AUGUST 6 - 10

KDD2023

LONG BEACH, CA

29TH ACM SIGKDD CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING

Hyperbolic Graph Neural Networks: A Tutorial of Methods and Applications

Min Zhou[1] , Menglin Yang[2], Bo Xiong[3], Hui Xiong [4], Irwin King[2]

[1]Huawei Cloud [2] The Chinese University of Hong Kong

[3] University of Stuttgart [4] HKUST (Guangzhou)

https://hyperbolicgnn.github.io/

Min Zhou

Huawei Cloud

Menglin Yang

The Chinese University of Hong Kong

Bo Xiong

University of Stuttgart

Hui Xiong

HKUST (Guangzhou)

Irwin King

The Chinese University of Hong Kong

Slack channel

- Join our slack channel for Q&A

https://hyperbolicgnn.github.io/

Agenda

Part 1. INTRODUCTION (Min, 9:00 - 9:30)

1.1 An Overview of Graph Learning

1.2 Brief Introduction of Riemannian Geometry

1.3 Motivation for Hyperbolic Graph Learning

Coffee Break (9:30 – 10:00)

Part 2. HYPERBOLIC GRAPH NETWORKS (Menglin, 10:00 – 10:30)

2.1 Preliminary

2.2 Hyperbolic Shallow Models

2.2 Hyperbolic Neural Networks

2.3 Hyperbolic Graph Neural Networks

Part 3. APPLICATIONS (Menglin, 10:30 – 11:00)

3.1 HGNN for Recommendation Systems

3.2 HGNN for Knowledge Graph

3.3 HGNN for Bioinformatics

Part 4. ADVANCED TOPICS (Bo, 11:00 – 12:00)

4.1 Complex Structures

4.2 Curvature-aware Learning

4.3 Evolving Interactions

4.4 Trustworthy and Scalability

Part 1. Introduction

Agenda

Part 1. INTRODUCTION (Min, 9:00 - 9:30)

1.1 An Overview of Graph Learning

1.2 Brief Introduction of Riemannian Geometry

1.3 Motivation for Hyperbolic Graph Learning

Coffee Break (9:30 – 10:000)

Part 2. HYPERBOLIC GRAPH NETWORKS (Menglin, 10:00 – 10:30)

2.1 Preliminary

2.2 Hyperbolic Shallow Models

2.2 Hyperbolic Neural Networks

2.3 Hyperbolic Graph Neural Networks

Part 3. APPLICATIONS (Menglin, 10:30 – 11:00)

3.1 HGNN for Recommendation Systems

3.2 HGNN for Knowledge Graph

3.3 HGNN for Healthcare

Part 4. ADVANCED TOPICS (Bo, 11:00 – 12:00)

4.1 Complex Structures

4.2 Curvature-aware Learning

4.3 Evolving Interactions

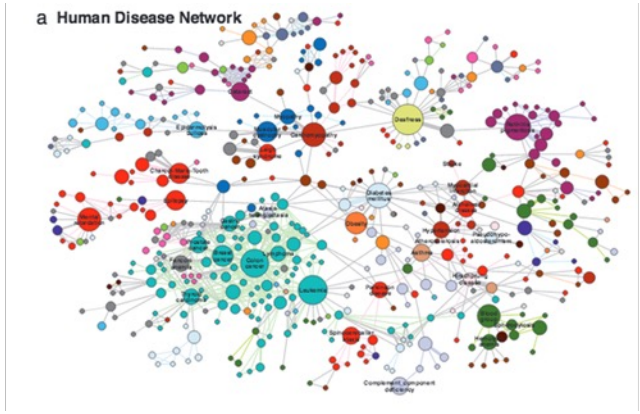
4.4 Trustworthy and Scalability

An Overview of Graph Learning

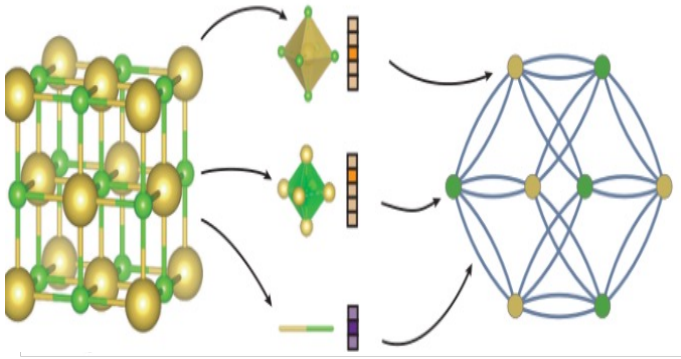
Graphs

Social networks

Biology network



Atom network

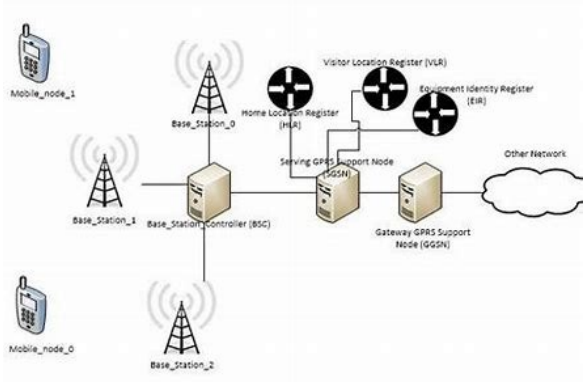


Financial networks

Source from Internet

Logistic networks

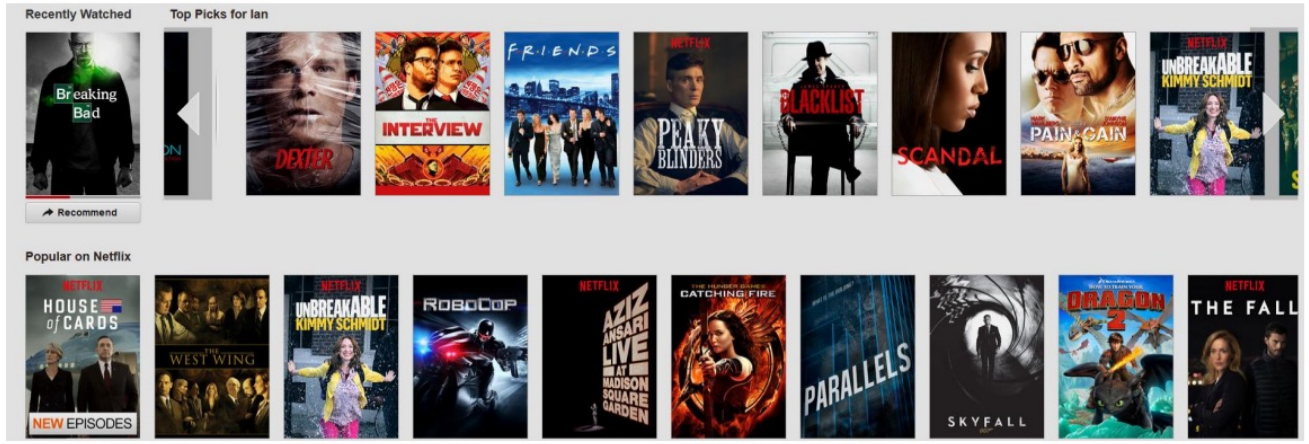
Telecom network



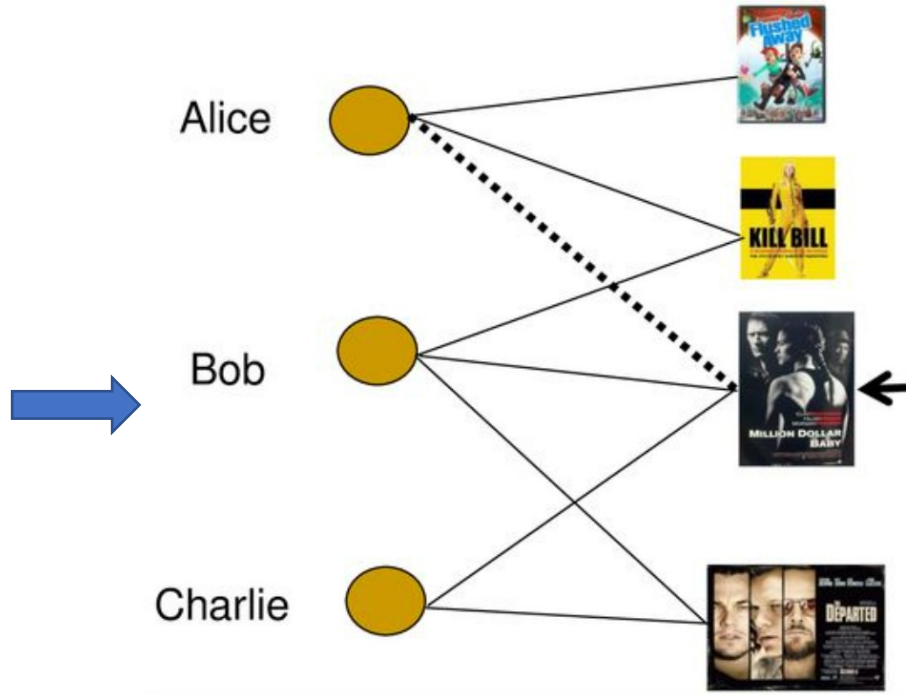
An Overview of Graph Learning

Graphs

Recommender Systems



Link Prediction

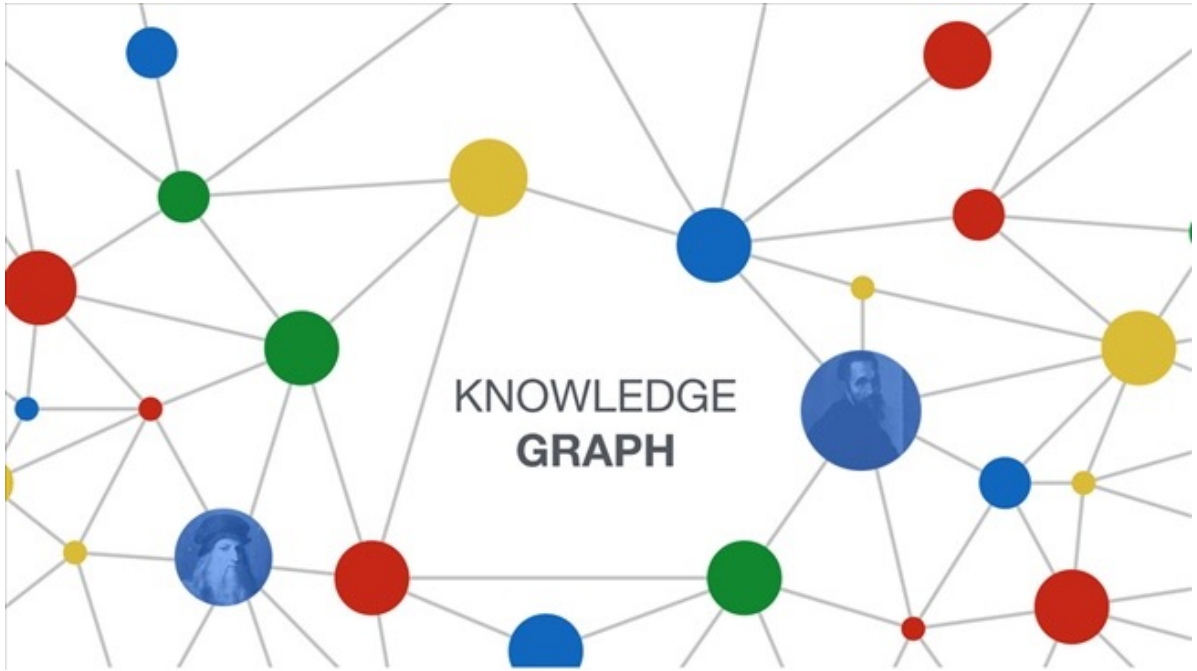


Source from Internet

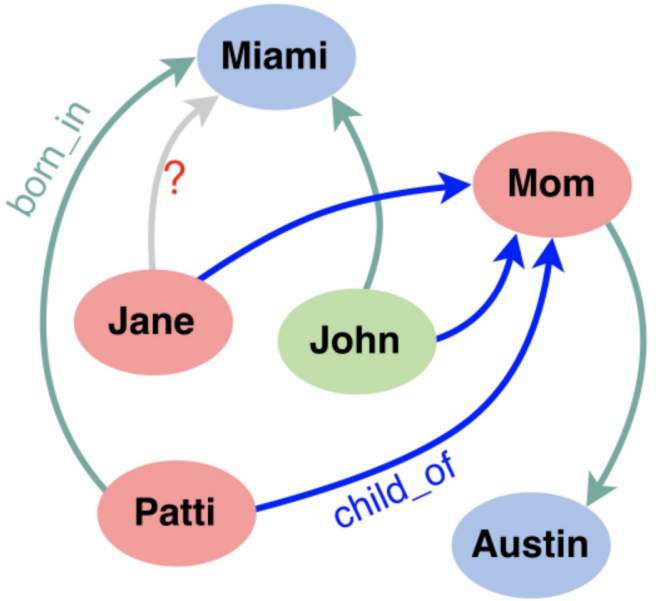
An Overview of Graph Learning

Graphs

Knowledge Graph Completion



Link prediction

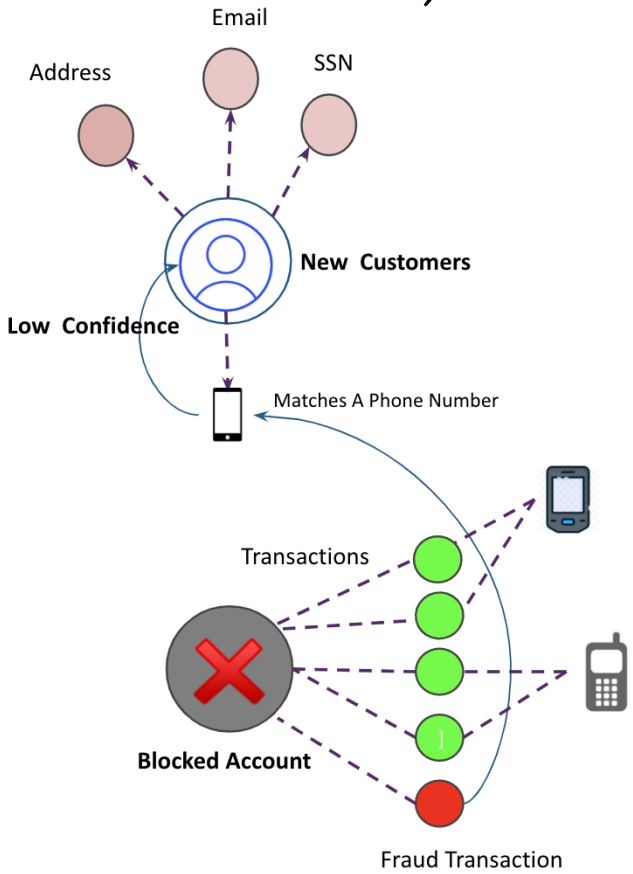


An Overview of Graph Learning

Graphs

Fraud Detection

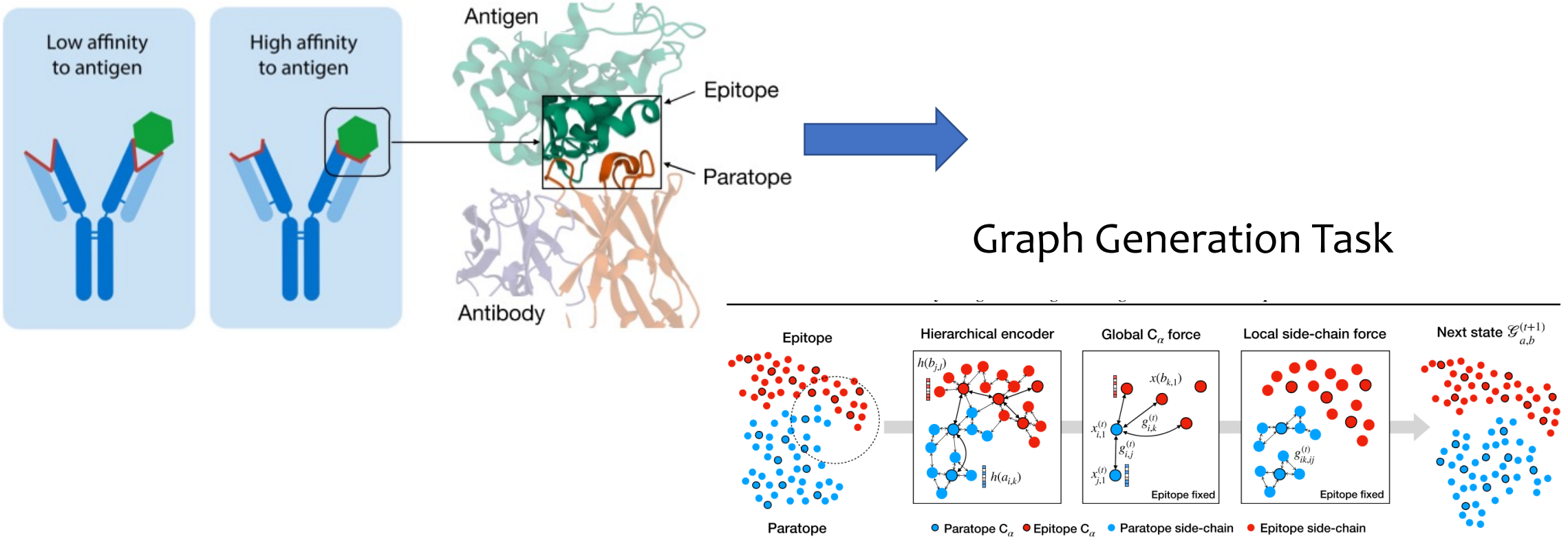
Link prediction, community detection, etc.



An Overview of Graph Learning

Graphs

- Antibody-Antigen Docking and Design

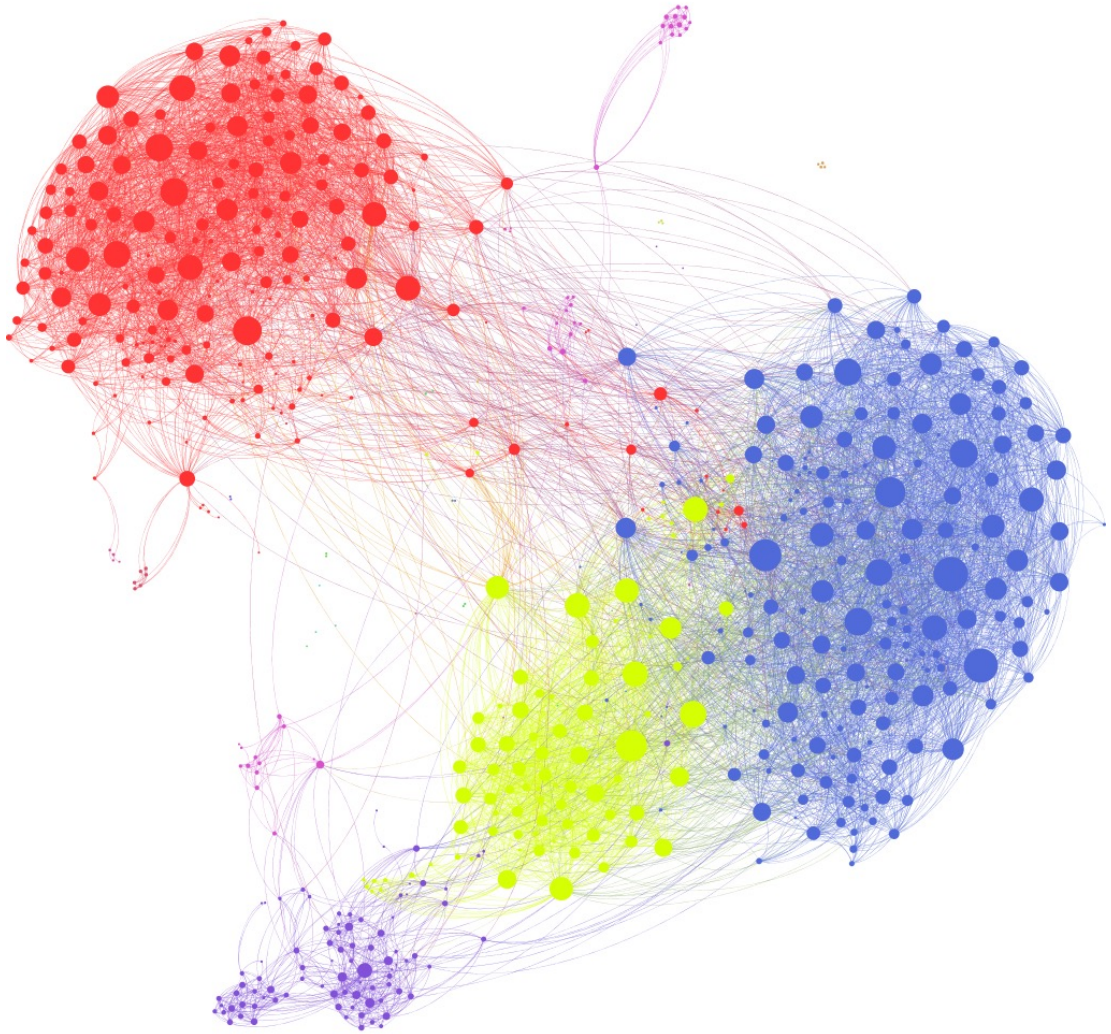


Jin W. et. al. Antibody-Antigen Docking and Design via Hierarchical Equivariant Refinement. ICML 2022

An Overview of Graph Learning

Graph Analysis

- Degree distribution
- Graph diameter
- Shortest path length
- Sparseness
- Curvature
- Symmetry

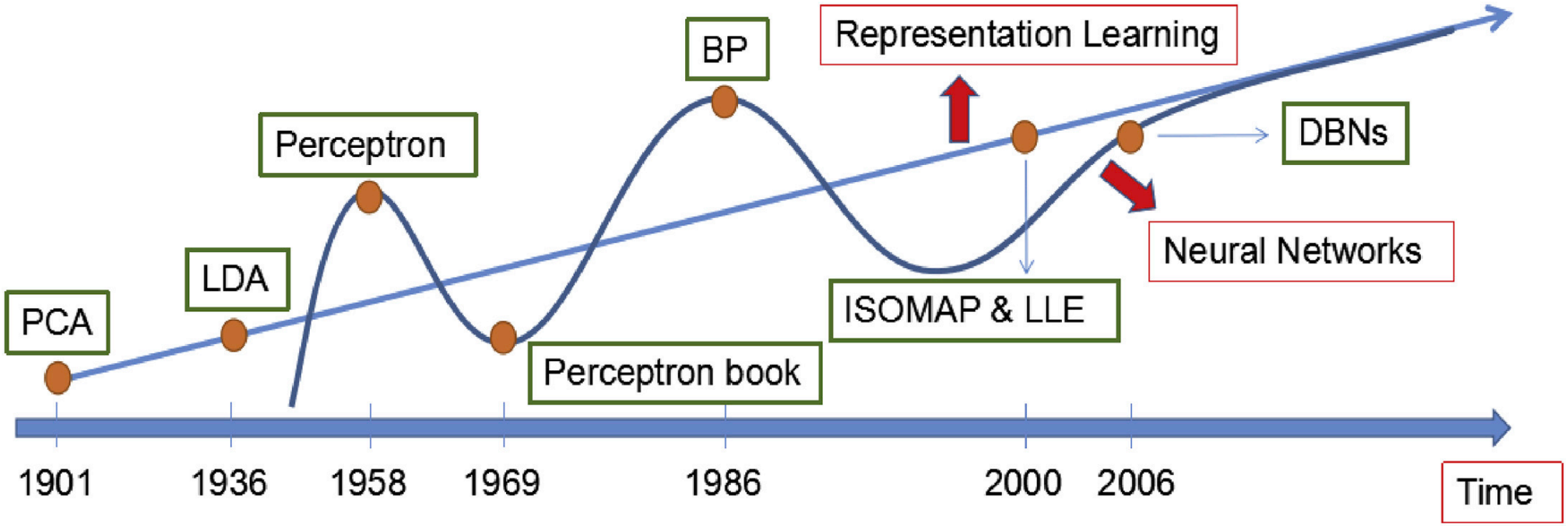


Representation Learning

- Graph embedding
- Graph neural network

An Overview of Graph Learning

"Can we automate the learning of useful features from raw data?"



"*Deep learning algorithms seek to exploit the unknown structure in the input distribution in order to discover good representations, often at multiple levels, with higher-level learned features defined in terms of lower-level features*"

Yoshua Bengio

An Overview of Graph Learning

Graph Embedding + Graph Neural Network

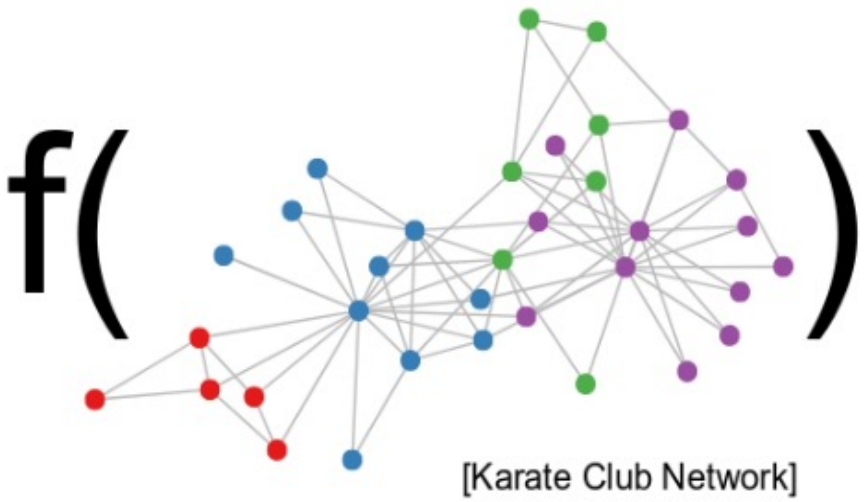
An Overview of Graph Learning

Graph Embedding

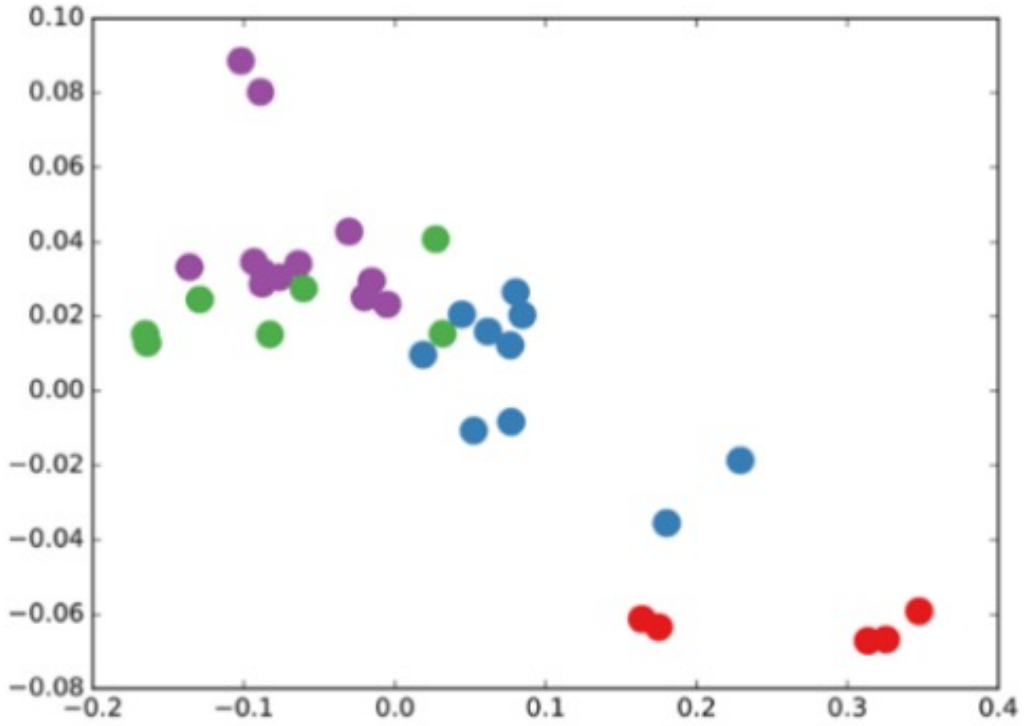
(1) Graph Embedding

Projecting graph data into a continuum space while the graph properties are preserved.

G=(V,E)

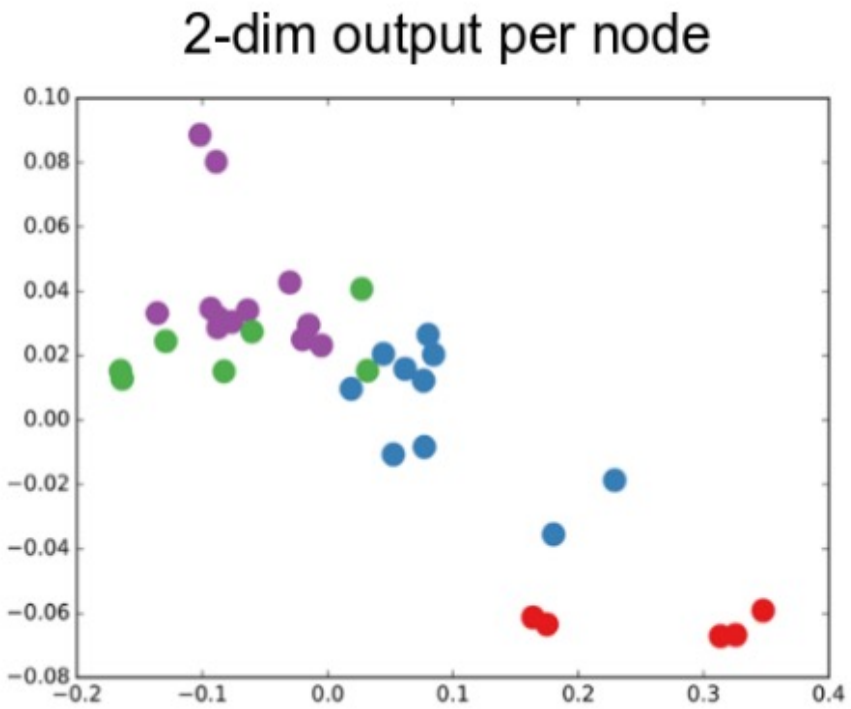


G=(V)
Vector space



(1) Graph Embedding

Goal: Perform network inference in the low-dimensional vector space



- Node importance
- Community detection
- Link prediction
- Node classification
- Network evolution
- ….

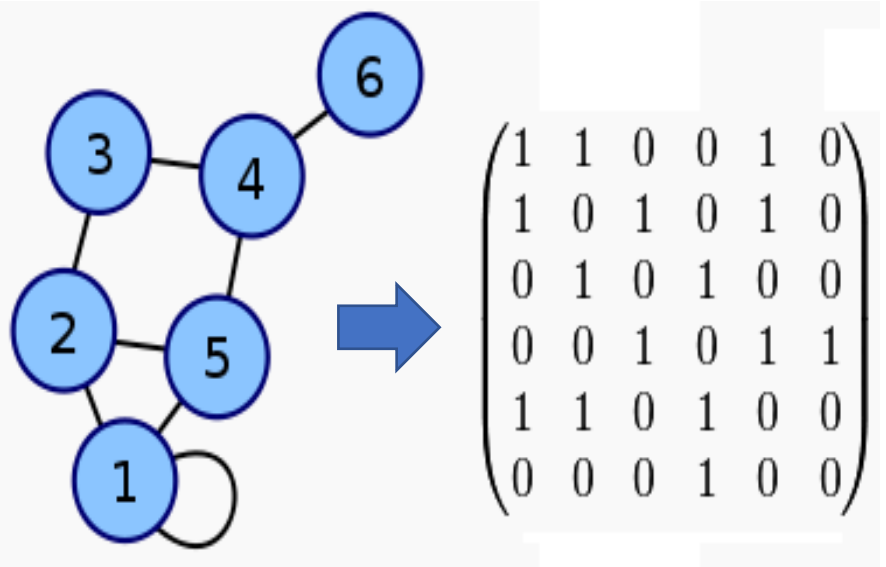
(1) Graph Embedding

Adjacency List encoding of *edges* vs one-hot encoding of *text tokens*

Vocabulary:
Man, woman, boy, girl, prince, princess, queen, king, monarch

	1	2	3	4	5	6	7	8	9
man	1	0	0	0	0	0	0	0	0
woman	0	1	0	0	0	0	0	0	0
boy	0	0	1	0	0	0	0	0	0
girl	0	0	0	1	0	0	0	0	0
prince	0	0	0	0	1	0	0	0	0
princess	0	0	0	0	0	1	0	0	0
queen	0	0	0	0	0	0	1	0	0
king	0	0	0	0	0	0	0	1	0
monarch	0	0	0	0	0	0	0	0	1

Each word gets a 1x9 vector representation



(1) Graph Embedding

Learns a vector representation for each word that maximized the probability of that word given the previous word.

Input: one hot encoded vector

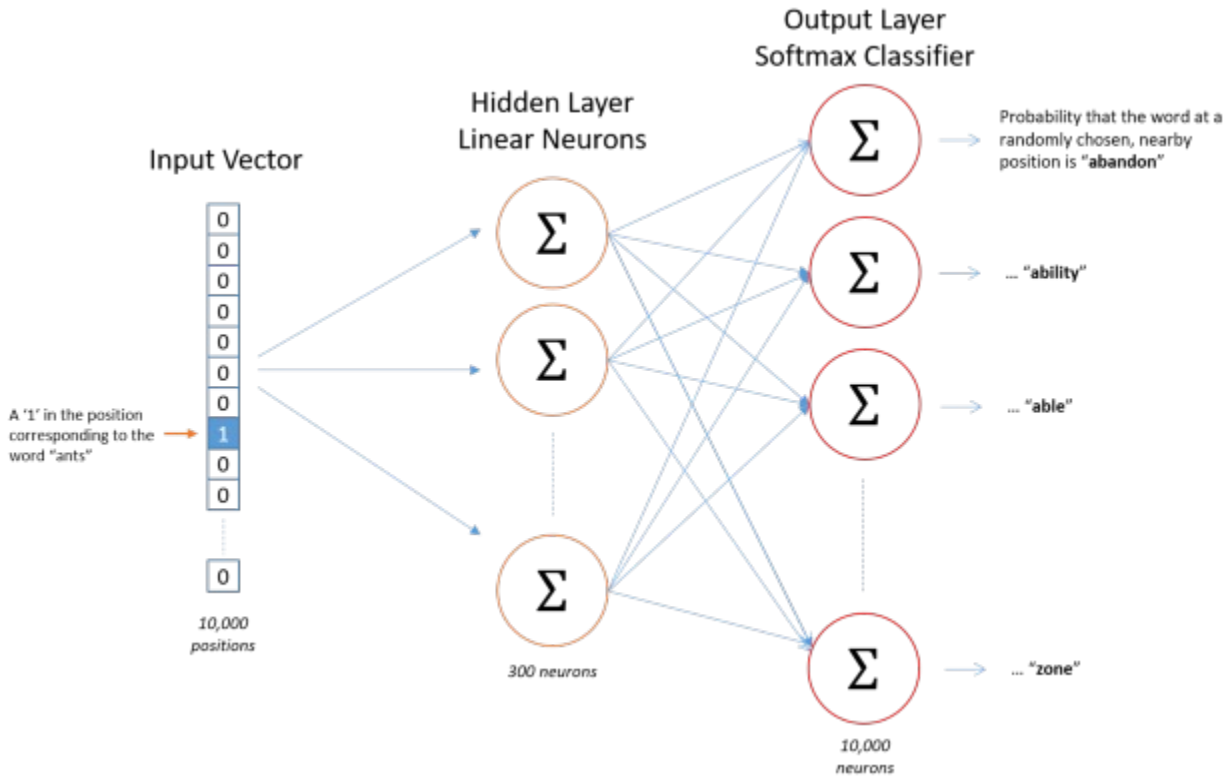


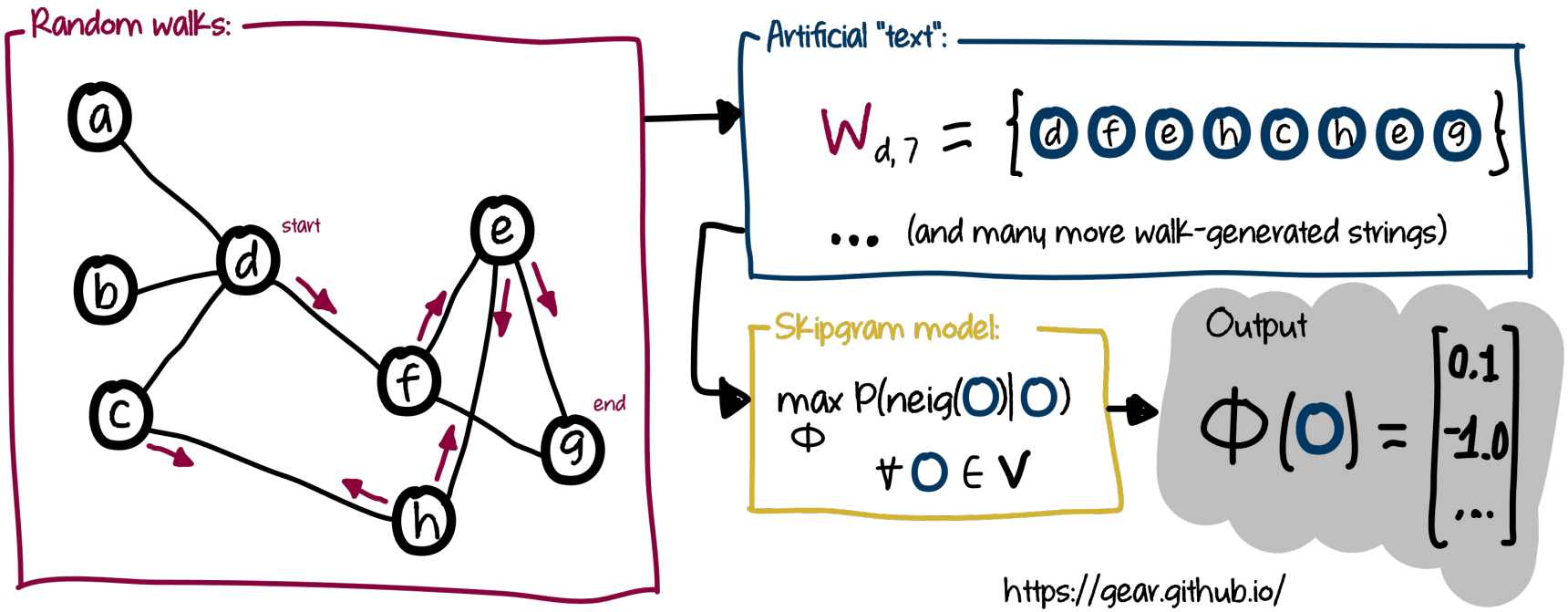
Illustration of SkipGram

Output: probability for each word in the corpus

https://towardsdatascience.com/word2vec-skip-gram-model-part-1-intuition-78614e4d6e0b

(1) Graph Embedding

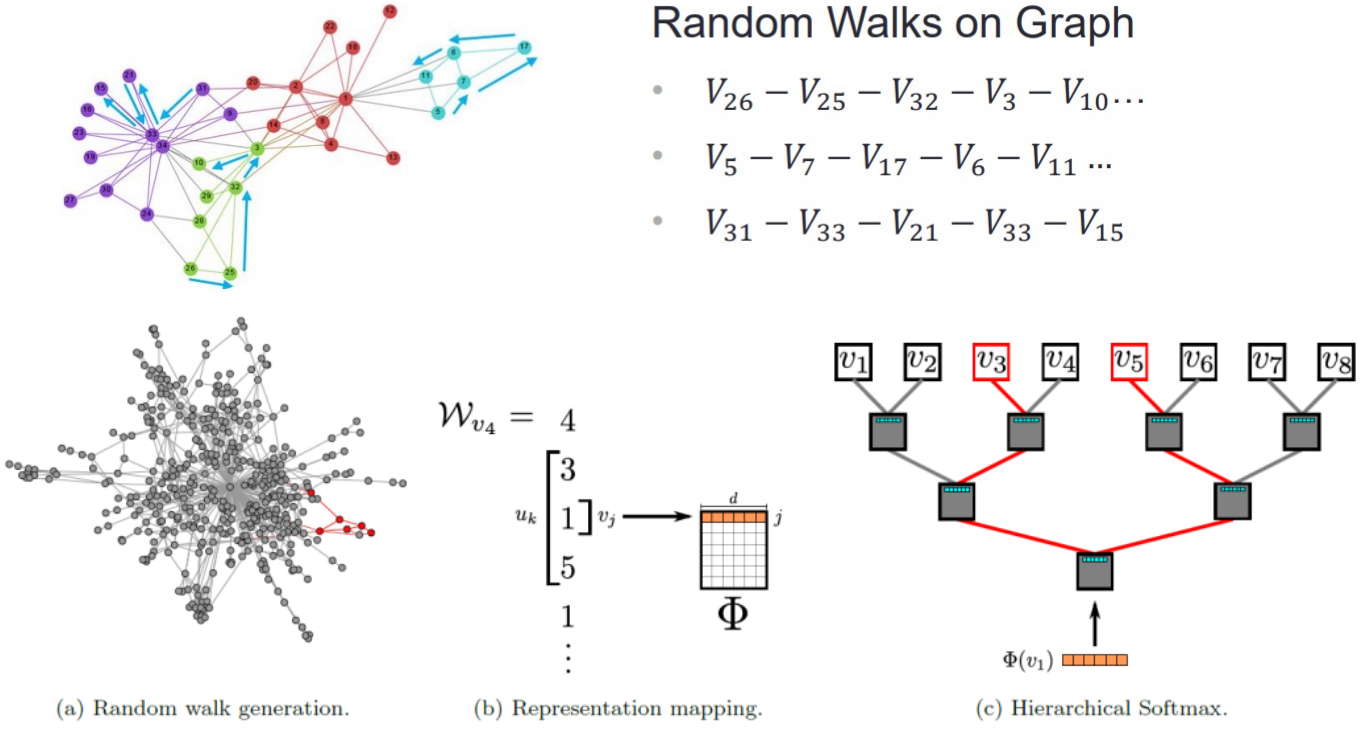
- How should we represent a node in a graph mathematically?
- Can we mimic word embedding ?
  - Treat each node treat as a word
  - Neighborhood around the node as the context window



(1) Graph Embedding

Deep Walk

- Exploit truncated random walks to define neighborhoods of a node.



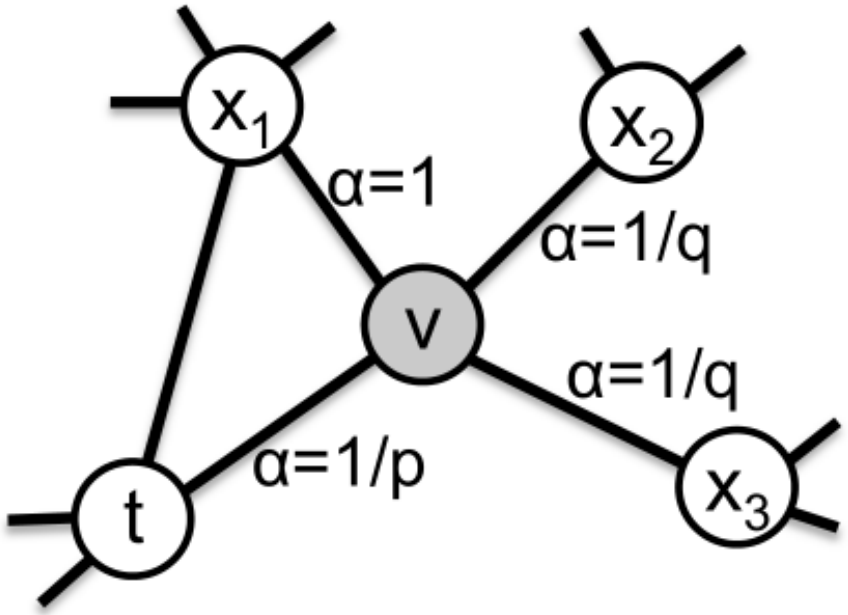
(a) Random walk generation. (b) Representation mapping. (c) Hierarchical Softmax.

B. Perozzi et al. Deepwalk: Online learning of social representations. KDD 2014

(1) Graph Embedding

Node2vec

- Generate representations of nodes in the graph via 2nd order (biased) random walk.



- Apply a bias factor alpha to reweigh the edge weights depending on the previous state

$$\alpha_{pq}(t,x) = \begin{cases} \frac{1}{p} & \text{if } d_{tx} = 0 \\ 1 & \text{if } d_{tx} = 1 \\ \frac{1}{q} & \text{if } d_{tx} = 2 \end{cases}$$

- Second order transition probability:

$$p(u|v,t) = \frac{\alpha_{pq}(t,u)w(u,v)}{\sum_{u' \in \mathcal{N}_v} \alpha_{pq}(t,u')w(u',v)}$$

Grover, Aditya, and Jure Leskovec. "node2vec: Scalable feature learning for networks." KDD 2016

(1) Graph Embedding

Summary

Many different methods:

- Matrix Factorization
- Random walk
- Others

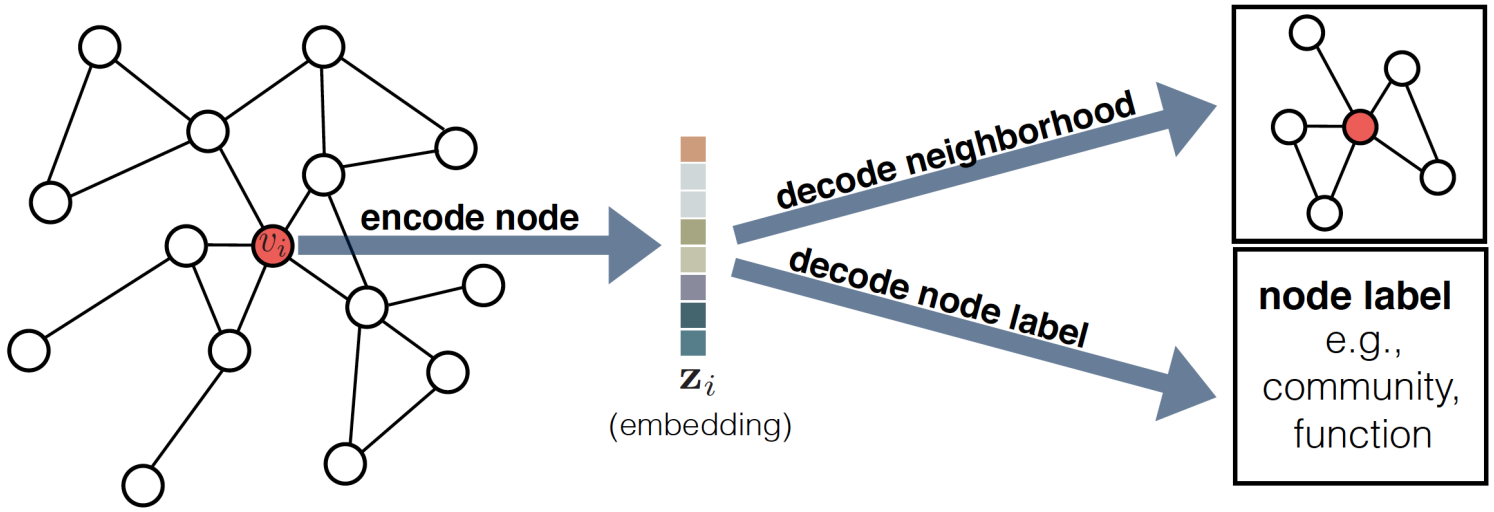
Category	Year	Published	Method	Time Complexity	Properties preserved
Factorization	2000	Science[26]	LLE	$O(\|E\|d^2)$	1^{st} order proximity
	2001	NIPS[25]	Laplacian Eigenmaps	$O(\|E\|d^2)$	
	2013	WWW[21]	Graph Factorization	$O(\|E\|d)$	
	2015	CIKM[27]	GraRep	$O(\|V\|^3)$	$1 - k^{th}$ order proximities
	2016	KDD[24]	HOPE	$O(\|E\|d^2)$	
Random Walk	2014	KDD[28]	DeepWalk	$O(\|V\|d)$	
	2016	KDD[29]	*node2vec*	$O(\|V\|d)$	$1 - k^{th}$ order proximities, structural equivalence

Hamilton, W. L. et.al. Representation Learning on Graphs: Methods and Applications. 2017

(1) Graph Embedding

Summary

- Most techniques consist of:



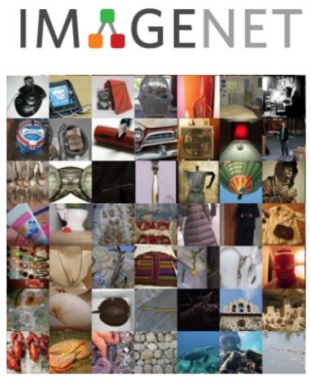
- A pairwise similarity function measures the similarity between nodes
- An encoder function to generate the node embedding
- A decoder function to reconstruct pairwise similarity
- A loss function to measure the quality of the pairwise reconstructions

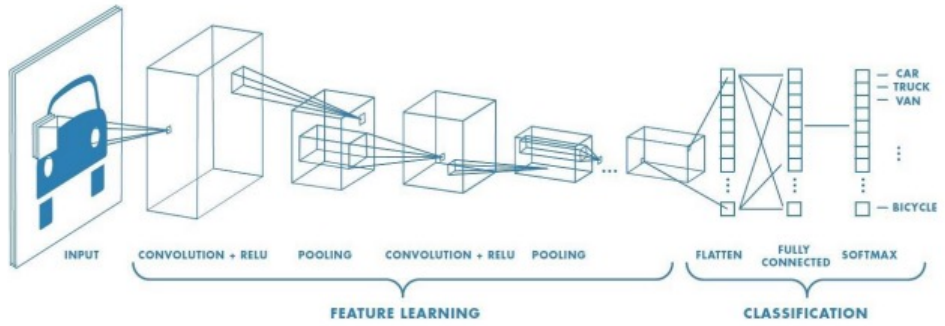
Hamilton, W. L. et.al. Representation Learning on Graphs: Methods and Applications. 2017

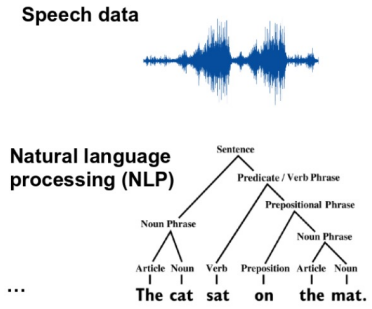
1.1 An Overview of Graph Learning

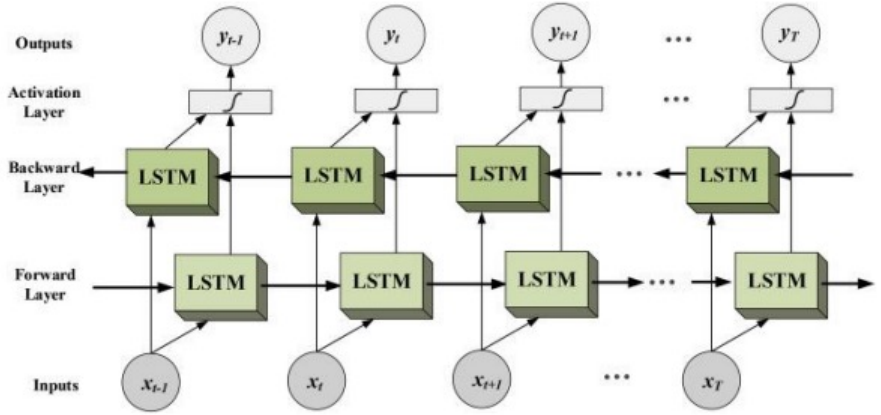
Graph Neural Network

(2) Graph Neural Network

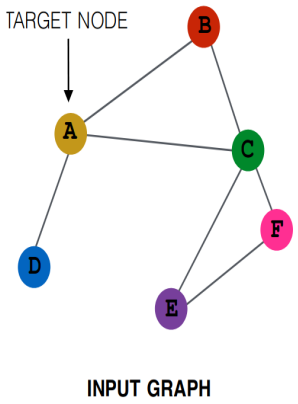


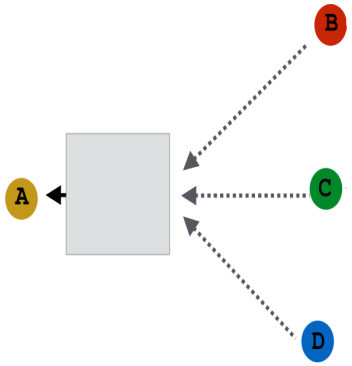






(2) Graph Neural Network

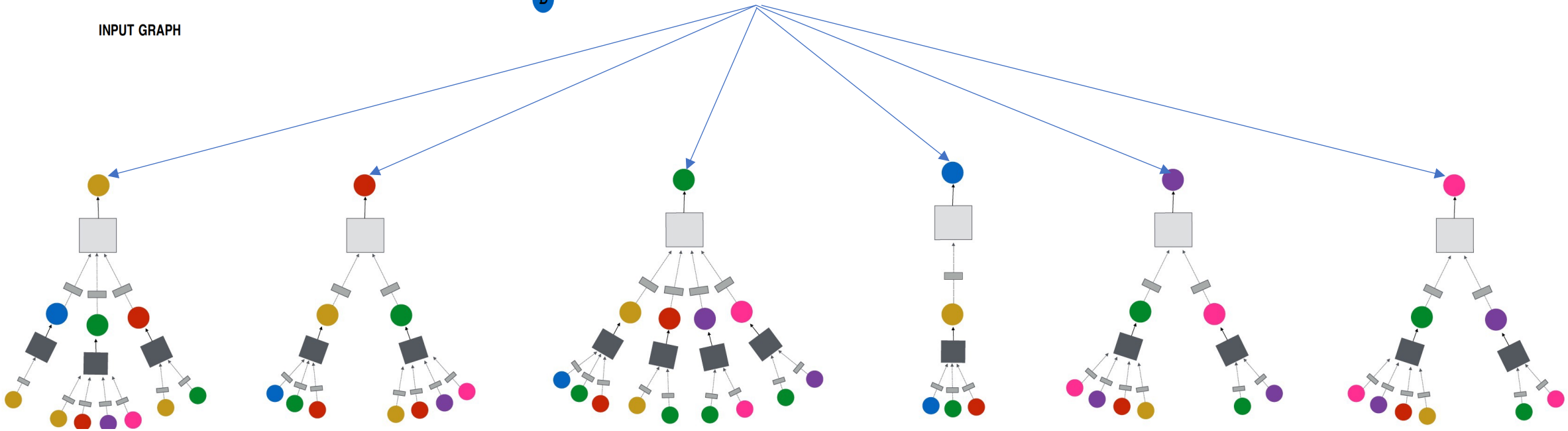




a. Neighborhood aggregation
b. Information propagation

$$H^{(l+1)} = f(H^{(l)}, A)$$

Every node defines a computation graph

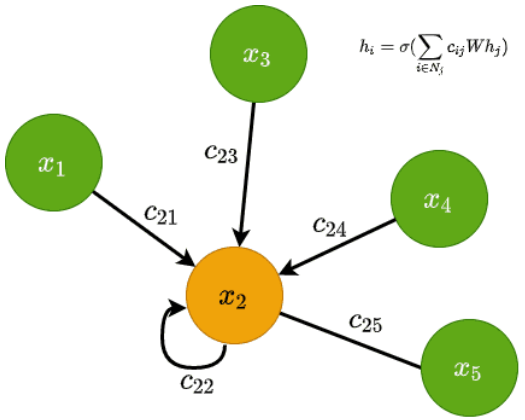


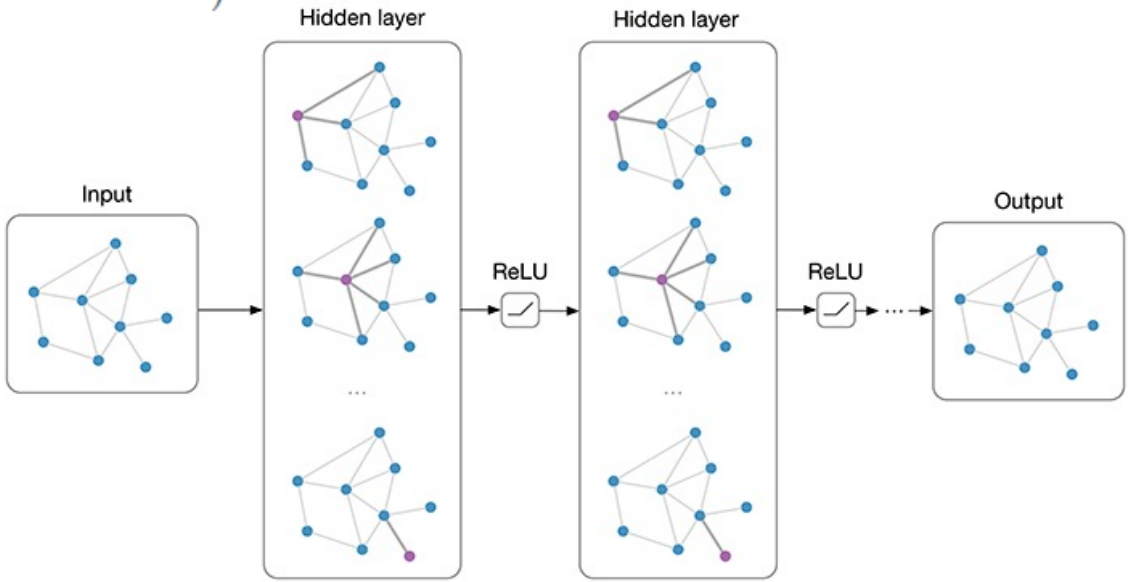
(2) Graph Neural Network

GCN

- Deal with the representation of a graph in the spectral domain
- Aggregate info from neighborhood via the normalized Laplacian matrix

$$H^{(l+1)} = \sigma(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{(l)}W^{(l)})$$





- Spectral method & Transductive learning
  - Easy to apply
  - Conduct on the entire graph and computationally inefficient

T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. ICLR, 2017.

