computational Topology* 3.4 (2019).

Gebhart, Thomas, Jakob Hansen, and Paul Schrater. "Knowledge Sheaves: A Sheaf-Theoretic Framework for Knowledge Graph Embedding." *International Conference on Artificial Intelligence and Statistics*. PMLR, 2023.

Hansen, Jakob, and Thomas Gebhart. "Sheaf neural networks." *NeurIPS 2020 Workshop on Topological Data Analysis and Beyond* (2020).

Bodnar, Cristian, et al. "Neural sheaf diffusion: A topological perspective on heterophily and oversmoothing in gnns." *Advances in Neural Information Processing Systems* 35 (2022).