(2) Graph Neural Network

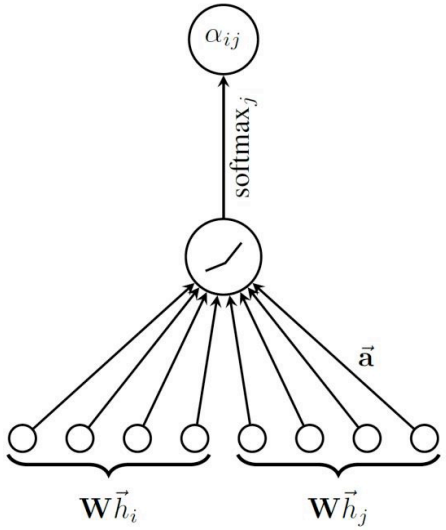
GAT

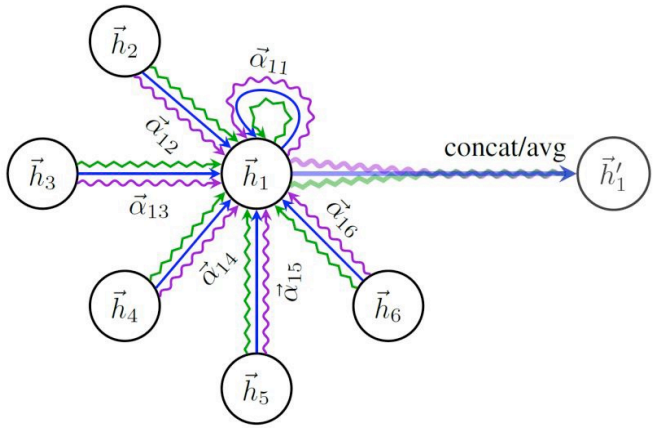
- Aggregate info from neighborhood via attention mechanism

$$a_{ij} = attention(h_i, h_j)$$

$$a_{ij} = \frac{exp(a_{ij})}{\sum_{k \in N_i} exp(a_{ik})}$$

$$h_i^{(l)} = \sigma(\sum_{i \in N_j} a_{ij} W h_j)$$





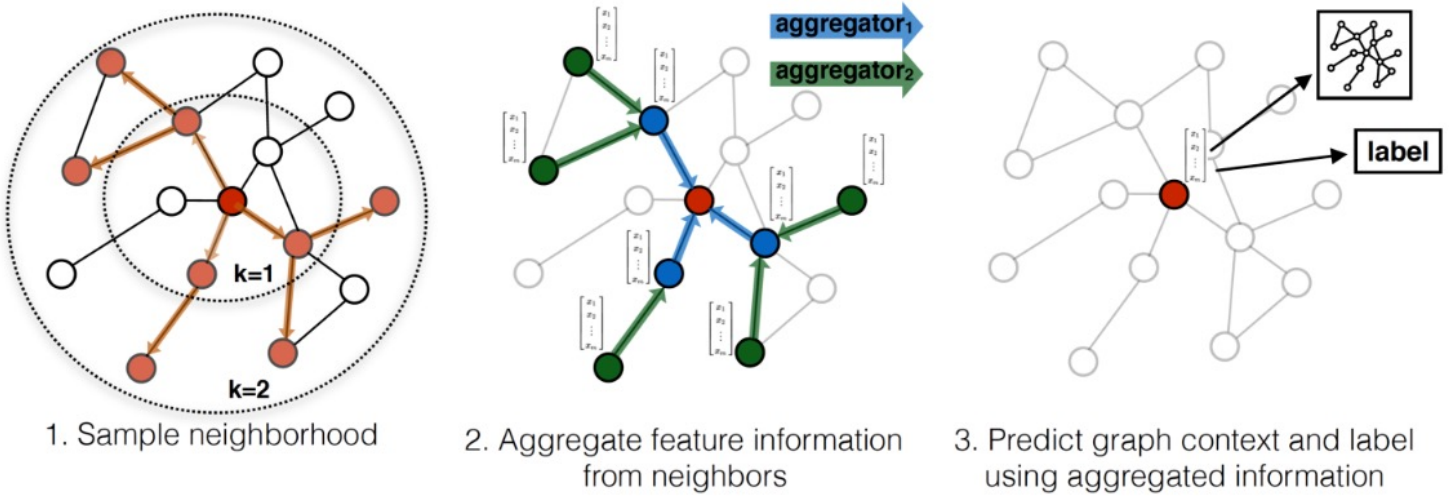
- Spatial method
- More flexible to handle multisource graph inputs

Veličković, Petar, et al. "Graph Attention Networks." ICLR. 2018.

(2) Graph Neural Network

GraphSAGE

- Sample a subset of node to conduct propagation
- Generalized aggregation: any differentiable function that maps set of vectors to a single vector



1. Sample neighborhood

2. Aggregate feature information from neighbors

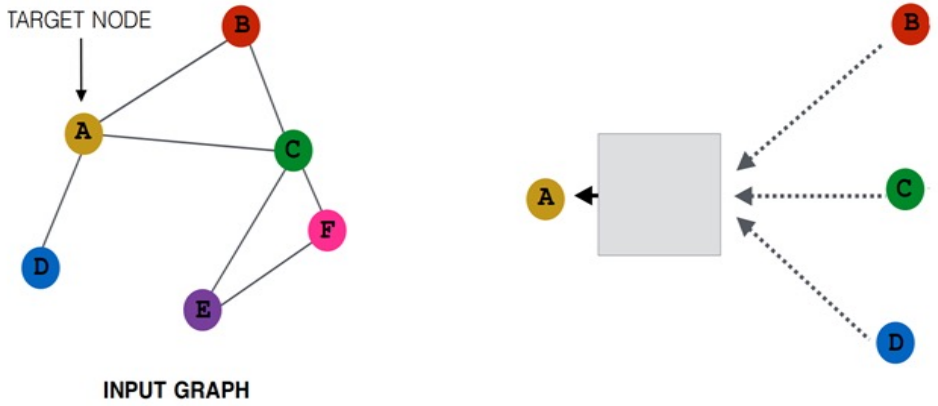
3. Predict graph context and label using aggregated information

- Spatial method with inductive capability

T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. ICLR, 2017.

(2) Graph Neural Network

Summary



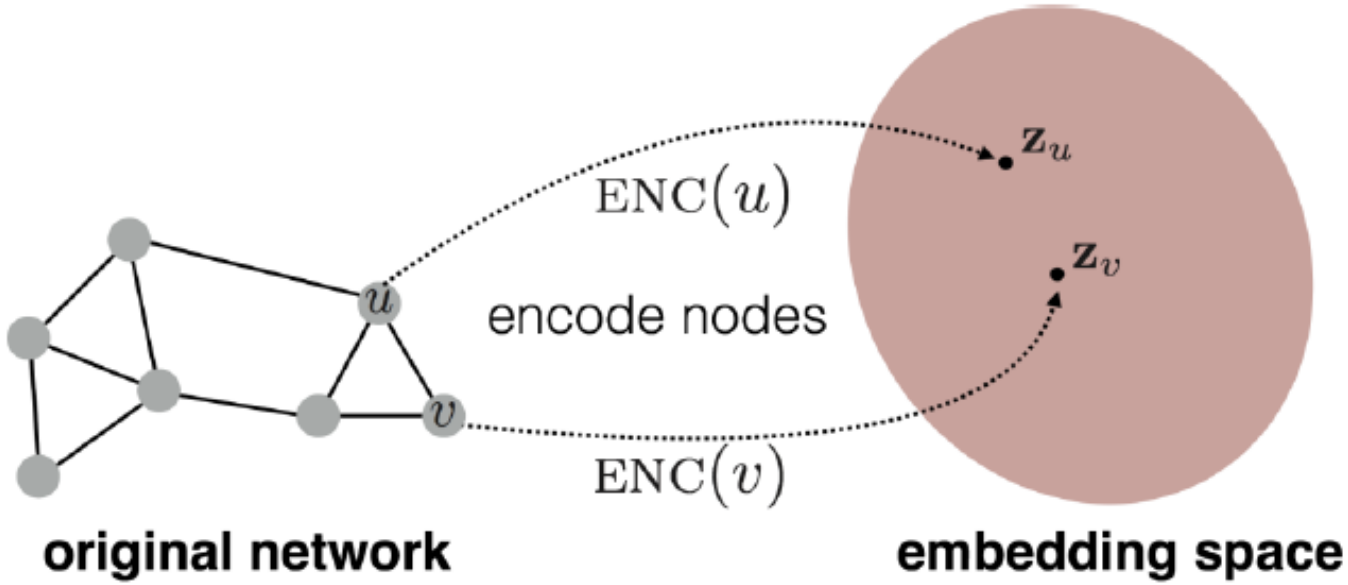
- Characteristics of GNNs
  - Information aggregation and prorogation
  - Trained via end-to-end manner according to downstream task
    - Supervised or semi-supervised
  - Utilize graph structure and node features simultaneously

Representation Learning on Networks, snap.stanford.edu/proj/embeddings-www, WWW 2018

(2) Graph Neural Network

Pipeline for Graph Learning

- **Step 1:** Obtain edge or/and node information , possibly with some augmentation
- **Step 2:** Use a parameterized encoder to map nodes in the graph(s) to an embedding space
- **Step 3:** Make predictions on nodes, edges or graphs based on embeddings
- **Step 4:** Compute loss and optimize the parameters



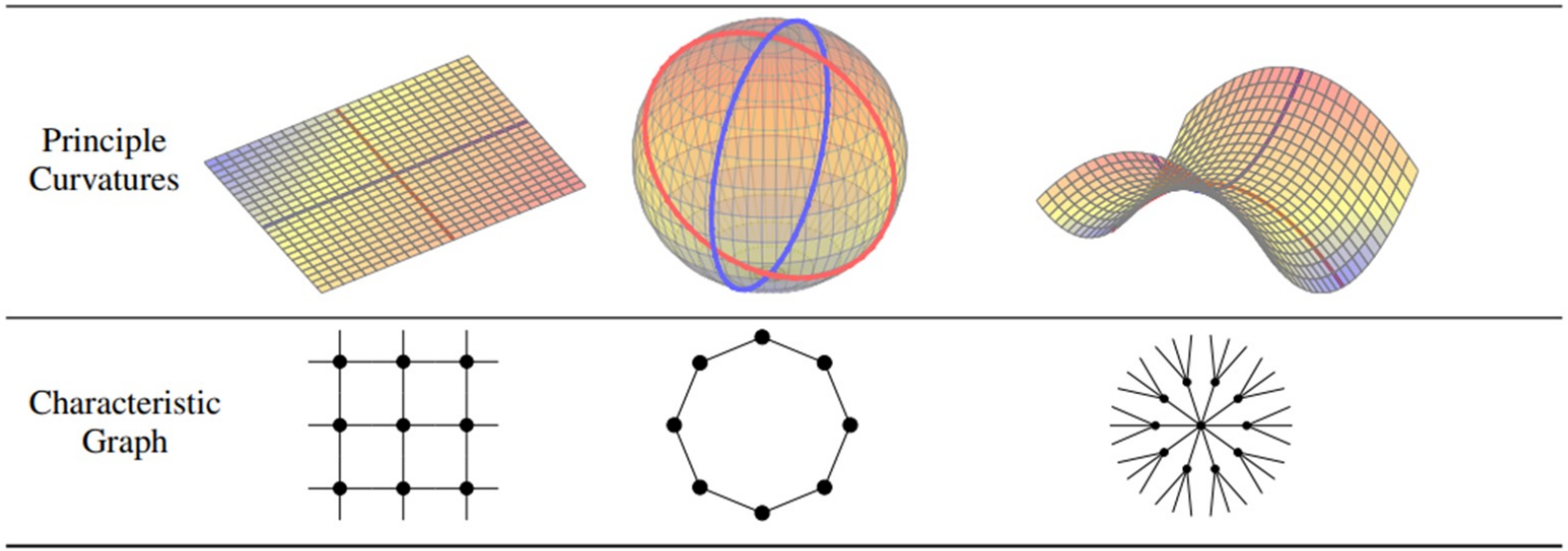
- Graph G = (V, E)
- V: Node set,
- E: Edge set
- f_G: Mapping function
- Z: Node embedding

Rex Ying. Introduction on Graph Representation Learning: A Geometric Perspective. 2022

Problems

Where to Learn Graphs?

- The quality of embeddings crucially depends on whether the geometry of the learning space matches the dataset.
- What embedding space geometry is optimal for data?



Curvature and Representation Learning: Identifying Embedding Spaces for Relational Data

Agenda

Part 1. INTRODUCTION (Min, 9:00 - 9:30)

1.1 An Overview of Graph Learning

1.2 Brief Introduction of Riemannian Geometry

1.3 Motivation for Hyperbolic Graph Learning

Coffee Break (9:30 – 10:000)

Part 2. HYPERBOLIC GRAPH NETWORKS (Menglin, 10:00 – 10:30)

2.1 Preliminary

2.2 Hyperbolic Shallow Models

2.2 Hyperbolic Neural Networks

2.3 Hyperbolic Graph Neural Networks

Part 3. APPLICATIONS (Menglin, 10:30 – 11:00)

3.1 HGNN for Recommendation Systems

3.2 HGNN for Knowledge Graph

3.3 HGNN for Healthcare

Part 4. ADVANCED TOPICS (Bo, 11:00 – 12:00)

4.1 Complex Structures

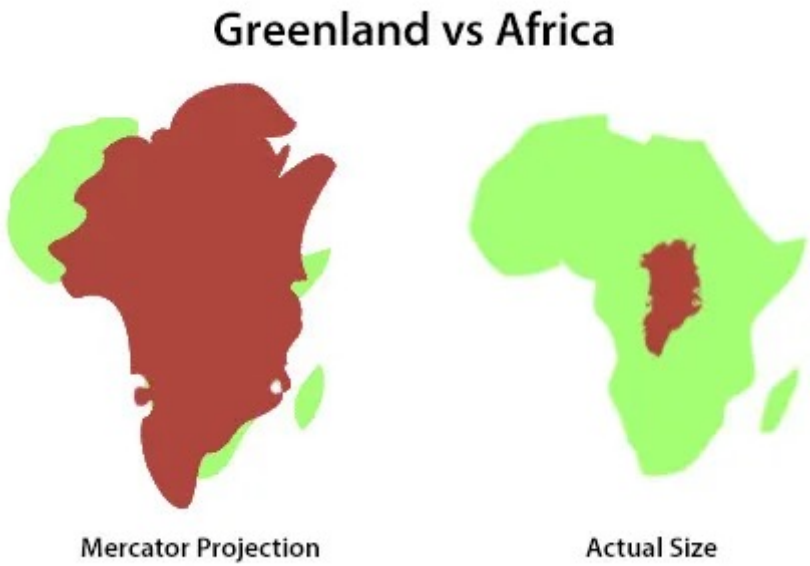
4.2 Curvature-aware Learning

4.3 Evolving Interactions

4.4 Trustworthy and Scalability

Riemannian Geometry

Constraints of Euclidean Geometry

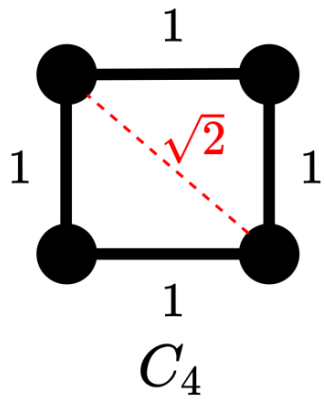


Many times, we have data lies on non-Euclidean manifold.

Riemannian Geometry

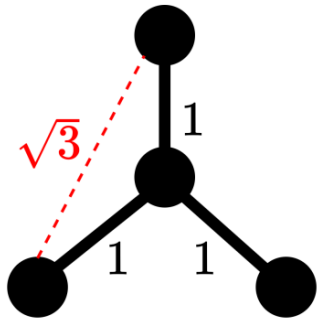
Constraints of Euclidean Geometry

- Cannot embed large classes of graph w/ low distortion or w/o loss of information
  - E.g. cycles, trees



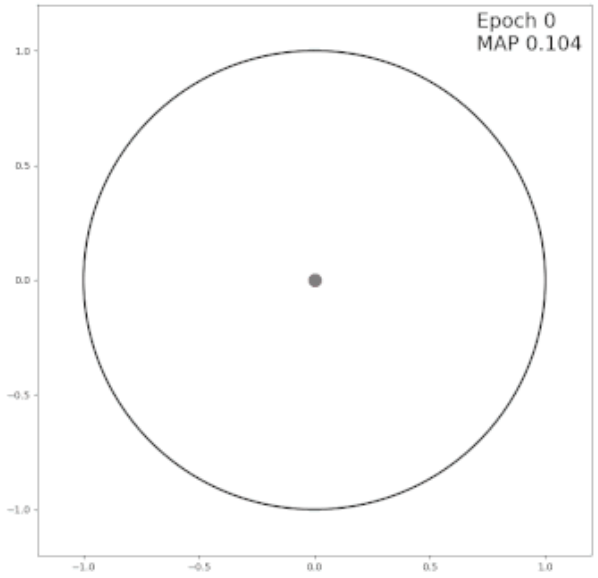
C_4

Distortion $\sqrt{2}$



$K_{1,3}$

Distortion $^{2}/_{\sqrt{3}}$



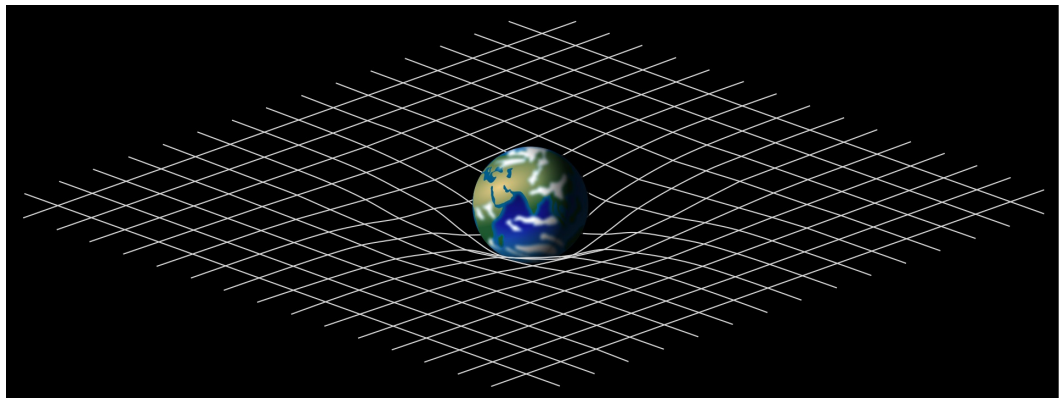
- Graph distance $d_G(i,j)$= "shortest path from i to j"

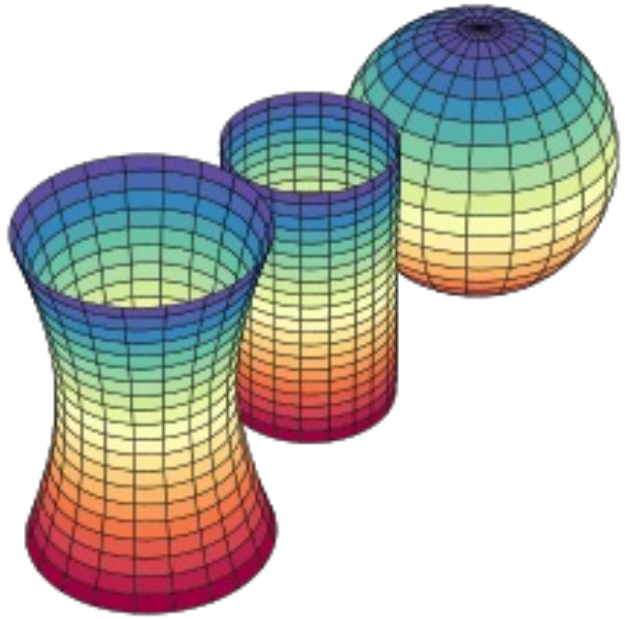
https://people.csail.mit.edu/oct/ctgnn_slides.pdf

Riemannian Geometry

Riemannian Manifolds

- Manifolds: high dimensional surface that looks locally-Euclidean
- Riemannian Manifold
  - Equipped with
    - *Inner product* $\langle .,. \rangle$: metric space
    - *Tangent space* $\mathcal{T}_x$: an $\mathbb{R}^n$ that approximates the manifold at any point x
  - Curvature: how much the surface deviates from being a plane
  - Geodesic: shortest path in manifold
    - Analogous to straight lines in $\mathbb{R}^n$

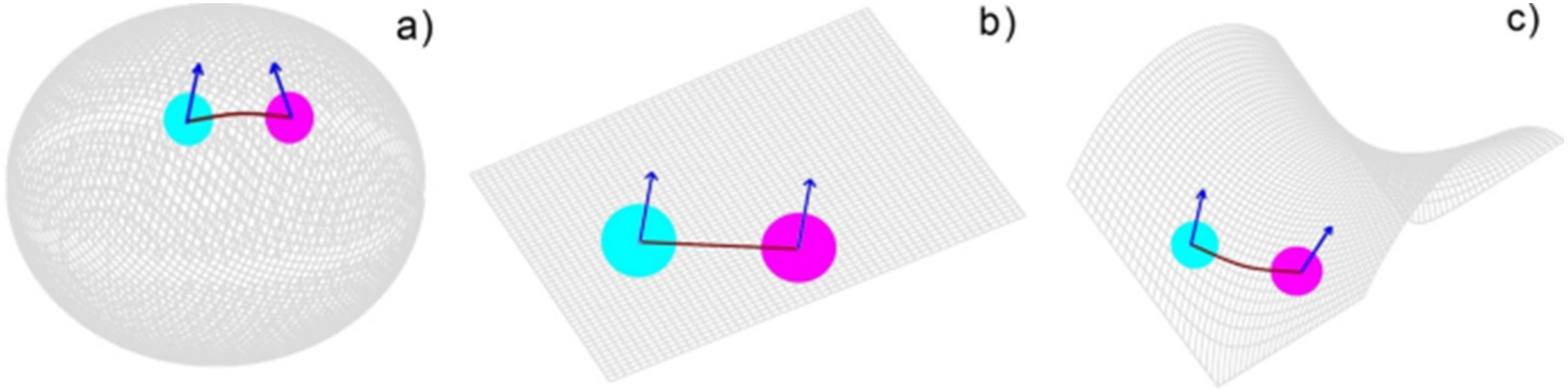




Riemannian Geometry

Curvature

- A geometric property: Flatness of an object
- Geometric intuition: Measure for growth rate of volume of distance ball→ "geodesic dispersion"



Positive Curvature

Flat Curvature

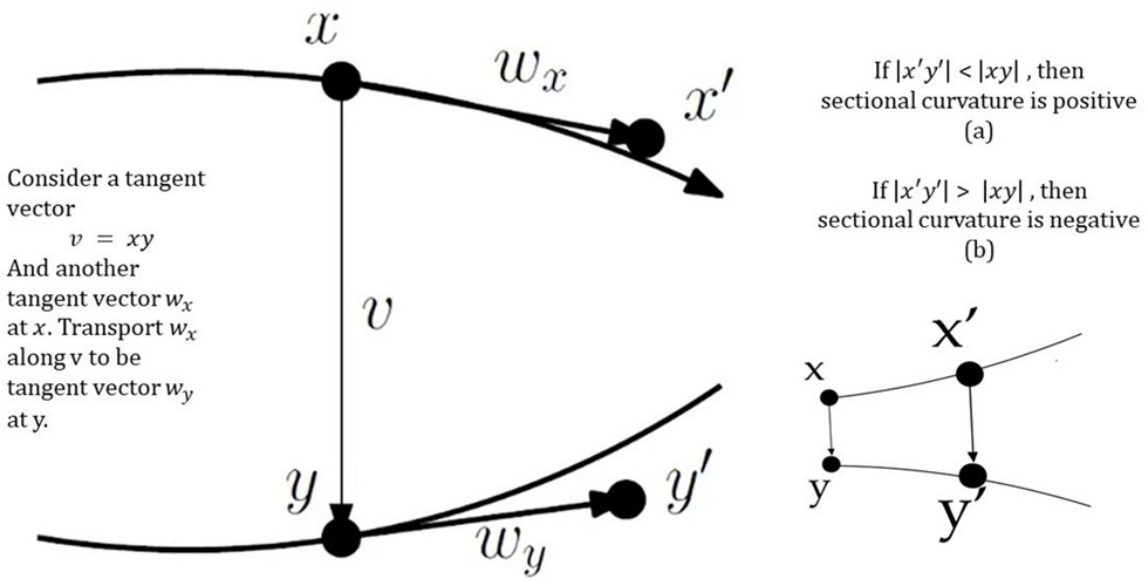
Negative Curvature

Eidi M, Jost J. Ollivier ricci curvature of directed hypergraphs[J]. Scientific Reports, 2020, 10(1): 1-14

Riemannian Geometry

Sectional Curvature & Ricci Curvature

- Consider a tangent vector $v = xy$ and another tangent vector w_x at x. Transport w_x along v to be a tangent vector w_x at y.



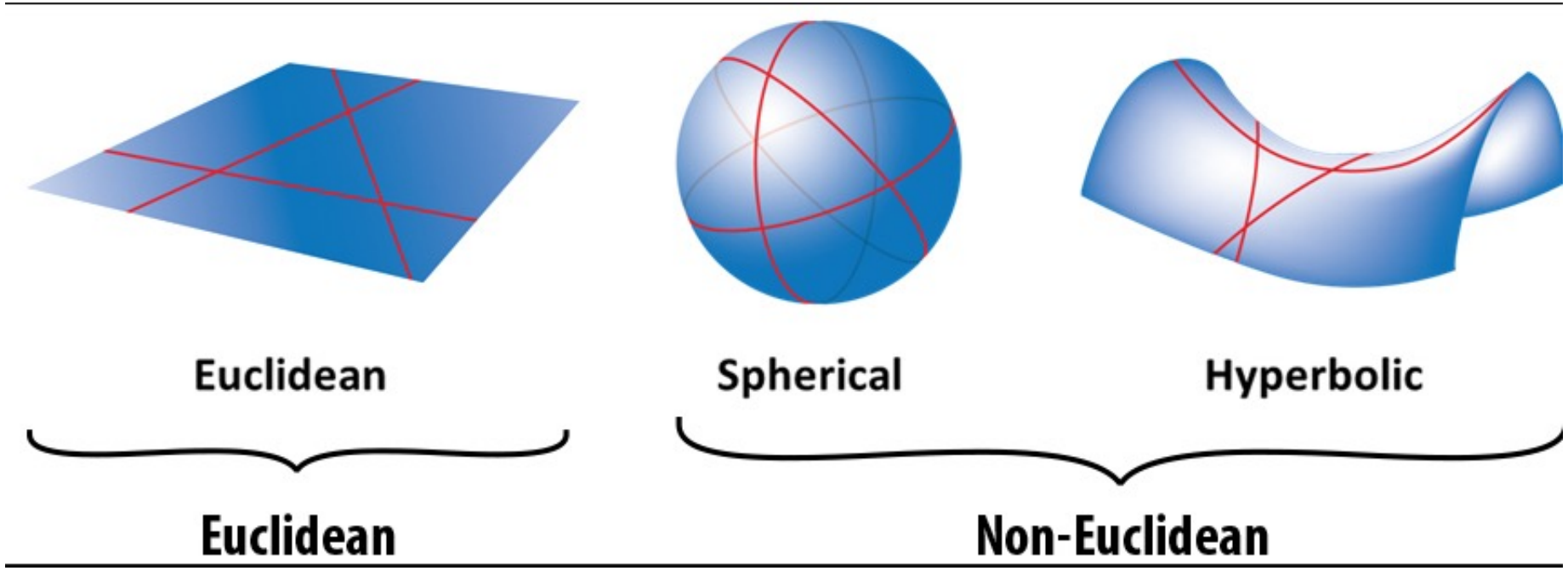
- Ricci Curvature: averaging over all directions w

Ni, Chien-Chun, et al. "Ricci curvature of the internet topology." *2015 IEEE conference on computer communications (INFOCOM)*. IEEE, 2015.

Riemannian Geometry

Geodesic

- Shortest path in the manifold



http://drmarkliu.com/noneuclidean

Agenda

Part 1. INTRODUCTION (Min, 9:00 - 9:30)

1.1 An Overview of Graph Learning

1.2 Brief Introduction of Riemannian Geometry

1.3 Motivation for Hyperbolic Graph Learning

Coffee Break (9:30 – 10:000)

Part 2. HYPERBOLIC GRAPH NETWORKS (Menglin, 10:00 – 10:30)

2.1 Preliminary

2.2 Hyperbolic Shallow Models

2.2 Hyperbolic Neural Networks

2.3 Hyperbolic Graph Neural Networks

Part 3. APPLICATIONS (Menglin, 10:30 – 11:00)

3.1 HGNN for Recommendation Systems

3.2 HGNN for Knowledge Graph

3.3 HGNN for Bioinformatics

Part 4. ADVANCED TOPICS (Bo, 11:00 – 12:00)

4.1 Complex Structures

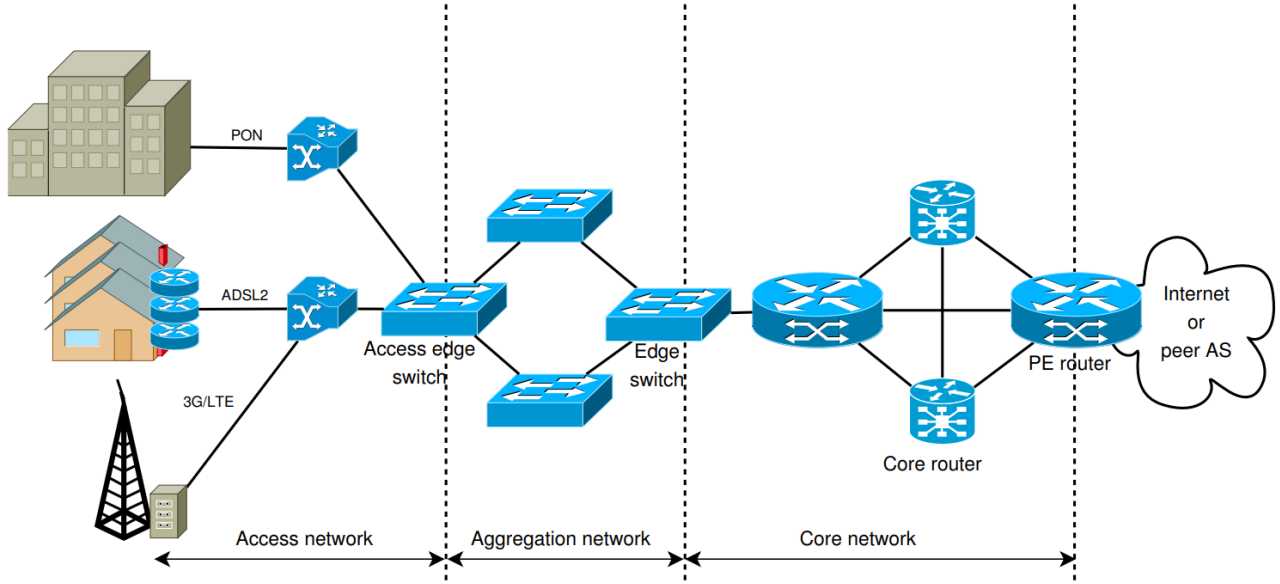
4.3 Curvature-aware Learning

4.2 Evolving Interactions

4.4 Trustworthy and Scalability

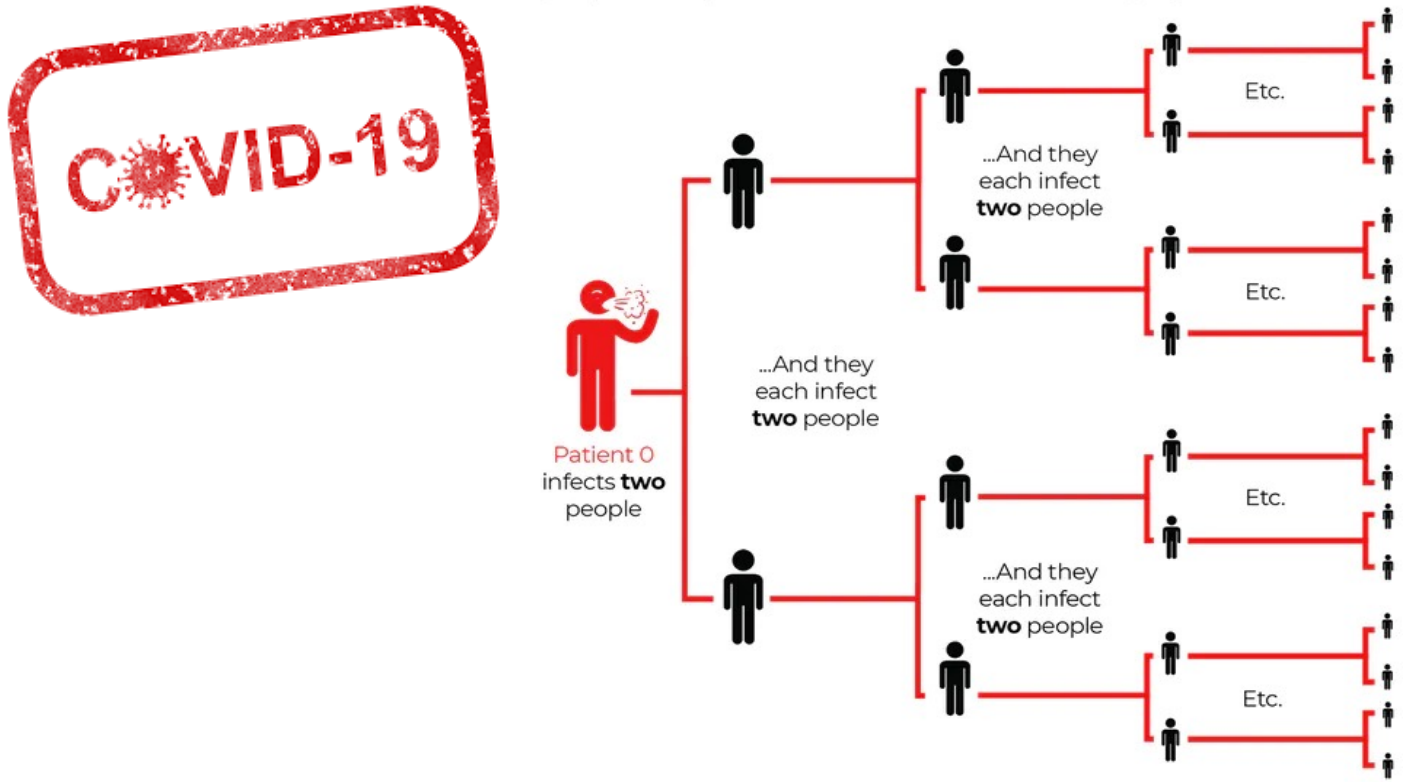
Motivations

Many network are with tree-like structures or power-law distributed.

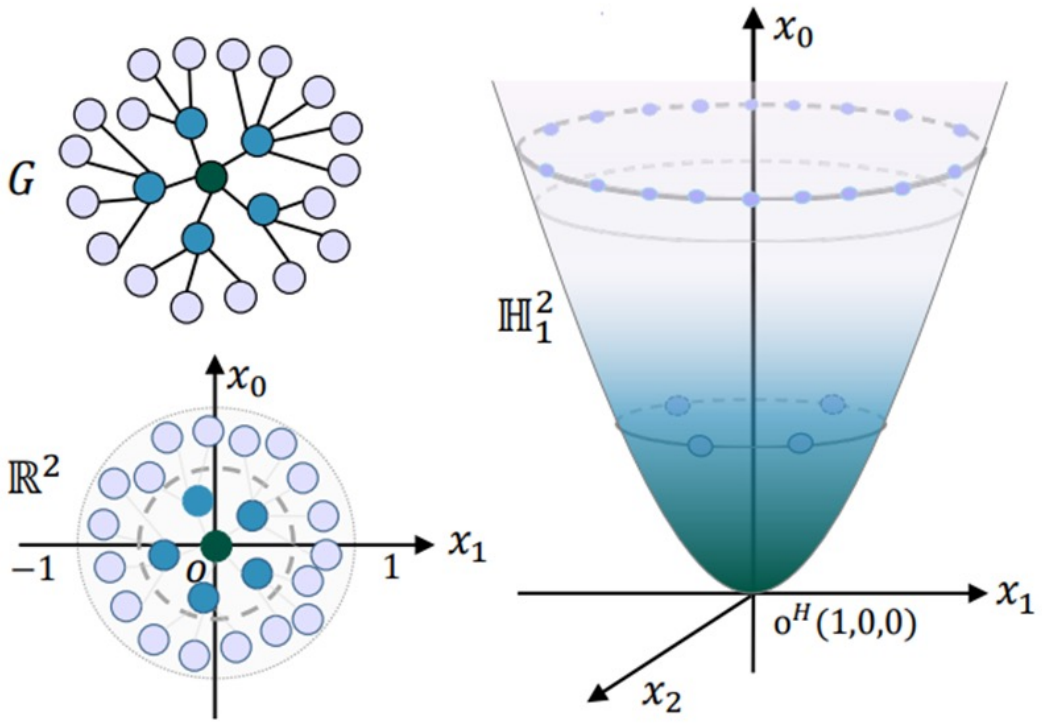


Motivations

Many network are with tree-like structures or power-law distributed.



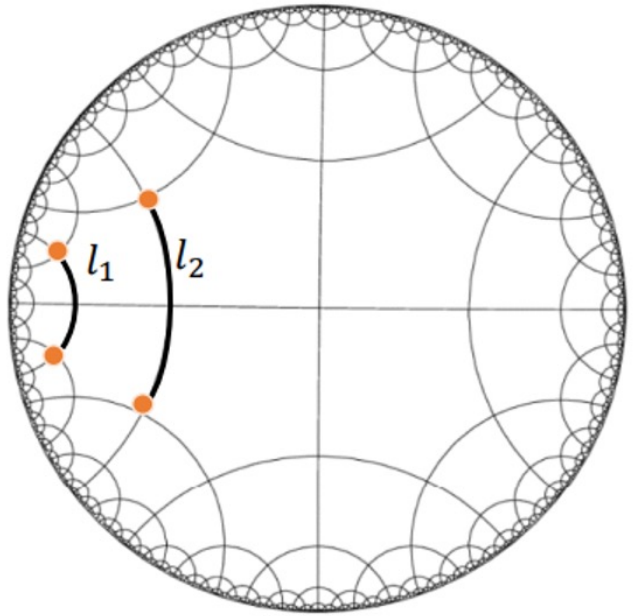
Nodes are exponentially increasing and hierarchical arranged.



(1) Acts as a geometric prior for hierarchical structures / tree graphs / heavy tailed distributions (e.g. scale-free, power-law). *

$$V_n^{\mathbb{E}}(r) = \frac{\pi^{\frac{n}{2}}}{\Gamma\left(\frac{n}{2}+1\right)} r^n.$$

$$V_n^{\mathbb{H}}(r) \sim \frac{\mathrm{Vol}\left(\mathbb{S}^n\right)}{2^{n-1}} e^{(n-1)r}, as \quad r \to \infty.$$



(2) Larger embedding space ---smaller embedding dimension and fewer parameter

Slack channel

- Join our slack channel for Q&A

https://hyperbolicgnn.github.io/

Part 2. Hyperbolic Graph Neural Network (HGNN)

Agenda

Part 1. INTRODUCTION (Min, 9:00 - 9:30)

1.1 An Overview of Graph Learning

1.2 Brief Introduction of Riemannian Geometry

1.3 Motivation for Hyperbolic Graph Learning

Coffee Break (9:30 – 10:00)

Part 2. HYPERBOLIC GRAPH NETWORKS (Menglin, 10:00 – 10:30)

2.1 Preliminary

2.2 Hyperbolic Shallow Models

2.2 Hyperbolic Neural Networks

2.3 Hyperbolic Graph Neural Networks

Part 3. APPLICATIONS (Menglin, 10:30 – 11:00)

3.1 HGNN for Recommendation Systems

3.2 HGNN for Knowledge Graph

3.3 HGNN for Bioinformatics

Part 4. ADVANCED TOPICS (Bo, 11:00 – 12:00)

4.1 Complex Structures

4.2 Curvature-aware Learning

4.3 Evolving Interactions

4.4 Trustworthy and Scalability

Hyperbolic geometry

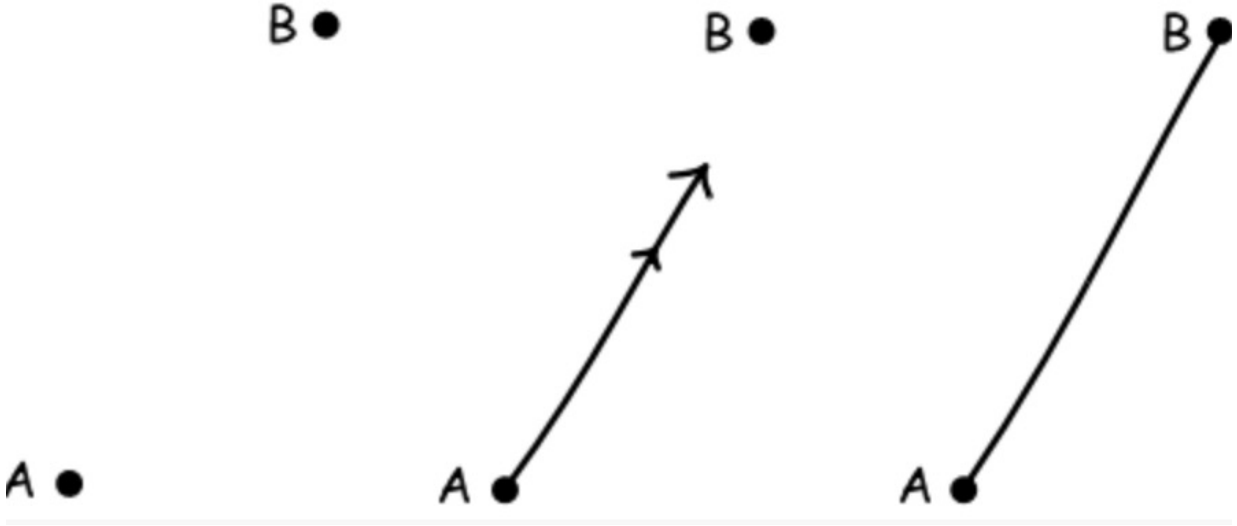
- A non-Euclidean geometry that reject the validity of Euclid's fifth, the “parallel”, postulate.

Figure source: https://www.quora.com/

Euclid fifth's postulate

1. A straight line segment can be drawn

joining any two points.



https://architectureflrc.weebly.com/euclid.html

Euclid fifth's postulate

1. A straight line segment can be drawn joining any two points.

2. Any straight line segment can be extended indefinitely in a straight ¨

Figure: https://architectureflrc.weebly.com/euclid.html

Euclid fifth's postulate

1. A straight line segment can be drawn joining any two points.

2. Any straight line segment can be extended indefinitely in a straight line.

3. Given any straight line segment, a circle can be drawn having the segment as the radius and one endpoint as center.

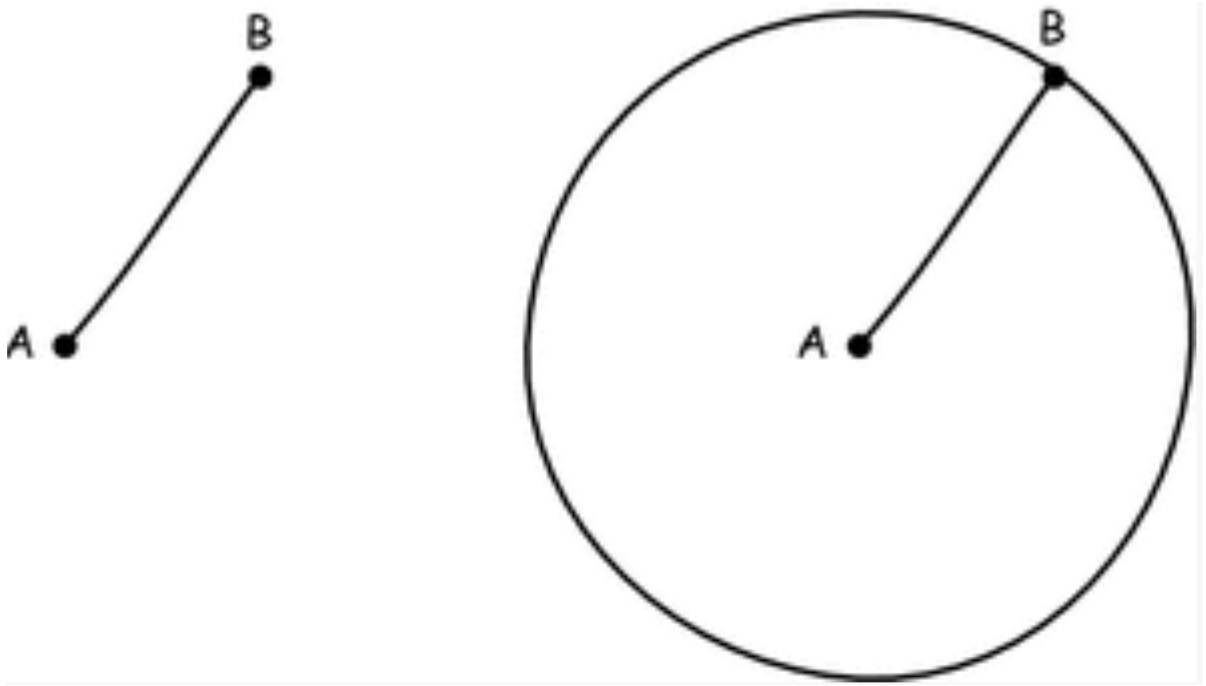


Figure: https://architectureflrc.weebly.com/euclid.html

Euclid fifth's postulate

1. A straight line segment can be drawn joining any two points.

2. Any straight line segment can be extended indefinitely in a straight line.

3. Given any straight line segment, a circle can be drawn having the segment as the radius and one endpoint as center.

4. All Right Angles are congruent.

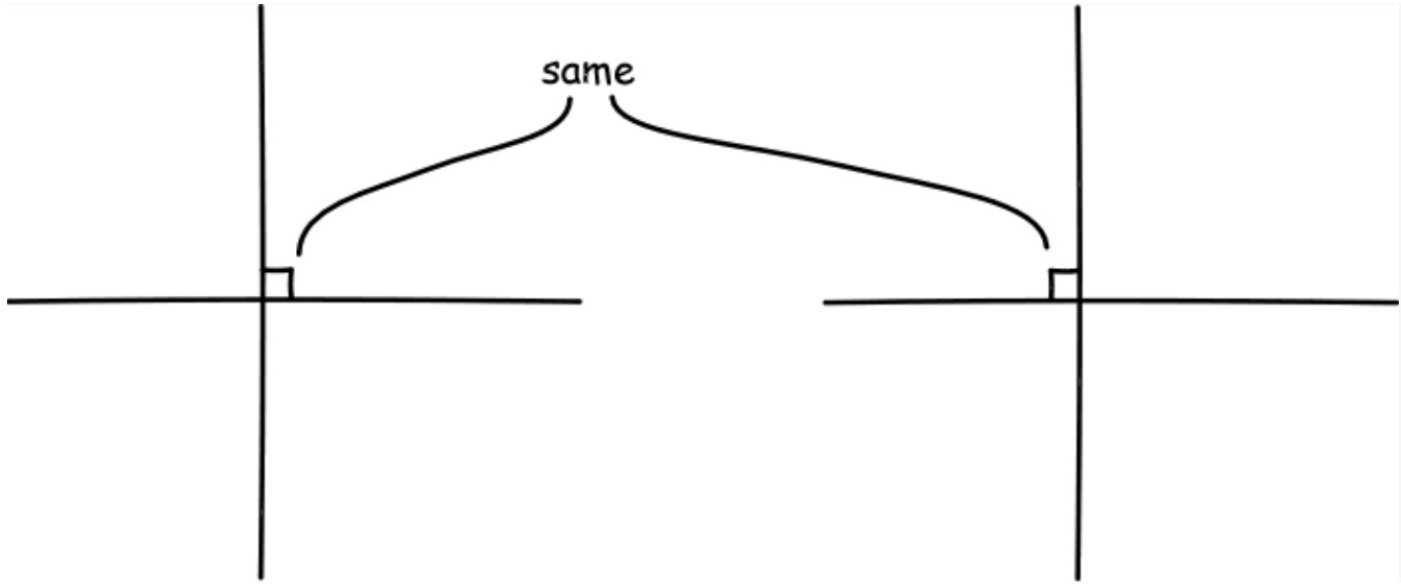


Figure: https://architectureflrc.weebly.com/euclid.html

Hyperbolic Geometry

- A non-Euclidean geometry that reject the validity of Euclid's fifth, the “parallel”, postulate.

In Euclidean space, there is only one line that paralls a line through a given point.

Figure: https://en.wikipedia.org/wiki/Poincar%C3%A9_disk_model

Hyperbolic Geometry

- A non-Euclidean geometry that reject the validity of Euclid's fifth, the “parallel”, postulate.

In Euclidean space, there is only one line that paralls a line through a given point.

In spherical space, there is no line that paralls a line through a given point.

Figure: https://en.wikipedia.org/wiki/Poincar%C3%A9_disk_model

Hyperbolic Geometry

- A non-Euclidean geometry that reject the validity of Euclid's fifth, the “parallel”, postulate.

In Euclidean space, there is only one line that paralls a line through a given point.

In spherical space, there is no line that paralls a line through a given point.

In hyperbolic space, there are at least two lines that parallel a line through a given point.

Figure: https://en.wikipedia.org/wiki/Poincar%C3%A9_disk_model

Hyperbolic Geometry

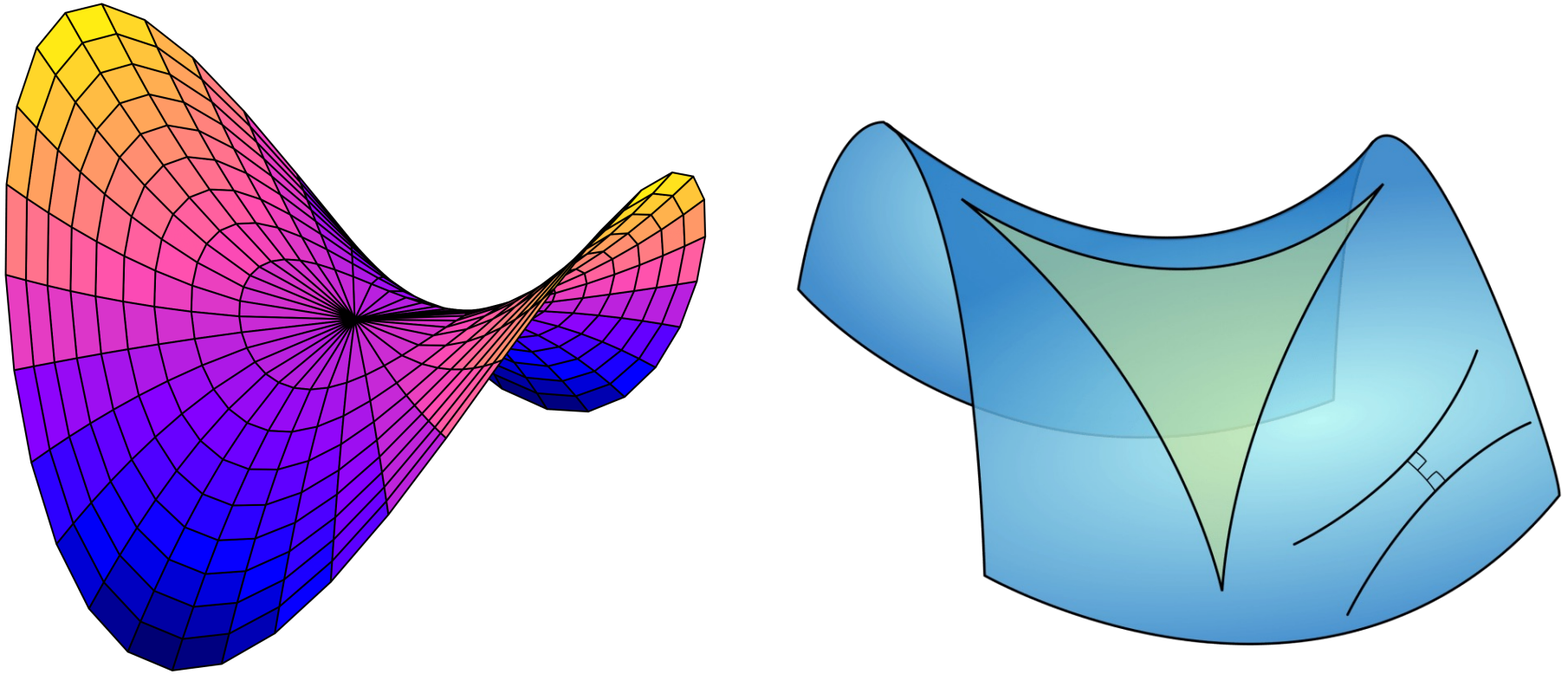
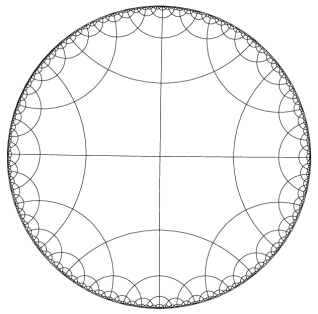


Figure: from web

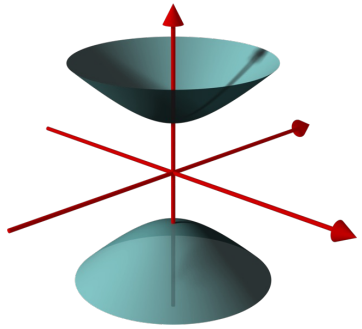
Mathematical Models for Hyperbolic Geometry

Hyperbolic Models

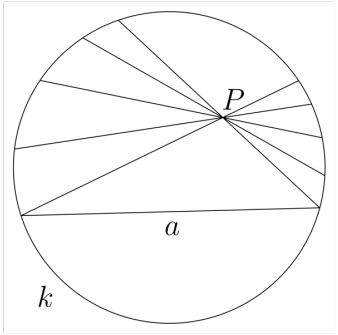
- **Poincaré Ball Model**



- **Lorentz (Hyperboloid) Model**

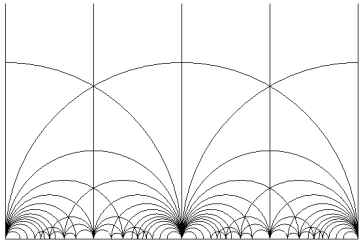


- **Klein Model**

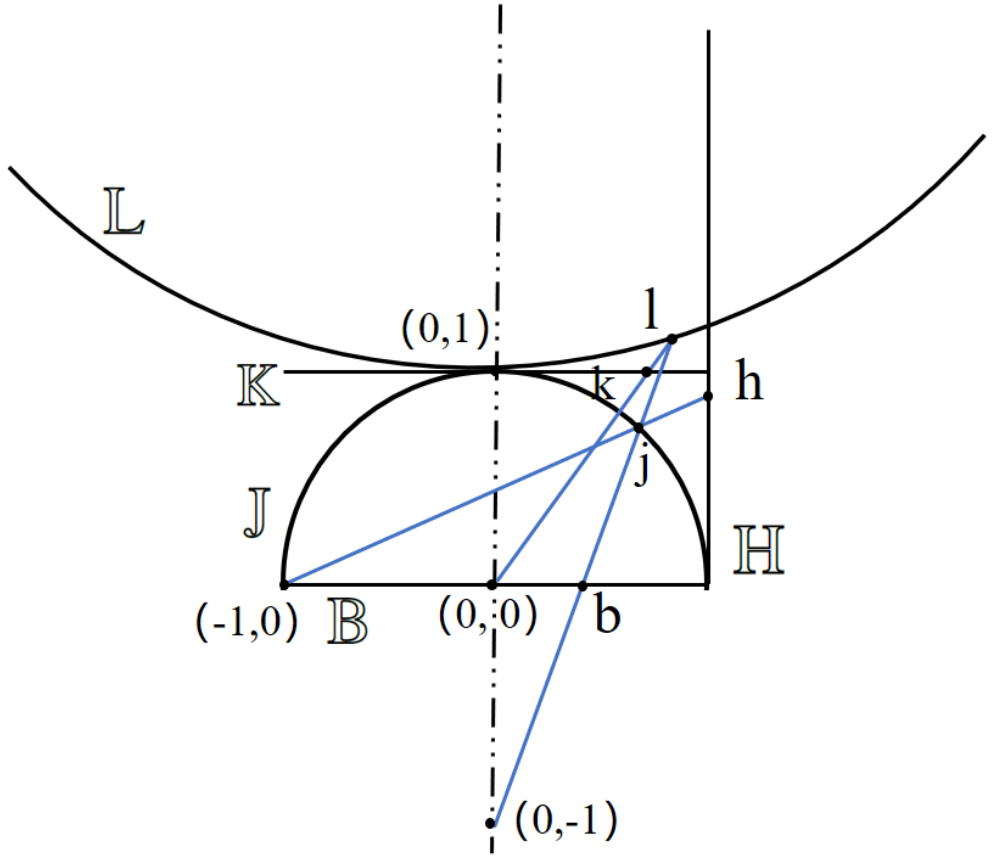


- **Hemisphere Model**

- **Poincaré Half Plane Model**



Isometrically Equivalent Models



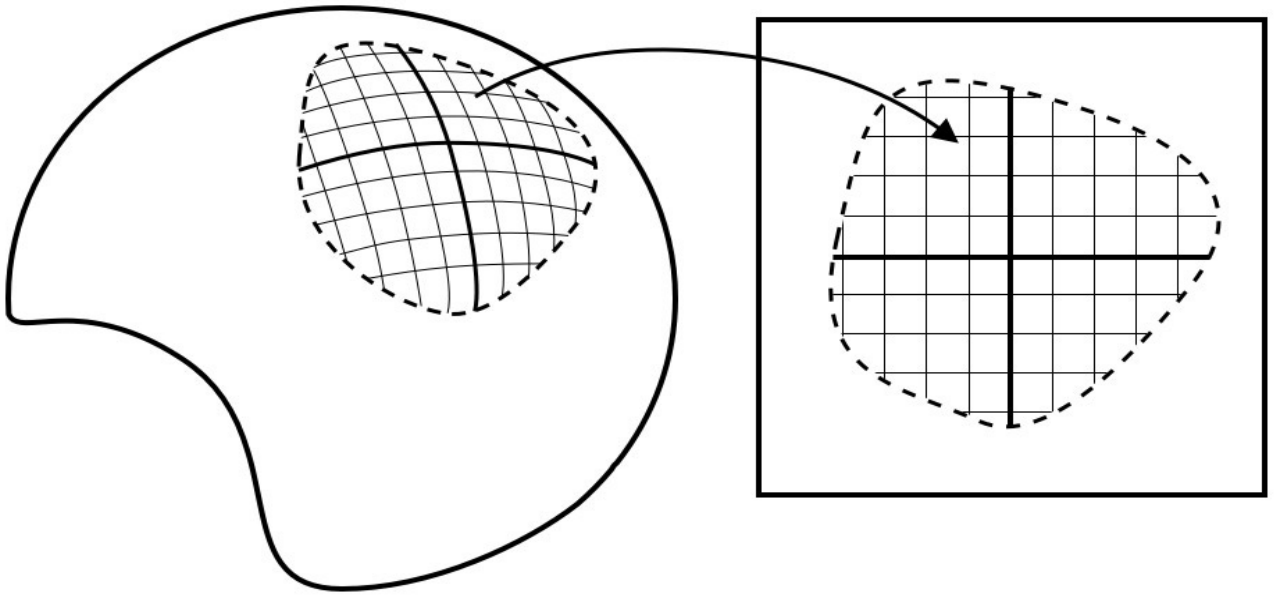
𝕃: Lorentz Model
𝕂: Klein Model
𝕁: Hemisphere model
𝔹: Poincaré Model
ℍ: Poincaré half plane Model

The five hyperbolic models are isometrically equivalent and the points h, b, j, k, l can be thought of as the same point in the hyperbolic space.

Peng, Wei, et al. "Hyperbolic deep neural networks: A survey." *arXiv preprint arXiv:2101.04562* (2021).

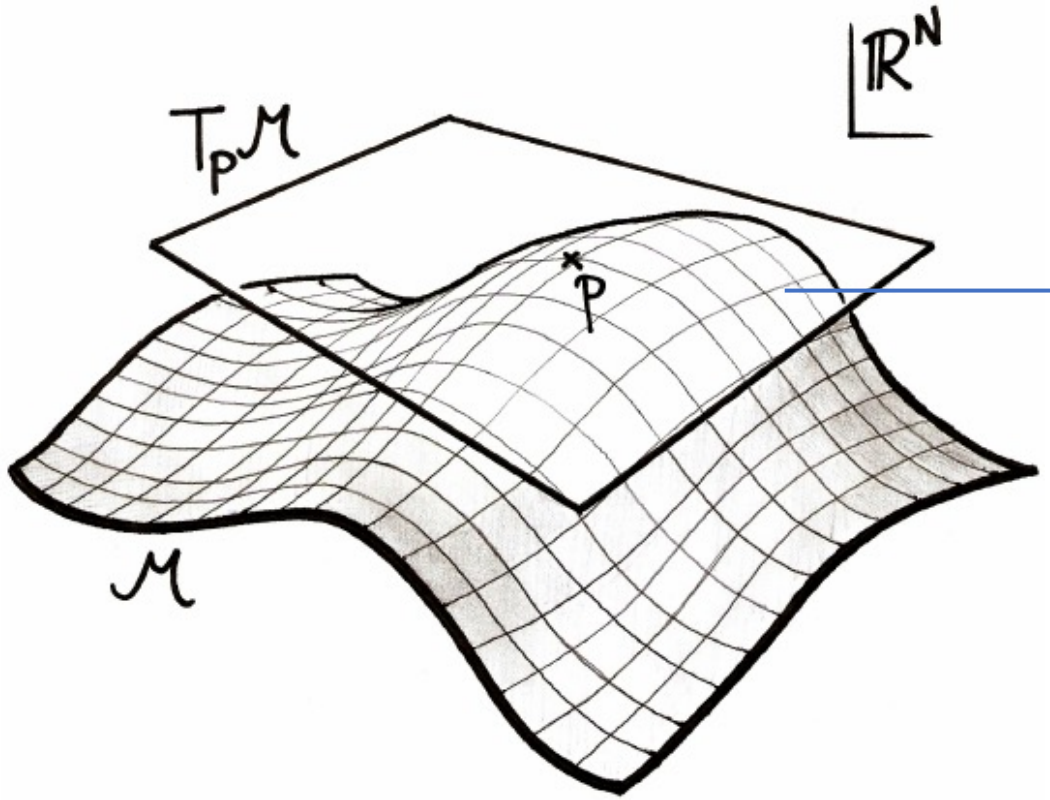
Manifold

- A manifold M of dimension n is a topological space of which each point's neighborhood can be locally approximated by the Euclidean space $\mathbb{R}^n$



Tangent space

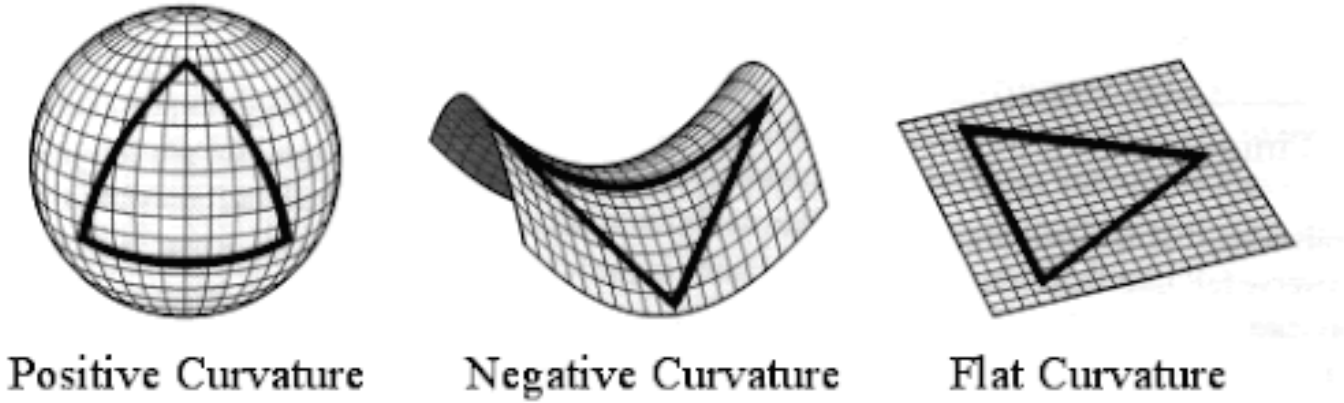
- For each point $x \in M$, the tangent space $\mathcal{T}_x M$ of M at x is defined as a n-dimensional vector-space approximating M at x at a first order.



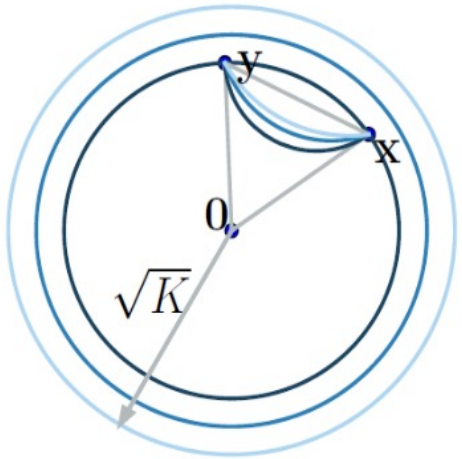
"achieve Feature transformation in neural network"

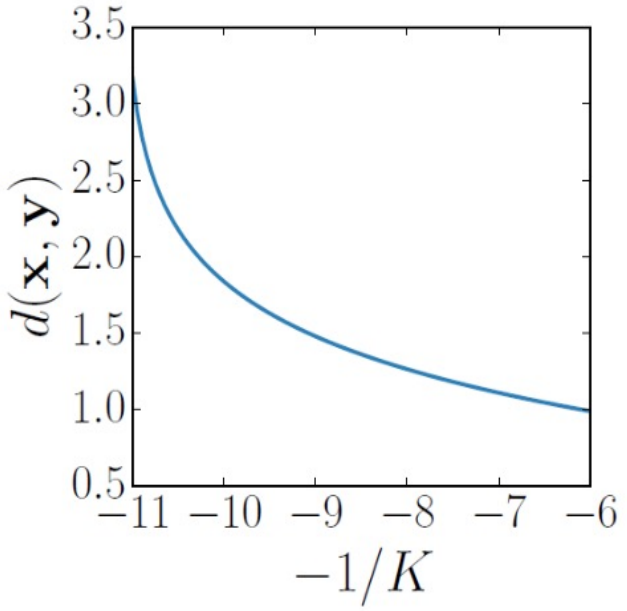
Curvature

- The curvature describes how much a geometric object deviates from the flat case. For example, a curve deviates from being a straight line, or a surface deviates from being a plane.



Negative curvature means that volumes grow faster than in the Euclidean space, and the positive curvature is the opposite of volumes growing slower





Dark blue: high curvature boundary and geodesics
Light blue: low curvature boundary and geodesics

- Hyperbolic Graph Convolutional Neural Networks, NeurIPS 2019

Riemannian metric

- For a manifold M, a Riemannian metric $g_M(x)$is a smooth collection of **inner products** on the associated tangent space $\mathcal{T}_x M$.
- For example, the Euclidean metric is

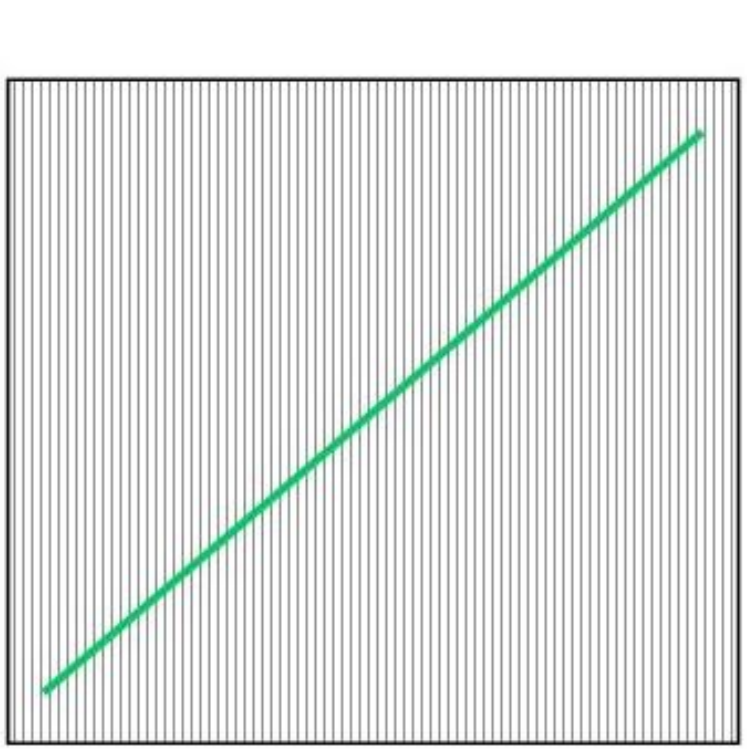
$$g_{E(x)} = I_n$$

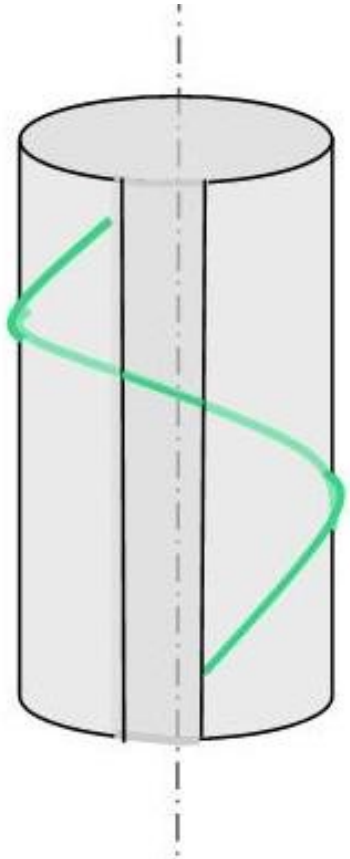
$$I_n = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix}$$

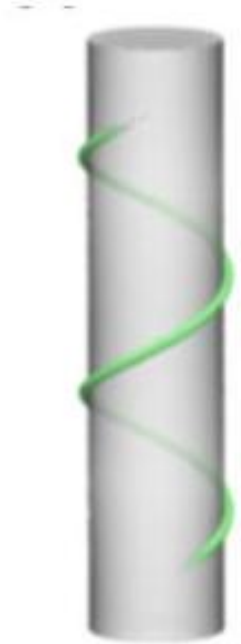
64

Geodesics

- Geodesics is the generalization of a straight line in the Euclidean space.







65

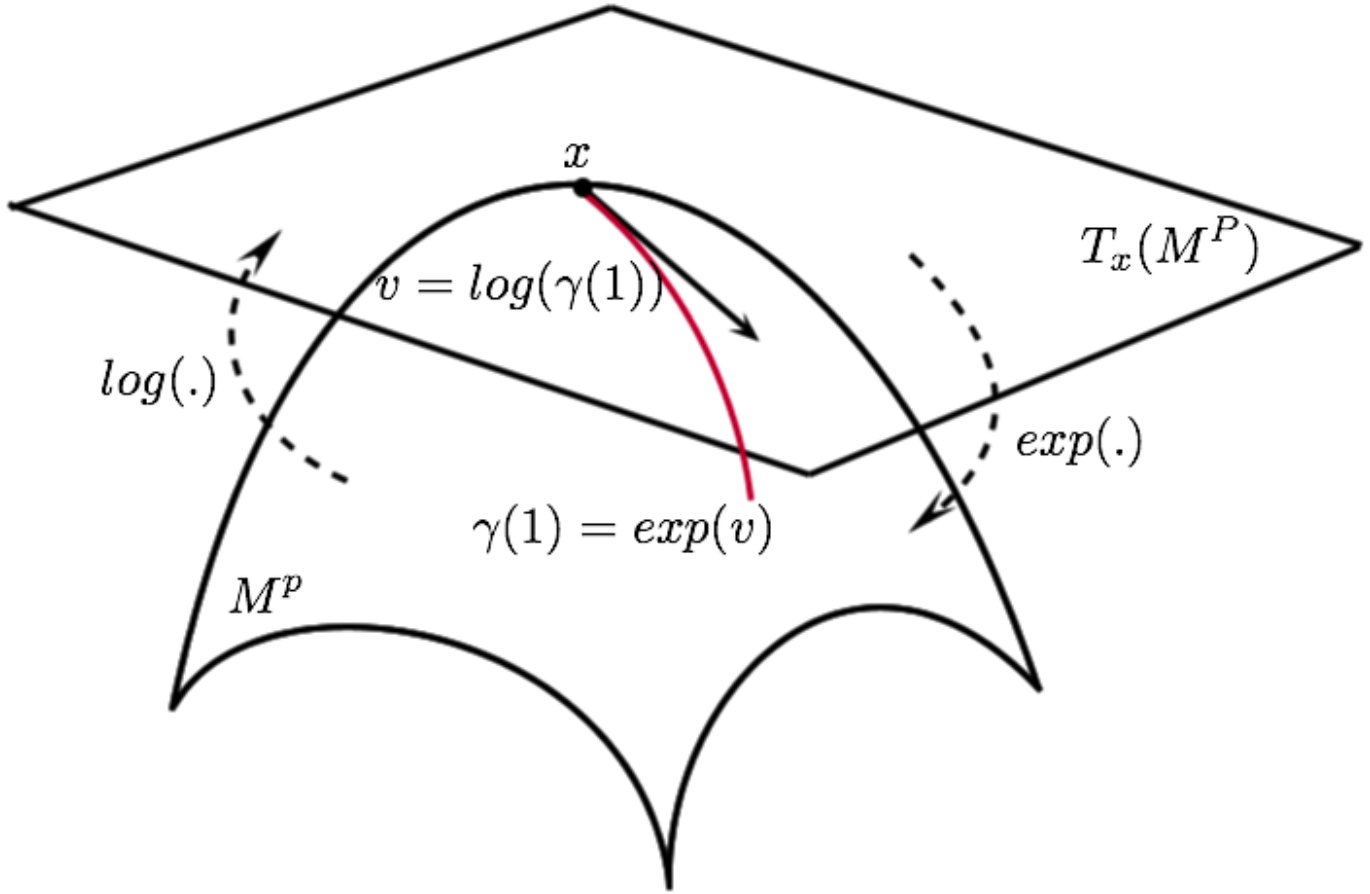
Preliminary

Exponential map

- $\mathcal{T}_x M \rightarrow M$

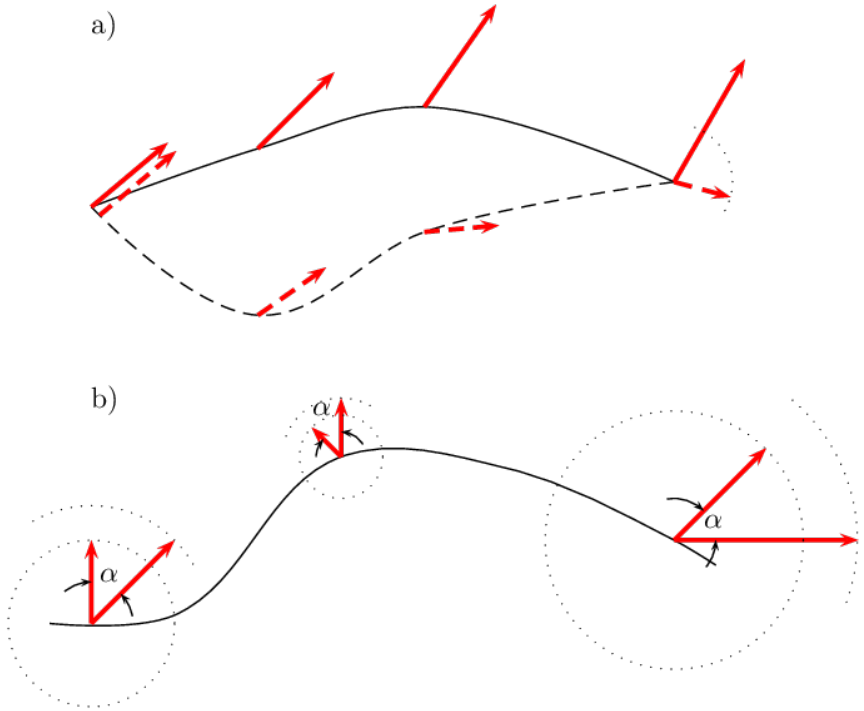
Logarithmic map

- $M \rightarrow \mathcal{T}_x M$



Parallel Transport

- Parallel Transport defines a way of transporting the local geometry along smooth curves that preserves the metric tensors



Preliminary

Table 6. Summary of operations in the Poincaré ball model and the Lorentz model ($\kappa < 0$)

	Poincaré Ball Model	**Lorentz Model (Hyperboloid Model)**
Manifold	$\mathcal{B}_\kappa^n = \left\{\mathbf{x} \in \mathbb{R}^n : \langle \mathbf{x}, \mathbf{x} \rangle_2 < -\frac{1}{\kappa}\right\}$	$\mathcal{L}_\kappa^n = \left\{\mathbf{x} \in \mathbb{R}^{n+1} : \langle \mathbf{x}, \mathbf{x} \rangle_{\mathcal{L}} = \frac{1}{\kappa}\right\}$
Metric	$g_{\mathbf{x}}^{\mathcal{B}_\kappa} = (\lambda_{\mathbf{x}}^\kappa)^2 \mathbf{I}_n$ where $\lambda_{\mathbf{x}}^\kappa = \frac{2}{1+\kappa\|\mathbf{x}\|_2^2}$	$g_{\mathbf{x}}^{\mathcal{L}_\kappa} = \eta$, where η is I except $\eta_{0,0} = -1$
Distance	$d_{\mathcal{B}}^\kappa(\mathbf{x}, \mathbf{y}) = \frac{1}{\sqrt{\lvert\kappa\rvert}} \cosh^{-1}\left(1 - \frac{2\kappa\|\mathbf{x}-\mathbf{y}\|_2^2}{(1+\kappa\|\mathbf{x}\|_2^2)(1+\kappa\|\mathbf{y}\|_2^2)}\right)$	$d_{\mathcal{L}}^\kappa(\mathbf{x}, \mathbf{y}) = \frac{1}{\sqrt{\lvert\kappa\rvert}} \cosh^{-1}\left(\kappa\langle \mathbf{x}, \mathbf{y} \rangle_{\mathcal{L}}\right)$
Logarithmic Map	$\log_{\mathbf{x}}^\kappa(\mathbf{y}) = \frac{2}{\sqrt{\lvert\kappa\rvert}\lambda^\kappa} \tanh^{-1}\left(\sqrt{\lvert\kappa\rvert}\,\|-\mathbf{x} \oplus_\kappa \mathbf{y}\|_2\right) \frac{-\mathbf{x}\oplus_\kappa \mathbf{y}}{\|-\mathbf{x}\oplus_\kappa \mathbf{y}\|_2}$	$\log_{\mathbf{x}}^\kappa(\mathbf{y}) = \frac{\cosh^{-1}(\kappa\langle \mathbf{x},\mathbf{y}\rangle_{\mathcal{L}})}{\sinh\left(\cosh^{-1}(\kappa\langle \mathbf{x},\mathbf{y}\rangle_{\mathcal{L}})\right)} (\mathbf{y} - \kappa\langle \mathbf{x}, \mathbf{y}\rangle_{\mathcal{L}}\mathbf{x})$
Exponential Map	$\exp_{\mathbf{x}}^\kappa(\mathbf{v}) = \mathbf{x} \oplus_\kappa \left(\tanh\left(\sqrt{\lvert\kappa\rvert}\frac{\lambda_{\mathbf{x}}^\kappa\|\mathbf{v}\|_2}{2}\right)\frac{\mathbf{v}}{\sqrt{\lvert\kappa\rvert}\|\mathbf{v}\|_2}\right)$	$\exp_{\mathbf{x}}^\kappa(\mathbf{v}) = \cosh\left(\sqrt{\lvert\kappa\rvert}\|\mathbf{v}\|_{\mathcal{L}}\right)\mathbf{x} + \mathbf{v}\frac{\sinh\left(\sqrt{\lvert\kappa\rvert}\|\mathbf{v}\|_{\mathcal{L}}\right)}{\sqrt{\lvert\kappa\rvert\|\mathbf{v}\|_{\mathcal{L}}}}$
Parallel Transport	$PT_{\mathbf{x}\to\mathbf{y}}^\kappa(\mathbf{v}) = \frac{\lambda_{\mathbf{x}}^\kappa}{\lambda_{\mathbf{y}}^\kappa}\,\mathrm{gyr}[\mathbf{y}, -\mathbf{x}]v$	$PT_{\mathbf{x}\to\mathbf{y}}^\kappa(\mathbf{v}) = \mathbf{v} - \frac{\kappa\langle \mathbf{y},\mathbf{v}\rangle_{\mathcal{L}}}{1+\kappa\langle \mathbf{x},\mathbf{y}\rangle_{\mathcal{L}}}(\mathbf{x}+\mathbf{y})$
Origin Point	$\mathbf{0}_n$	$[\frac{1}{\sqrt{\lvert\kappa\rvert}}, \mathbf{0}_n]$

YANG Menglin et.al. Hyperbolic Representation Learning: Revisiting and Advancing. ICML 2023

Agenda

Part 1. INTRODUCTION (Min, 9:00 - 9:30)

1.1 An Overview of Graph Learning

1.2 Brief Introduction of Riemannian Geometry

1.3 Motivation for Hyperbolic Graph Learning

Coffee Break (9:30 – 10:00)

Part 2. HYPERBOLIC GRAPH NETWORKS (Menglin, 10:00 – 10:30)

2.1 Preliminary

2.2 Hyperbolic Shallow Models

2.2 Hyperbolic Neural Networks

2.3 Hyperbolic Graph Neural Networks

Part 3. APPLICATIONS (Menglin, 10:30 – 11:00)

3.1 HGNN for Recommendation Systems

3.2 HGNN for Knowledge Graph

3.3 HGNN for Healthcare

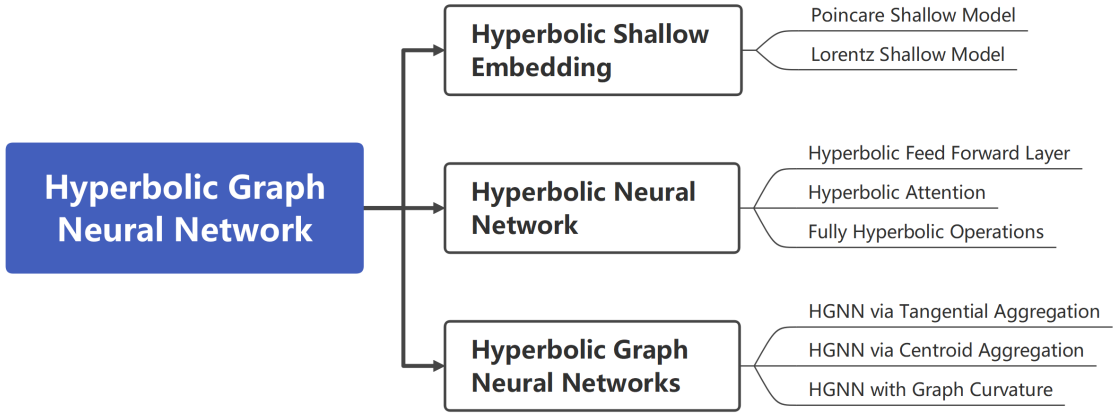
Part 4. ADVANCED TOPICS (Bo, 11:00 – 12:00)

4.1 Complex Structures

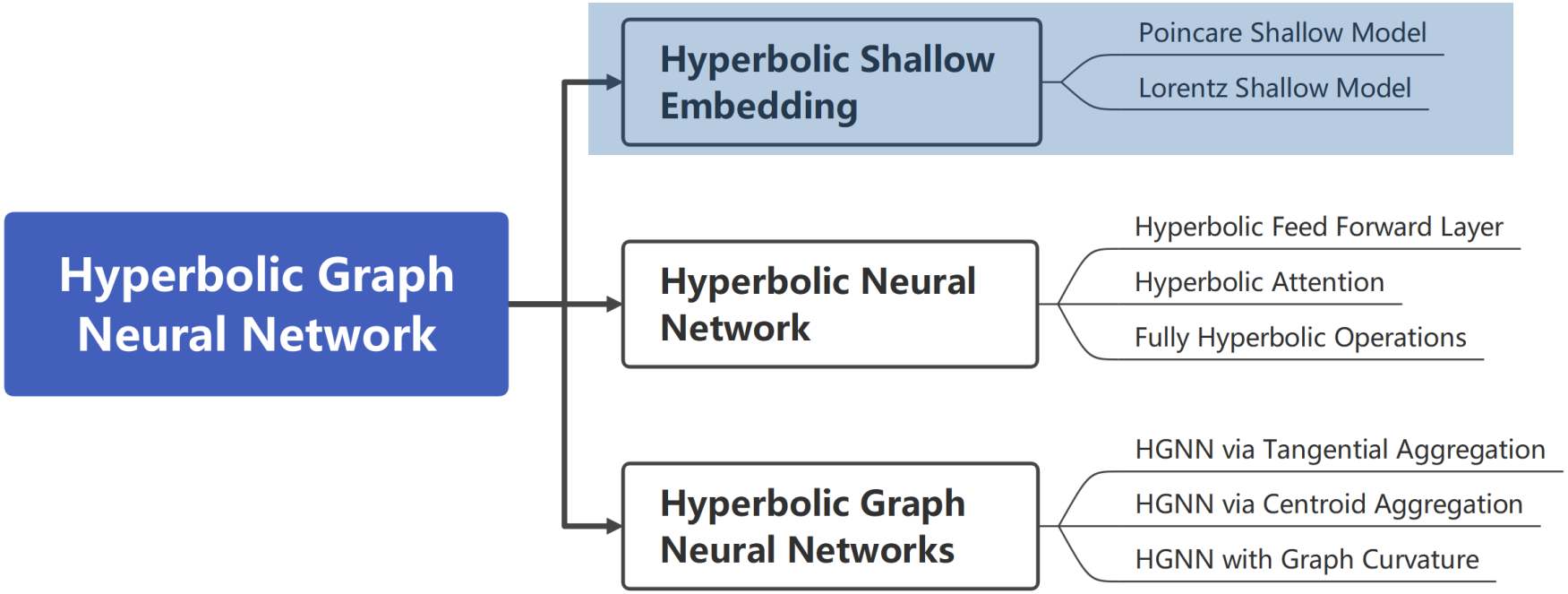
4.2 Curvature-aware Learning

4.3 Evolving Interactions

4.4 Trustworthy and Scalability

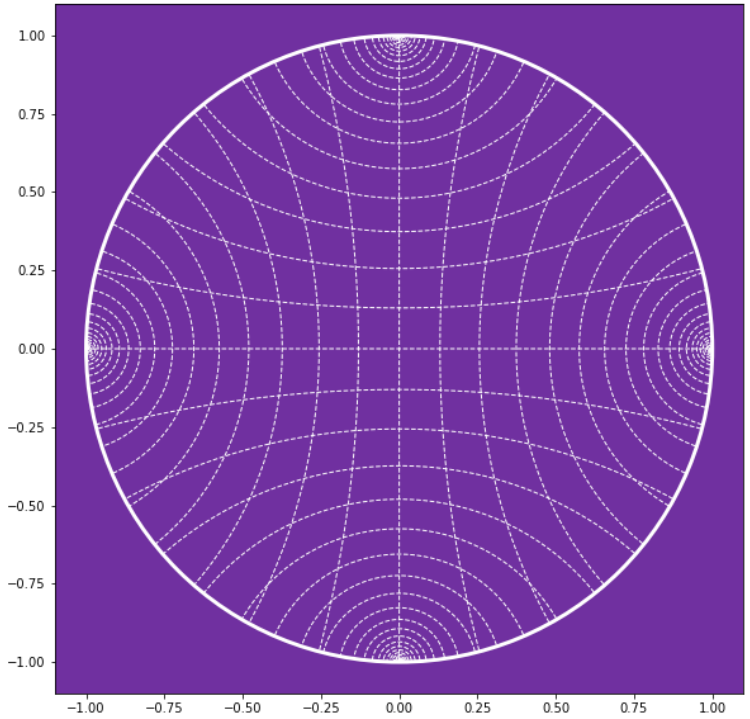


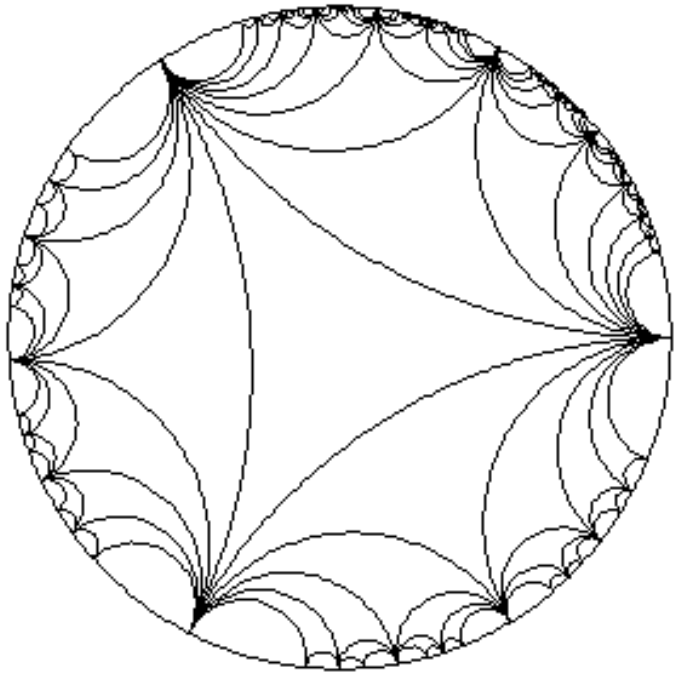
Hyperbolic Shallow Models

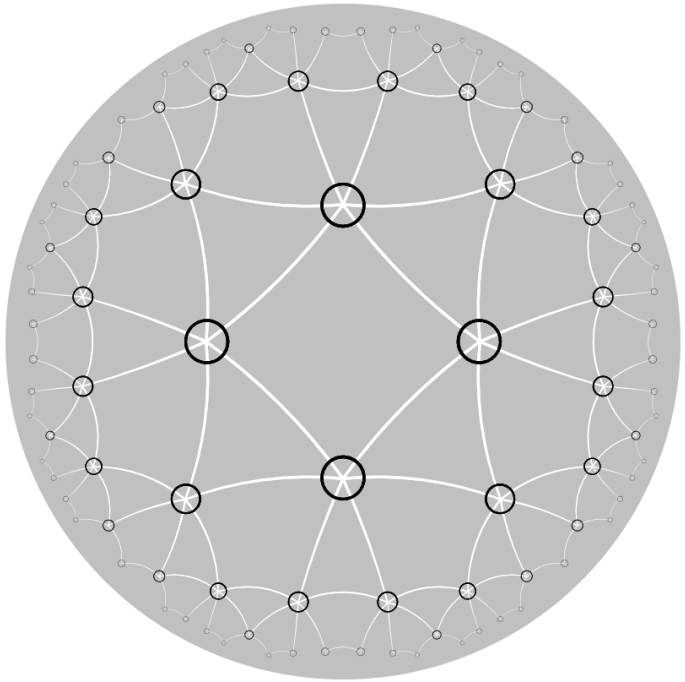


Poincaré Shallow Model

Poincaré Ball Model







Radius proportional to K (inverse of curvature), Open ball (exclude boundary)
In the right two figure, each triangle/circle has the same area.

https://mathworld.wolfram.com/PoincareHyperbolicDisk.html

Poincaré Shallow Model

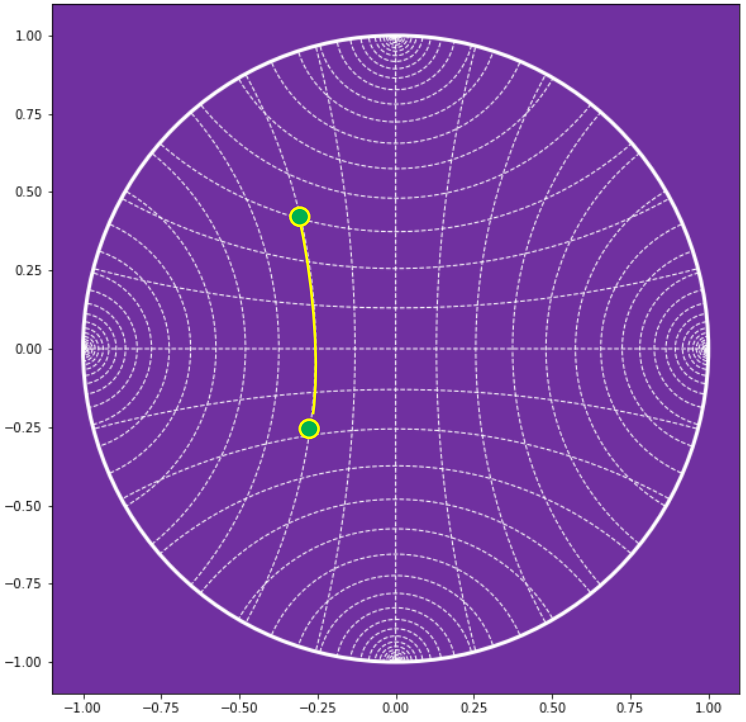
Poincaré Ball Model (Intuitive Visualization)

❖ **Definition** $\mathcal{B}^d = \{x \in \mathbb{R}^d \big| ||x|| < 1\}$,

where $||x|| = \sqrt{\sum_{i=1}^{d} x_i^2}$, a Euclidean norm.

❖ **Distance** between $u, v \in \mathcal{B}^d$:

$$d_{\mathcal{B}}(u,v) = arcosh\left(1 + 2 \cdot \frac{||u-v||^2}{(1-||u||^2)(1-||v||^2)}\right).$$



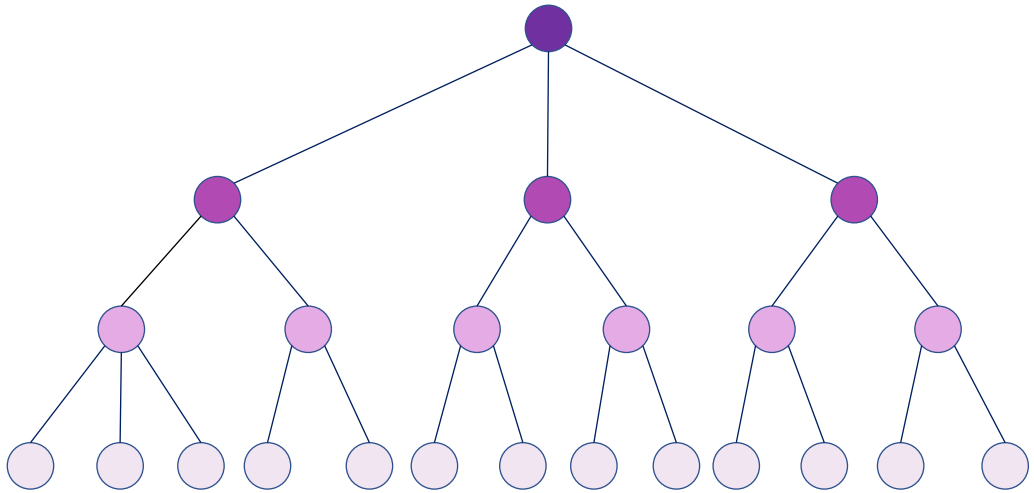
2-dimensional Poincaré ball

Poincaré Shallow Model

Poincaré Shallow Embedding

❖ **Problem** We have a tree-like graph with set of vertices $\mathcal{N} = \{\boldsymbol{x}_i\}_i^n$, and we want to find embeddings $\Theta = \{\boldsymbol{\theta}_i\}_i^n$, that they

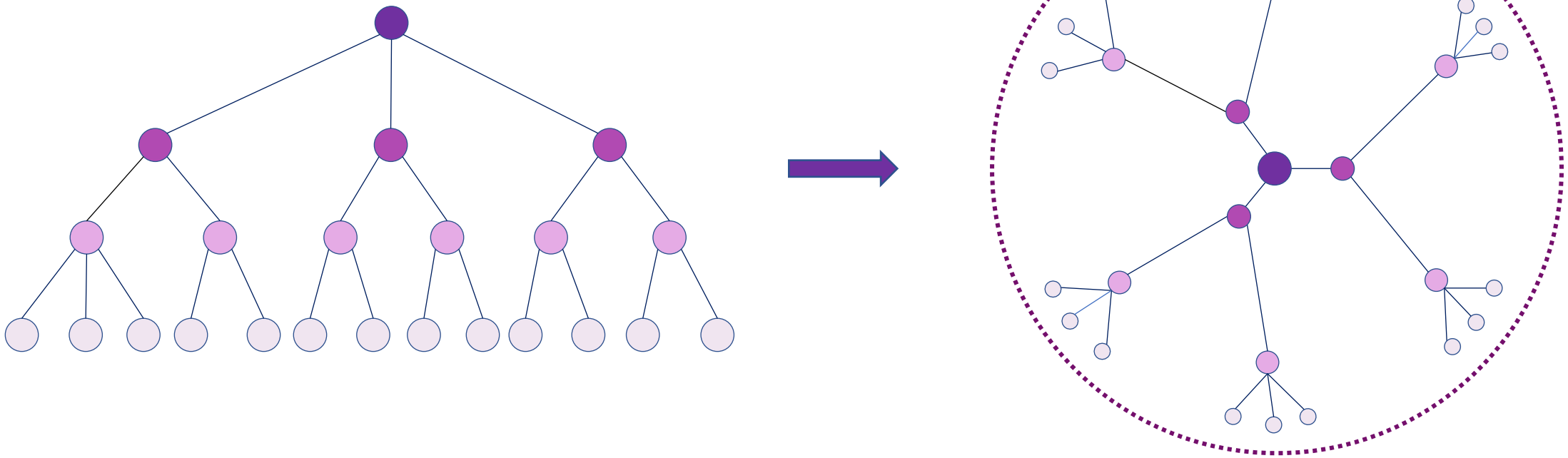
- capture the similarity between vertices (including semantics and structures)
- encode the latent hierarchical structure
- have feasible runtime and memory complexity



Poincaré Shallow Model

Poincaré Shallow Embedding

❖ **Problem** We have a tree-like graph with set of vertices $\mathcal{N} = \{\boldsymbol{x}_i\}_i^n$, and we want to find embeddings $\Theta = \{\boldsymbol{\theta}_i\}_i^n$, that they



Poincaré Shallow Model

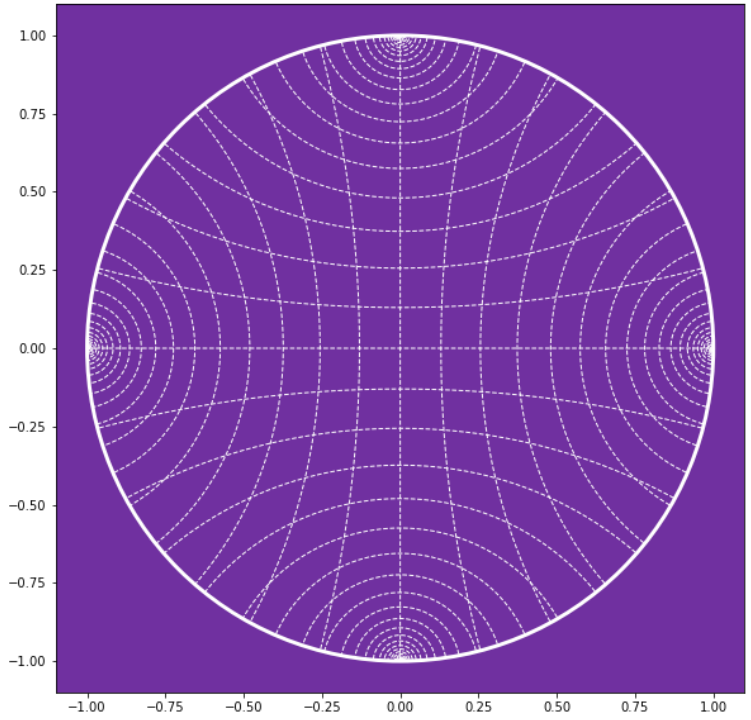
Poincaré Shallow Embedding

❖**Objective**

$$\Theta^* \leftarrow argmin_\theta \, L(\Theta),$$

$$\boldsymbol{s.t.} \, \forall \boldsymbol{\theta_i} \in \boldsymbol{\Theta}: \left|\left|\boldsymbol{\theta_i}\right|\right| < 1$$

where $L(\boldsymbol{\theta})$ is task-specific loss function.



2-dimensional Poincaré ball

Poincaré Shallow Model

Poincaré Shallow Embedding

1. **Initialization**: all node are projected to Poincaré Ball.

   1) Without node feature: randomly using uniform distribution

   2) With node feature: using exponential map

2. **Update** $\Theta: \boldsymbol{\theta_{t+1}} \rightarrow \text{proj}\left(\boldsymbol{\theta_t} - \eta_t \cdot \frac{\left(1-\left|\left|\boldsymbol{\theta_t}\right|\right|^2\right)^2}{4} \boldsymbol{\nabla_E}\right)$

where

$$\boldsymbol{\Delta_E} = \frac{\boldsymbol{\partial L(\theta_t)}}{\boldsymbol{\partial\left(d(\theta_t, x)\right)}} \cdot \frac{\boldsymbol{\partial d(\theta_t, x)}}{\boldsymbol{\partial \theta_t}} \qquad \text{proj}(\boldsymbol{\theta_t}) = \begin{cases} \frac{\boldsymbol{\theta_t}}{\left|\left|\boldsymbol{\theta_t}\right|\right|} - \boldsymbol{\epsilon\ if}\ \left|\left|\boldsymbol{\theta_t}\right|\right| \geq 1 \\ \boldsymbol{\theta_t} \quad \boldsymbol{otherwise} \end{cases}$$

Nickel, Maximillian, and Douwe Kiela. "Poincaré embeddings for learning hierarchical representations." *NeurIPS* 30 (2017).

Poincaré Shallow Model

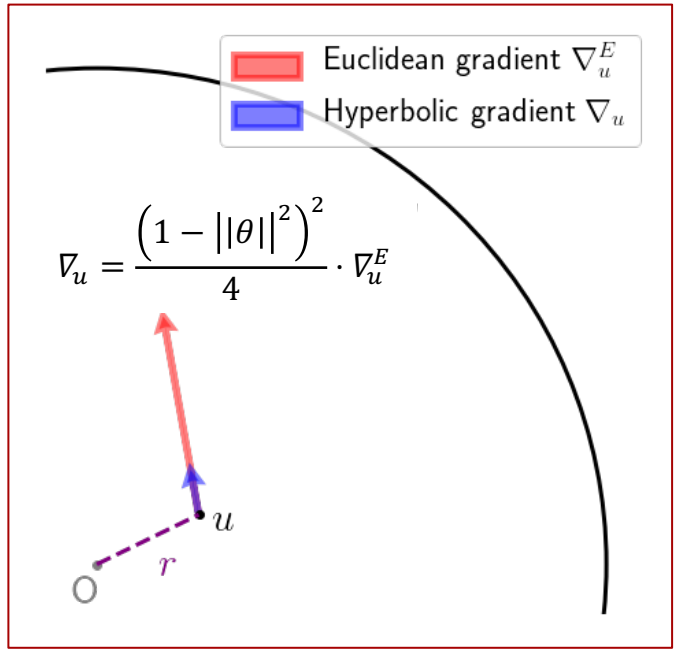
Poincaré Shallow Embedding

1. **Initialization**: all node are projected to Poincaré Ball.

   1) Without node feature: randomly using uniform distribution

   2) With node feature: using exponential map

2. **Update** $\Theta: \boldsymbol{\theta_{t+1}} \rightarrow \text{proj}\left(\boldsymbol{\theta_t} - \eta_t \cdot \frac{\left(1-\left|\left|\boldsymbol{\theta_t}\right|\right|^2\right)^2}{4} \boldsymbol{\nabla_E}\right)$



where

$$\boldsymbol{\Delta_E} = \frac{\boldsymbol{\partial L(\theta_t)}}{\boldsymbol{\partial\left(d(\theta_t, x)\right)}} \cdot \frac{\boldsymbol{\partial d(\theta_t, x)}}{\boldsymbol{\partial \theta_t}} \qquad \text{proj}(\boldsymbol{\theta_t}) = \begin{cases} \frac{\boldsymbol{\theta_t}}{\left|\left|\boldsymbol{\theta_t}\right|\right|} - \boldsymbol{\epsilon} \; \boldsymbol{if} \; \left|\left|\boldsymbol{\theta_t}\right|\right| \geq \mathbf{1} \\ \boldsymbol{\theta_t} \quad \boldsymbol{otherwise} \end{cases}$$

Nickel, Maximillian, and Douwe Kiela. "Poincaré embeddings for learning hierarchical representations." *NeurIPS* 30 (2017).

Results on Poincaré Ball Embedding

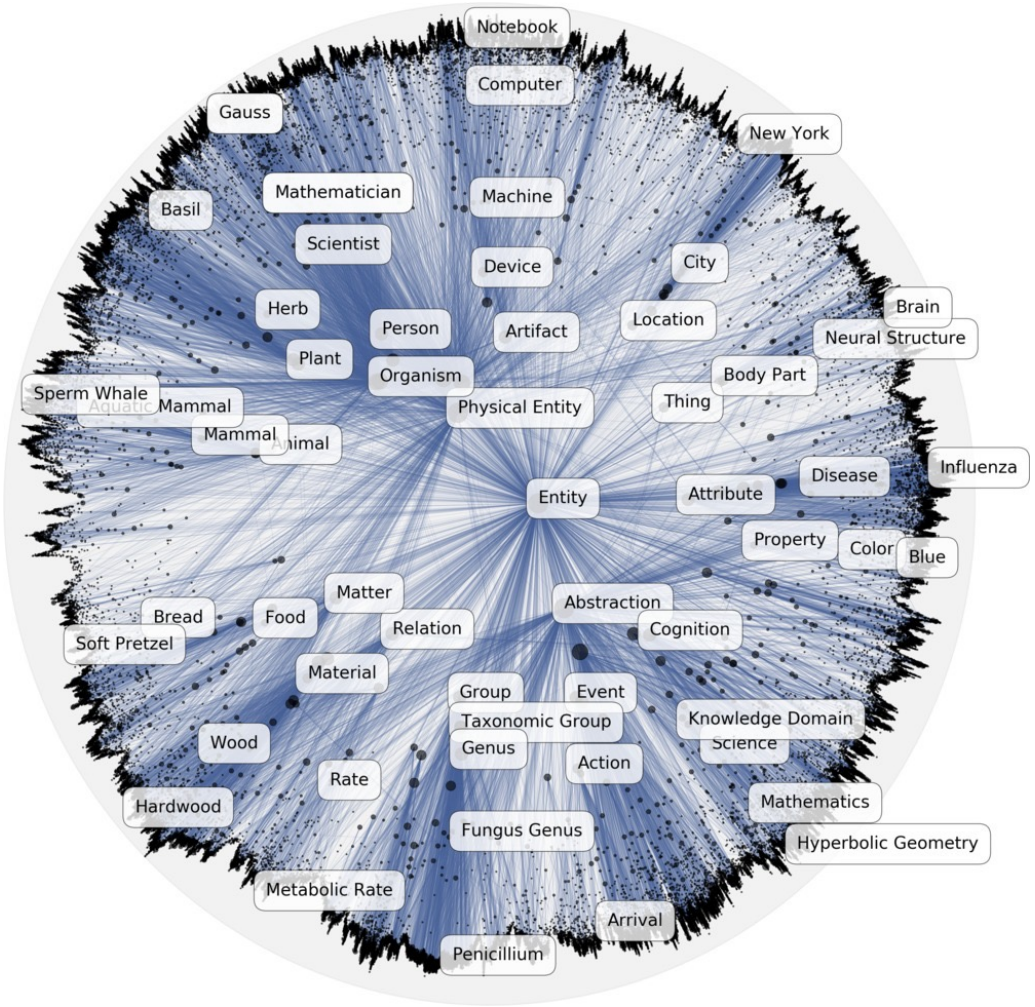


Table: Experimental results on the transitive closure of the WORDNET noun hierarchy. Highlighted cells indicate the best Euclidean embeddings as well as the Poincaré embeddings which achieve equal or better results. Bold numbers indicate absolute best results.

			Dimensionality					
			5	10	20	50	100	200
WORDNET Reconstruction	**Euclidean**	Rank	3542.3	2286.9	1685.9	1281.7	1187.3	1157.3
		MAP	0.024	0.059	0.087	0.140	0.162	0.168
	Translational	Rank	205.9	179.4	95.3	92.8	92.7	91.0
		MAP	0.517	0.503	0.563	0.566	0.562	0.565
	Poincaré	Rank	4.9	4.02	3.84	3.98	3.9	**3.83**
		MAP	0.823	0.851	0.855	0.86	0.857	**0.87**
WORDNET Link Pred.	**Euclidean**	Rank	3311.1	2199.5	952.3	351.4	190.7	81.5
		MAP	0.024	0.059	0.176	0.286	0.428	0.490
	Translational	Rank	65.7	56.6	52.1	47.2	43.2	40.4
		MAP	0.545	0.554	0.554	0.56	0.562	0.559
	Poincaré	Rank	5.7	**4.3**	4.9	4.6	4.6	4.6
		MAP	0.825	0.852	0.861	**0.863**	0.856	0.855

Nickel, Maximillian, and Douwe Kiela. "Poincaré embeddings for learning hierarchical representations." *NeurIPS* 30 (2017).

Lorentz Shallow Models

Lorentz Model

(Numerically more stable, simpler formula)

❖ **Definition** $\mathcal{L}^d = \{x \in \mathbb{R}^{d+1} : < x, x >_{\mathcal{L}} = -1, x_0 > 0\}$,

where $< x, y >_{\mathcal{L}} := -x_0 y_0 + x_1 y_1 + \cdots + x_d y_d$.

Time-like Space-like

❖ **Distance** between $u, v \in \mathcal{L}^d$:

$$d_{\mathcal{L}}(u, v) = arcosh(- < x, y >_{\mathcal{L}}).$$

Lorentz Shallow Models

Lorentz Model

(Numerically more stable, simpler formula)

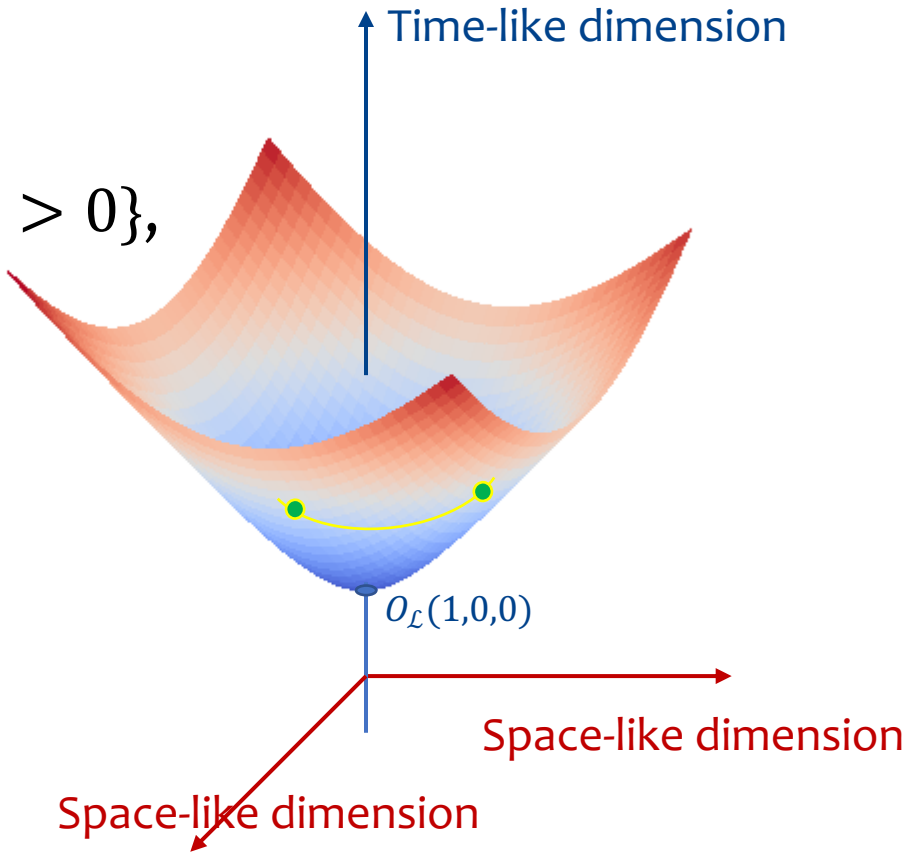
❖ **Definition** $\mathcal{L}^d = \{x \in \mathbb{R}^{d+1} : < x, x >_{\mathcal{L}} = -1, x_0 > 0\}$,

where $< x, y >_{\mathcal{L}} := \boxed{-x_0 y_0} + \boxed{x_1 y_1 + \cdots + x_d y_d}$.

Time-like Space-like

❖ **Distance** between $u, v \in \mathcal{L}^d$:

$$d_{\mathcal{L}}(u, v) = arcosh(-< x, y >_{\mathcal{L}}).$$



2-dimensional Lorentz Model

Lorentz Shallow Models

Lorentz Shallow Embedding

❖ **Problem** We have a tree-like graph with set of vertices $\mathcal{N} = \{x_i\}_i^n$, and we want to find embeddings $\Theta = \{\theta_i\}_i^n$, that they

- capture the similarity between vertices (including semantics and structure);
- encode the latent hierarchical structure;
- have feasible runtime and memory complexity.

Lorentz Shallow Models

Lorentz Shallow Embedding

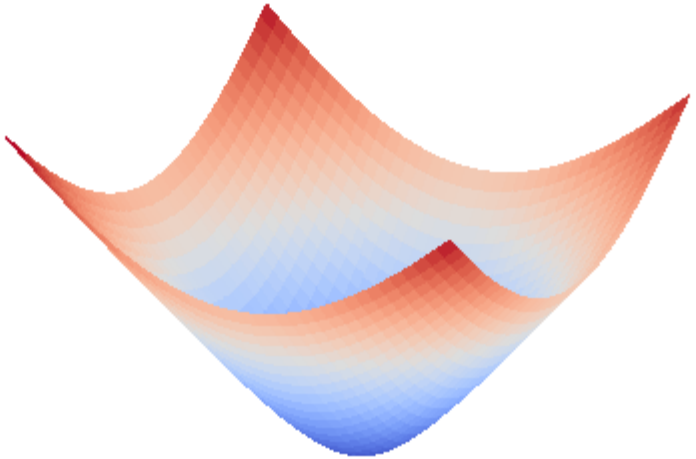
❖ **Objective**

$$\theta^* \leftarrow \arg min_\theta \, L(\boldsymbol{\Theta}),$$

$$s.t. \; \forall \boldsymbol{\theta_i} \in \boldsymbol{\Theta}: \boldsymbol{\theta_i} \in \mathcal{L}^{\boldsymbol{d}}$$

$$\boldsymbol{\theta_{i,0}^2 + \theta_{i,1}^2 + \cdots + \theta_{i,d}^2 = -1}$$

where $L(\theta)$ is task-specific loss function.



2-dimensional Lorentz Model

Lorentz Shallow Models

Lorentz Shallow Embedding

1. **Initialization:** all node are projected to Lorentz Model.

   1) Without node feature: randomly using uniform distribution

   2) With node feature: using exponential map

2. **Update Θ :** $\theta_{t+1} \rightarrow exp_{\theta_t}(-\eta_t \cdot \mathrm{grad} f(\theta_t))$,

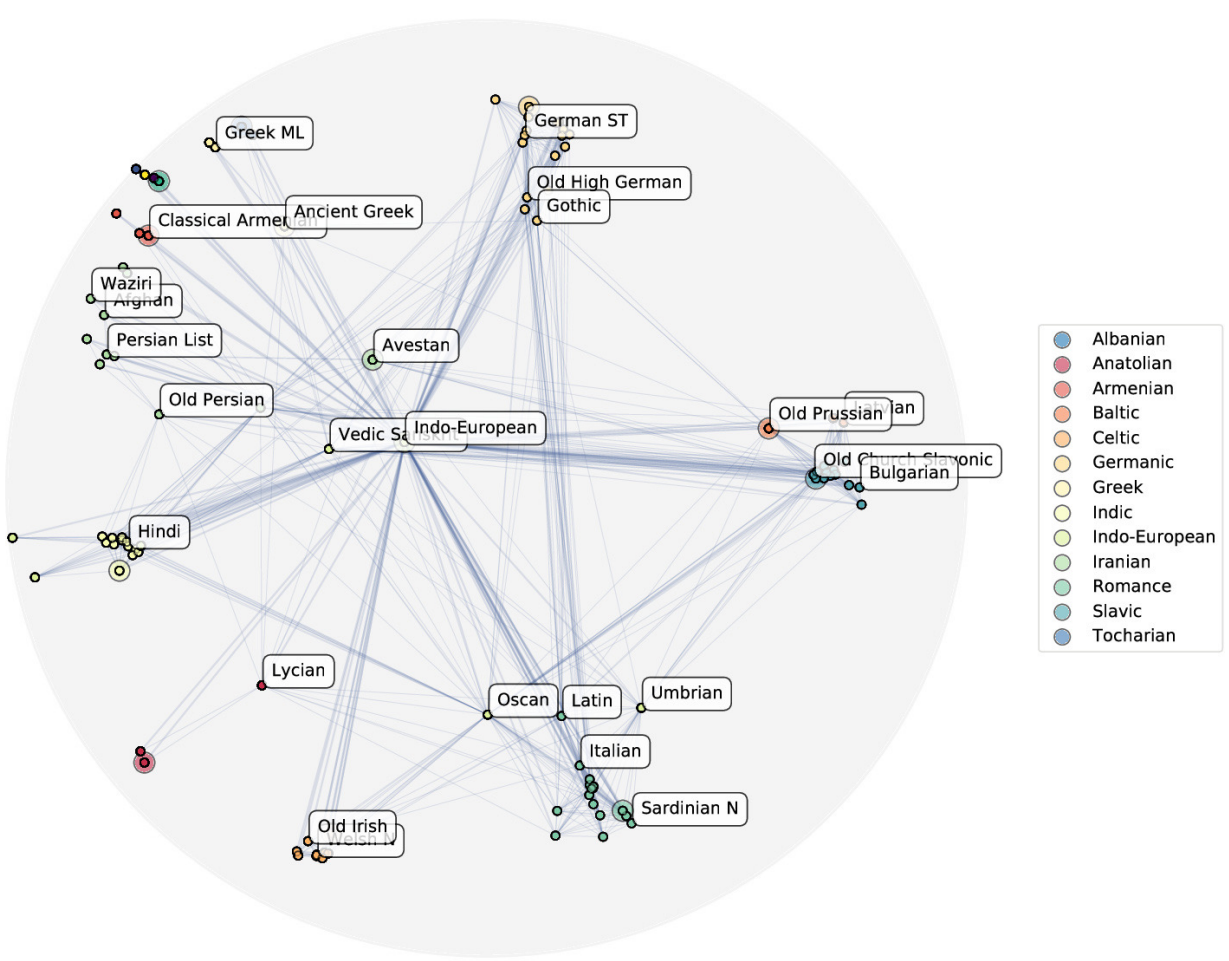
Where $\mathrm{grad}\, f(\theta_t) \in \mathcal{T}_\theta \mathcal{M}$ denotes the Riemannian gradient and η denotes the learning rate.

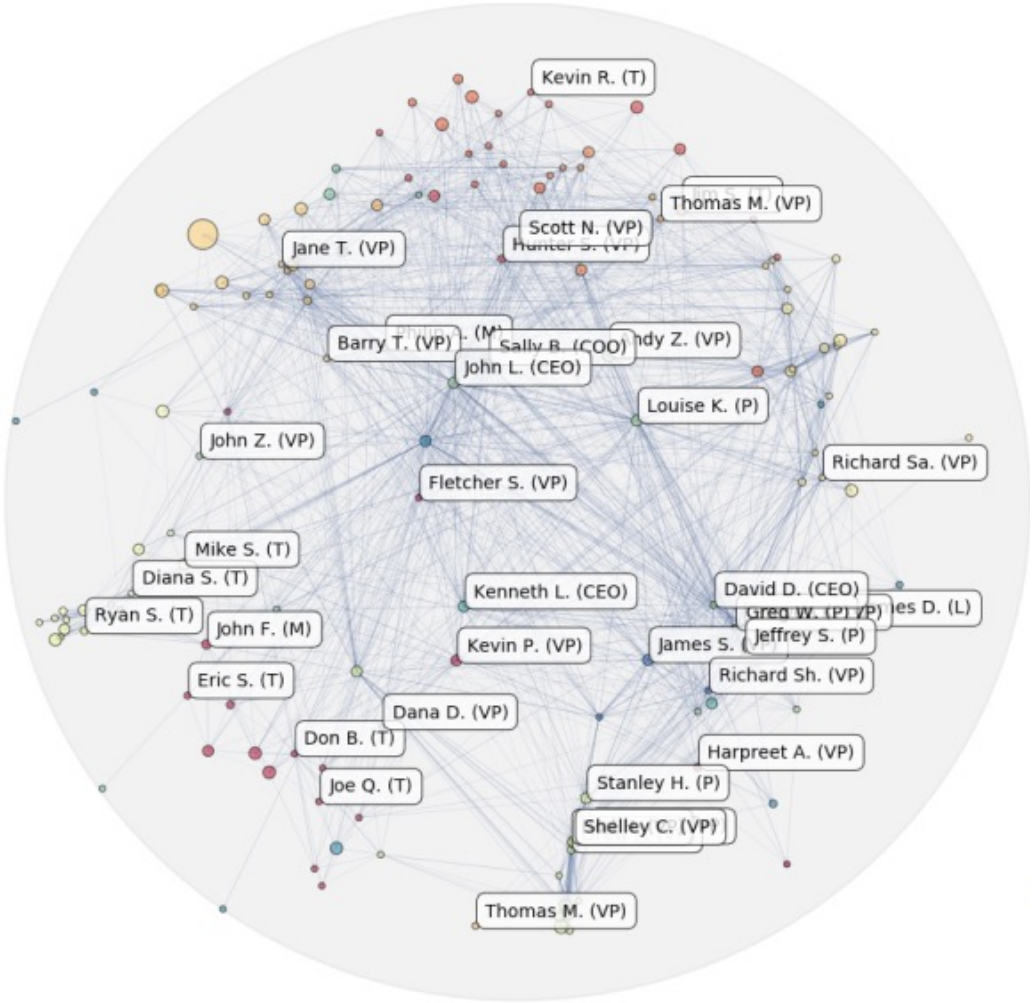
Algorithm 1 Riemannian Stochastic Gradient Descent

Input Learning rate η, number of epochs T.
for $t = 1, \ldots, T$

$$\begin{aligned} \boldsymbol{h}_t &\leftarrow g_{\theta_t}^{-1} \nabla f(\theta_t) \\ \mathrm{grad}\, f(\theta_t) &\leftarrow \mathrm{proj}_{\theta_t}(\boldsymbol{h}_t) \\ \theta_{t+1} &\leftarrow \exp_{\theta_t}(-\eta\, \mathrm{grad}\, f(\theta_t)) \end{aligned}$$

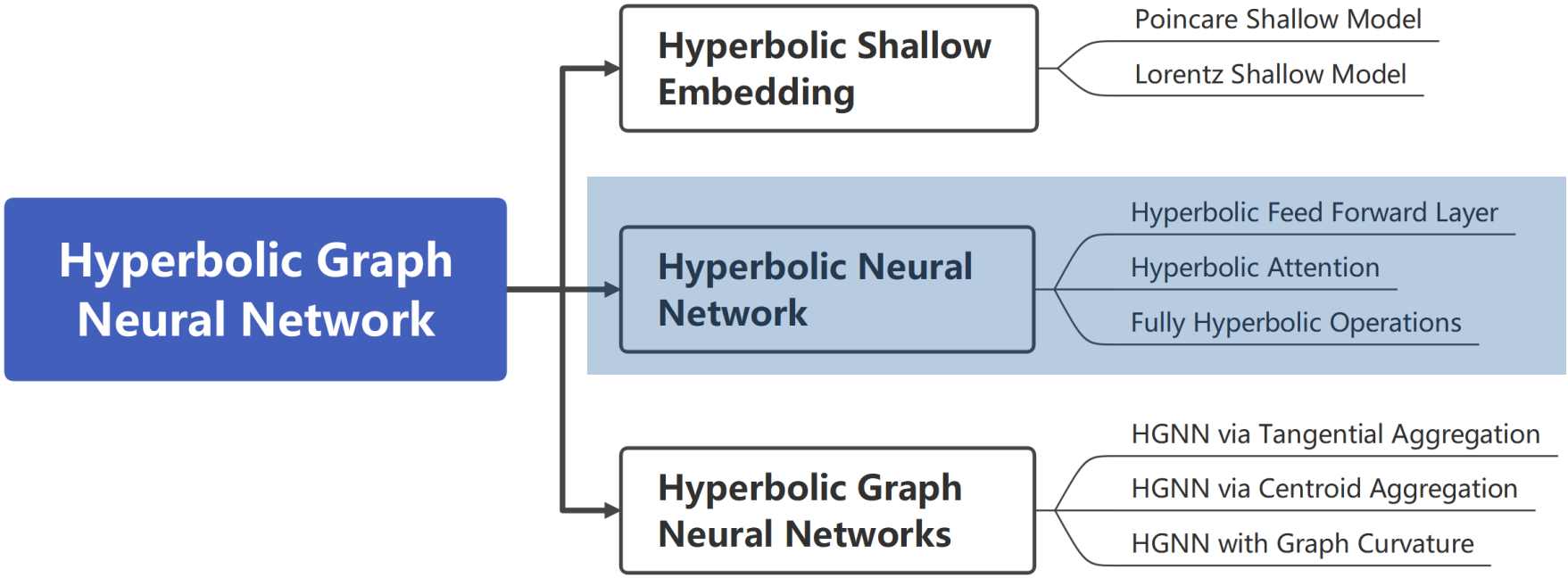
Results on Lorentz Model





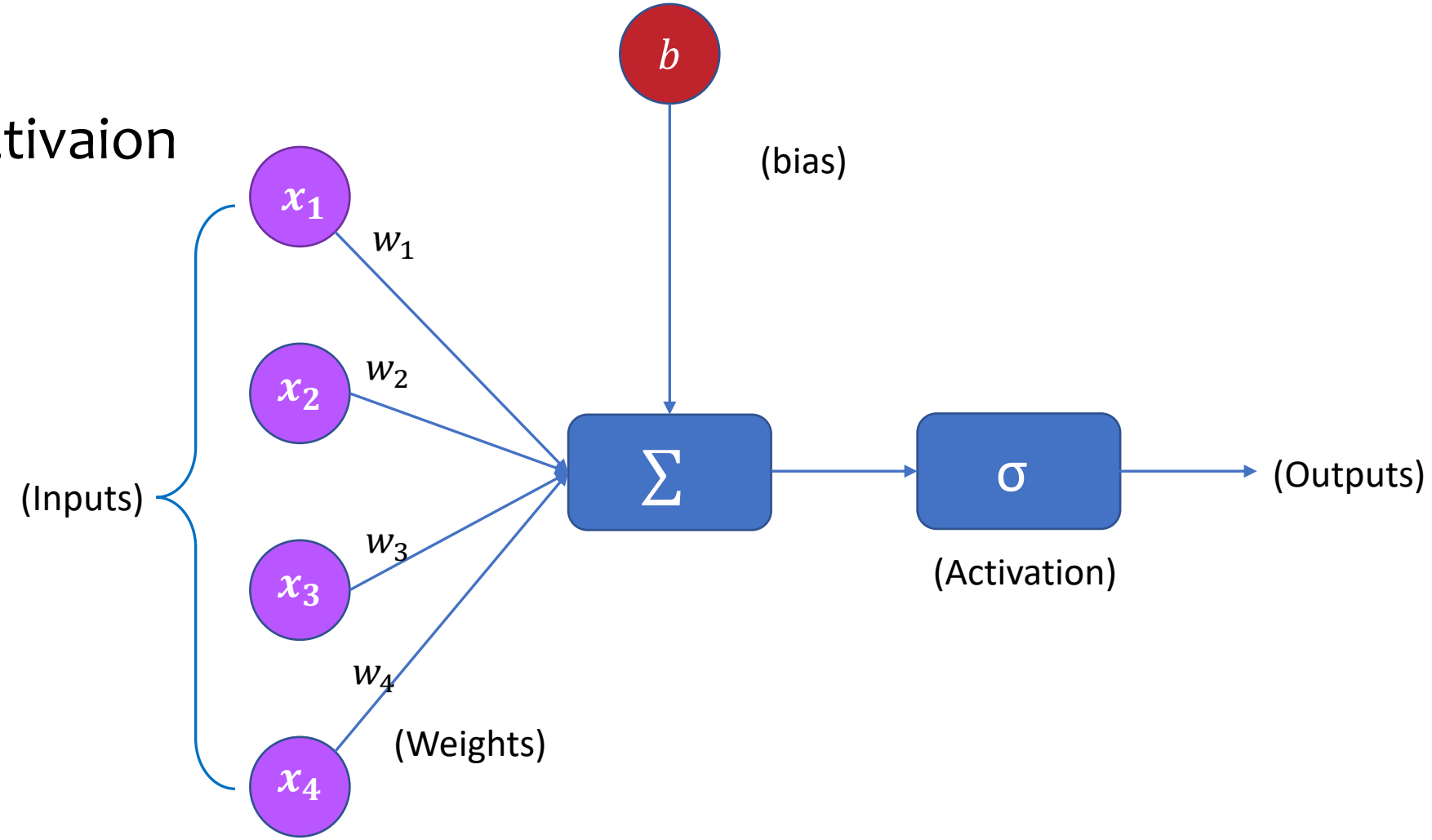
https://mnick.github.io/project/geometric-representation-learning/

Hyperbolic Neural Networks



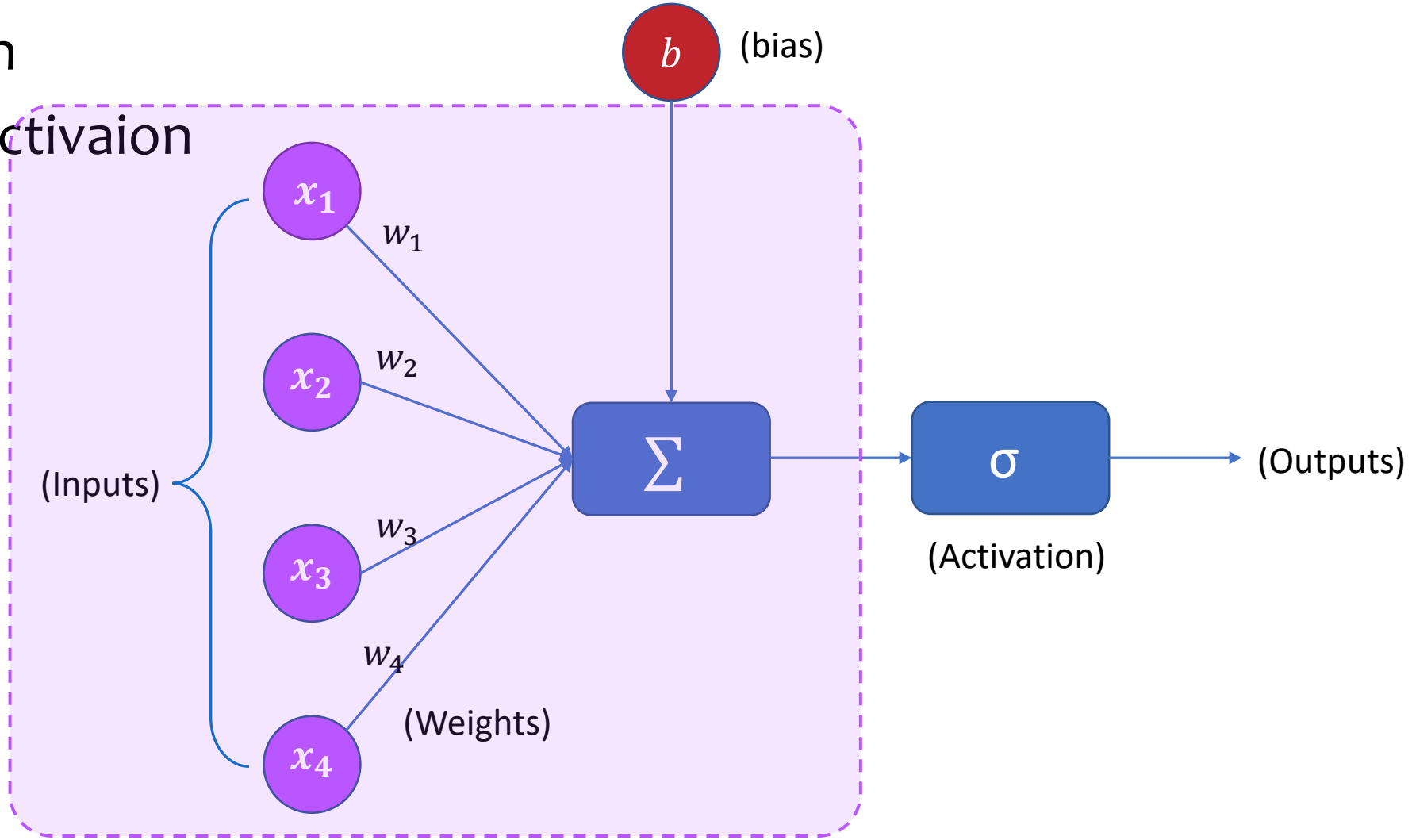
Feed Forward Layer

- Matrix-vector Multiplication
- Bias Addition
- Non-linear Activaion



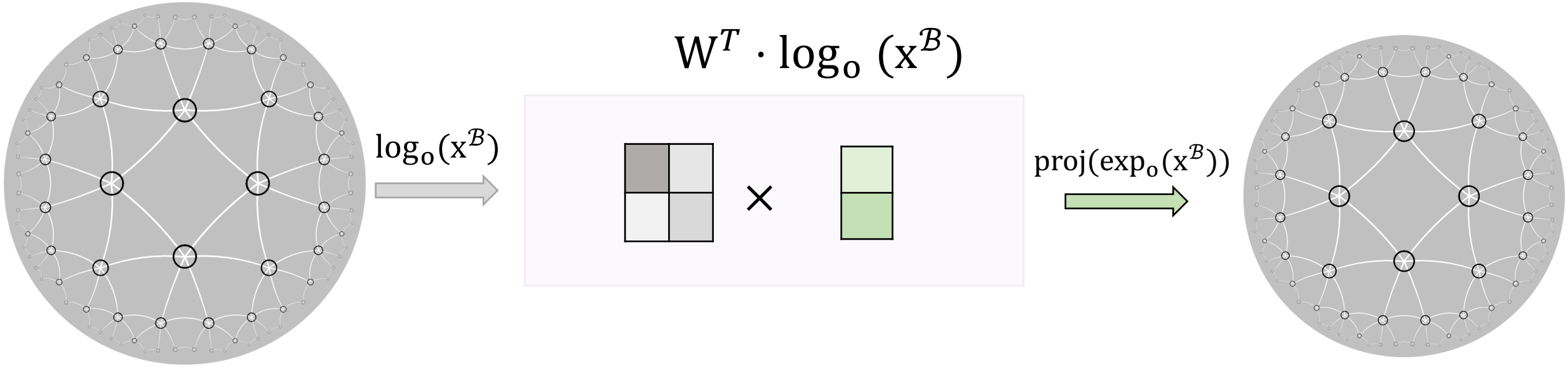
Feed Forward Layer

- **Matrix-vector Multiplication**
- Bias Addition
- Non-linear Activaion



Feed Forward Layer

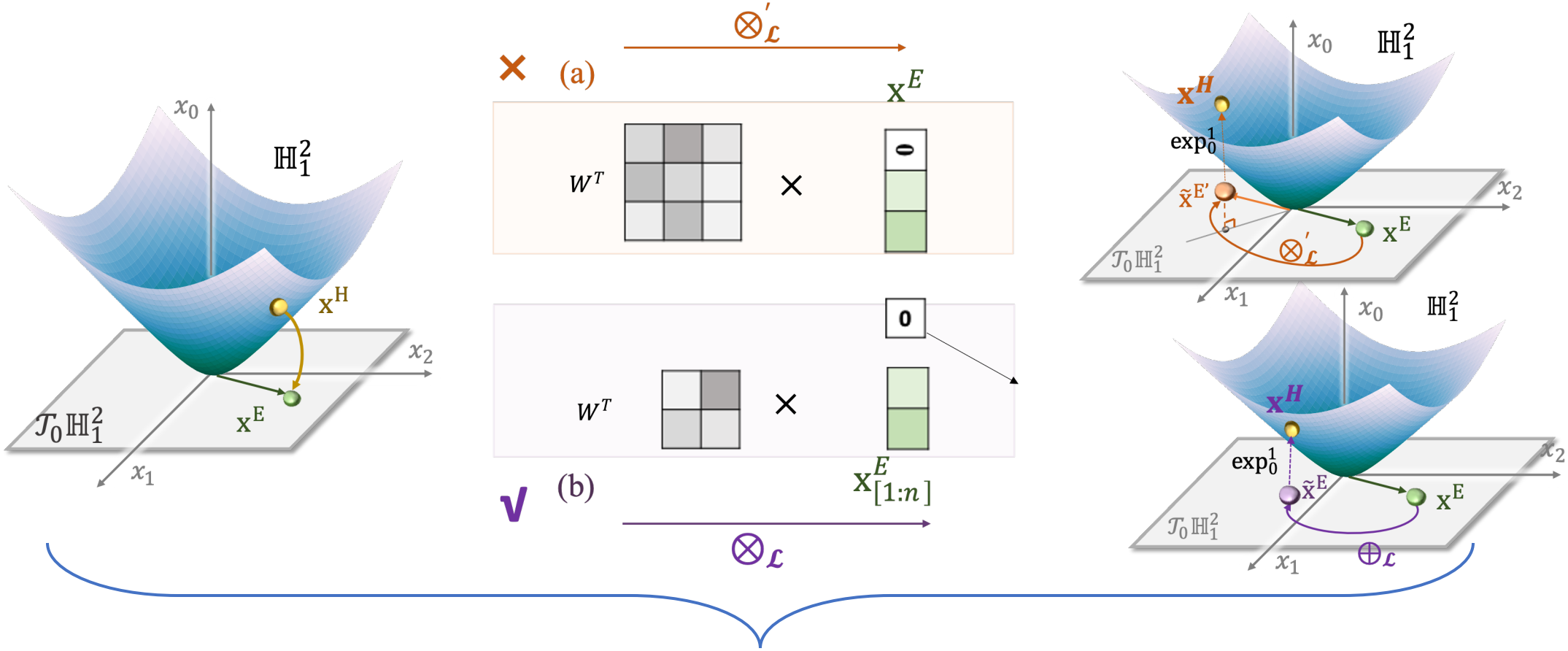
- Hyperbolic Matrix-vector Multiplication on Poincaré Ball Model



$$\text{Hyperbolic Matrix Multiplication}: W \otimes_{\mathcal{B}} x^{\mathcal{B}} = \text{proj}(\exp_{\mathbf{o}}(W^{\mathrm{T}} \log_o(x^{\mathcal{B}})))$$

Feed Forward Layer

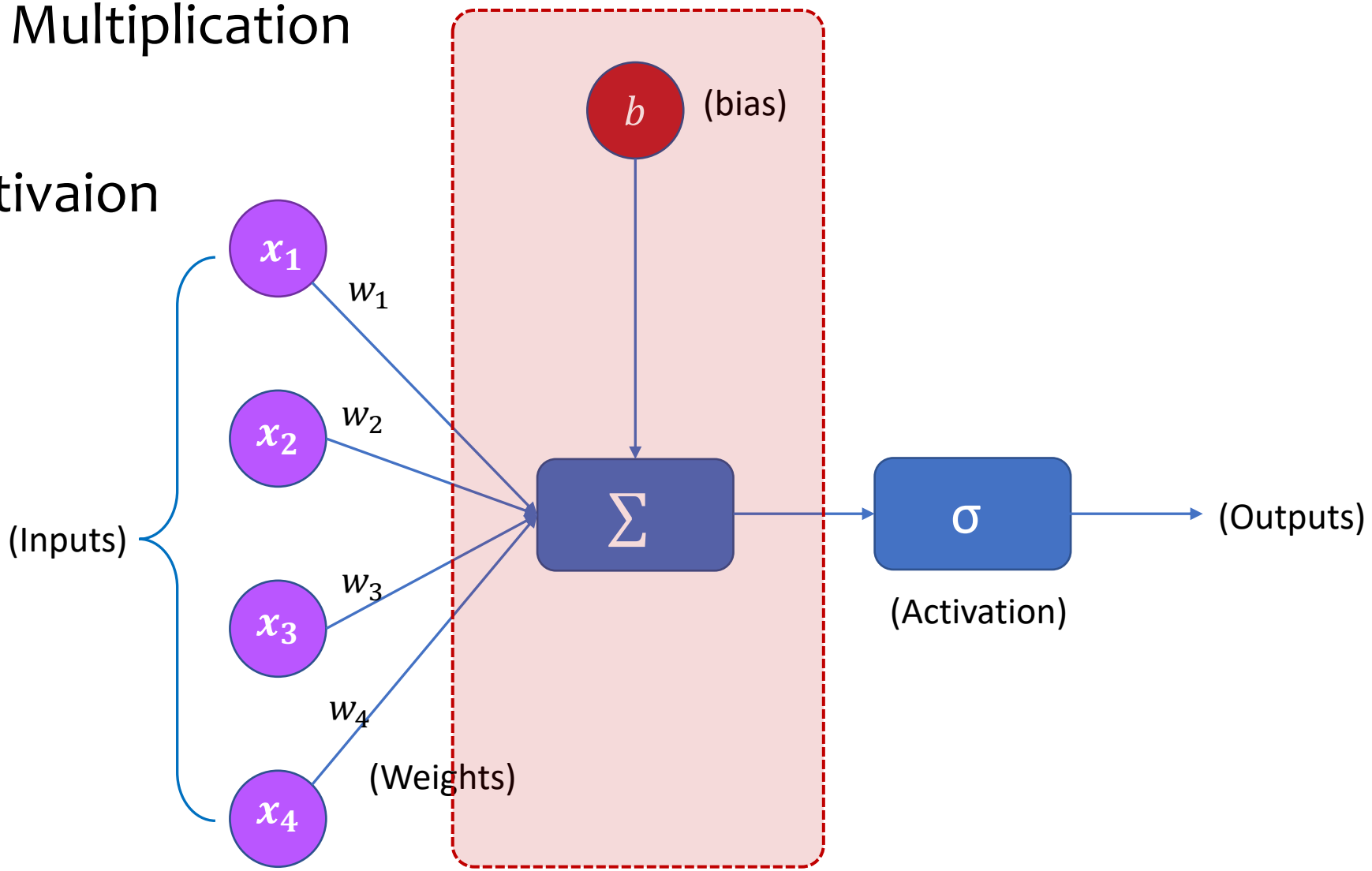
- Hyperbolic Matrix-vector Multiplication on Lorentz Model



$$\text{Hyperbolic Matrix Multiplication}: \mathrm{W} \otimes_{\mathcal{B}} \mathrm{x}^{\mathcal{B}} = \mathrm{proj}(\exp_{\mathbf{o}}([\mathbf{0}: W^{\mathrm{T}} \log_{\mathrm{o}}(\mathrm{x}^{\mathcal{B}}_{[1:n]})]))$$

Feed Forward Layer

- Matrix-vector Multiplication
- **Bias Addition**
- Non-linear Activaion



Feed Forward Layer

- Bias addition on Poincaré Ball Model and Lorentz Model
  - In Euclidean Space

$$x \leftarrow x + b$$

  - In Hyperbolic Space

$$x^H \leftarrow x^H \oplus b = \exp_{x^H}\left(P_{0 \to x^H}(\log_0(b))\right)$$

Fig: https://imgur.com/gallery/tqplYeT

Feed Forward Layer

- Bias addition on Poincaré Ball Model and Lorentz Model
  - In Euclidean Space

$$x \leftarrow x + b$$

  - In Hyperbolic Space

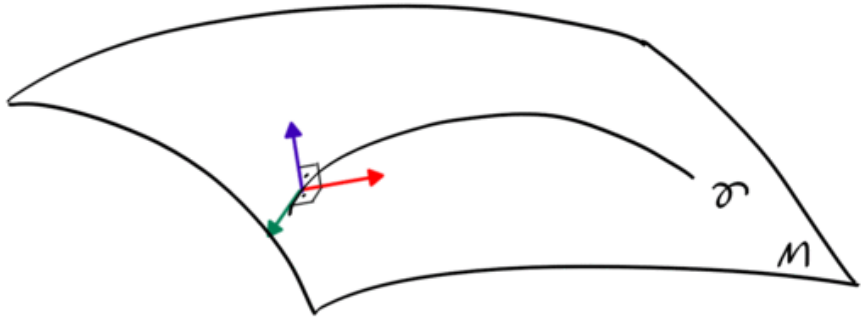


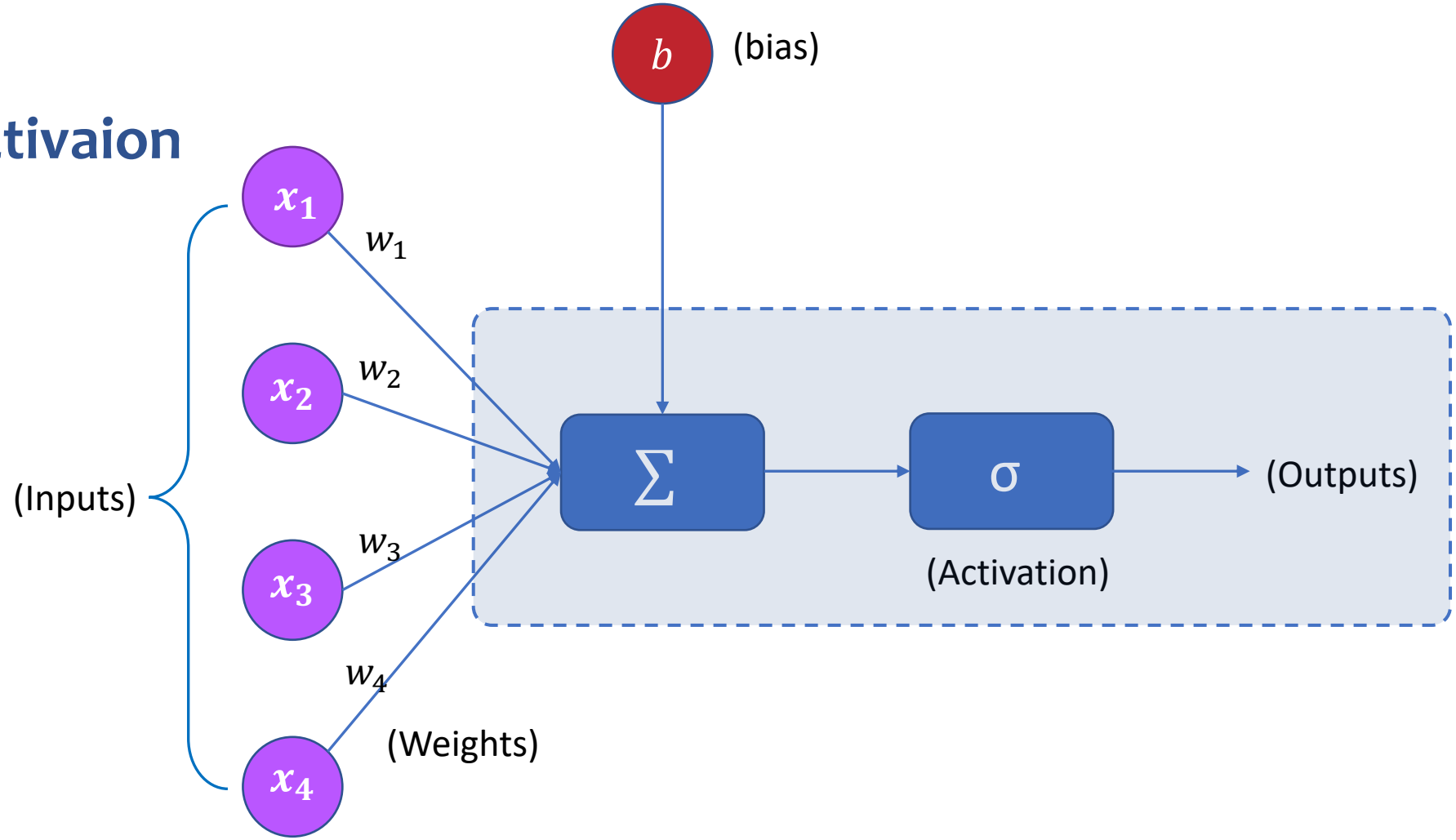
Fig: Parallel Transport

$$x^H \leftarrow x^H \oplus b = \exp_{x^H}\left(P_{0 \to x^H}(\log_0(b))\right)$$

Fig: https://imgur.com/gallery/tqplYeT

Feed Forward Layer

- Matrix-vector Multiplication
- Bias Addition
- **Non-linear Activaion**



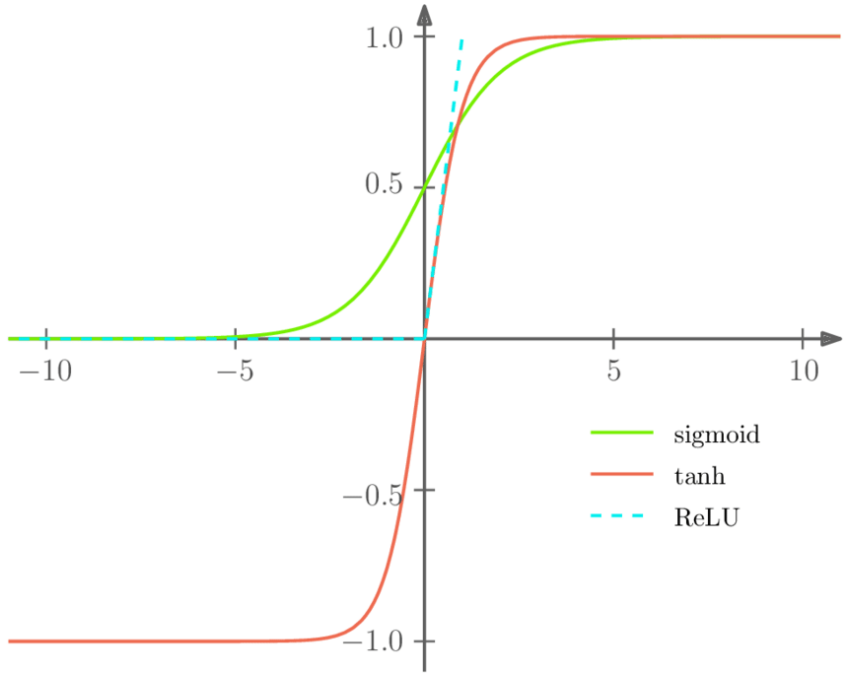
Feed Forward Layer

- Non-linear Activation on Poincaré Ball Model and Lorentz Model
  - In Euclidean Space

$$x \leftarrow f(x)$$

  - In Hyperbolic Space

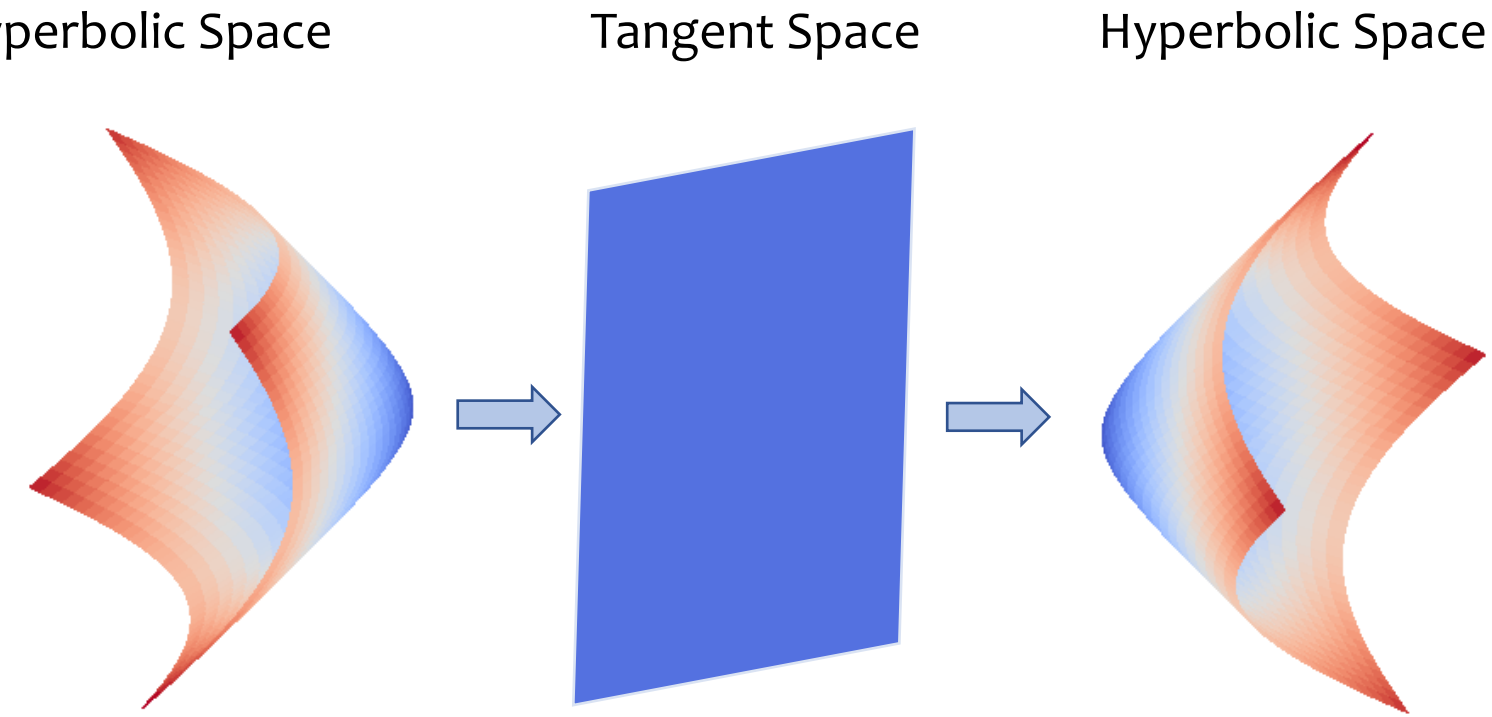
$$x^H \leftarrow f(x^H)$$



$$\text{Update}^{K_{l-1},K_l}(x^H) := \exp_0^{K_l}\left(f\left(\log_0^{K_{l-1}}(x^H)\right)\right)$$

Fig: https://imgur.com/gallery/tqplYeT

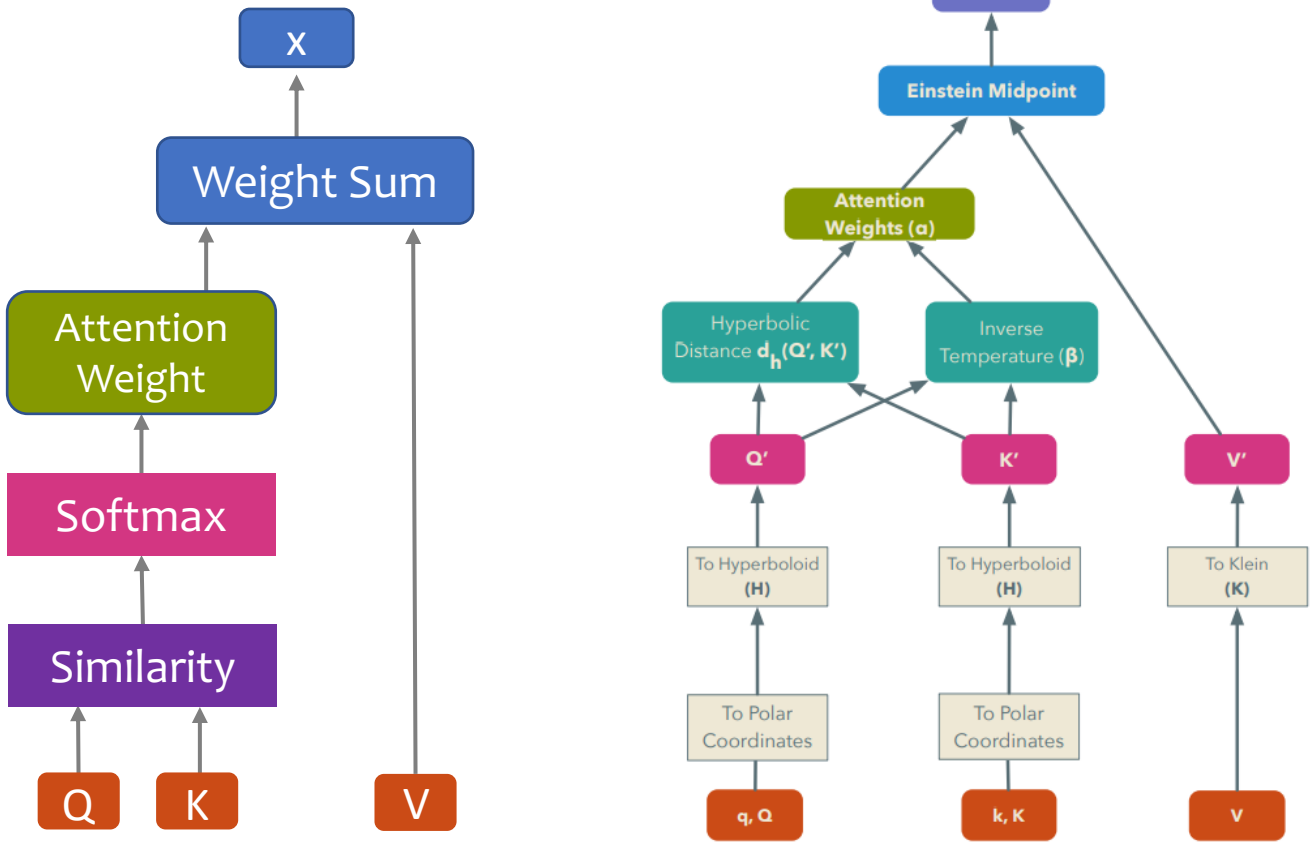
Uniform Paradim



- Hyperbolic space to tangent space
- Operations on tangent space
- Tangent space to Hyperbolic space

Hyperbolic Attention Layer

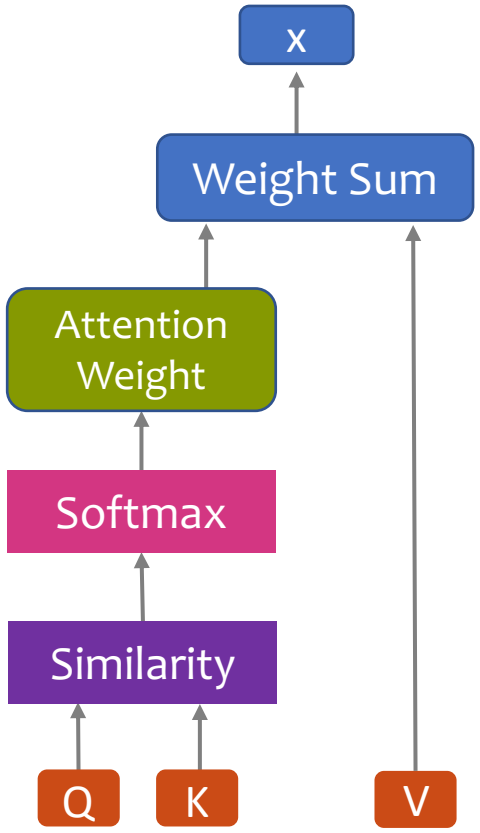
- Hyperbolic Attention Layer

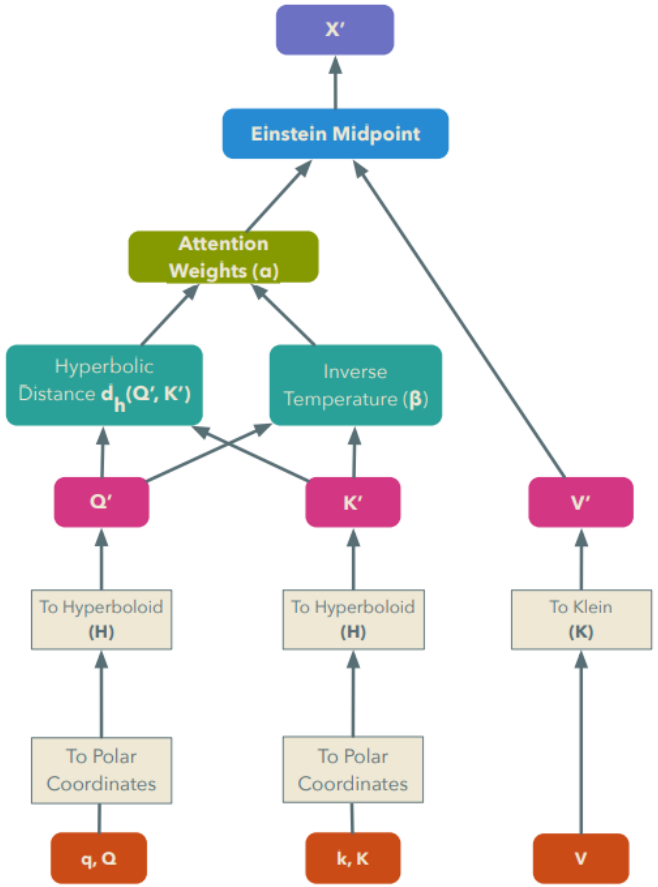


Gulcehre, Caglar, et al. "Hyperbolic attention networks." *arXiv preprint arXiv:1805.09786* (2018).

Hyperbolic Attention Layer

- Hyperbolic Attention Layer





Final Results x:

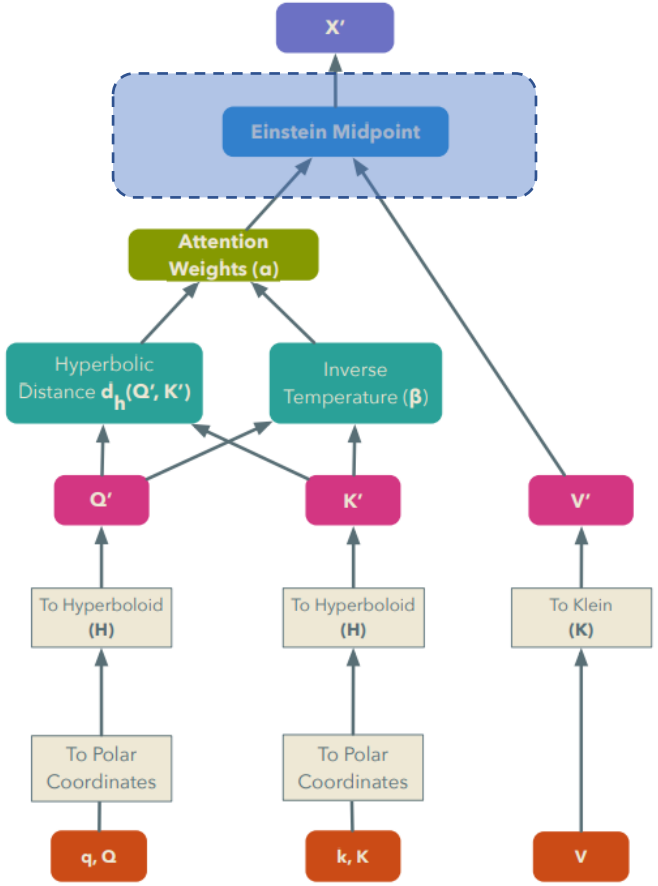
$$x' = \sum_j \left[\frac{\alpha_{ij} \gamma(\boldsymbol{v}_{ij})}{\sum_\ell \alpha_{i\ell} \gamma(\boldsymbol{v}_{i\ell})} \right] \boldsymbol{v}_{ij}$$

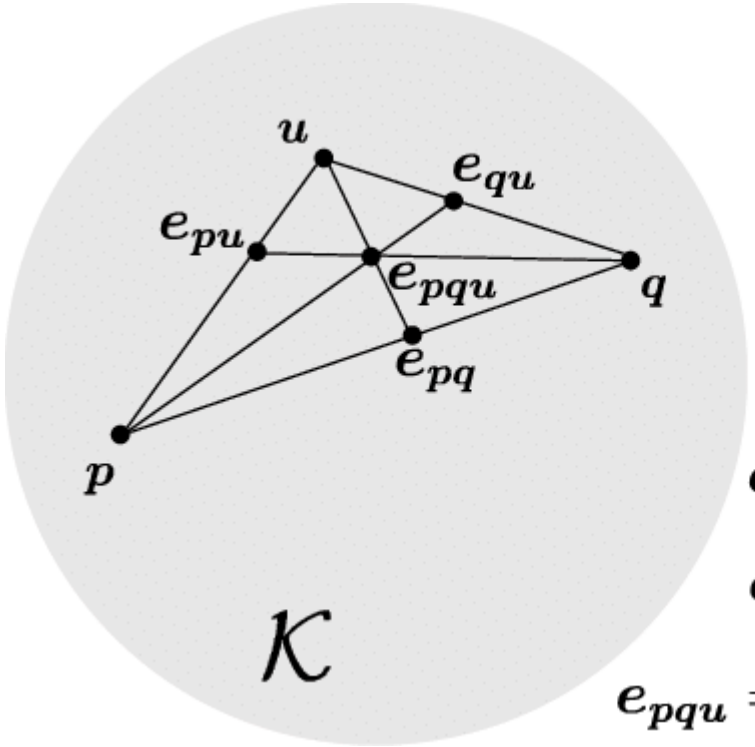
Attention Weights:

$$\alpha(\mathbf{q_i}, \mathbf{k_i}) = f(-\beta d_h(\mathbf{q_i}, \mathbf{k_j}) - c)$$

Hyperbolic Attention Layer

- Midpoint





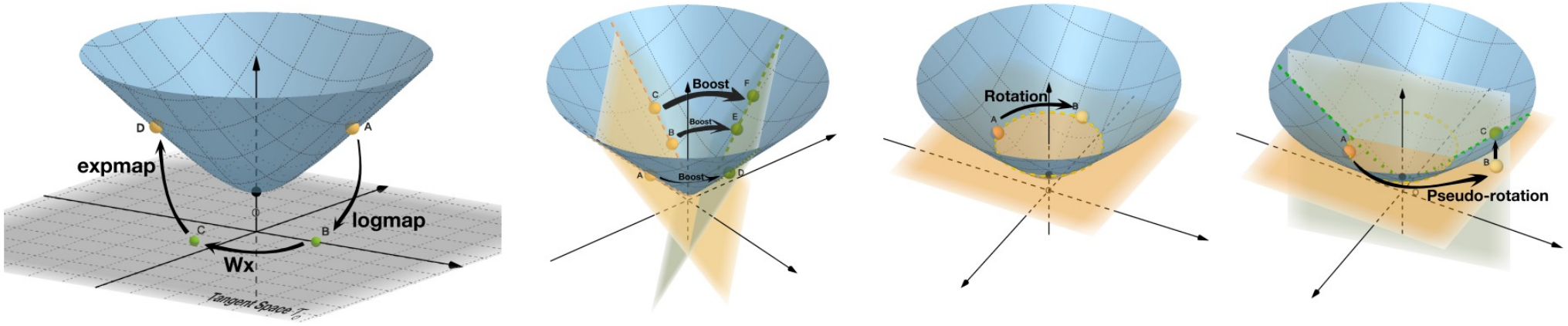
$$e_{pq} = \frac{\gamma_p p + \gamma_q q}{\gamma_p + \gamma_q}$$

$$e_{pu} = \frac{\gamma_p p + \gamma_u u}{\gamma_p + \gamma_u}$$

$$e_{qu} = \frac{\gamma_q q + \gamma_u u}{\gamma_q + \gamma_u}$$

$$e_{pqu} = \frac{\gamma_p p + \gamma_q q + \gamma_u u}{\gamma_p + \gamma_q + \gamma_u}$$

Fully Hyperbolic Neural Network



(a) Linear layer formalized in tangent space (b) Lorentz boost (c) Lorentz rotation (d) Pseudo-rotation

Fully Hyperbolic Neural Network

$$\mathbf{y} = \mathrm{HL}(\mathbf{x}) = \begin{bmatrix} \sqrt{\|\phi(\mathbf{W}\mathbf{x}, \mathbf{v})\|^2 - 1/K} \\ \phi(\mathbf{W}\mathbf{x}, \mathbf{v}) \end{bmatrix}$$

$$\phi(\mathbf{W}\mathbf{x}, \mathbf{v}) = \mathbf{W}\,\mathrm{dropout}(\mathbf{x})$$

$$\phi(\mathbf{W}\mathbf{x}, \mathbf{v}) = \frac{\lambda\sigma\left(\mathbf{v}^\top \mathbf{x} + b'\right)}{\|\mathbf{W}h(\mathbf{x}) + \mathbf{b}\|}(\mathbf{W}h(\mathbf{x}) + \mathbf{b})$$

- Hyperbolic Space ($\mathbb{L}_K^n$): an n-dimensional hyperbolic space with curvature K.
- Transformation ($\mathbf{W}\mathbf{x}$): The weight matrix $\mathbf{W}$ transforms the input data $\mathbf{x}$.
- Function (ϕ): This function dictates how the transformed input interacts with the additional vector $\mathbf{v}$ and further modifies the data. Function (ϕ) can be used for dropout, non-linear activation, and normalization.

Fully Hyperbolic Neural Network

$$\mathbf{y} = \mathrm{HL}(\mathbf{x}) = \begin{bmatrix} \sqrt{\|\phi(\mathbf{W}\mathbf{x}, \mathbf{v})\|^2 - 1/K} \\ \phi(\mathbf{W}\mathbf{x}, \mathbf{v}) \end{bmatrix}$$

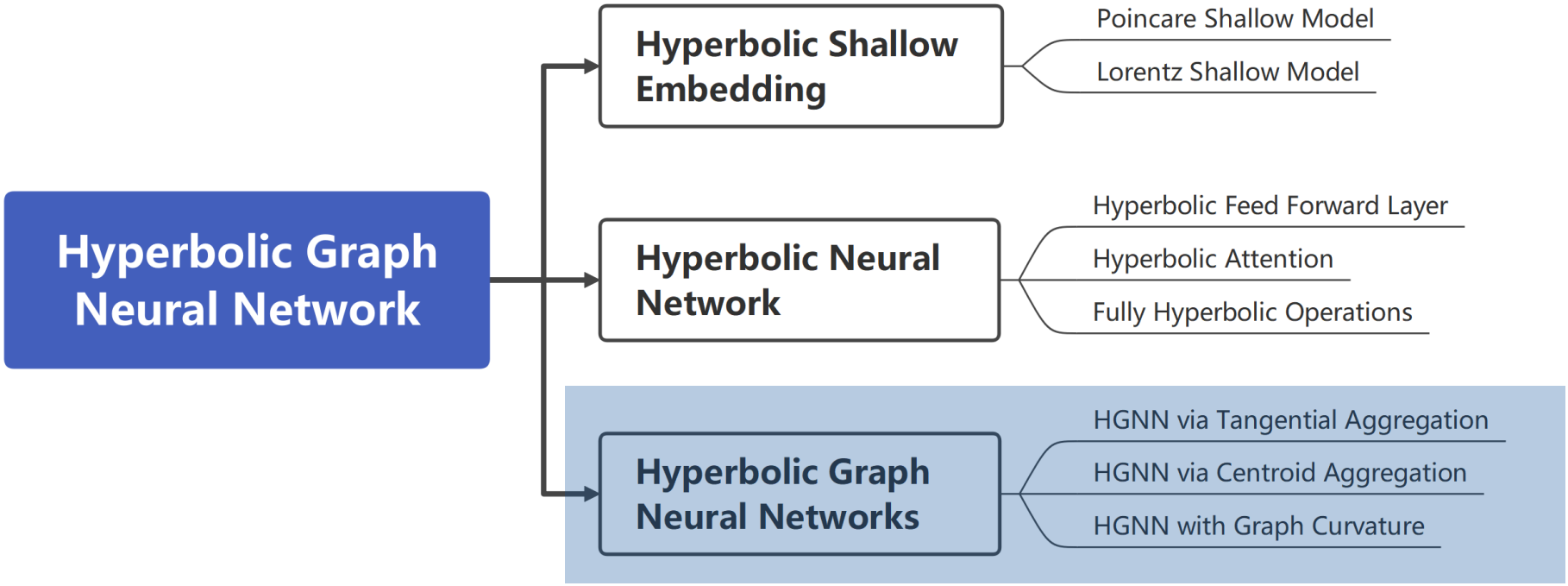
$$\phi(\mathbf{W}\mathbf{x}, \mathbf{v}) = \mathbf{W}\,\mathrm{dropout}(\mathbf{x})$$

$$\phi(\mathbf{W}\mathbf{x}, \mathbf{v}) = \frac{\lambda\sigma\left(\mathbf{v}^\top\mathbf{x} + b'\right)}{\|\mathbf{W}h(\mathbf{x}) + \mathbf{b}\|}(\mathbf{W}h(\mathbf{x}) + \mathbf{b})$$

- Hyperbolic Space ($\mathbb{L}_K^n$): an n-dimensional hyperbolic space with curvature K.
- Transformation ($\mathbf{W}\mathbf{x}$): The weight matrix $\mathbf{W}$ transforms the input data $\mathbf{x}$.
- Function (ϕ): This function dictates how the transformed input interacts with the additional vector $\mathbf{v}$ and further modifies the data. Function (ϕ) can be used for dropout, non-linear activation, and normalization.
- Output ($\mathbf{y}$): The output consists of two components. The first is derived from the norm of the operation function's result, adjusted by the curvature K. The second component is the direct result of the operation function.

$$\{\boldsymbol{\mu}_1^i, \boldsymbol{\mu}_2^i, \ldots\} = \mathtt{ATT}^i(\mathtt{HL}_{\mathcal{Q}}^i(\mathcal{Q}), \mathtt{HL}_{\mathcal{K}}^i(\mathcal{K}), \mathtt{HL}_{\mathcal{V}}^i(\mathcal{V}))$$

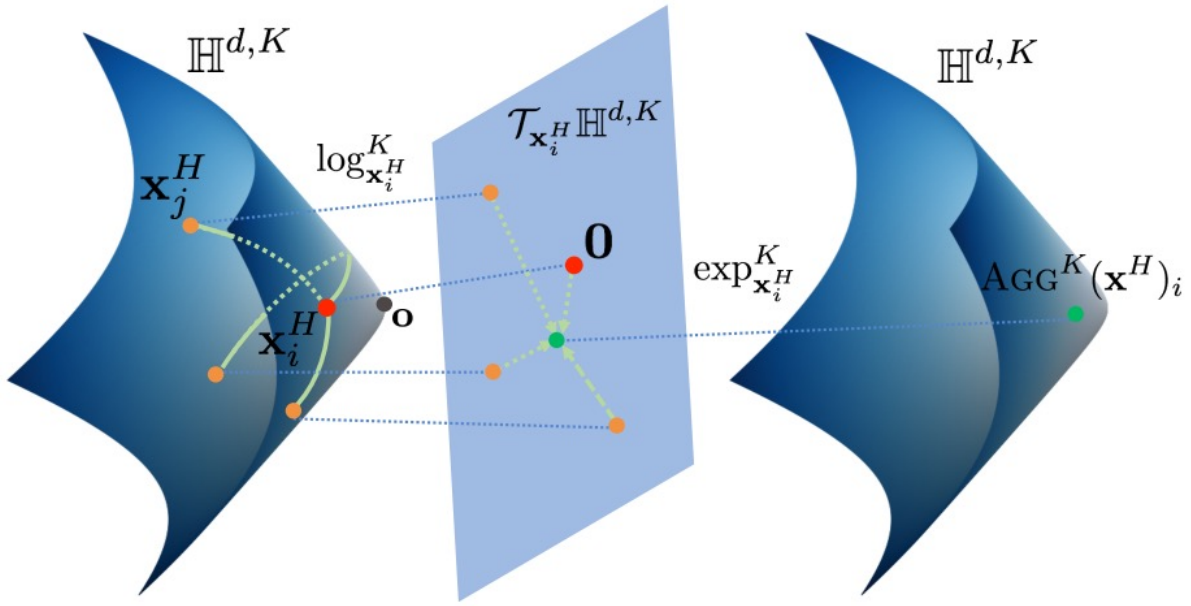
2.2 Hyperbolic Neural Networks



HGNN via Tangential Aggregation

- Hyperbolic Graph Convolutional Neural Networks
  - Feature Transformation (**Message**)
  - Neighborhood Aggregation (**Aggregation**)
  - Non-linear Activation (**Update**)

- $\boldsymbol{h}_i^{l,H} = \mathrm{Msg}(\boldsymbol{x}_i^{l-1,H})$
- $\boldsymbol{y}_i^{l,H} = \mathrm{AGG}^{K_{l-1}}(\boldsymbol{h}^{l,H})_i$
- $\boldsymbol{x}_i^{l,H} = \mathrm{Update}^{K_{l-1},K_l}(\boldsymbol{y}_i^{l,H})$



Chami, Ines, et al. "Hyperbolic graph convolutional neural networks." NeurIPS (2019).

HGNN via Tangential Aggregation

- Input Transformation: $x^{0,H} := \exp_0^K\left((0, x^{0,E})\right)$

- **Message:**

$$\boldsymbol{h}_i^{l,H} = \left(W^l \otimes^{K_{l-1}} \boldsymbol{x}_i^{l-1,H}\right) \oplus^{K_{l-1}} \boldsymbol{b}^l$$

$$W \otimes^K \boldsymbol{x}^H := \exp_0^K(W \log_0^K(\boldsymbol{x}))$$

$$\boldsymbol{x}^H \oplus \boldsymbol{b} := \exp_{x^H}^K\left(P_{o \to x^H}^K(\boldsymbol{b})\right)$$

- **Aggregation:**

$$\mathrm{AGG}(\boldsymbol{x}^H)_i := \exp_{\boldsymbol{x}_i^H}^{\mathrm{K}}\Big(\sum_{j \in N(j)} w_{ij} \log_{\boldsymbol{x}_i^H}^{\mathrm{K}}(\boldsymbol{x}_j^H)\Big)$$

Attention weights

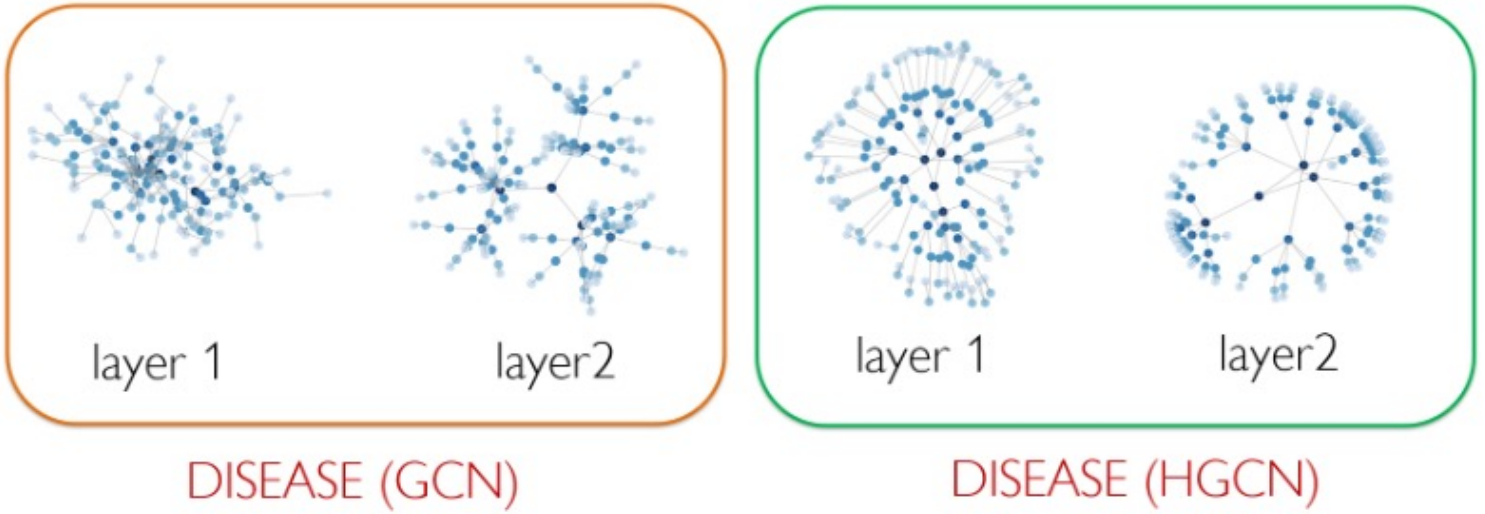
- **Update:**

Trainable curvature

$$\mathrm{Update}^{K_{l-1}, K_l}(x^H) := \exp_0^{K_l}\left(\sigma\left(\log_{\mathrm{o}}^{K_{l-1}}(x^H)\right)\right)$$

Chami, Ines, et al. "Hyperbolic graph convolutional neural networks." NeurIPS (2019).

HGNN via Tangential Aggregation



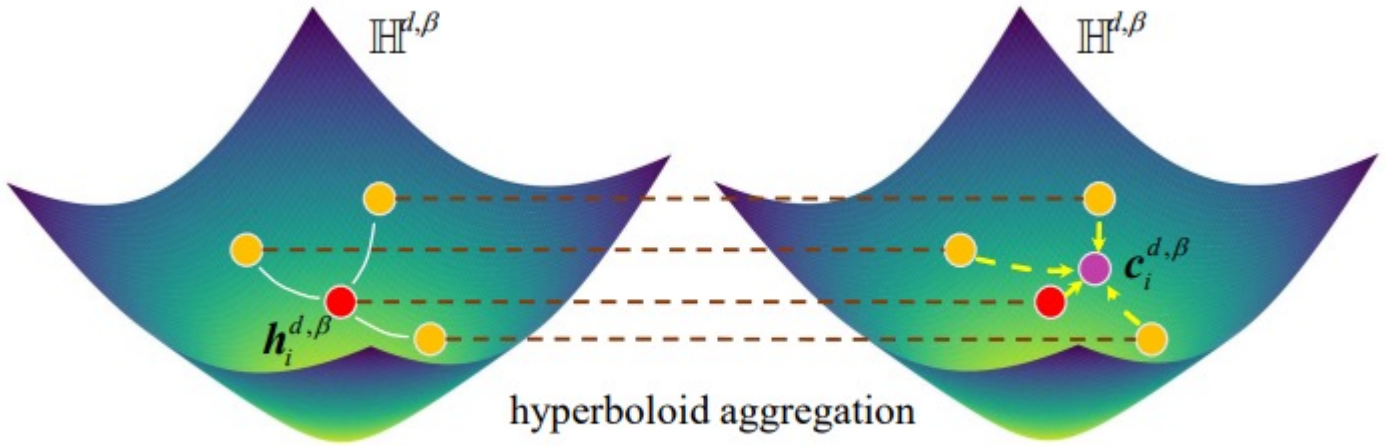
The above show that the 1st layer and 2nd layer embedding for both GCN and HGCN.

HGCN can map the nodes to embeddings of different hyperbolic space with different curvature at each layer, resulting in low distortion embedding for hierarchical data.

Chami, Ines, et al. "Hyperbolic graph convolutional neural networks." NeurIPS (2019).

HGNN via Centroid Aggregation

- Hyperbolic Graph Neural Networks
  - Feature Transformation
  - Neighborhood Aggregation
  - Non-linear Activation



(b) Hyperbolic aggregation on hyperboloid model via centroid

Zhang, Yiding, et al. "Lorentzian graph convolutional networks." *The WebConf*. 2021.

HGNN via Centroid Aggregation

- Input Transformation: $x^{0,H} := \exp_0^K\left((0, x^{0,E})\right)$
- **Message:**

$$\boldsymbol{h}_i^{l,H} = \left(W^l \otimes^{K_{l-1}} \boldsymbol{x}_i^{l-1,H}\right)$$

- **Aggregation:**

$$W \otimes^K \boldsymbol{x}^H := \exp_0^K\left(\left(0, \mathrm{W}\log_0^K\left(\boldsymbol{x}_{[1:n]}\right)\right)\right)$$

$$\mathrm{AGG}(\boldsymbol{x}^H)_i := \frac{\sum_{j\in N(i)} w_{ij} x_j^H}{\left|\left|\sum_{j\in N(i)} w_{ij} x_j^H\right|\right|_{\mathcal{L}}\left|\right.}$$

Edge weights/attention weights

- **Update:**

Trainable curvature

$$\mathrm{Update}^{K_{l-1},K_l}(x^H) := \exp_0^{K_l}\left(0, \sigma\left(\log_{\mathrm{o}}^{K_{l-1}}\left(x_{[1;n]}^H\right)\right)\right)$$

Zhang, Yiding, et al. "Lorentzian graph convolutional networks." *The WebConf*. 2021.

HGNN with Graph Curvature

- Input Transformation: $x^{0,H} := \exp_0^K\left((0, x^{0,E})\right)$
- **Message:**

$$\boldsymbol{h}_i^{l,H} = \left(W^l \otimes^{K_{l-1}} \boldsymbol{x}_i^{l-1,H}\right)$$

$$W \otimes^K \boldsymbol{x}^H := \exp_0^K\left(\left(0, \mathrm{W}\log_0^K\left(\boldsymbol{x}_{[1:n]}\right)\right)\right)$$

- **Aggregation:**

$$\mathrm{AGG}(\boldsymbol{x}^H)_i := \exp_{\boldsymbol{x}_i^H}^{\mathrm{K}}\left(\sum_{j \in N(j)} k_{ij} \log_{x_i^H}^{\mathrm{K}}(\boldsymbol{x}_j^H)\right)$$

Edge curvature &| attention weights

Trainable curvature

- **Update:**

$$\mathrm{Update}^{K_{l-1}, K_l}(x^H) := \exp_0^{K_l}\left(0, \sigma\left(\log_{\mathrm{o}}^{K_{l-1}}\left(x_{[1;n]}^H\right)\right)\right)$$

Yang, Menglin et.al. kHGCN: Tree-likeness Modeling via Continuous and Discrete Curvature Learning. KDD 2023.

Slack channel

- Join our slack channel for Q&A

https://hyperbolicgnn.github.io/

Part 3. Applications

Agenda

Part 1. INTRODUCTION (Min, 9:00 - 9:30)

1.1 An Overview of Graph Learning

1.2 Brief Introduction of Riemannian Geometry

1.3 Motivation for Hyperbolic Graph Learning

Coffee Break (9:30 – 10:00)

Part 2. HYPERBOLIC GRAPH NETWORKS (Menglin, 10:00 – 10:30)

2.1 Preliminary

2.2 Hyperbolic Shallow Models

2.2 Hyperbolic Neural Networks

2.3 Hyperbolic Graph Neural Networks

Part 3. APPLICATIONS (Menglin, 10:30 – 11:00)

3.1 HGNN for Recommendation Systems

3.2 HGNN for Knowledge Graph

3.3 HGNN for Bioinformatics

Part 4. ADVANCED TOPICS (Bo, 11:00 – 12:00)

4.1 Complex Structures

4.2 Curvature-aware Learning

4.3 Evolving Interactions

4.4 Trustworthy and Scalability

Recommendation Systems

Recommender System are Everywhere

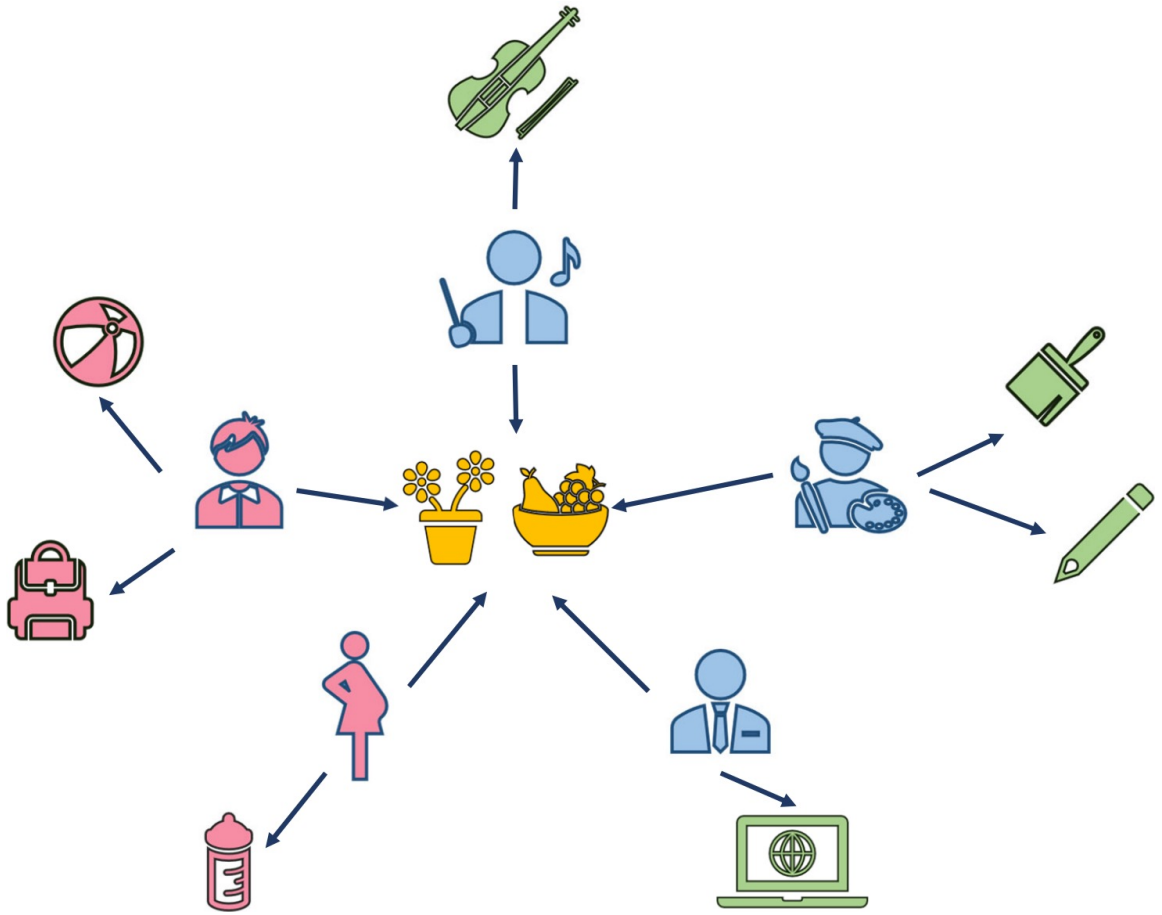
Music Recommendation

Products Recommendation

News Recommendation

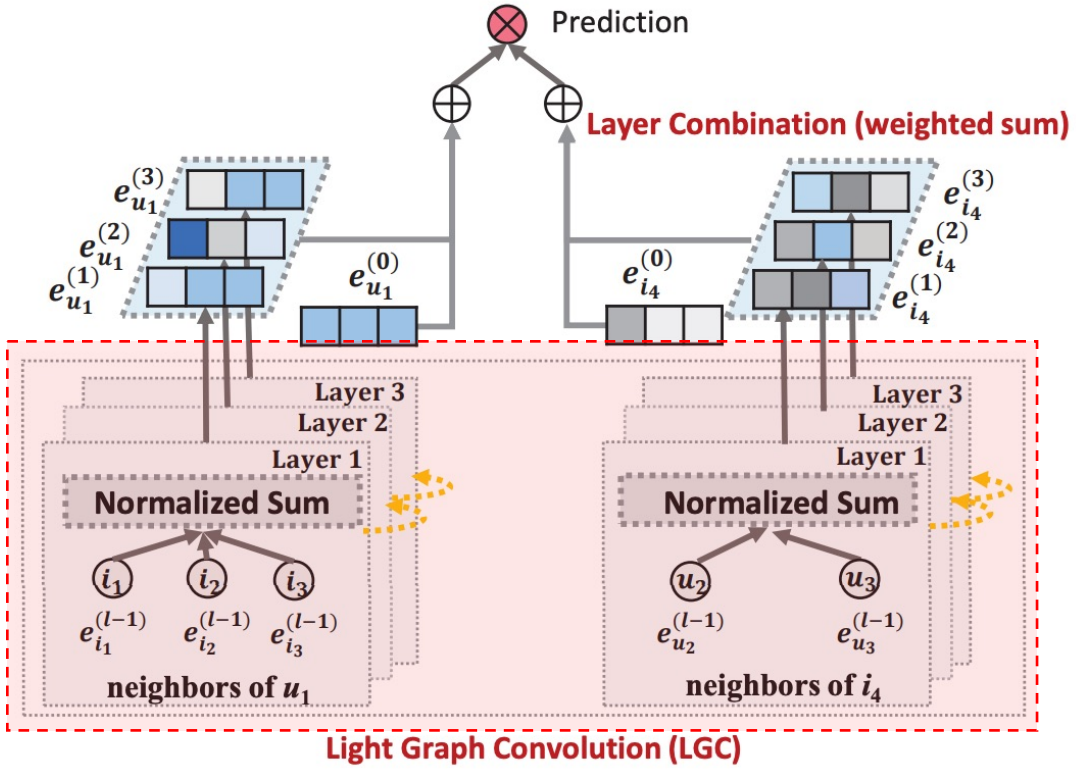
Recommendation Systems

User-item Networks



Users and items are nodes and the interactions are edges.

GNN for User-item Network Modeling *in Euclidean Space*



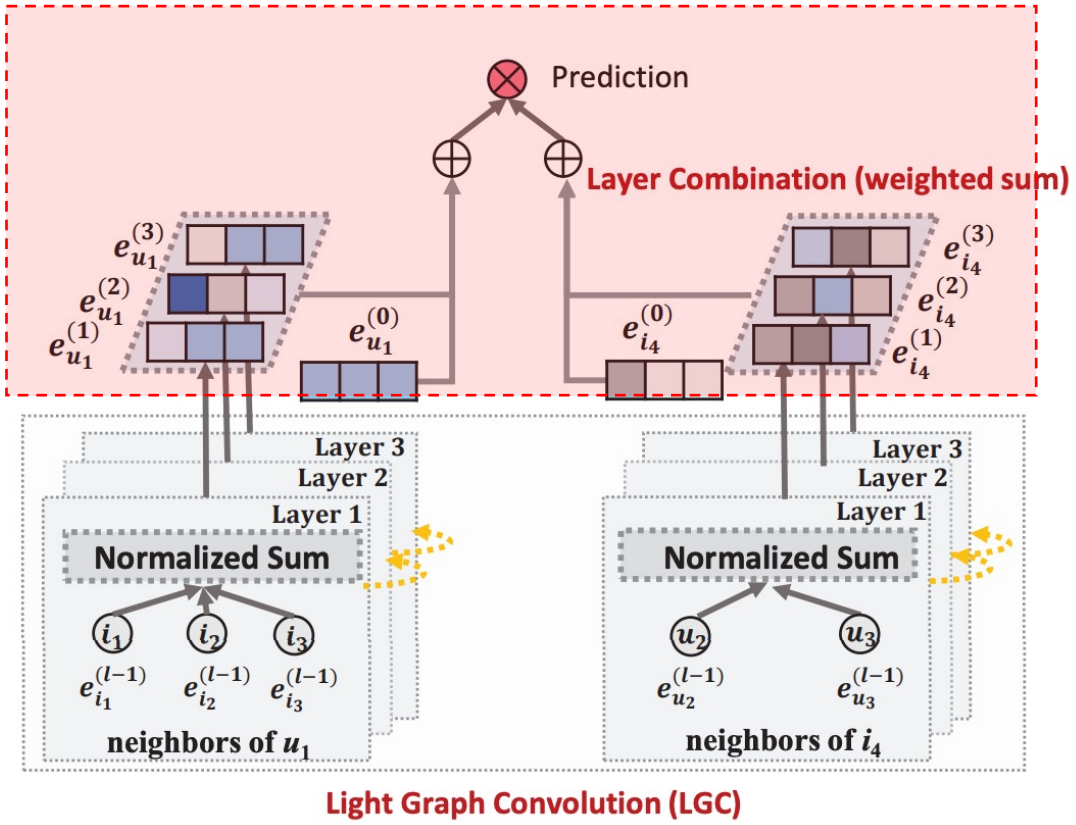
$$\mathbf{e}_u^{(k+1)} = \sum_{i \in \mathcal{N}_u} \frac{1}{\sqrt{|\mathcal{N}_u|}\sqrt{|\mathcal{N}_i|}} \mathbf{e}_i^{(k)},$$

$$\mathbf{e}_i^{(k+1)} = \sum_{u \in \mathcal{N}_i} \frac{1}{\sqrt{|\mathcal{N}_i|}\sqrt{|\mathcal{N}_u|}} \mathbf{e}_u^{(k)}.$$

Note: where $\mathbf{e}_u^k$ and $\mathbf{e}_i^k$ respectively denote the refined embedding of user u and item i after k layers propagation. N_u denotes the set of items that are interacted by user u, N_i denotes the set of users that interact with item i.

Xiangnan He, et al. "Lightgcn: Simplifying and powering graph convolution network for recommendation." SIGIR. 2020.

GNN for User-item Network Modeling *in Euclidean Space*



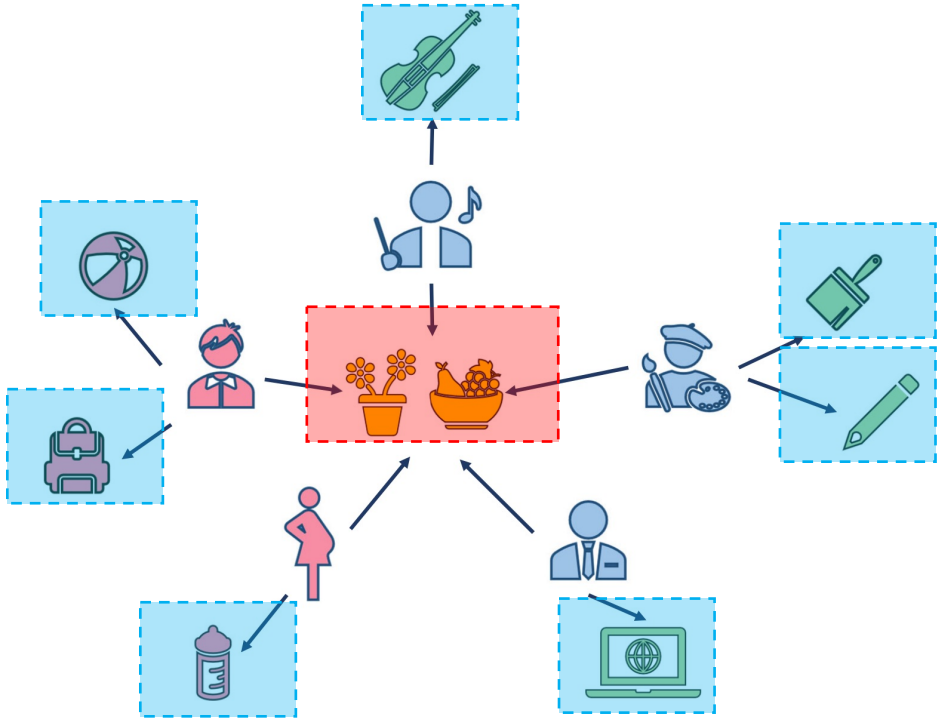
$$\hat{y}_{ui} = \mathbf{e}_u^T \mathbf{e}_i,$$

Note: The model prediction is defined as the inner product of user and item final representations.

Xiangnan He, et al. "Lightgcn: Simplifying and powering graph convolution network for recommendation." SIGIR. 2020.

Recommendation Systems

User-item Networks



(plotted by Jiahong Liu)

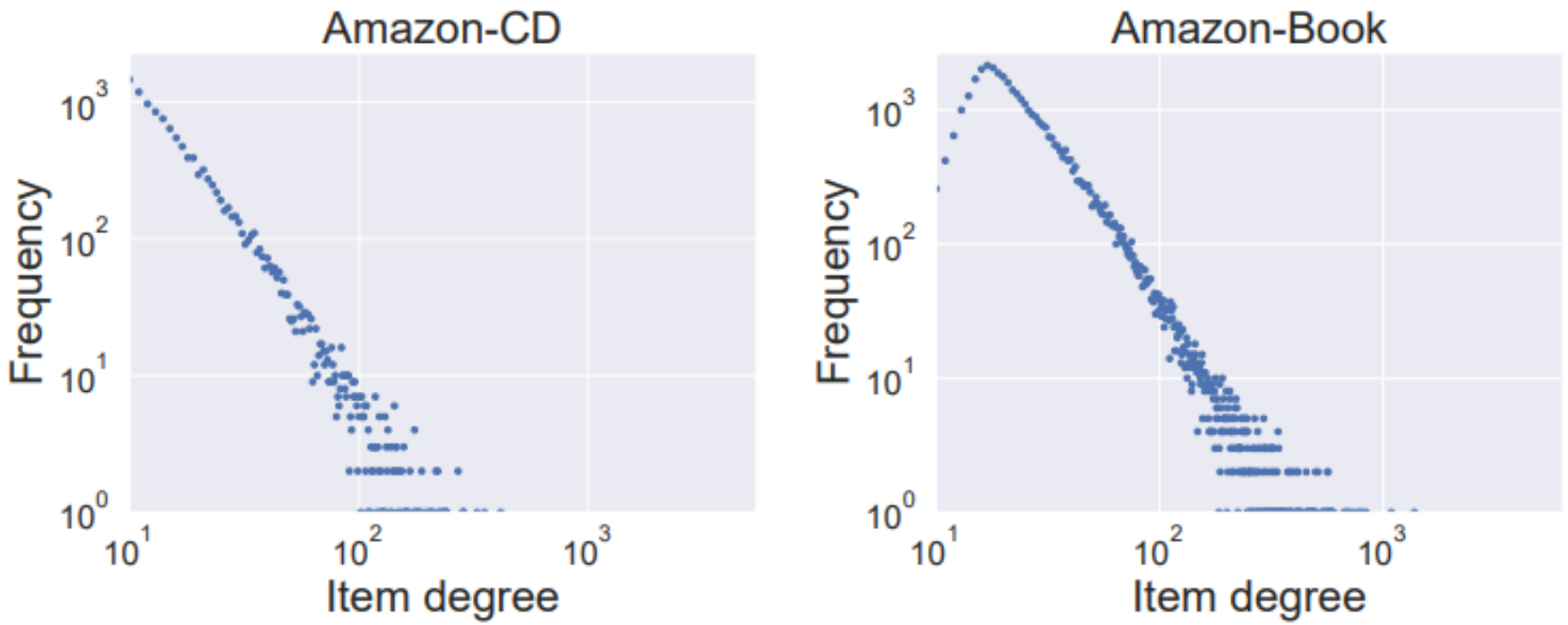
Note: Popular items are competitive while the long-tail item reflects personalized preference or something new.

The popular items are in the minority, whereas the individualized or unpopular items are in the majority.

Menglin Yang, et.al. Hyperbolic Informative Collaborative Filtering. KDD 2022

Recommendation Systems

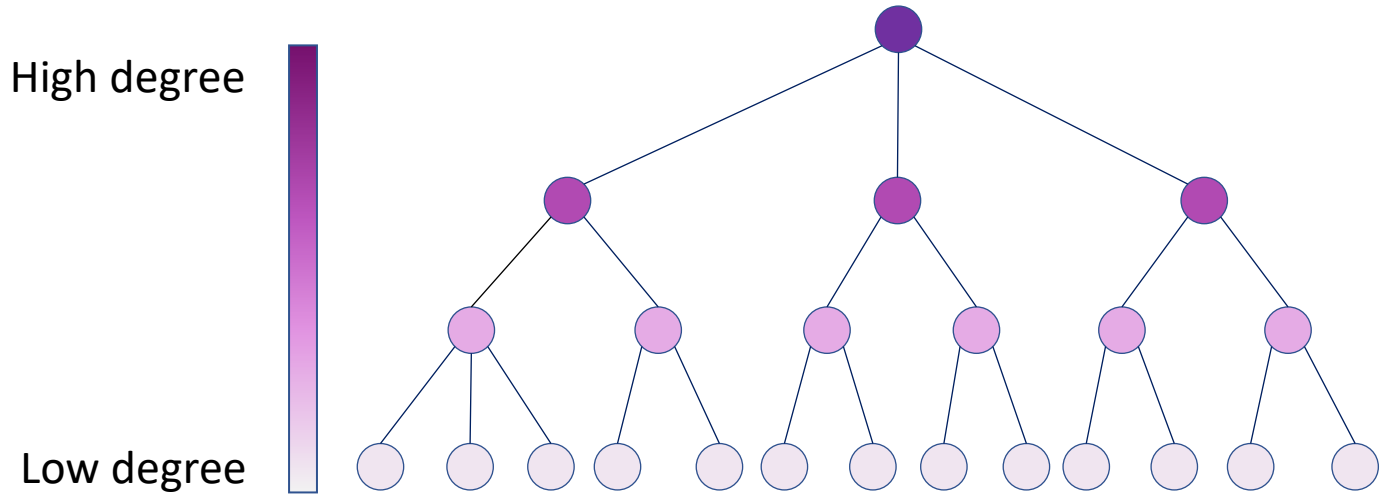
Degree Distribution



Power-law distribution !!!

Menglin Yang, et al. HRCF: Enhancing collaborative filtering via hyperbolic geometric regularization. *Proceedings of the ACM Web Conference 2022*.

Recommendation Systems



Tree-like structure (Power-law distribution)

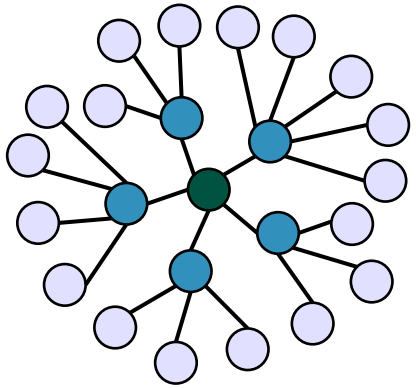
Note:

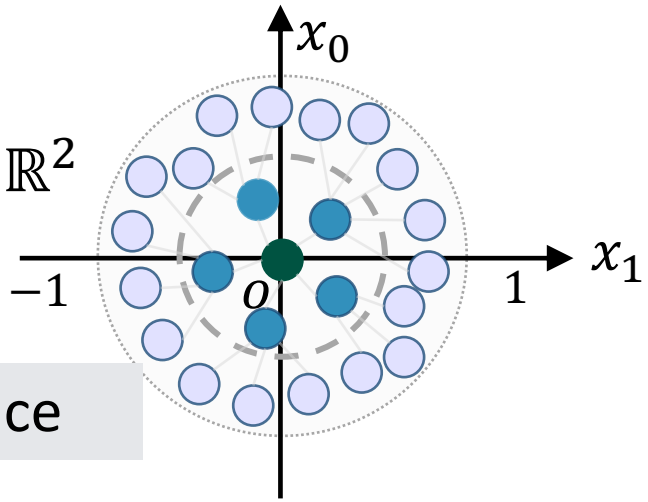
High degree nodes are liked by the majority which indicates their popularity is high.

Low degree nodes are liked by the minority which demonstrates **their popularity is low.**

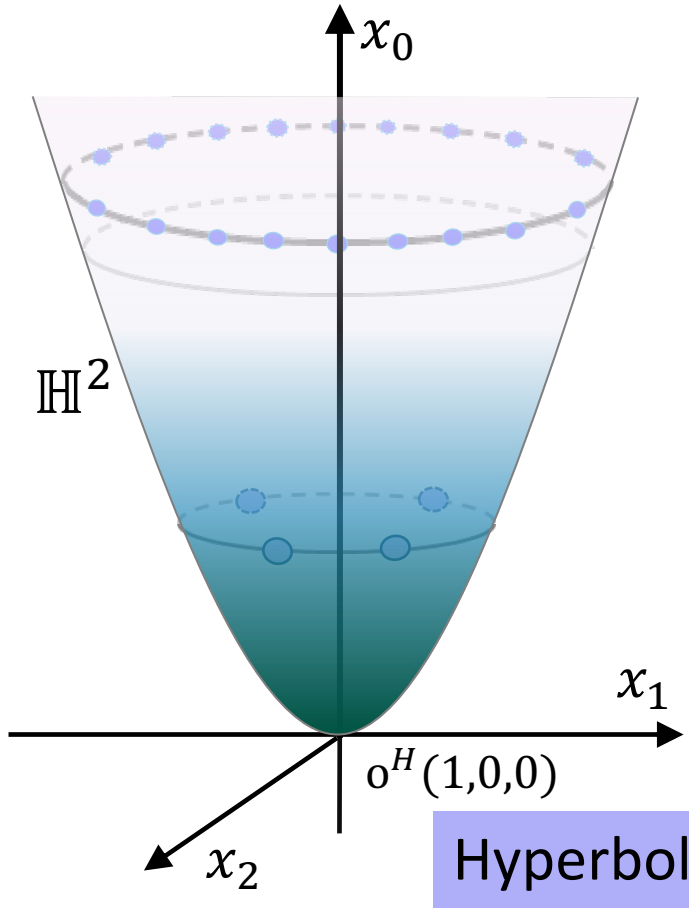
Embedding Tree-like Graphs

Tree-like G





Euclidean space

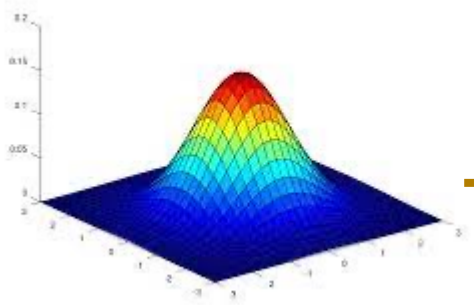


Hyperbolic space

Menglin Yang, et al. HRCF: Enhancing collaborative filtering via hyperbolic geometric regularization. The WebConf 2021

Recommendation Systems

- Hyperbolic Embedding Initializing Layer

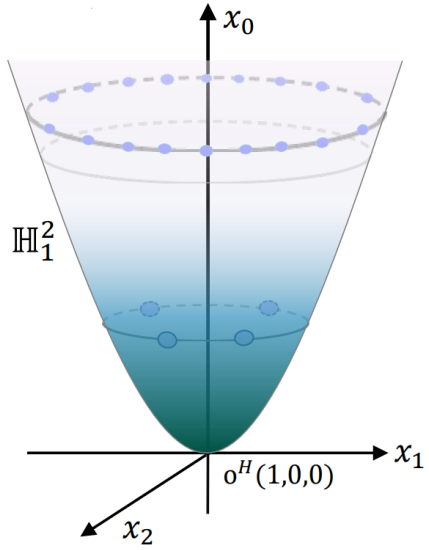


Multivariate Gaussian Distribution

Items' initialized embedding

Users' initialized embedding

$\mathcal{T}_o\mathbb{H}$



$N(\mathbf{0}, \sigma\boldsymbol{I})$

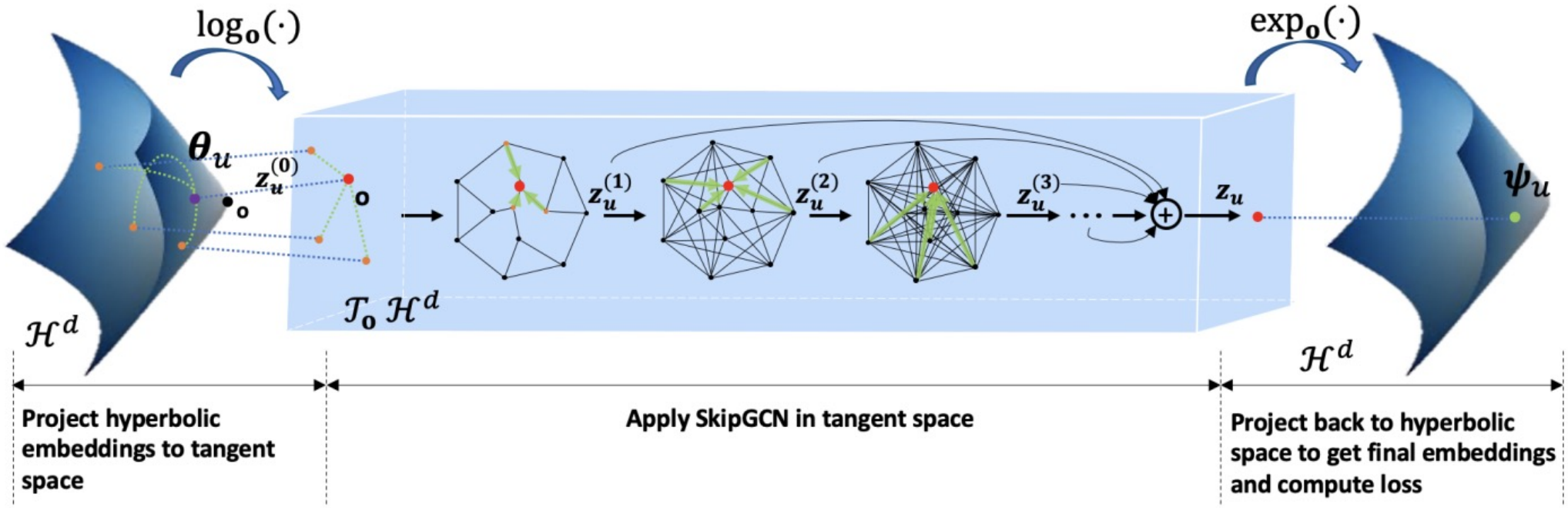
Item: $\mathrm{x}_i \sim N(\mathbf{0}, \sigma\boldsymbol{I})$
User: $\mathrm{x}_u \sim N(\mathbf{0}, \sigma\boldsymbol{I})$

Item: $(0, \mathrm{x}_i)$
User: $(0, \mathrm{x}_u)$

Item: $\mathrm{x}_\mathrm{i}^\mathrm{H} = \exp_\mathrm{o}(0, \mathrm{x}_\mathrm{i})$
User: $\mathrm{x}_\mathrm{i}^\mathrm{H} = \exp_\mathrm{o}(0, \mathrm{x}_\mathrm{u})$

HGCF (WWW '21)

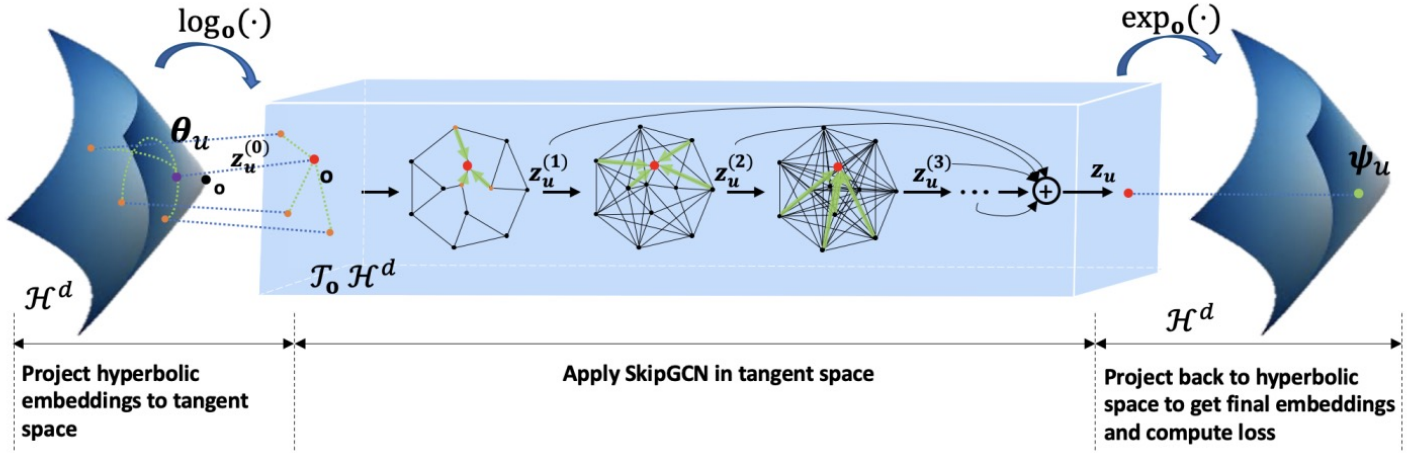
- Hyperbolic Embedding Initializing Layer
- Hyperbolic Message Aggregation



Sun, Jianing, et al. "Hgcf: Hyperbolic graph convolution networks for collaborative filtering." *Proceedings of the Web Conference 2021*.

HGCF (WWW '21)

- Hyperbolic Embedding Initializing Layer
- Hyperbolic Message Aggregation
- Loss Function

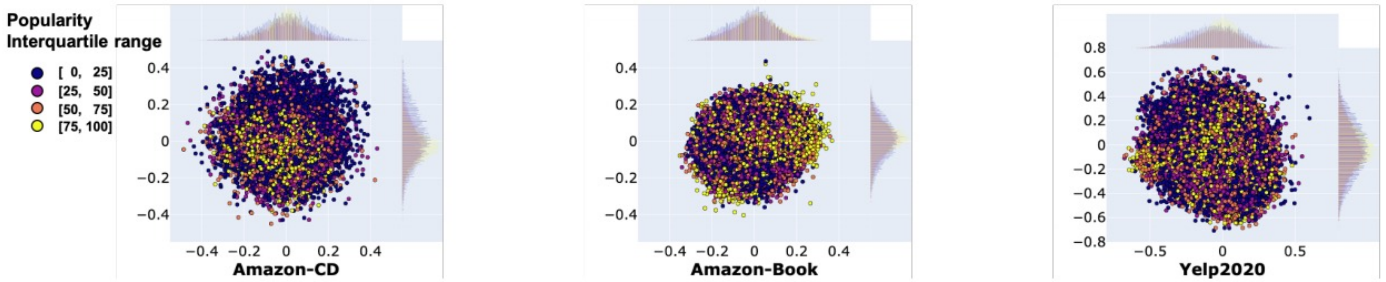


Loss Function

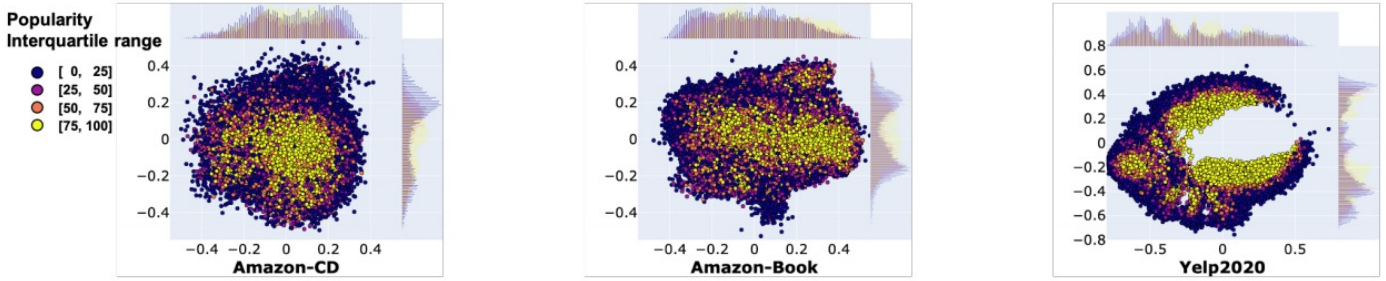
$$L_{\text{margin}}(u,i,j) = \max(d_{\mathcal{L}}(\mathbf{e}_u^H, \mathbf{e}_i^H)^2 - d_{\mathcal{L}}(\mathbf{e}_u^H, \mathbf{e}_j^H)^2 + m, 0)$$

Sun, Jianing, et al. "Hgcf: Hyperbolic graph convolution networks for collaborative filtering." *The WebConf 2021*

HGCF (WWW '21)



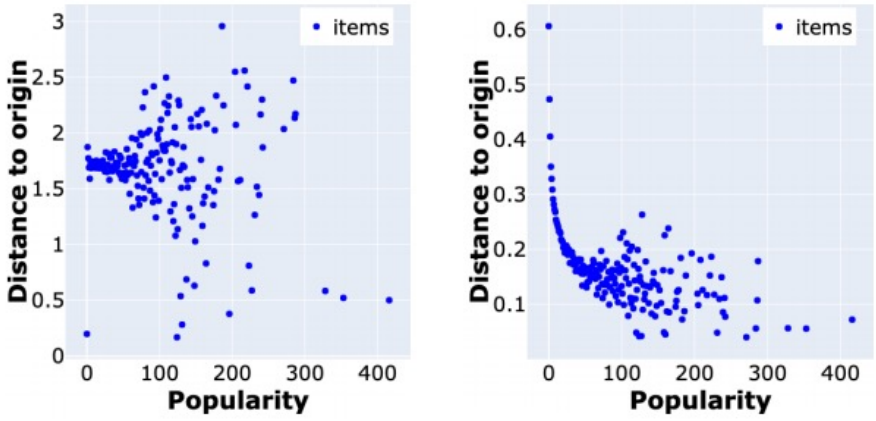
(a) Item embeddings **before** the graph convolution layers.



(b) Item embeddings **after** the graph convolution layers.

- After the graph convolution layers, a clear hierarchy appears according to popularity.
- In the Amazon datasets this hierarchy is reflected in an almost circular way with the most popular items at the center and the less popular ones away from it.

- BPR tends to cluster less popular items at specific distance from the origin.
- HGCF on the other hand has a clear exponential trend where distance to the origin increases exponentially for less popular items



(a) BPR (b) HGCF

Sun, Jianing, et al. "Hgcf: Hyperbolic graph convolution networks for collaborative filtering." *The WebConf 2021*

LKGR (WSDM '22)

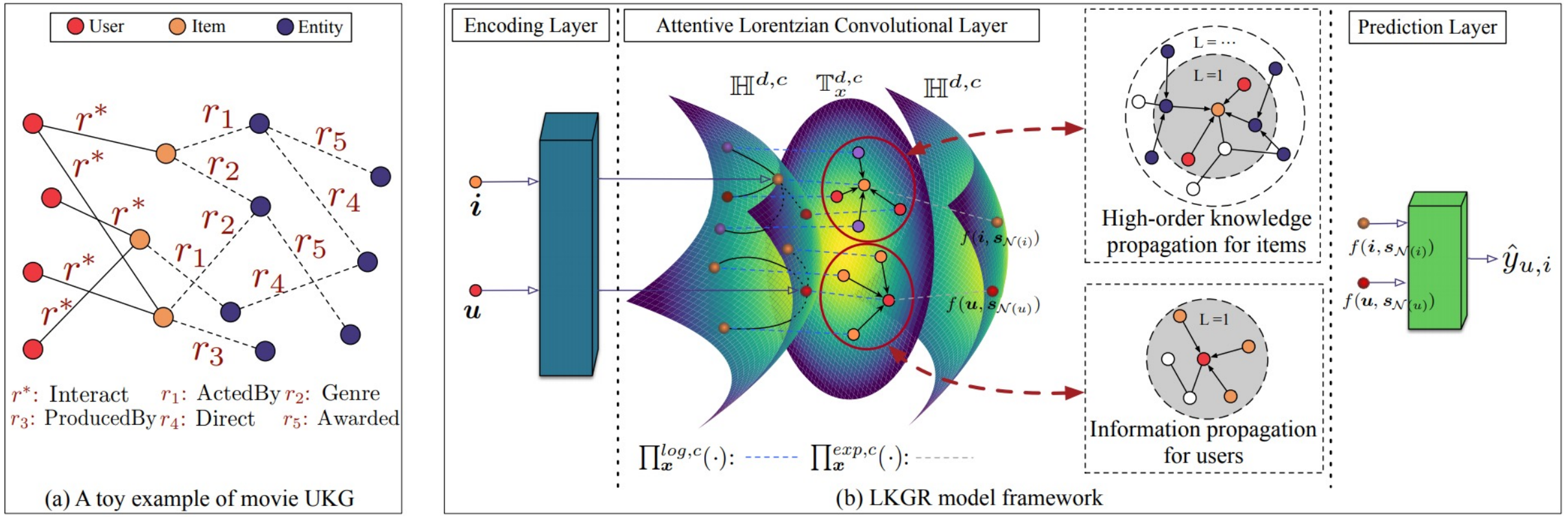
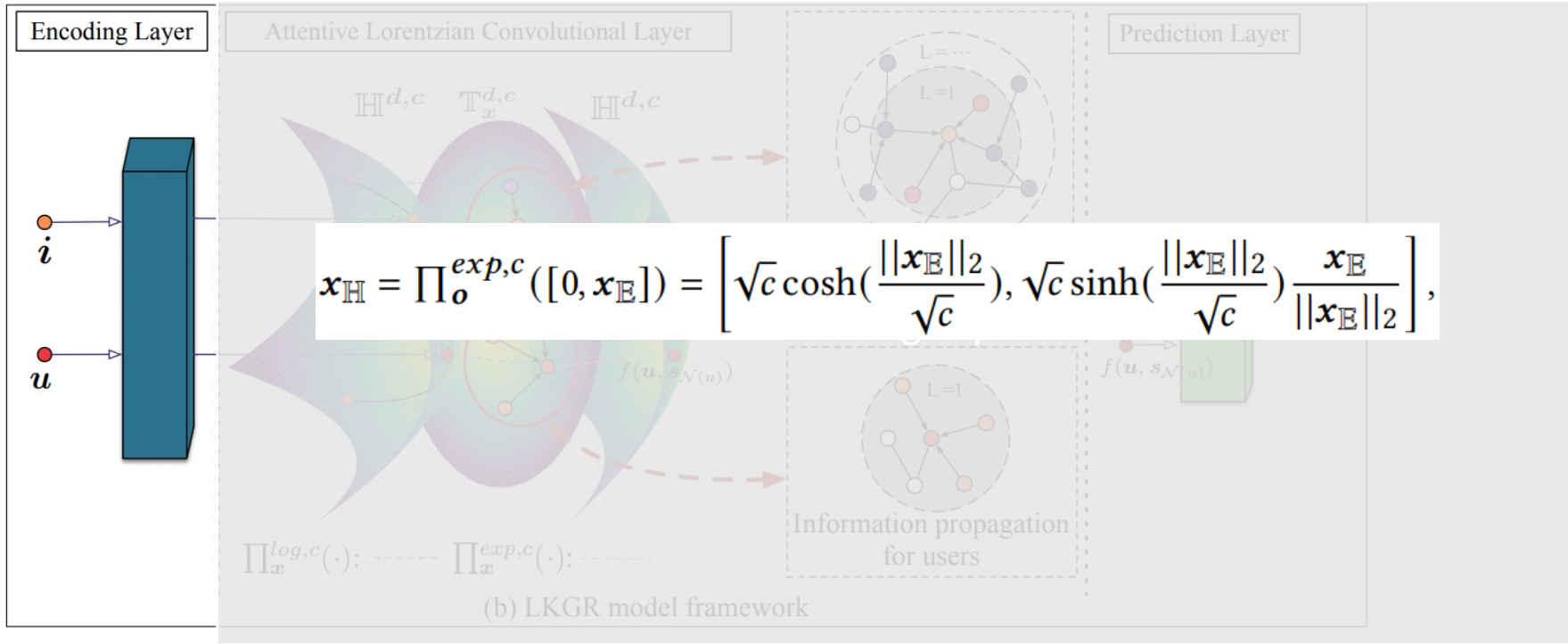


Figure 2: (a) The tripartite graph modeling users, items and KG entities. (b) Illustration of the proposed LKGR model.

Chen, Yankai, et al. "Modeling Scale-free Graphs with Hyperbolic Geometry for Knowledge-aware Recommendation." WSDM '22.

LKGR (WSDM '22)

- **Encoding Layer**

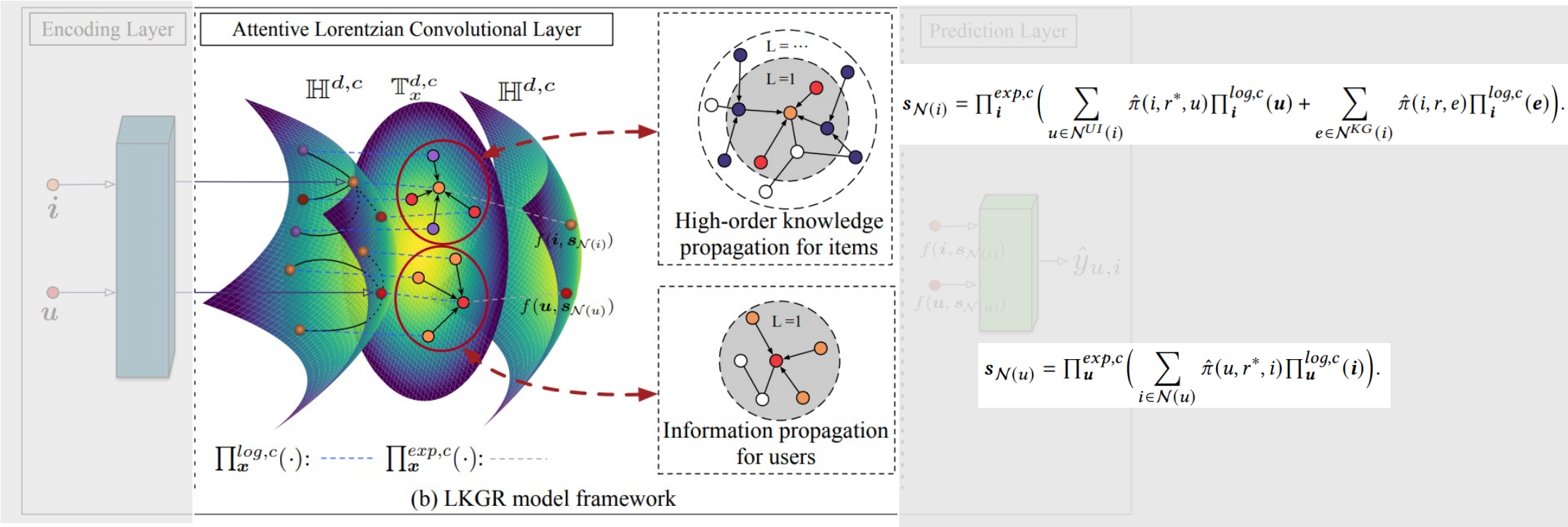


$$x_{\mathbb{H}} = \prod_o^{exp,c}([0, x_{\mathbb{E}}]) = \left[\sqrt{c}\cosh(\frac{||x_{\mathbb{E}}||_2}{\sqrt{c}}), \sqrt{c}\sinh(\frac{||x_{\mathbb{E}}||_2}{\sqrt{c}})\frac{x_{\mathbb{E}}}{||x_{\mathbb{E}}||_2}\right],$$

Chen, Yankai, et al. "Modeling Scale-free Graphs with Hyperbolic Geometry for Knowledge-aware Recommendation." WSDM '22.

LKGR (WSDM '22)

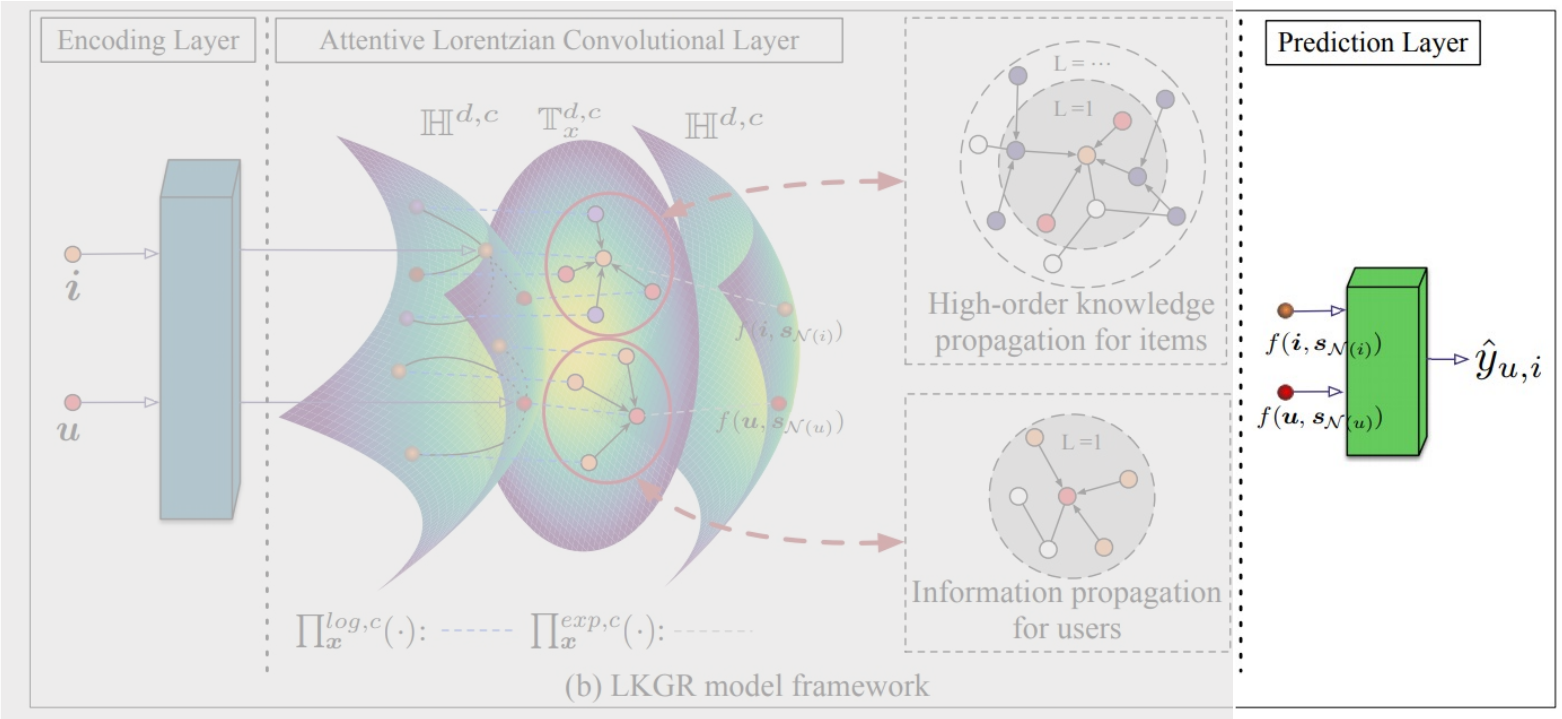
- **Attentive Lorentzian Convolutional Layer**



Chen, Yankai, et al. "Modeling Scale-free Graphs with Hyperbolic Geometry for Knowledge-aware Recommendation." WSDM '22.

LKGR (WSDM '22)

- **Prediction Layer**



$$\hat{y}_{u,i} = g(\boldsymbol{u}, \boldsymbol{i}) = \left(\prod_o^{log,c}(\boldsymbol{u})\right)^T \prod_o^{log,c}(\boldsymbol{i}).$$

Chen, Yankai, et al. "Modeling Scale-free Graphs with Hyperbolic Geometry for Knowledge-aware Recommendation." WSDM '22.

LKGR (WSDM '22)

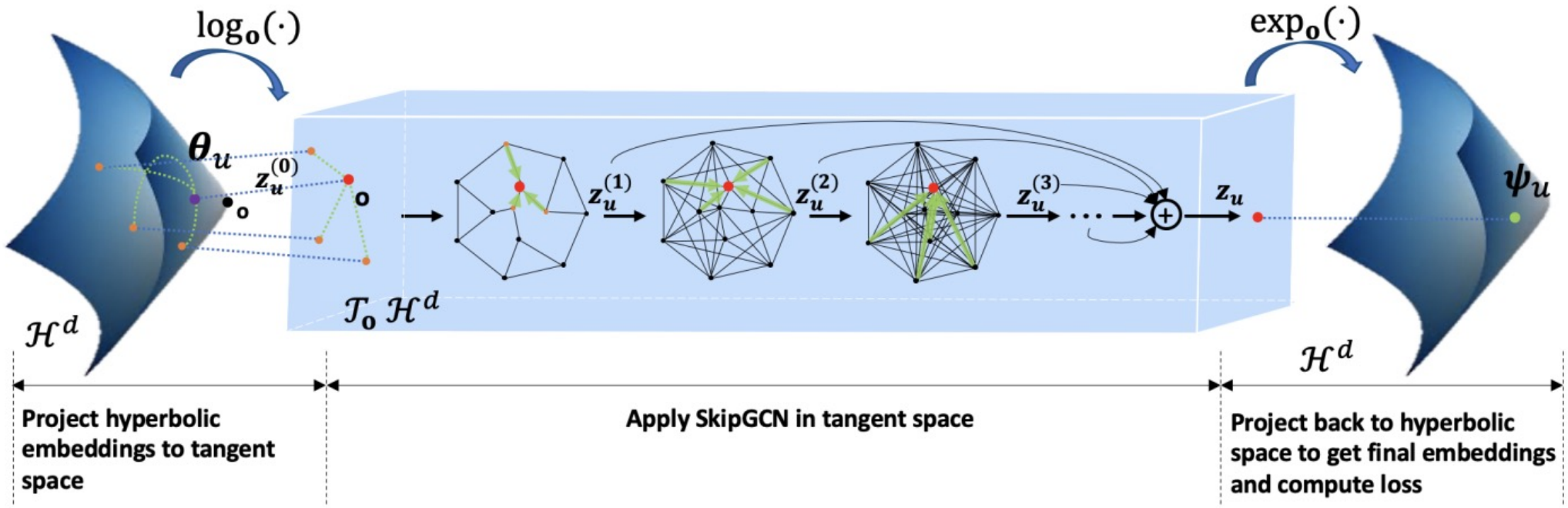
Table 2: Average results of Top@20 recommendation task. Underline indicates the second-best model performance. Bold denotes the empirical improvements against second-best models, and * denotes scenarios where a Wilcoxon signed-rank test indicates a statistically significant improvement under 95% confidence level between our model and second-best models.

Model	Book-Crossing		MovieLens-20M		Dianping-Food	
	Recall@20(%)	NDCG@20(%)	Recall@20(%)	NDCG@20(%)	Recall@20(%)	NDCG@20(%)
BPRMF	4.67 (8.7e-3)	2.80 (4.3e-3)	20.48 (1.6e-2)	15.77 (9.1e-3)	19.90 (3.0e-2)	10.79 (2.0e-2)
NFM	3.93 (2.2e-2)	2.17 (1.5e-2)	19.79 (3.3e-2)	14.28 (1.1e-2)	23.85 (3.9e-2)	<u>12.48</u> (3.0e-2)
CKE	4.38 (9.6e-3)	2.24 (4.2e-2)	21.52 (1.2e-2)	15.73 (1.3e-2)	22.24 (3.1e-2)	12.09 (1.5e-2)
RippleNet	7.12 (2.1e-2)	5.09 (1.7e-2)	13.74 (2.6e-2)	9.77 (1.7e-2)	21.20 (4.1e-2)	10.99 (2.0e-2)
KGNN-LS	<u>8.51</u> (2.2e-2)	<u>6.06</u> (1.7e-2)	20.20 (1.0e-2)	15.49 (1.3e-2)	15.52 (4.9e-2)	7.92 (2.9e-2)
KGCN	7.85 (2.9e-2)	5.93 (2.3e-2)	19.24 (3.2e-2)	13.87 (1.6e-2)	19.03 (3.0e-2)	9.34 (1.5e-2)
KGAT	5.34 (6.1e-3)	3.01 (7.9e-3)	<u>21.80</u> (7.7e-3)	<u>16.81</u> (1.1e-2)	15.57 (2.4e-2)	7.67 (1.7e-2)
CKAN	6.19 (1.1e-2)	3.47 (5.3e-3)	17.48 (1.7e-2)	12.48 (1.4e-2)	<u>24.10</u> (3.9e-2)	**13.33** (2.0e-2)
LKGR	**9.48*** (2.3e-2)	**6.50*** (1.6e-2)	**25.14*** (4.1e-2)	**18.34*** (4.7e-2)	**24.97*** (4.4e-2)	10.45 (1.9e-2)
% Improv.	11.40%	7.26%	15.32%	9.10%	3.61%	N/A

Chen, Yankai, et al. "Modeling Scale-free Graphs with Hyperbolic Geometry for Knowledge-aware Recommendation." WSDM '22.

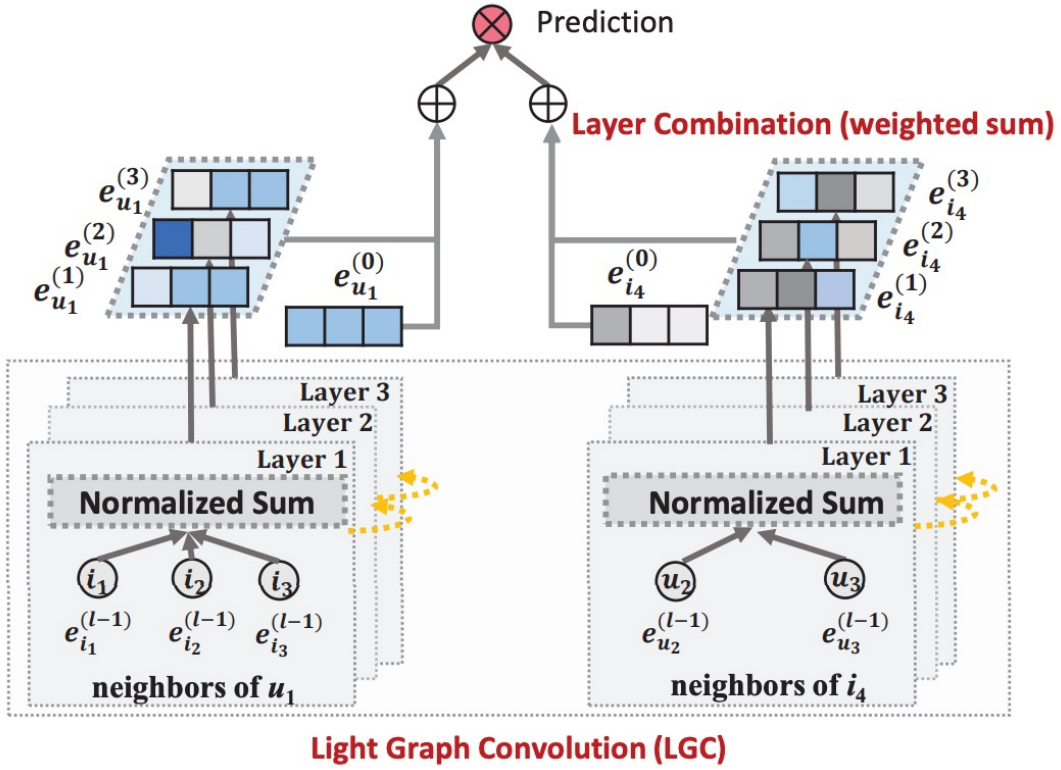
Revisit: HGCF (WWW '21)

- GNN for user-item network modeling in hyperbolic space



Sun, Jianing, et al. "Hgcf: Hyperbolic graph convolution networks for collaborative filtering." *The WebConf 2021*

Revisit: LightGCN (SIGIR '20)



$$\mathbf{e}_u^{(k+1)} = \sum_{i \in \mathcal{N}_u} \frac{1}{\sqrt{|\mathcal{N}_u|}\sqrt{|\mathcal{N}_i|}} \mathbf{e}_i^{(k)},$$

$$\mathbf{e}_i^{(k+1)} = \sum_{u \in \mathcal{N}_i} \frac{1}{\sqrt{|\mathcal{N}_i|}\sqrt{|\mathcal{N}_u|}} \mathbf{e}_u^{(k)}.$$

Note: where $\mathbf{e}_u^k$ and $\mathbf{e}_i^k$ respectively denote the refined embedding of user u and item i after k layers propagation. N_u denotes the set of items that are interacted by user u, N_i denotes the set of users that interact with item i.

Xiangnan He, et al. "Lightgcn: Simplifying and powering graph convolution network for recommendation." SIGIR. 2020.

Revisit: Experimental Comparison

Table 2: Recall (top table) and NDCG (bottom table) results for all datasets. The best performing model on each dataset and metric is highlighted in bold, and second best model is underlined. Asterisks denote statistically significant Wilcoxon signed rank test for the difference in scores between the best and second best models.

Datasets		BPRMF	WRMF	VAE-CF	TransCF	CML	LRML	SML	NGCF	LightGCN	HAE	HVAE	Ours
Amazon-CD	R@10	0.0779	0.0863	0.0786	0.0518	0.0864	0.0502	0.0475	0.0758	0.0929	0.0666	0.0781	**0.0962***
	R@20	0.1200	0.1313	0.1155	0.0791	0.1341	0.0771	0.0734	0.1150	0.1404	0.0963	0.1147	**0.1455***
Amazon-Book	R@10	0.0611	0.0623	0.0740	0.0407	0.0665	0.0522	0.0479	0.0658	0.0799	0.0634	0.0774	**0.0867***
	R@20	0.0974	0.0919	0.1066	0.0632	0.1023	0.0834	0.0768	0.1050	0.1248	0.0912	0.1125	**0.1318***
Yelp2020	R@10	0.0325	0.0470	0.0429	0.0247	0.0363	0.0326	0.0319	0.0458	0.0522	0.0360	0.0421	**0.0543***
	R@20	0.0556	0.0793	0.0706	0.0424	0.0638	0.0562	0.0544	0.0764	0.0866	0.0588	0.0691	**0.0884***

Datasets		BPRMF	WRMF	VAE-CF	TransCF	CML	LRML	SML	NGCF	LightGCN	HAE	HVAE	Ours
Amazon-CD	N@10	0.0610	0.0651	0.0615	0.0396	0.0639	0.0405	0.0361	0.0591	0.0726	0.0565	0.0629	**0.0751***
	N@20	0.0974	0.0817	0.0752	0.0488	0.0813	0.0492	0.0456	0.0718	0.0881	0.0657	0.0749	**0.0909***
Amazon-Book	N@10	0.0594	0.0563	0.0716	0.0392	0.0624	0.0515	0.0422	0.0655	0.0780	0.0709	0.0778	**0.0869***
	N@20	0.0971	0.0730	0.0878	0.0474	0.0808	0.0626	0.0550	0.0791	0.0938	0.0789	0.0901	**0.1022***
Yelp2020	N@10	0.0283	0.0372	0.0353	0.0214	0.0310	0.0287	0.0255	0.0405	**0.0461**	0.0331	0.0371	0.0458
	N@20	0.0512	0.0506	0.0469	0.0277	0.0428	0.0369	0.0347	0.0513	0.0582	0.0409	0.0465	**0.0585***

Questions: In what aspect, hyperbolic model is superior to the Euclidean counterpart?

Sun, Jianing, et al. "Hgcf: Hyperbolic graph convolution networks for collaborative filtering." *The WebConf 2021*

HICF (KDD '22)

Table 1: Statistics of the experimental data.

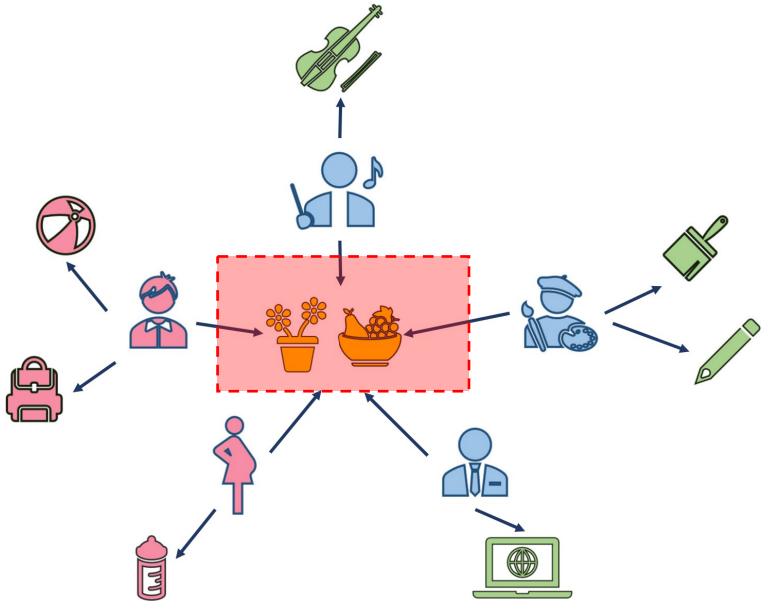
Dataset	#User	#Item			#Interactions	Density
		All	H20(%)	T80(%)		
Amazon-CD	22,947	18,395	46	54	422,301	0.10%
Amazon-Book	52,406	41,264	47	53	1,861,118	0.09%
Yelp2020	71,135	45,063	62	37	1,940,014	0.05%

Yang, Menglin, Zhihao Li, Min Zhou, Jiahong Liu, and Irwin King. "HICF: Hyperbolic Informative Collaborative Filtering." KDD 2022.

HICF (KDD '22)

Table 1: Statistics of the experimental data.

Dataset	#User	#Item			#Interactions	Density
		All	H20(%)	T80(%)		
Amazon-CD	22,947	18,395	46	54	422,301	0.10%
Amazon-Book	52,406	41,264	47	53	1,861,118	0.09%
Yelp2020	71,135	45,063	62	37	1,940,014	0.05%



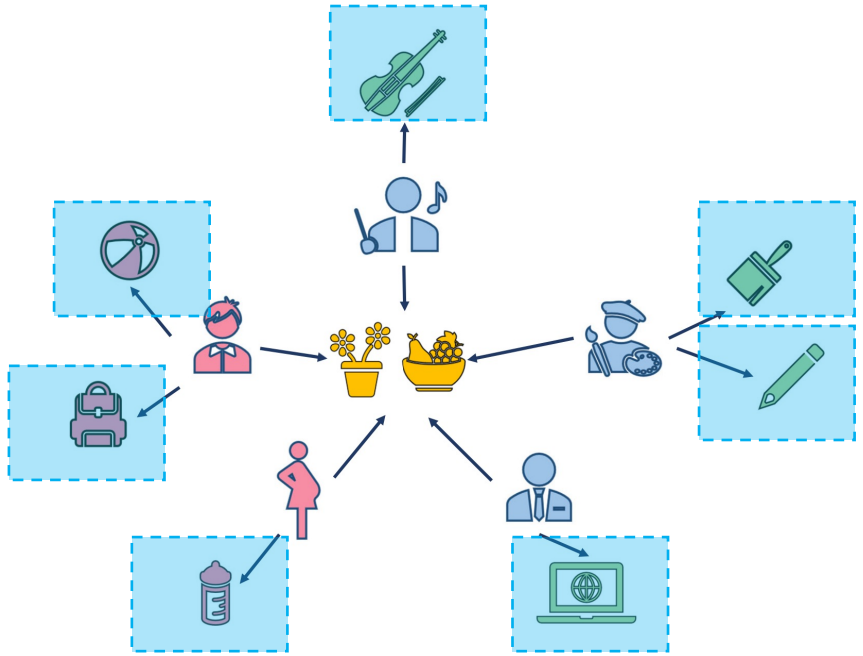
In general, the H20 items account for the minority.

Yang, Menglin, Zhihao Li, Min Zhou, Jiahong Liu, and Irwin King. "HICF: Hyperbolic Informative Collaborative Filtering." KDD 2022.

HICF (KDD '22)

Table 1: Statistics of the experimental data.

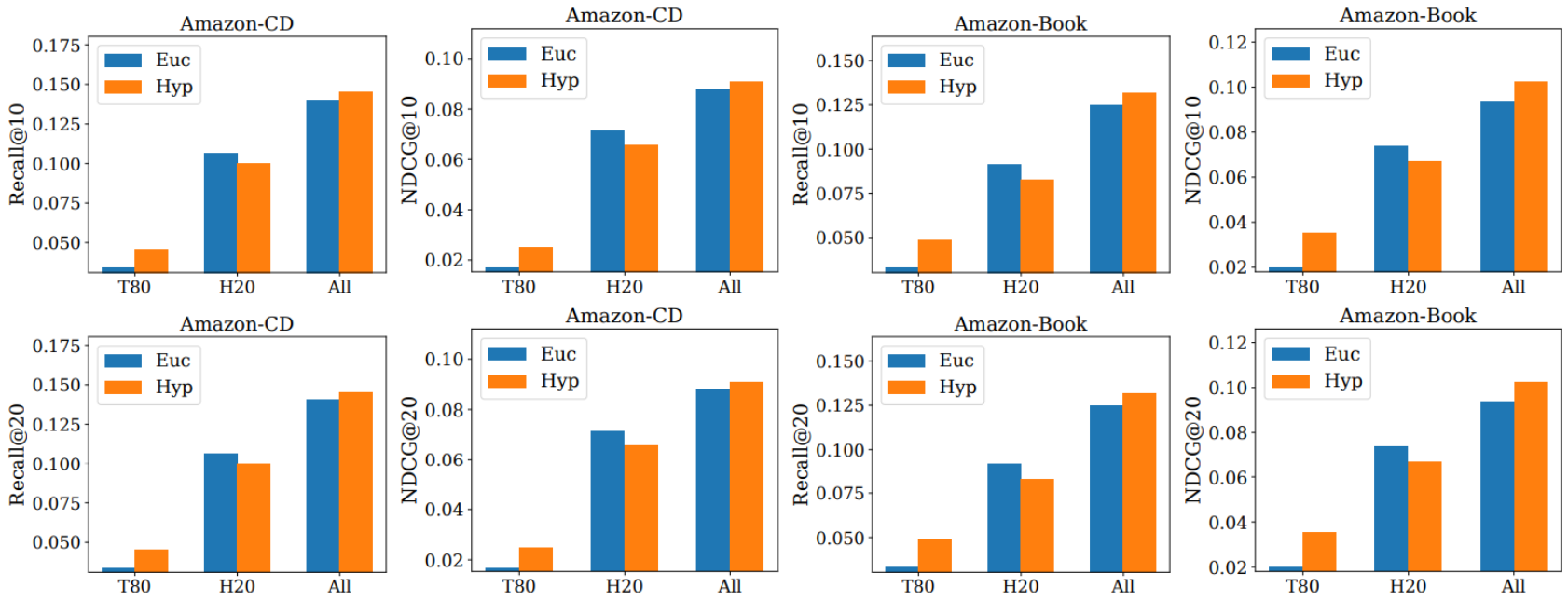
Dataset	#User	#Item			#Interactions	Density
		All	H20(%)	T80(%)		
Amazon-CD	22,947	18,395	46	54	422,301	0.10%
Amazon-Book	52,406	41,264	47	53	1,861,118	0.09%
Yelp2020	71,135	45,063	62	37	1,940,014	0.05%



In general, the T80 items account for the majority.

Yang, Menglin, Zhihao Li, Min Zhou, Jiahong Liu, and Irwin King. "HICF: Hyperbolic Informative Collaborative Filtering." KDD 2022.

HICF (KDD '22)

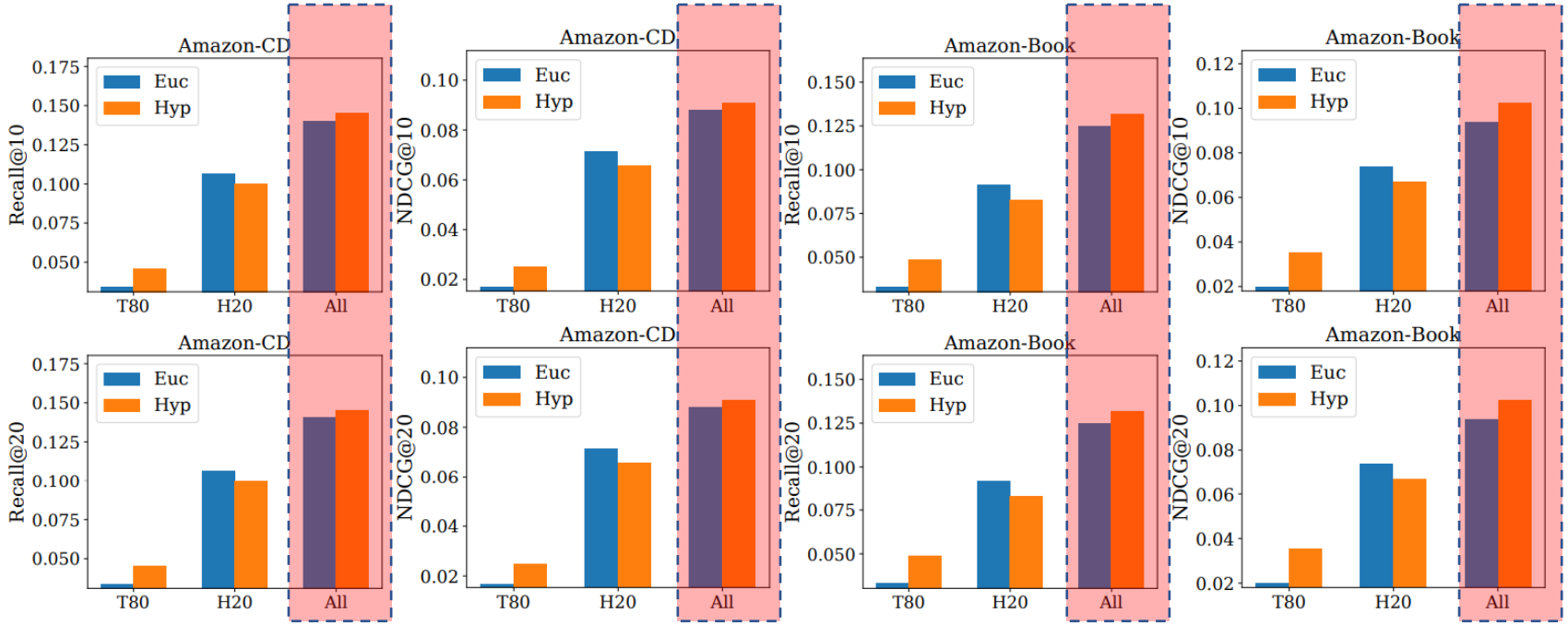


Observations

- The overall recommendation performance of the hyperbolic model is better than that of the Euclidean model.
- Tail items get greater emphasis in the hyperbolic model as the results on tail items are far beyond that of the Euclidean counterpart.
- Head items receive moderate attention in the hyperbolic model as the performance of HGCF is sightly lower than that of LightGCN.

Yang, Menglin, Zhihao Li, Min Zhou, Jiahong Liu, and Irwin King. "HICF: Hyperbolic Informative Collaborative Filtering." KDD 2022.

HICF (KDD '22)

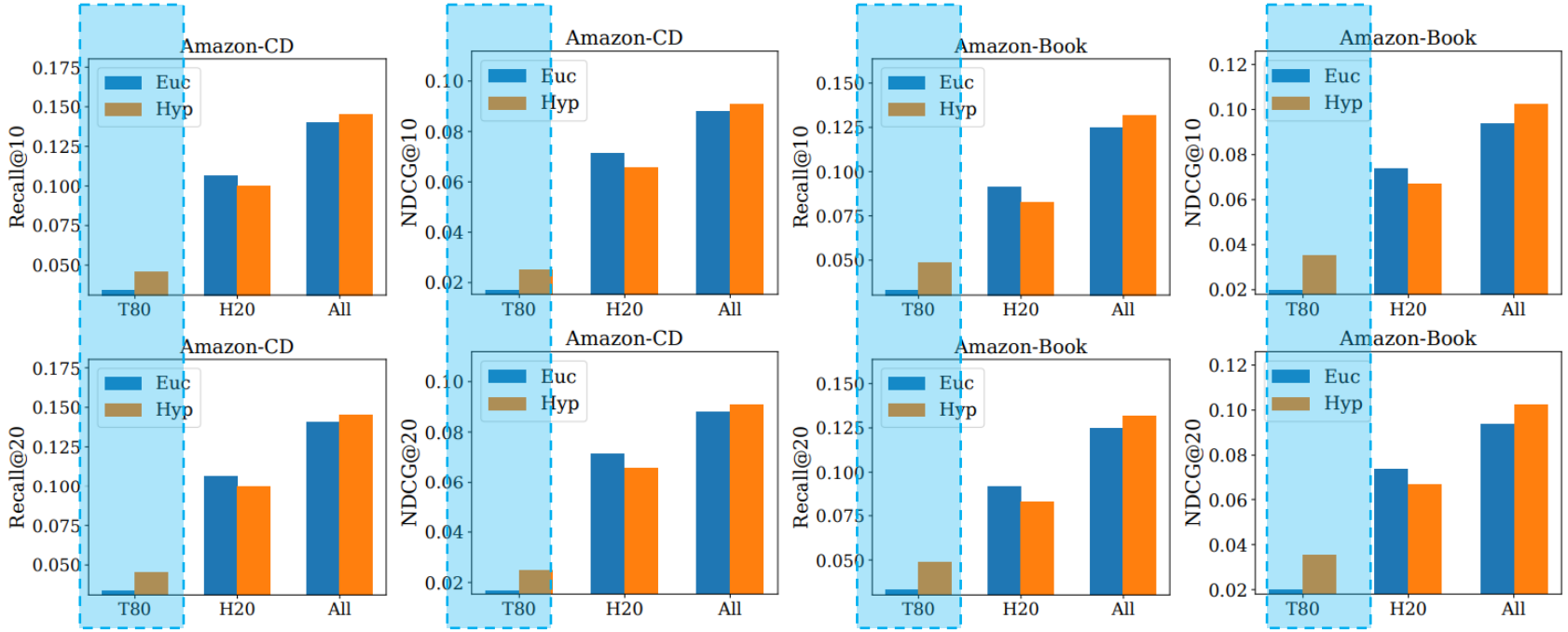


Observations

- The overall recommendation performance of the hyperbolic model is better than that of the Euclidean model.

Yang, Menglin, Zhihao Li, Min Zhou, Jiahong Liu, and Irwin King. "HICF: Hyperbolic Informative Collaborative Filtering." KDD 2022.

HICF (KDD '22)

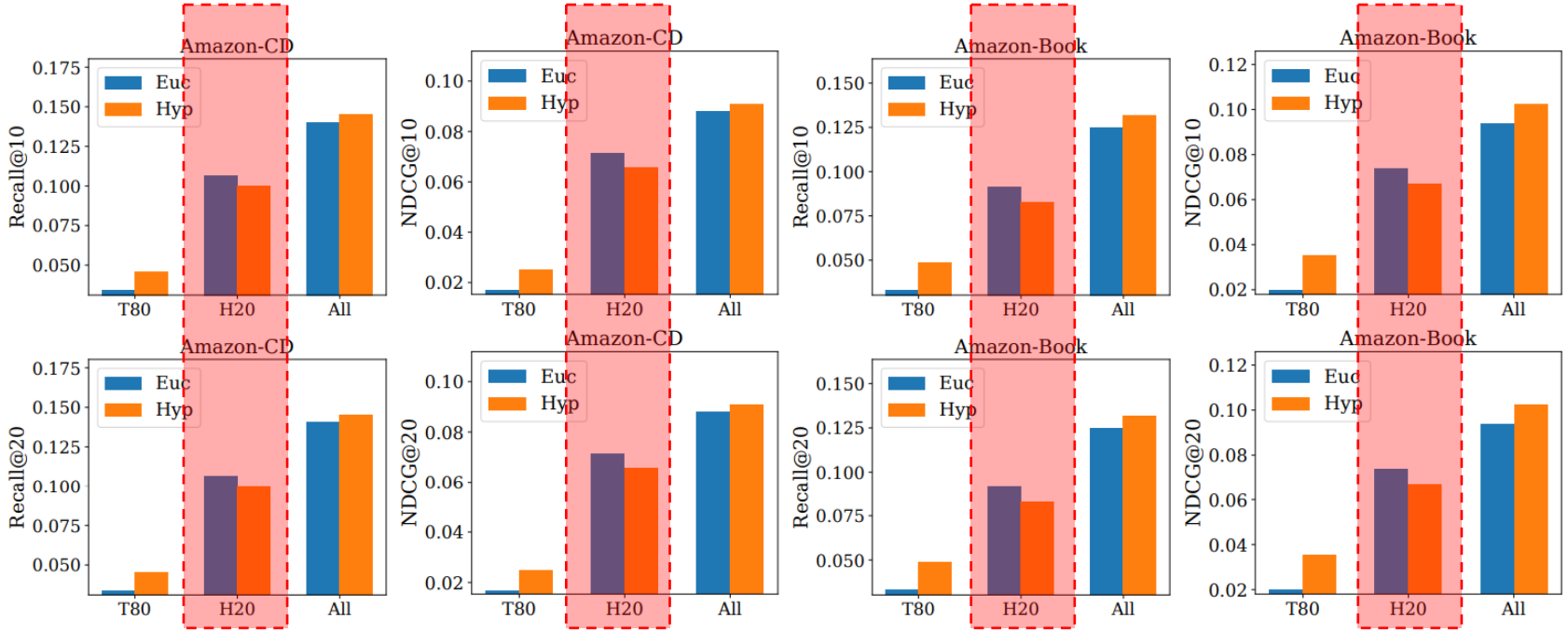


Observations

- The overall recommendation performance of the hyperbolic model is better than that of the Euclidean model.
- Tail items get greater emphasis in the hyperbolic model as the results on tail items are far beyond that of the Euclidean counterpart.

Yang, Menglin, Zhihao Li, Min Zhou, Jiahong Liu, and Irwin King. "HICF: Hyperbolic Informative Collaborative Filtering." KDD 2022.

HICF (KDD '22)

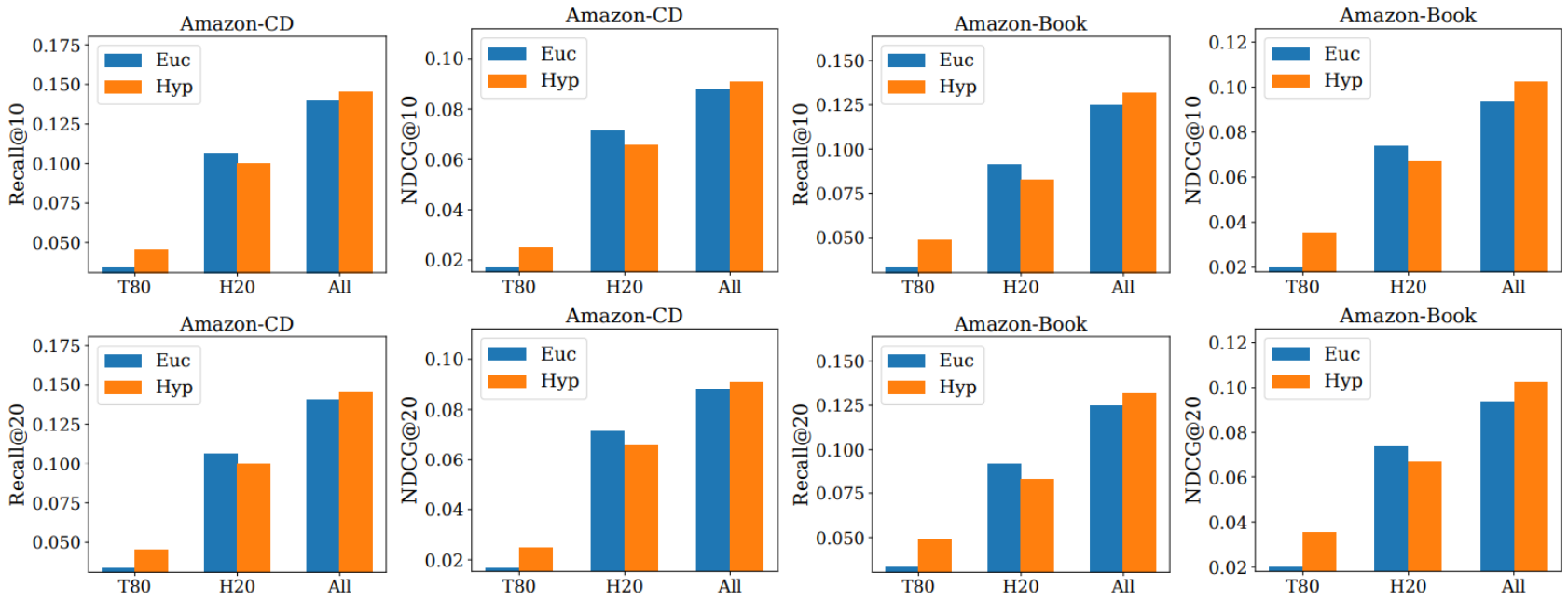


Observations

- The overall recommendation performance of the hyperbolic model is better than that of the Euclidean model.
- Tail items get greater emphasis in the hyperbolic model as the results on tail items are far beyond that of the Euclidean counterpart.
- Head items receive moderate attention in the hyperbolic model as the performance of HGCF is sightly lower than that of LightGCN.

Yang, Menglin, Zhihao Li, Min Zhou, Jiahong Liu, and Irwin King. "HICF: Hyperbolic Informative Collaborative Filtering." KDD 2022.

HICF (KDD '22)

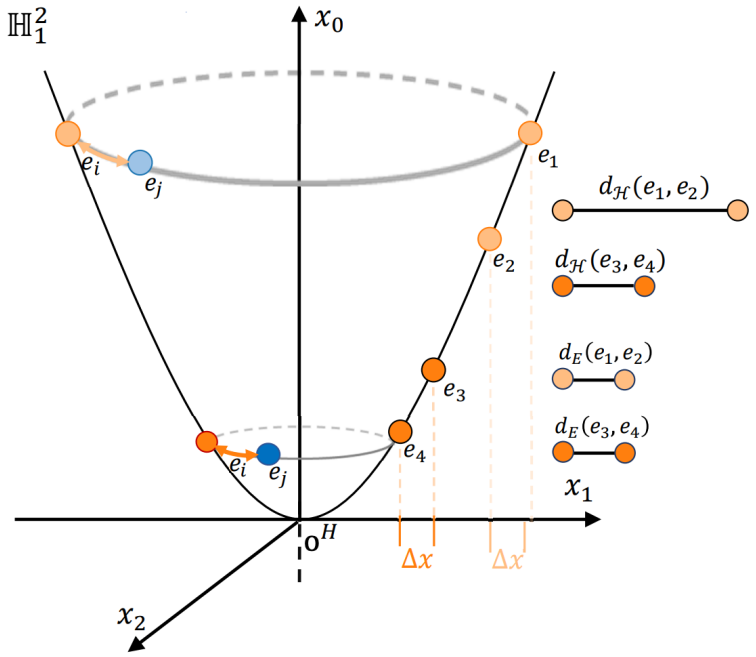


Problems

- There is still large room for improvement for tail items.
- There is an urgent need to improve the performance on head items.

Yang, Menglin, Zhihao Li, Min Zhou, Jiahong Liu, and Irwin King. "HICF: Hyperbolic Informative Collaborative Filtering." KDD 2022.

HICF (KDD '22)



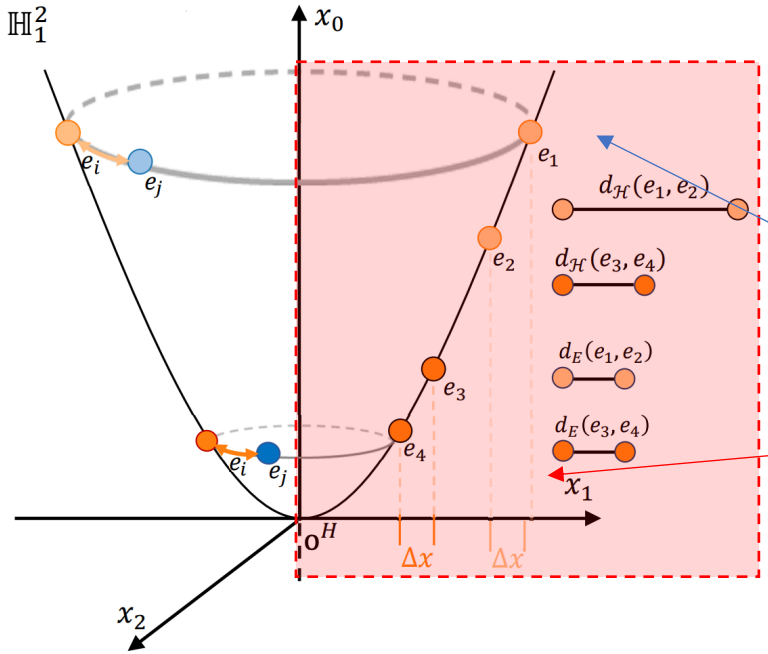
$$\ell(u, i, j) = \max(\underbrace{d^2_{\mathcal{H}}(\mathbf{e}_u, \mathbf{e}_i)}_{\text{Pull}} - \underbrace{d^2_{\mathcal{H}}(\mathbf{e}_u, \mathbf{e}_j)}_{\text{Push}} + m, 0),$$

Note: d_H is the hyperbolic distance and m determines the margin between the distance difference between positive pair (u, i) and negative pair (u,j).

Yang, Menglin, Zhihao Li, Min Zhou, Jiahong Liu, and Irwin King. "HICF: Hyperbolic Informative Collaborative Filtering." KDD 2022.

HICF (KDD '22)

Hyperbolic-aware Margin Learning (HAML)



Large Margin

Small Margin

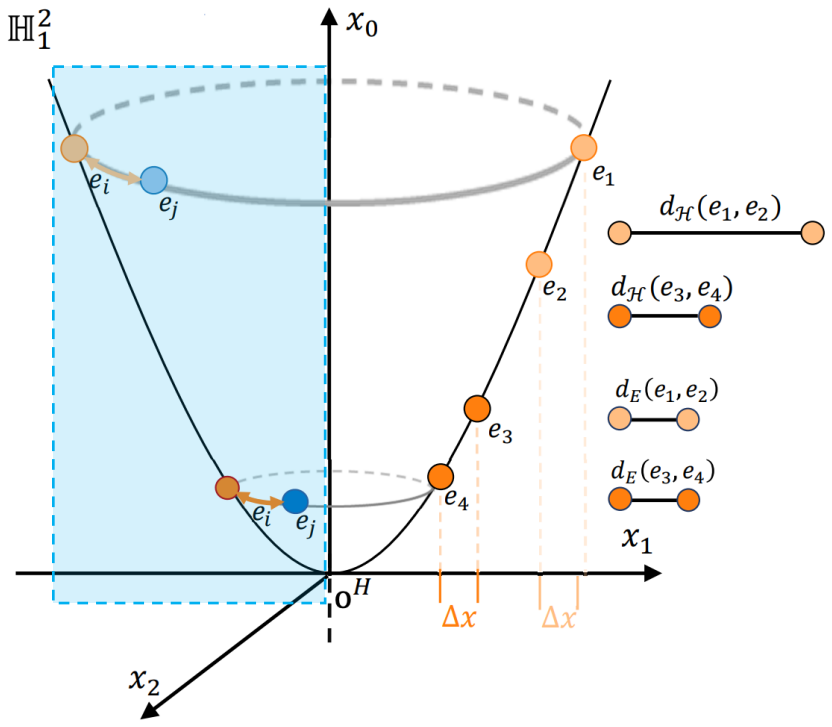
$$\ell(u,i,j) = \max(\underbrace{d^2_{\mathcal{H}}(\mathbf{e}_u, \mathbf{e}_i)}_{\text{Pull}} - \underbrace{d^2_{\mathcal{H}}(\mathbf{e}_u, \mathbf{e}_j)}_{\text{Push}} + m, 0),$$

$$m^{\mathcal{H}}_{ui} = \text{sigmoid}(\delta),$$

$$\delta = \frac{d^2_{\mathcal{H}}(\mathbf{e}_u, \mathbf{o}) + d^2_{\mathcal{H}}(\mathbf{e}_i, \mathbf{o}) - d^2_{\mathcal{H}}(\mathbf{e}_u, \mathbf{e}_i)}{\mathbf{e}_{u,0}\mathbf{e}_{i,0}},$$

Yang, Menglin, Zhihao Li, Min Zhou, Jiahong Liu, and Irwin King. "HICF: Hyperbolic Informative Collaborative Filtering." KDD 2022.

HICF (KDD '22)



Informative Sampling

Hyperbolic Informative Negative Sampling

Algorithm 1: HINS algorithm

Input: Hyper parameters n_{neg}; Item set $\mathcal{I}$; the embedding matrix E; The index of the user u, its current positive item i and its other positive item $\mathcal{N}_u$ in the training set.

Output: The informative item index j.

Random sample n_{neg} items $\mathcal{I}_u^{[n]}$ from $\mathcal{I} \backslash \mathcal{N}_u$;

forall *item index* $\bar{j}$ *in* $\mathcal{I}_u^{[n]}$ **do**
- Get the embeddings of the $\bar{j}$ from E, i.e., $\mathbf{e}_{\bar{j}}$;
- Let $j = -1, d_{\min} = +\infty$;
- Compute the hyperbolic distance $d_{\mathcal{H}}(\mathbf{e}_{\bar{j}}, \mathbf{e}_i)$;
- **if** $d_{\mathcal{H}}(\mathbf{e}_{\bar{j}}, \mathbf{e}_i) < d_{\min}$ **then**
  - $d_{\min} = d_{\mathcal{H}}(\mathbf{e}_{\bar{j}}, \mathbf{e}_i)$;
  - $j = \bar{j}$;
- **end**

end

Return j;

Yang, Menglin, Zhihao Li, Min Zhou, Jiahong Liu, and Irwin King. "HICF: Hyperbolic Informative Collaborative Filtering." KDD 2022.

HICF (KDD '22)

Datasets		WRMF	VAE-CF	TransCF	CML	LRML	SML	NGCF	LightGCN	HAE	HAVE	HGCF	Ours	Δ(%)
Amazon-CD	R@10	0.0863	0.0786	0.0518	0.0864	0.0502	0.0475	0.0758	0.0929	0.0666	0.0781	0.0962	**0.1079***	+12.16
	R@20	0.1313	0.1155	0.0791	0.1341	0.0771	0.0734	0.1150	0.1404	0.0963	0.1147	0.1455	**0.1586***	+9.00
Amazon-Book	R@10	0.0623	0.0740	0.0407	0.0665	0.0522	0.0479	0.0658	0.0799	0.0634	0.0774	0.0867	**0.0965***	+11.30
	R@20	0.0919	0.1066	0.0632	0.1023	0.0834	0.0768	0.1050	0.1248	0.0912	0.1125	0.1318	**0.1449***	+9.94
Yelp2020	R@10	0.0470	0.0429	0.0247	0.0363	0.0326	0.0319	0.0458	0.0522	0.0360	0.0421	0.0527	**0.0570***	+8.16
	R@20	0.0793	0.0706	0.0424	0.0638	0.0562	0.0544	0.0764	0.0866	0.0588	0.0691	0.0884	**0.0948***	+7.24

Datasets		WRMF	VAE-CF	TransCF	CML	LRML	SML	NGCF	LightGCN	HAE	HAVE	HGCF	Ours	Δ(%)
Amazon-CD	N@10	0.0651	0.0615	0.0396	0.0639	0.0405	0.0361	0.0591	0.0726	0.0565	0.0629	0.0751	**0.0848***	+12.92
	N@20	0.0817	0.0752	0.0488	0.0813	0.0492	0.0456	0.0718	0.0881	0.0657	0.0749	0.0909	**0.1010***	+11.11
Amazon-Book	N@10	0.0563	0.0716	0.0392	0.0624	0.0515	0.0422	0.0655	0.0780	0.0709	0.0778	0.0869	**0.0978***	+12.54
	N@20	0.0730	0.0878	0.0474	0.0808	0.0626	0.0550	0.0791	0.0938	0.0789	0.0901	0.1022	**0.1142***	+11.74
Yelp2020	N@10	0.0372	0.0353	0.0214	0.0310	0.0287	0.0255	0.0405	0.0461	0.0331	0.0371	0.0458	**0.0502***	+9.13
	N@20	0.0506	0.0469	0.0277	0.0428	0.0369	0.0347	0.0513	0.0582	0.0409	0.0465	0.0585	**0.0633***	+8.21

The proposed method outperforms all baselines in both Recall and NDCG metrics.

Yang, Menglin, Zhihao Li, Min Zhou, Jiahong Liu, and Irwin King. "HICF: Hyperbolic Informative Collaborative Filtering." KDD 2022.

HICF (KDD '22)

Datasets	Models	R@20		R@10		R@5		N@20		N@10		N@5	
		H20	T80	H20	T80	H20	T80	H20	T80	H20	T80	H20	T80
Amazon-CD	LightGCN	0.1062	0.0342	0.0741	0.0188	0.0493	0.0104	0.0712	0.0169	0.0608	0.0118	0.0529	0.0084
	HGCF	0.0998	0.0457	0.0667	0.0295	0.0439	0.0179	0.0658	0.0251	0.0550	0.0201	0.0486	0.0157
	HICF(Ours)	**0.1027**	**0.0559**	**0.0717**	**0.0362**	**0.0476**	**0.0222**	**0.0692**	**0.0318**	**0.0596**	**0.0252**	**0.0527**	**0.0197**
	$\Delta_{\mathcal{H}}(\%)$	+2.91	+22.32	+7.50	+22.71	+8.43	+24.02	+5.17	+26.69	+8.36	+25.37	+8.44	+25.48
Amazon-Book	LightGCN	0.0915	0.0333	0.0624	0.0175	0.0390	0.0104	0.0740	0.0198	0.0635	0.0145	0.0589	0.0104
	HGCF	0.0829	0.0489	0.0550	0.0317	0.0344	0.0197	0.0670	0.0352	0.0578	0.0291	0.0539	0.0251
	HICF(Ours)	**0.0898**	**0.0551**	**0.0603**	**0.0362**	**0.0387**	**0.0227**	**0.0738**	**0.0404**	**0.0642**	**0.0336**	**0.0605**	**0.0293**
	$\Delta_{\mathcal{H}}(\%)$	+8.32	+12.68	+9.64	+14.20	+12.50	+15.23	+10.15	+14.77	+11.07	+15.46	+12.24	+16.73
Yelp2020	LightGCN	0.0836	0.0030	0.0512	0.0010	0.0298	0.0003	0.0567	0.0015	0.0448	0.0006	0.0380	0.0004
	HGCF	0.0788	**0.0096**	0.0473	**0.0054**	0.0270	0.0030	0.0526	**0.0059**	0.0417	0.0043	0.0354	**0.0033**
	HICF(Ours)	**0.0854**	0.0094	**0.0518**	0.0052	**0.0299**	0.0029	**0.0576**	0.0057	**0.0461**	0.0041	**0.0395**	0.0032
	$\Delta_{\mathcal{H}}(\%)$	+8.38	-2.08	+9.51	-3.70	+10.74	-3.33	+9.51	-3.39	+10.55	-4.65	+11.58	-3.03

For head items, the performances of HICF comprehensively outperform that of HGCF and the improvements are up to 8.43% on Amazon-CD, 12.50% on Amazon-Book, and 11.58% on Yelp.

In addition, HICF narrows the performance gap of the tail items with Euclidean LightGCN on Amazon-CD and Amazon-Book and impressively surpasses the performance of LightGCN on Yelp.

Yang, Menglin, Zhihao Li, Min Zhou, Jiahong Liu, and Irwin King. "HICF: Hyperbolic Informative Collaborative Filtering." KDD 2022.

HICF (KDD '22)

Datasets	Models	R@20		R@10		R@5		N@20		N@10		N@5	
		H20	T80	H20	T80	H20	T80	H20	T80	H20	T80	H20	T80
Amazon-CD	LightGCN	0.1062	0.0342	0.0741	0.0188	0.0493	0.0104	0.0712	0.0169	0.0608	0.0118	0.0529	0.0084
	HGCF	0.0998	0.0457	0.0667	0.0295	0.0439	0.0179	0.0658	0.0251	0.0550	0.0201	0.0486	0.0157
	HICF(Ours)	**0.1027**	**0.0559**	**0.0717**	**0.0362**	**0.0476**	**0.0222**	**0.0692**	**0.0318**	**0.0596**	**0.0252**	**0.0527**	**0.0197**
	$\Delta_{\mathcal{H}}(\%)$	+2.91	+22.32	+7.50	+22.71	+8.43	+24.02	+5.17	+26.69	+8.36	+25.37	+8.44	+25.48
Amazon-Book	LightGCN	0.0915	0.0333	0.0624	0.0175	0.0390	0.0104	0.0740	0.0198	0.0635	0.0145	0.0589	0.0104
	HGCF	0.0829	0.0489	0.0550	0.0317	0.0344	0.0197	0.0670	0.0352	0.0578	0.0291	0.0539	0.0251
	HICF(Ours)	**0.0898**	**0.0551**	**0.0603**	**0.0362**	**0.0387**	**0.0227**	**0.0738**	**0.0404**	**0.0642**	**0.0336**	**0.0605**	**0.0293**
	$\Delta_{\mathcal{H}}(\%)$	+8.32	+12.68	+9.64	+14.20	+12.50	+15.23	+10.15	+14.77	+11.07	+15.46	+12.24	+16.73
Yelp2020	LightGCN	0.0836	0.0030	0.0512	0.0010	0.0298	0.0003	0.0567	0.0015	0.0448	0.0006	0.0380	0.0004
	HGCF	0.0788	**0.0096**	0.0473	**0.0054**	0.0270	0.0030	0.0526	**0.0059**	0.0417	0.0043	0.0354	**0.0033**
	HICF(Ours)	**0.0854**	0.0094	**0.0518**	0.0052	**0.0299**	0.0029	**0.0576**	0.0057	**0.0461**	0.0041	**0.0395**	0.0032
	$\Delta_{\mathcal{H}}(\%)$	+8.38	-2.08	+9.51	-3.70	+10.74	-3.33	+9.51	-3.39	+10.55	-4.65	+11.58	-3.03

For head items, the performances of HICF comprehensively outperform that of HGCF and the improvements are up to 8.43% on Amazon-CD, 12.50% on Amazon-Book, and 11.58% on Yelp.

In addition, HICF narrows the performance gap of the tail items with Euclidean LightGCN on Amazon-CD and Amazon-Book and impressively surpasses the performance of LightGCN on Yelp.

Yang, Menglin, Zhihao Li, Min Zhou, Jiahong Liu, and Irwin King. "HICF: Hyperbolic Informative Collaborative Filtering." KDD 2022.

HICF (KDD '22)

Datasets	Models	R@20		R@10		R@5		N@20		N@10		N@5	
		H20	T80	H20	T80	H20	T80	H20	T80	H20	T80	H20	T80
Amazon-CD	LightGCN	0.1062	0.0342	0.0741	0.0188	0.0493	0.0104	0.0712	0.0169	0.0608	0.0118	0.0529	0.0084
	HGCF	0.0998	0.0457	0.0667	0.0295	0.0439	0.0179	0.0658	0.0251	0.0550	0.0201	0.0486	0.0157
	HICF(Ours)	**0.1027**	**0.0559**	**0.0717**	**0.0362**	**0.0476**	**0.0222**	**0.0692**	**0.0318**	**0.0596**	**0.0252**	**0.0527**	**0.0197**
	$\Delta_{\mathcal{H}}(\%)$	+2.91	+22.32	+7.50	+22.71	+8.43	+24.02	+5.17	+26.69	+8.36	+25.37	+8.44	+25.48
Amazon-Book	LightGCN	0.0915	0.0333	0.0624	0.0175	0.0390	0.0104	0.0740	0.0198	0.0635	0.0145	0.0589	0.0104
	HGCF	0.0829	0.0489	0.0550	0.0317	0.0344	0.0197	0.0670	0.0352	0.0578	0.0291	0.0539	0.0251
	HICF(Ours)	**0.0898**	**0.0551**	**0.0603**	**0.0362**	**0.0387**	**0.0227**	**0.0738**	**0.0404**	**0.0642**	**0.0336**	**0.0605**	**0.0293**
	$\Delta_{\mathcal{H}}(\%)$	+8.32	+12.68	+9.64	+14.20	+12.50	+15.23	+10.15	+14.77	+11.07	+15.46	+12.24	+16.73
Yelp2020	LightGCN	0.0836	0.0030	0.0512	0.0010	0.0298	0.0003	0.0567	0.0015	0.0448	0.0006	0.0380	0.0004
	HGCF	0.0788	**0.0096**	0.0473	**0.0054**	0.0270	0.0030	0.0526	**0.0059**	0.0417	0.0043	0.0354	**0.0033**
	HICF(Ours)	**0.0854**	0.0094	**0.0518**	0.0052	**0.0299**	0.0029	**0.0576**	0.0057	**0.0461**	0.0041	**0.0395**	0.0032
	$\Delta_{\mathcal{H}}(\%)$	+8.38	-2.08	+9.51	-3.70	+10.74	-3.33	+9.51	-3.39	+10.55	-4.65	+11.58	-3.03

For head items, the performances of HICF comprehensively outperform that of HGCF and the improvements are up to 8.43% on Amazon-CD, 12.50% on Amazon-Book, and 11.58% on Yelp.

In addition, HICF narrows the performance gap of the tail items with Euclidean LightGCN on Amazon-CD and Amazon-Book and impressively surpasses the performance of LightGCN on Yelp.

Yang, Menglin, Zhihao Li, Min Zhou, Jiahong Liu, and Irwin King. "HICF: Hyperbolic Informative Collaborative Filtering." KDD 2022.

Agenda

Part 1. INTRODUCTION (Min, 9:00 - 9:30)

1.1 An Overview of Graph Learning

1.2 Brief Introduction of Riemannian Geometry

1.3 Motivation for Hyperbolic Graph Learning

Coffee Break (9:30 – 10:00)

Part 2. HYPERBOLIC GRAPH NETWORKS (Menglin, 10:00 – 10:30)

2.1 Preliminary

2.2 Hyperbolic Shallow Models

2.2 Hyperbolic Neural Networks

2.3 Hyperbolic Graph Neural Networks

Part 3. APPLICATIONS (Menglin, 10:30 – 11:00)

3.1 HGNN for Recommendation Systems

3.2 HGNN for Knowledge Graph

3.3 HGNN for Bioinformatics

Part 4. ADVANCED TOPICS (Bo, 11:00 – 12:00)

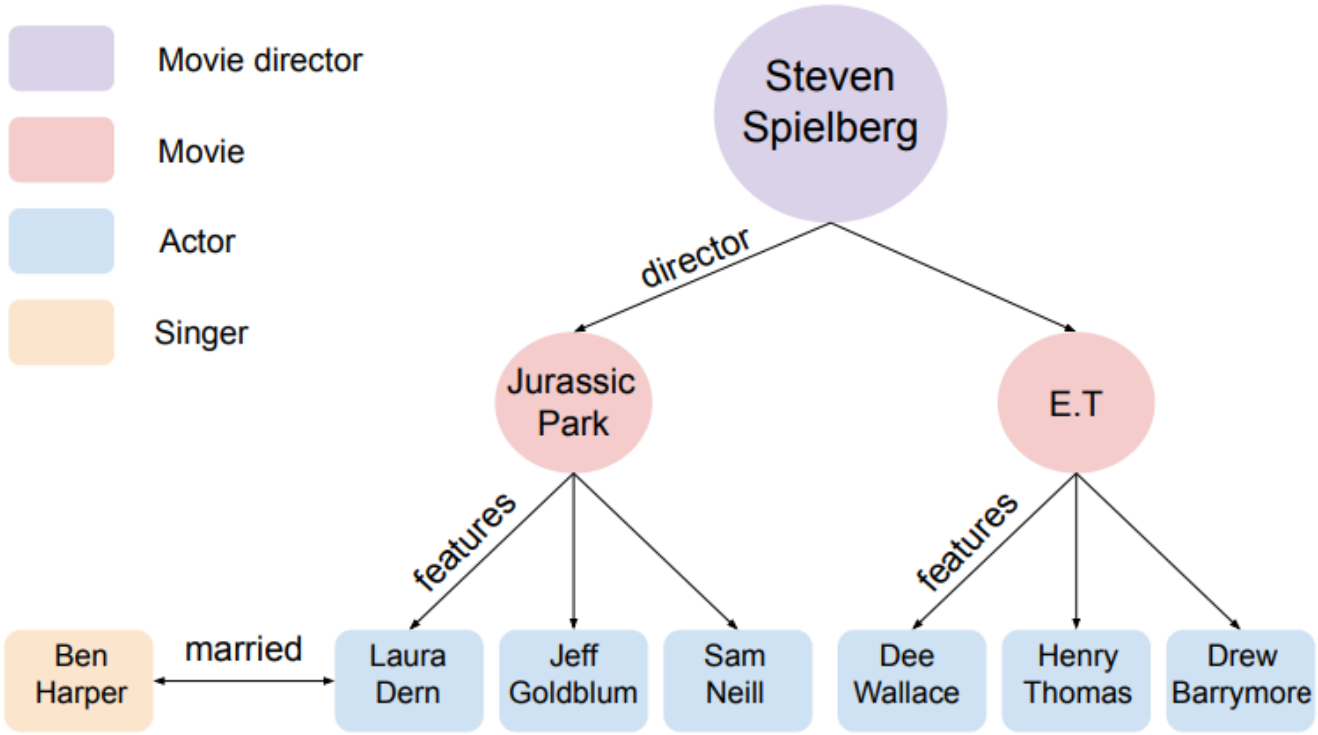
4.1 Complex Structures

4.3 Curvature-aware Learning

4.2 Evolving Interactions

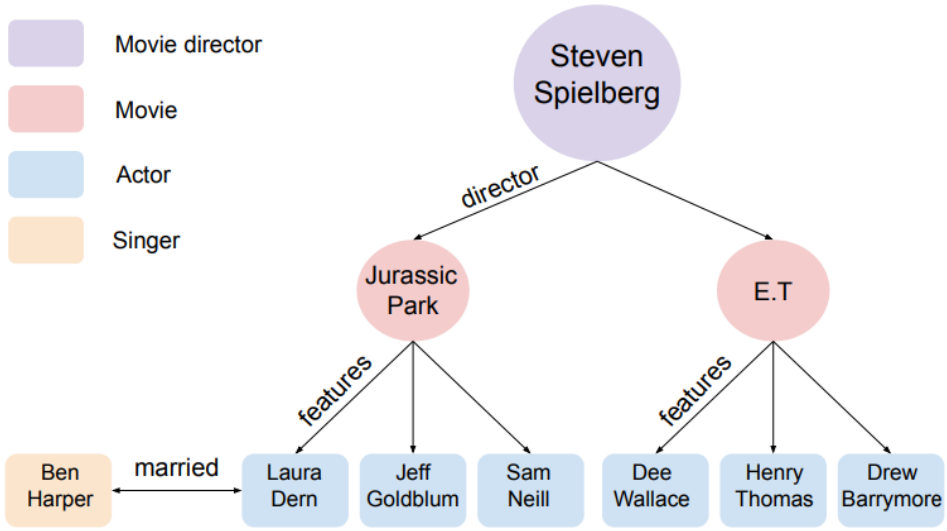
4.4 Trustworthy and Scalability

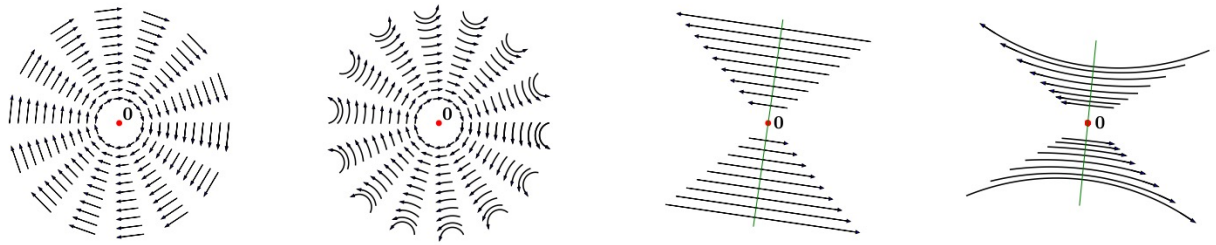
Hyperbolic Knowledge Graph (ACL '19)



Chami, Ines, et al. "Low-dimensional hyperbolic knowledge graph embeddings." *ACL 2019*

Hyperbolic Knowledge Graph (ACL '19)

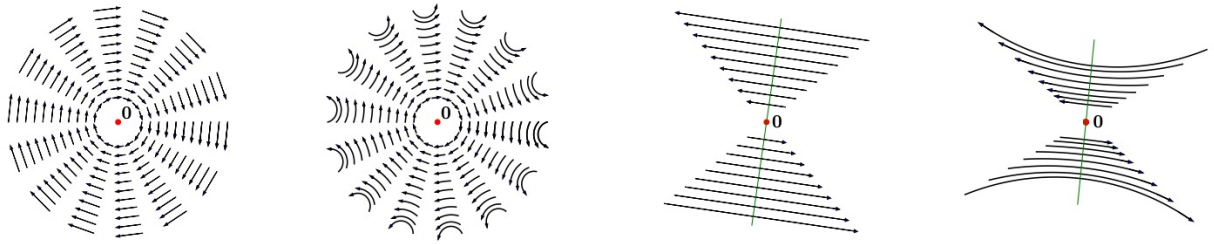




(a) Rotations (b) Reflections

Chami, Ines, et al. "Low-dimensional hyperbolic knowledge graph embeddings." *ACL 2019*

Hyperbolic Knowledge Graph (ACL '19)



(a) Rotations (b) Reflections

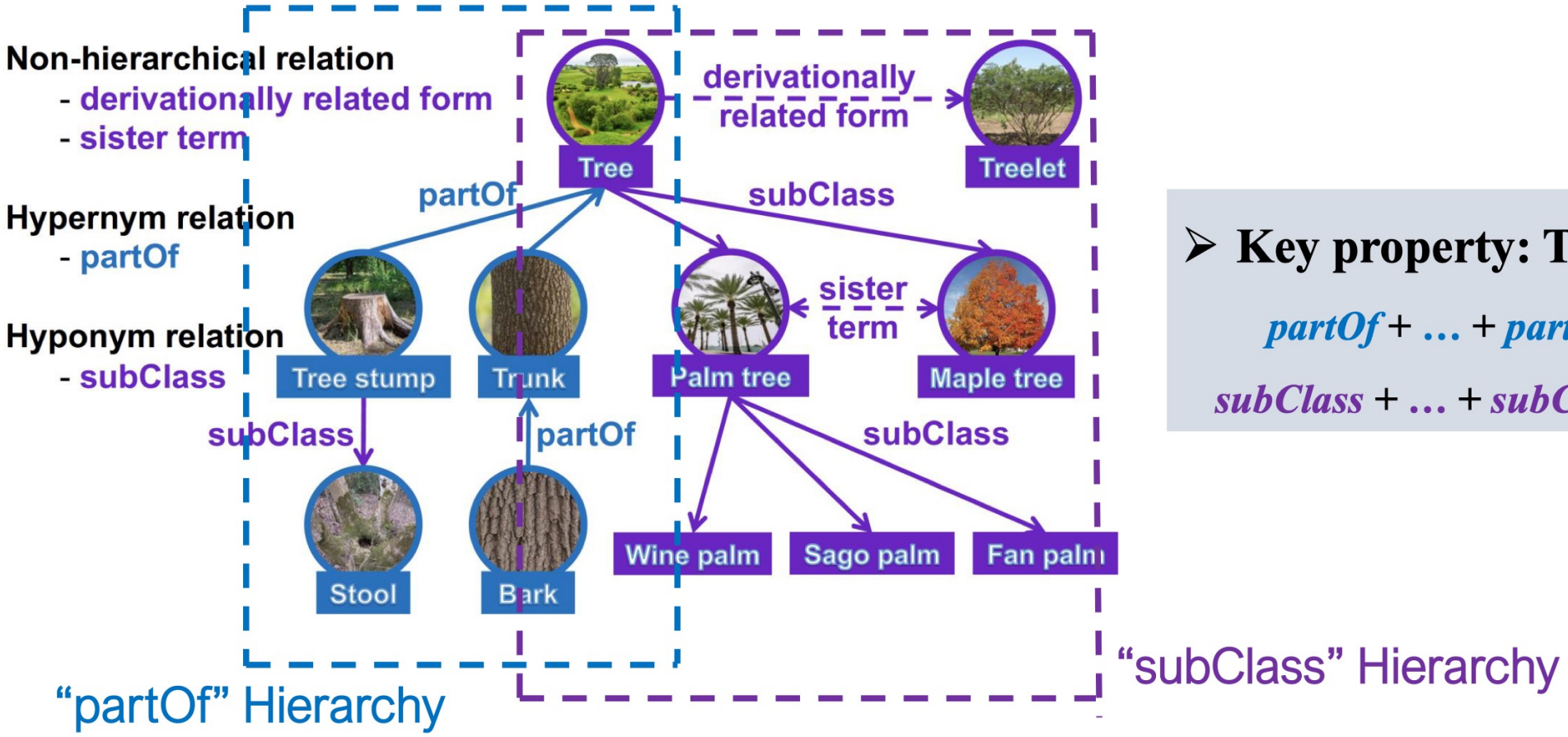
$$\mathrm{Rot}(\Theta_r) = \mathrm{diag}(G^+(\theta_{r,1}), \ldots, G^+(\theta_{r,\frac{d}{2}})), \quad (4)$$

$$\mathrm{Ref}(\Phi_r) = \mathrm{diag}(G^-(\phi_{r,1}), \ldots, G^-(\phi_{r,\frac{n}{2}})), \quad (5)$$

$$\text{where } G^{\pm}(\theta) := \begin{bmatrix} \cos(\theta) & \mp\sin(\theta) \\ \sin(\theta) & \pm\cos(\theta) \end{bmatrix}. \quad (6)$$

Chami, Ines, et al. "Low-dimensional hyperbolic knowledge graph embeddings." *ACL 2019*

ConE (NeurIPS '21)



- **Key property: Transitivity**

$$partOf + \ldots + partOf = partOf$$

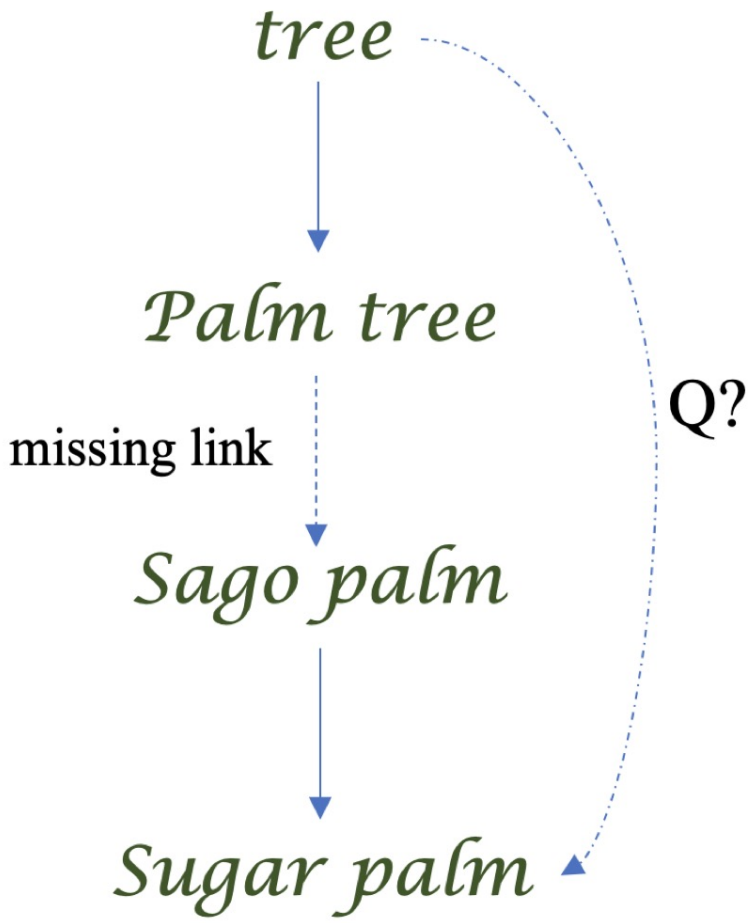
$$subClass + \ldots + subClass = subclass$$

Bai, Yushi, et al. "Modeling heterogeneous hierarchies with relation-specific hyperbolic cones." *NeurIPS* 2021.

ConE (NeurIPS '21)

- To do hierarchical reasoning that involves cross-layer connection
  - Ancestor-Descendant Prediction

- Such task requires:
  - Model Transitivity
  - Model non-hierarchical Properties



Bai, Yushi, et al. "Modeling heterogeneous hierarchies with relation-specific hyperbolic cones." *NeurIPS* 2021

ConE (NeurIPS '21)

Entity ➡ Hyperbolic Cone

Relation

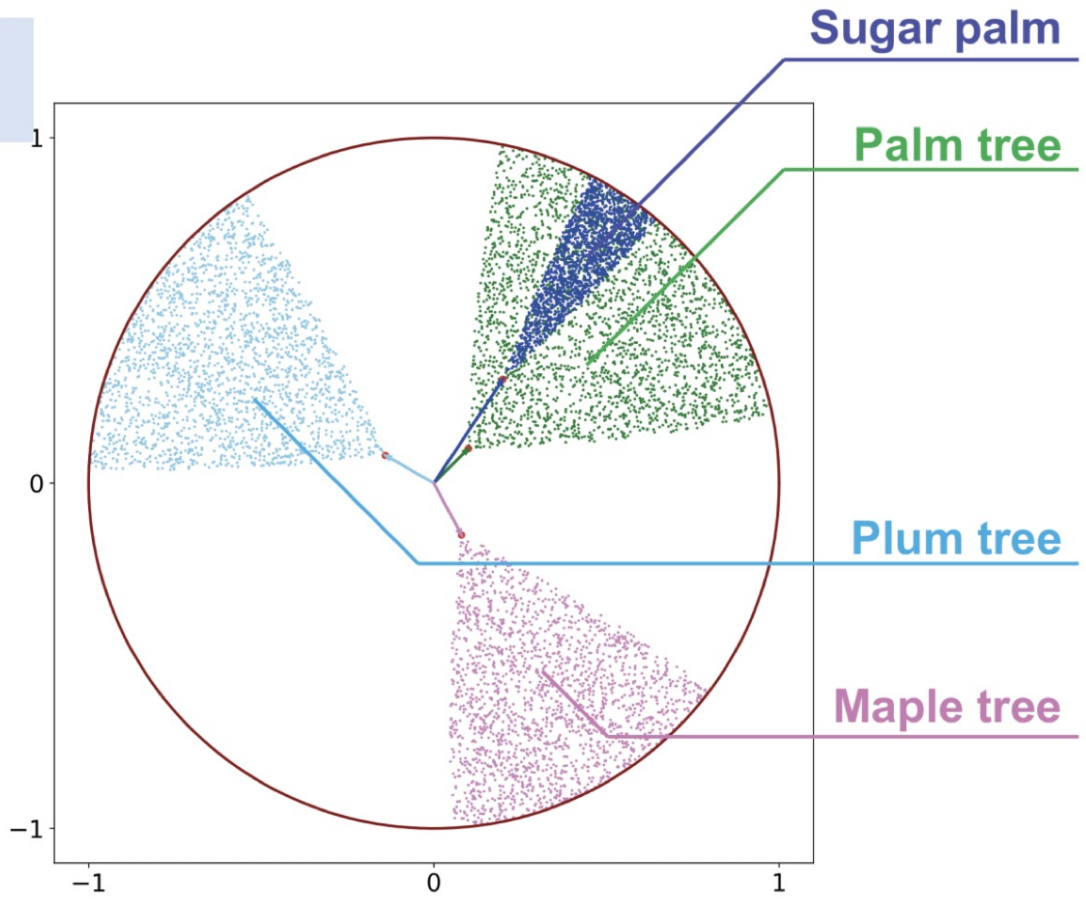
⬇

Cone Transformation

Partial ordering

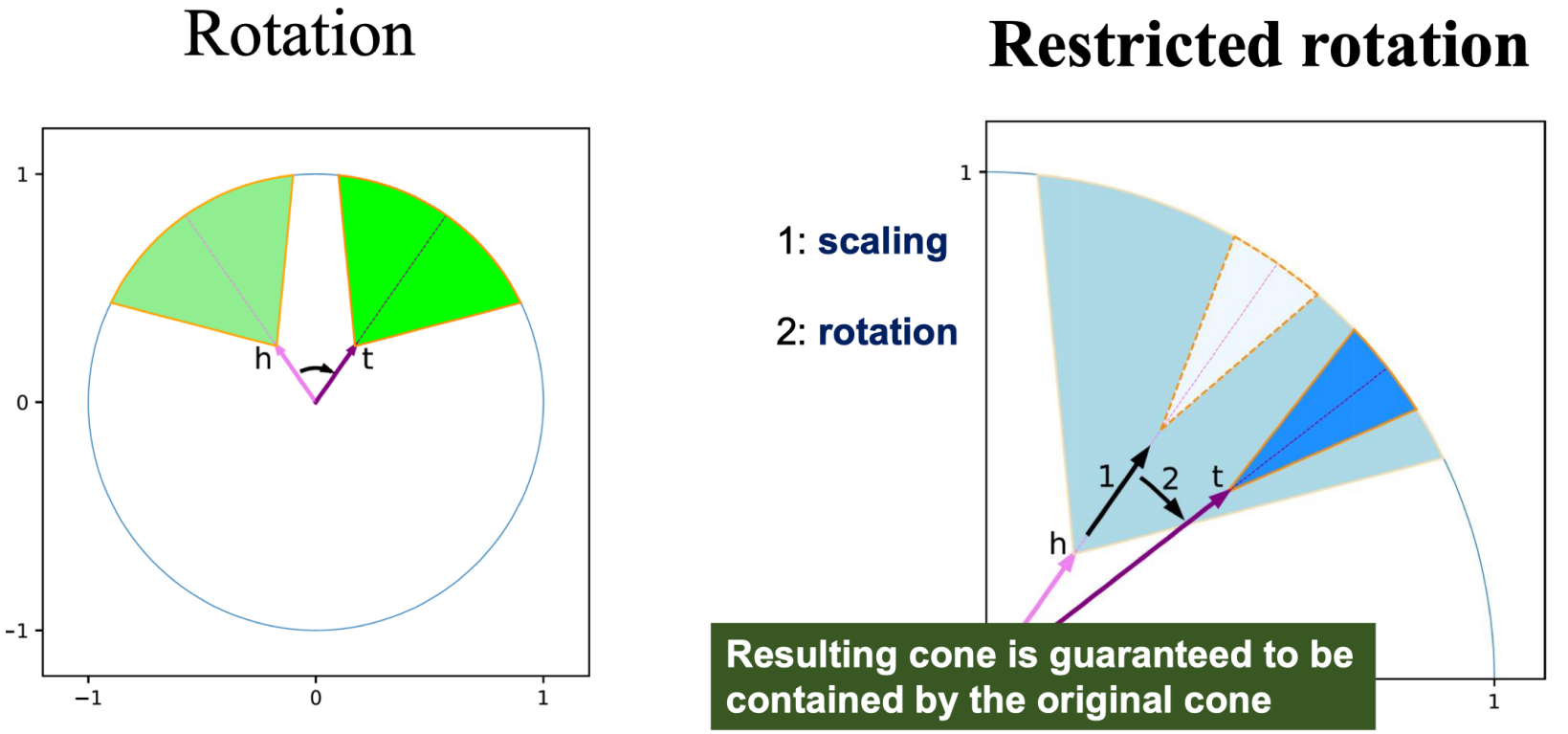
⬇

Cone containment



Bai, Yushi, et al. "Modeling heterogeneous hierarchies with relation-specific hyperbolic cones." *NeurIPS* 2021.

ConE (NeurIPS '21)

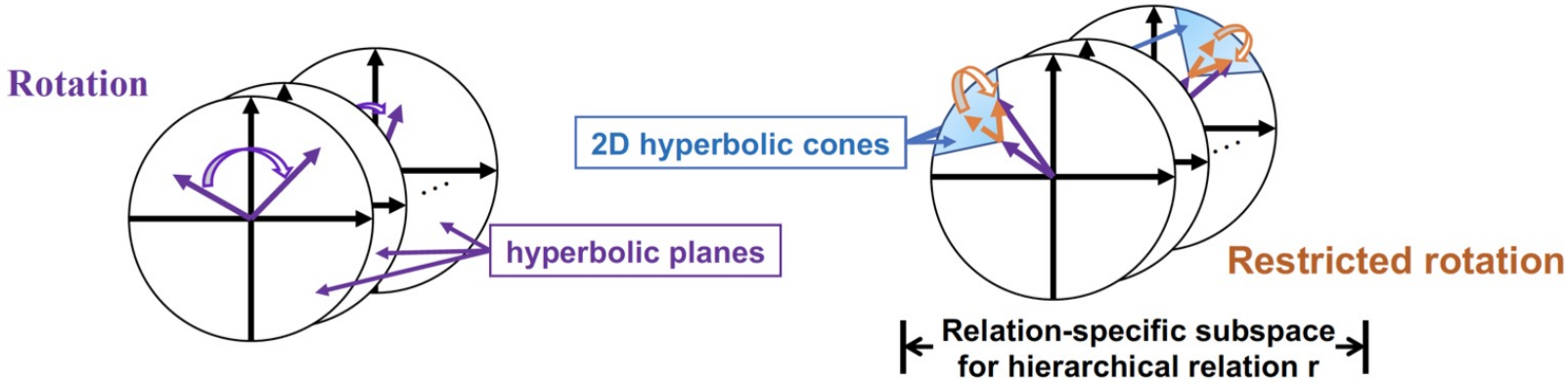


Bai, Yushi, et al. "Modeling heterogeneous hierarchies with relation-specific hyperbolic cones." *NeurIPS* 2021.

ConE (NeurIPS '21)

- Motivation behind two kinds of transformations?
  - Restricted rotation: Capture hierarchical property (transitivity)
  - Rotation: Capture other non-hierarchical properties (composition)

- Why allocating subspace for each hierarchical relation?
  - Partial ordering of different hierarchical relations are preserved in different subspaces



Bai, Yushi, et al. "Modeling heterogeneous hierarchies with relation-specific hyperbolic cones." *NeurIPS* 2021.

HyperKA (EMNLP '20)

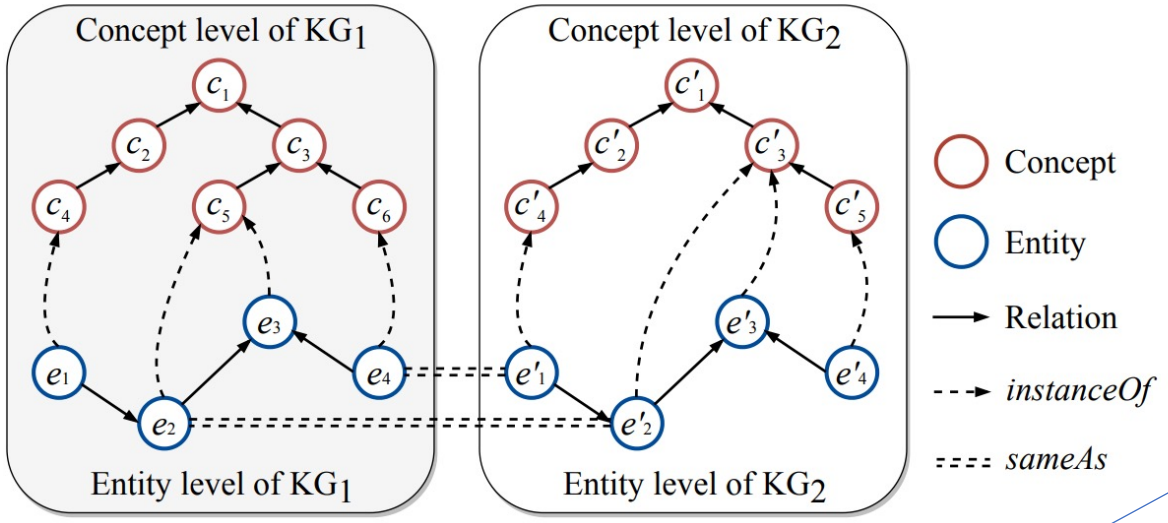


Figure 1: Illustration of two kinds of knowledge associations (i.e., *sameAs* and *instanceOf*) in KGs.

Knowledge association (i and j) can be at different spaces

$$\pi(i,j) = d_{\mathbb{D}}(\mathbf{M} \otimes \mathbf{u}_i, \mathbf{u}_j),$$

$$\mathcal{L}_{\text{proj}} = \sum_{(i,j)\in\mathcal{A}^+} \pi(i,j) + \sum_{(i',j')\in\mathcal{A}^-} [\lambda_2 - \pi(i',j')]_+, \quad (10)$$

Agenda

Part 1. INTRODUCTION (Min, 9:00 - 9:30)

1.1 An Overview of Graph Learning

1.2 Brief Introduction of Riemannian Geometry

1.3 Motivation for Hyperbolic Graph Learning

Coffee Break (9:30 – 10:00)

Part 2. HYPERBOLIC GRAPH NETWORKS (Menglin, 10:00 – 10:30)

2.1 Preliminary

2.2 Hyperbolic Shallow Models

2.2 Hyperbolic Neural Networks

2.3 Hyperbolic Graph Neural Networks

Part 3. APPLICATIONS (Menglin, 10:30 – 11:00)

3.1 HGNN for Recommendation Systems

3.2 HGNN for Knowledge Graph

3.3 HGNN for Bioinformatics

Part 4. ADVANCED TOPICS (Bo, 11:00 – 12:00)

4.1 Complex Structures

4.2 Curvature-aware Learning

4.3 Evolving Interactions

4.4 Trustworthy and Scalability

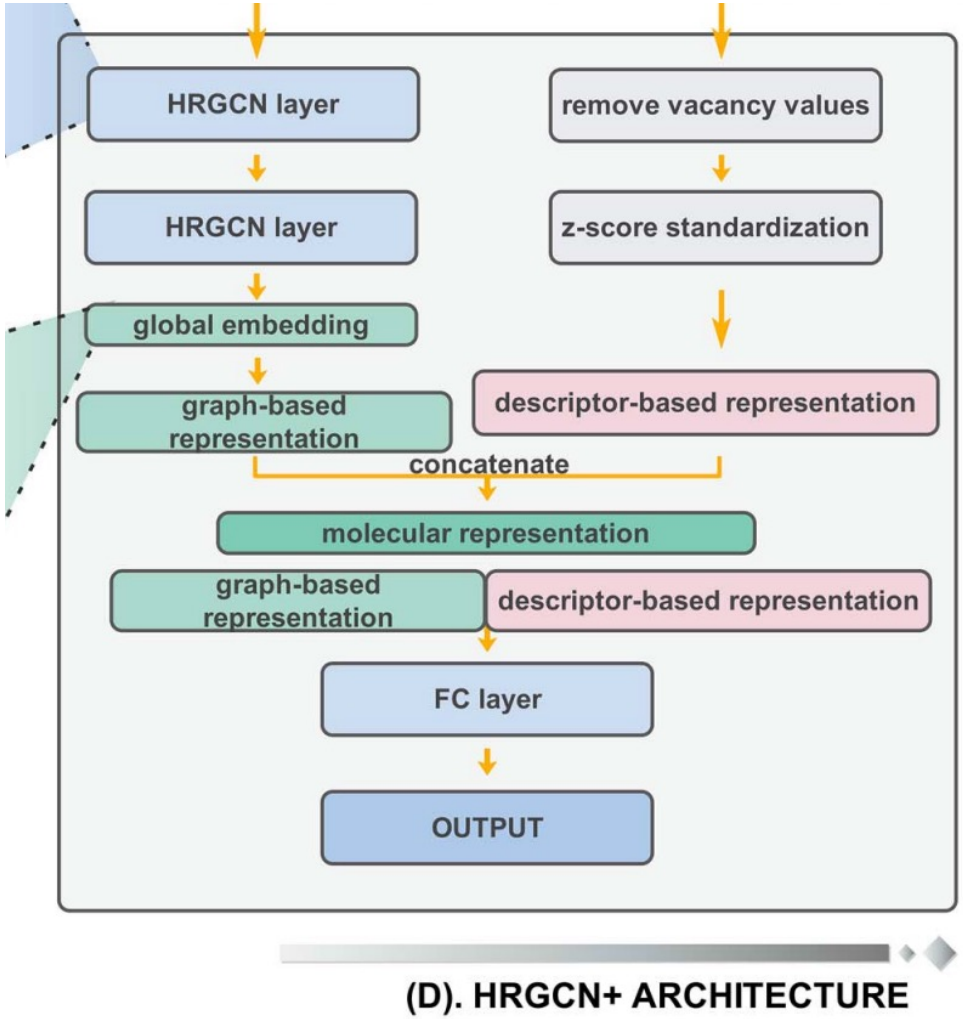
HRGCN (BIB '21)

Input Representation

- Molecular graphs with atom and bond features (e.g. atom type, bond type)
- Additional vector of molecular descriptors (192 in this paper)

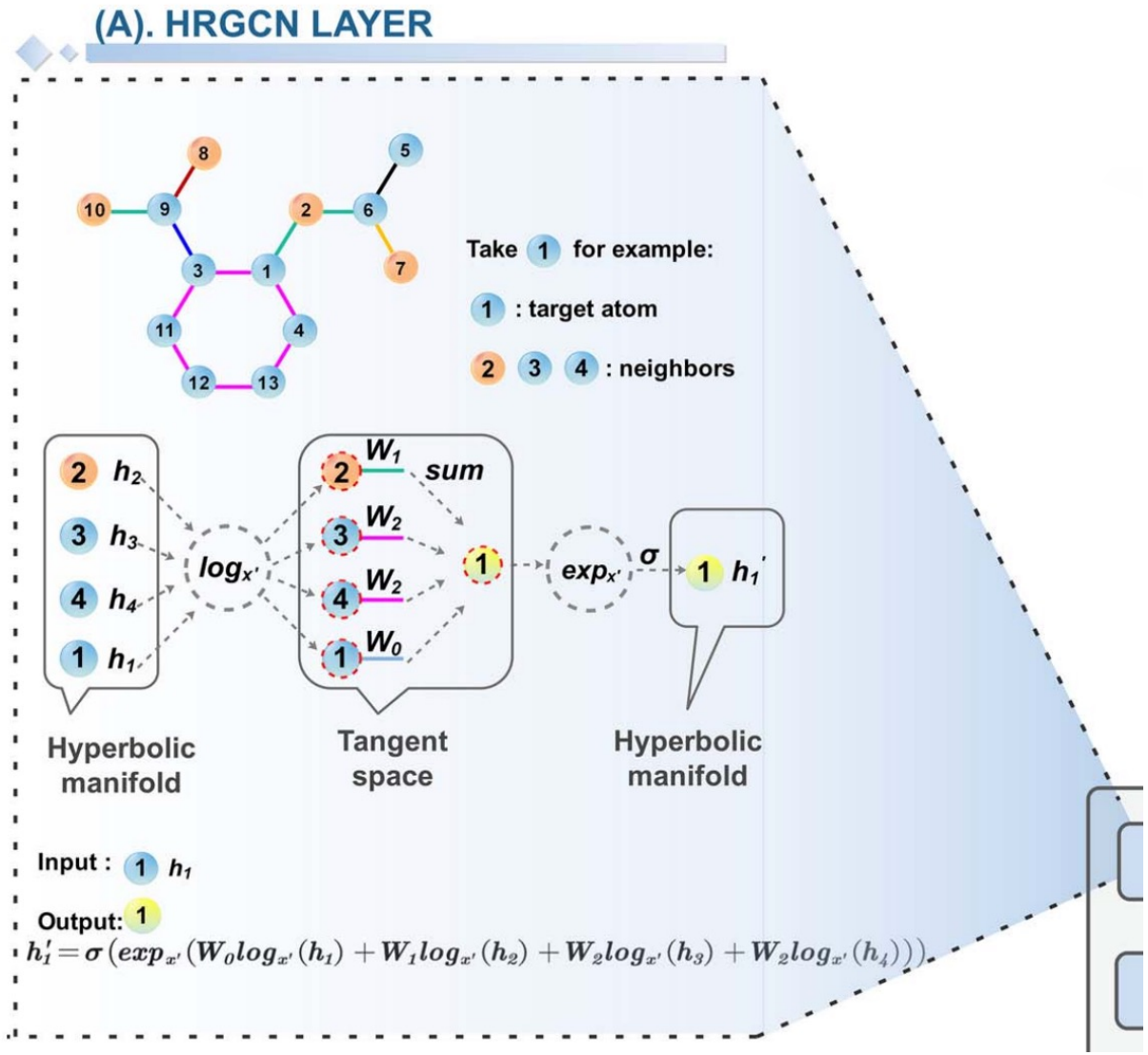
Model Architecture

- 2 layers of Hyperbolic Relational Graph Convolution (HRGC) to encode graph structure
- Global pooling (sum and max pooling) to summarize graph embedding
- Descriptor values passed through a dense layer
- Graph embedding and descriptor embedding concatenated
- Final prediction layer



Wu, Zhenxing, et al. "Hyperbolic relational graph convolution networks plus: a simple but highly efficient QSAR-modeling method." *Briefings in Bioinformatics* 22.5 (2021): bbab112.

HRGCN (BIB '21)



- It is based on relational graph convolution (RGC), which incorporates edge/relation features in graph convolutions.

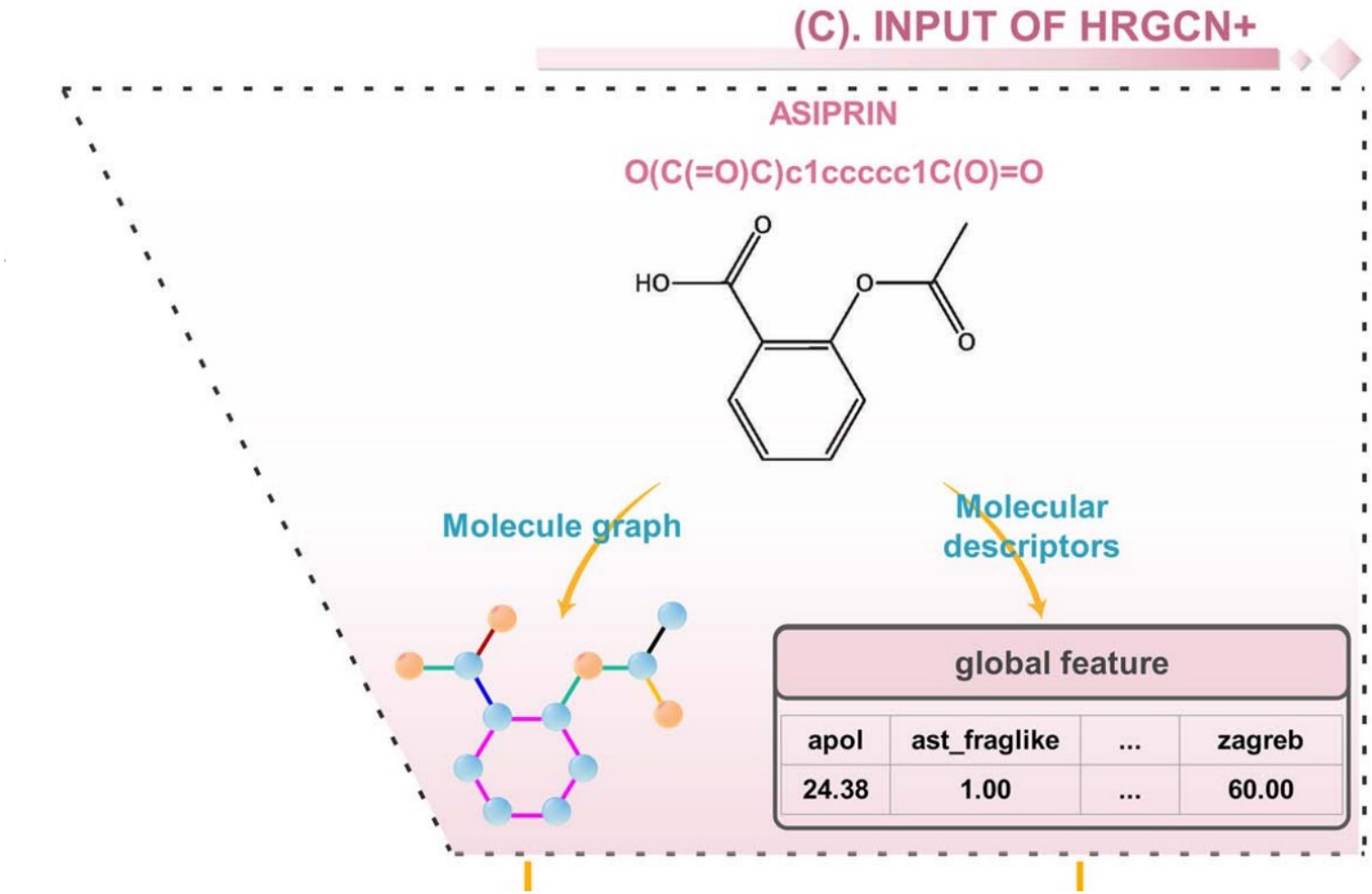
- RGC is extended to hyperbolic space, which is more suitable for tree-structured molecular graphs.

- Each layer propagates node features using messages passed along edges. Messages are transformed using relation-specific weight matrices.

- Messages are mapped from hyperbolic space to tangent space, transformed linearly, and mapped back.

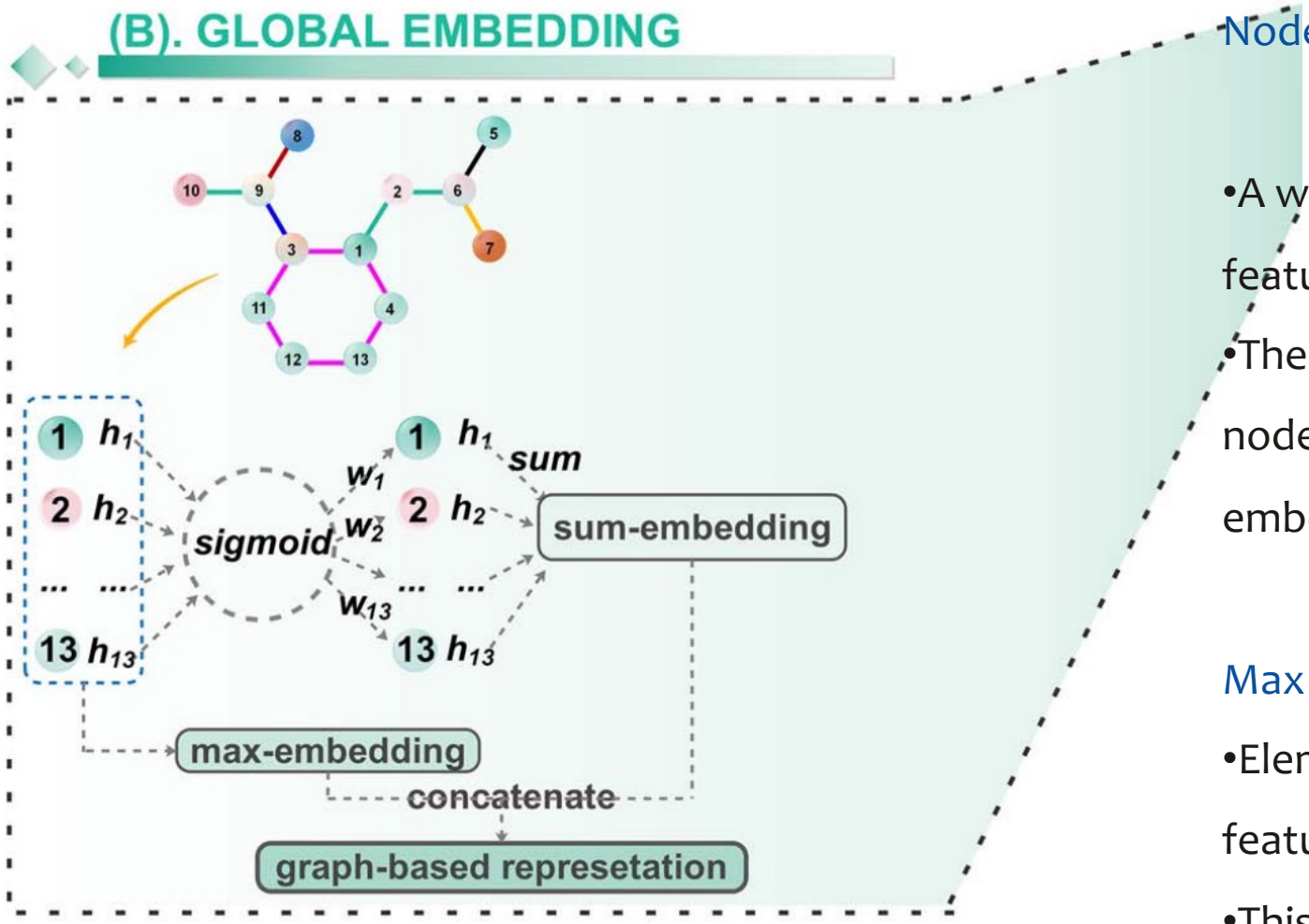
Wu, Zhenxing, et al. "Hyperbolic relational graph convolution networks plus: a simple but highly efficient QSAR-modeling method." *Briefings in Bioinformatics* 22.5 (2021): bbab112.

HRGCN (BIB '21)



Wu, Zhenxing, et al. "Hyperbolic relational graph convolution networks plus: a simple but highly efficient QSAR-modeling method." *Briefings in Bioinformatics* 22.5 (2021): bbab112.

HRGCN (BIB '21)



Node Weighted Summation

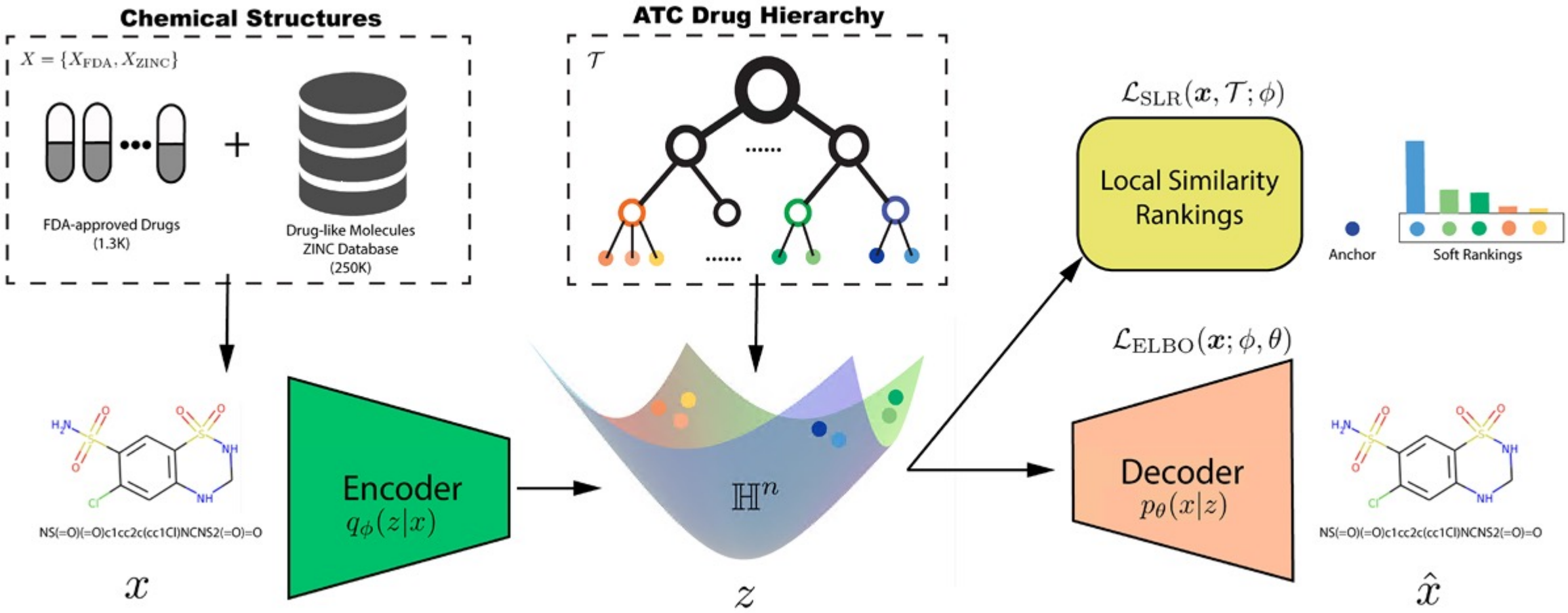
- A weight is calculated for each node by passing its feature vector through a dense layer and sigmoid.
- The node features are summed, weighted by these node weights, to obtain a summed graph embedding.

Max Pooling

- Element-wise maximum is taken over all node feature vectors.
- This gives a max pooled graph embedding.

Wu, Zhenxing, et al. "Hyperbolic relational graph convolution networks plus: a simple but highly efficient QSAR-modeling method." *Briefings in Bioinformatics* 22.5 (2021): bbab112.

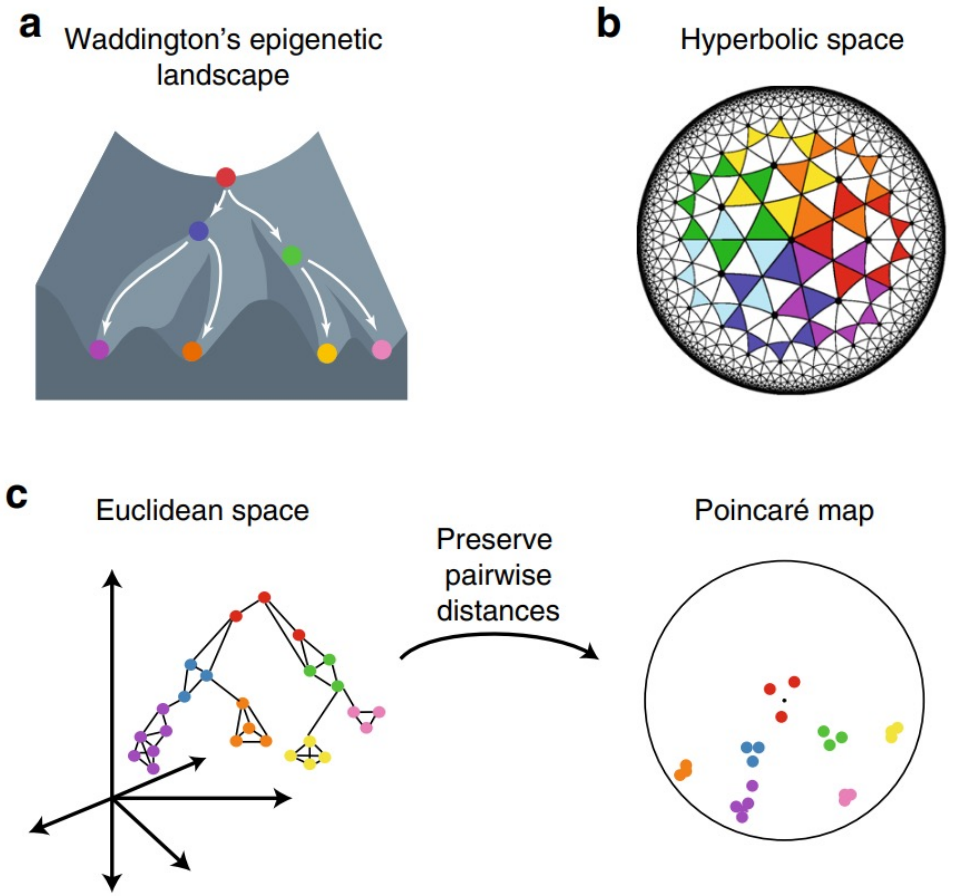
Drug Embedding (JCIM '20)

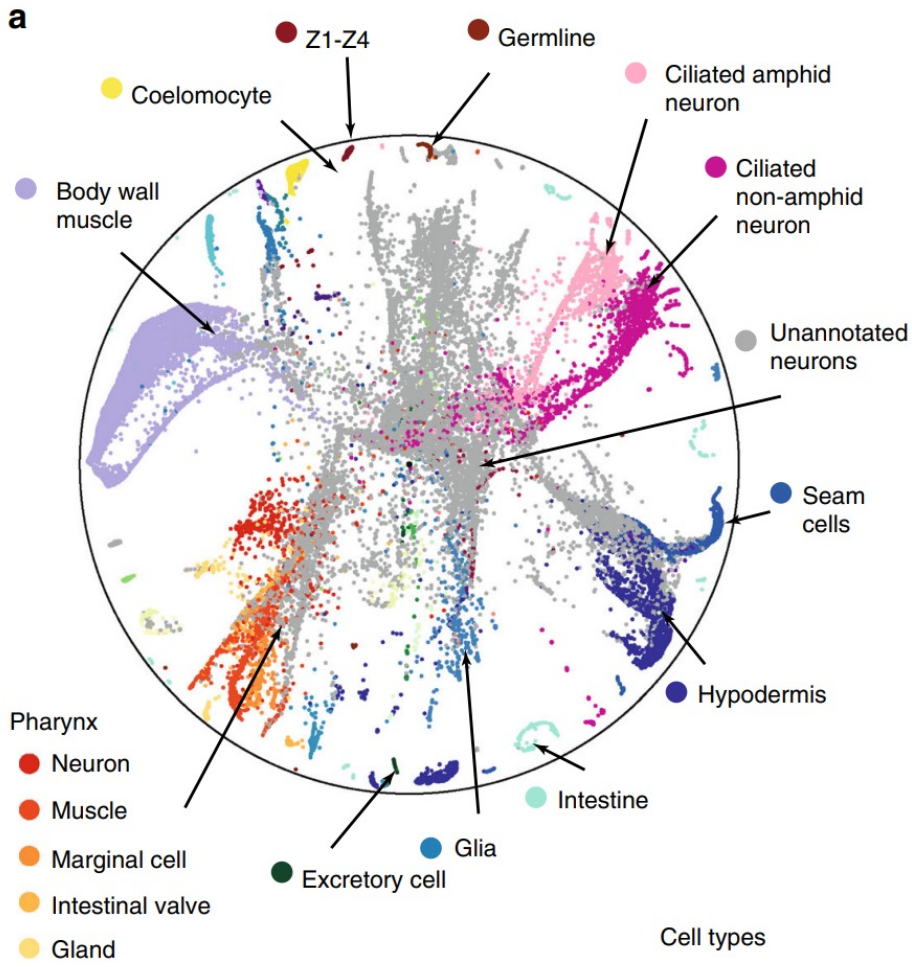


Hyperbolic spaces help inject hierarchies knowledge for drug generation.

Yu, Ke, Shyam Visweswaran, and Kayhan Batmanghelich. "Semi-supervised hierarchical drug embedding in hyperbolic space." Journal of chemical information and modeling 60.12 (2020): 5647-5657.

Cell Analyzing (NC '20)





Hyperbolic maps discover hierarchies and branching processes.

Klimovskaia, Anna, et al. "Poincaré maps for analyzing complex hierarchies in single-cell data." Nature communications 11.1 (2020): 1-9.

Hyperbolic Image Embedding

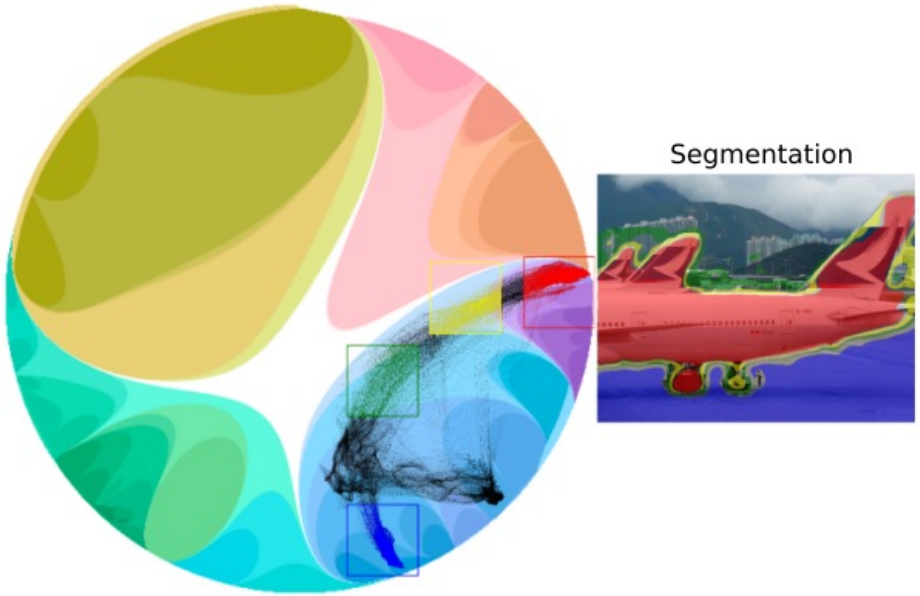


Image segmentation

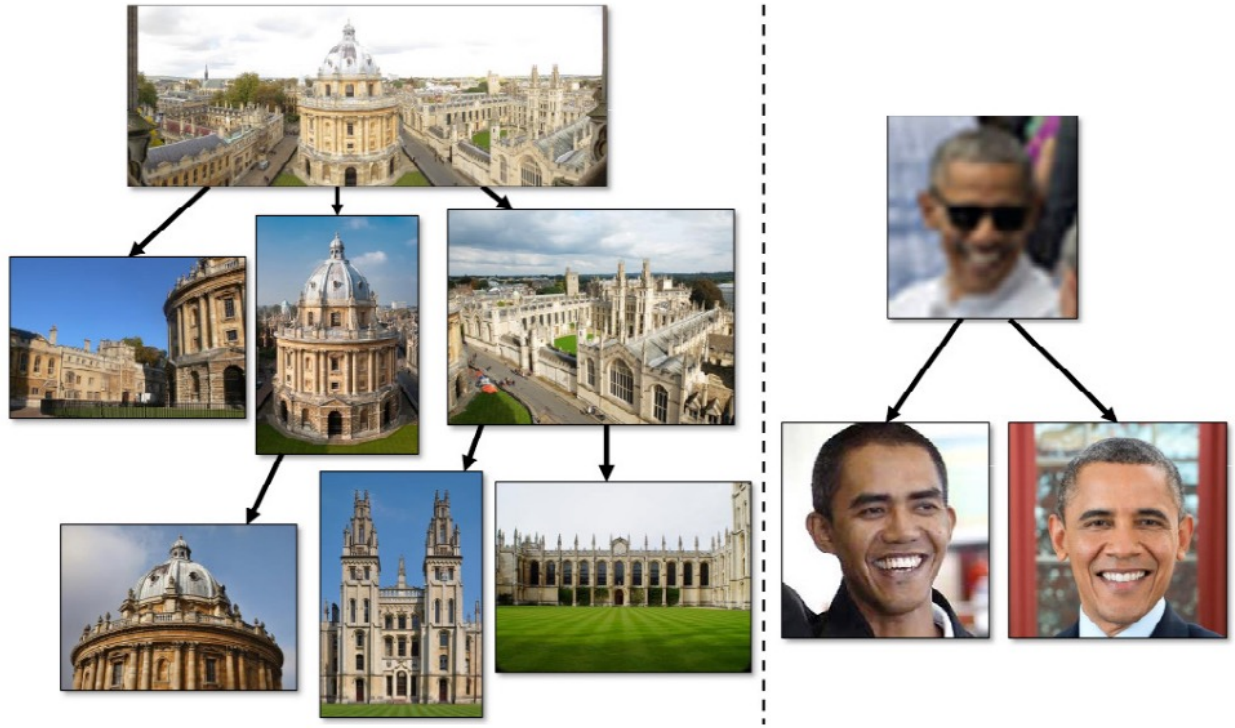
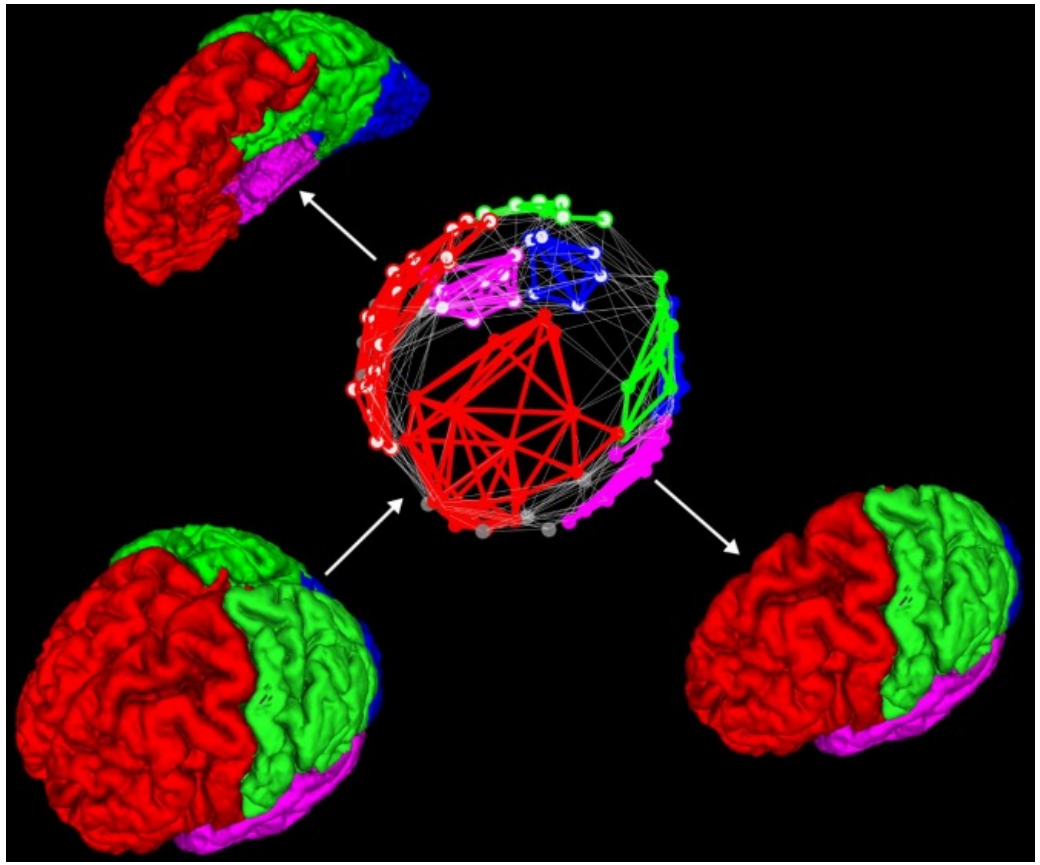


Image classification

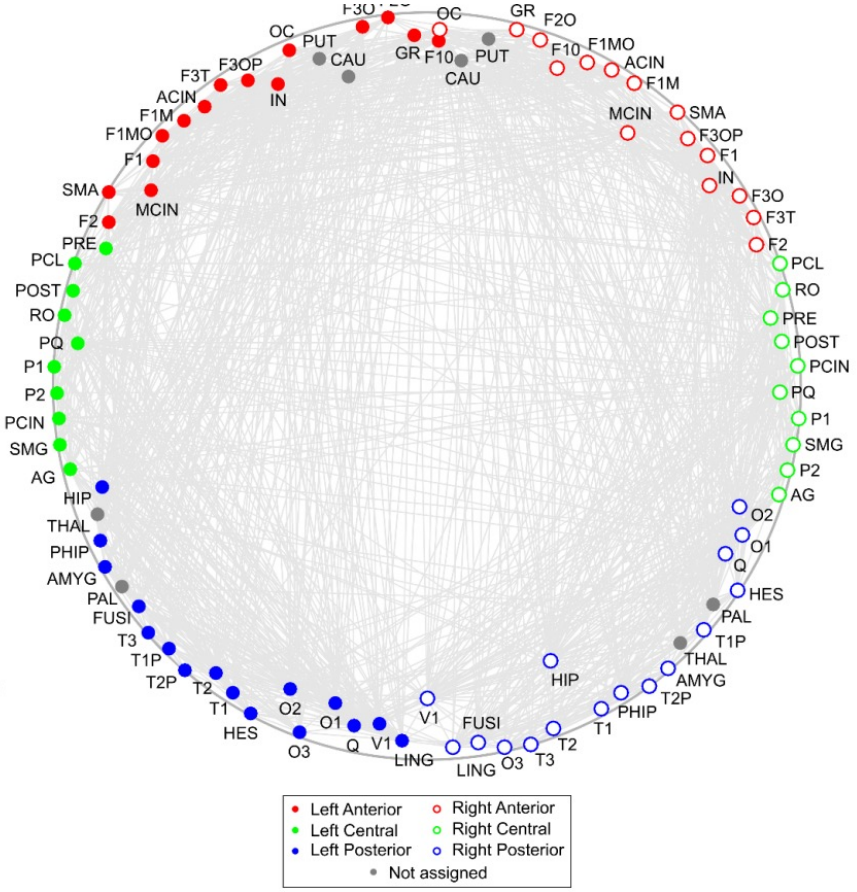
Hyperbolic space provides spacious room and additionally present uncertainty information.

Left: GhadimiAtigh, Mina, et al. "Hyperbolic Image Segmentation." CVPR (2022). Right: Khrulkov, Valentin, et al. "Hyperbolic image embeddings." CVPR (2020).

Hyperbolic Space for Brain Structure



PRE	Precentral
F1	Frontal-Sup
F10	Frontal-Sup-Orb
F2	Frontal-Mid
F2O	Frontal-Mid-Orb
F3OP	Frontal-Inf-Oper
F3T	Frontal-Inf-Tri
F3O	Frontal-Inf-Orb
RO	Rolandic-Oper
SMA	Supp-Motor-Area
OC	Olfactory
F1M	Frontal-Sup-Medial
F1MO	Frontal-Med-Orb
GR	Rectus
IN	Insula
ACIN	Cingulum-Ant
MCIN	Cingulum-Mid
PCIN	Cingulum-Post
HIP	Hippocampus
PHIP	ParaHippocampal
AMYG	Amygdala
V1	Calcarine
Q	Cuneus
LING	Lingual
O1	Occipital-Sup
O2	Occipital-Mid
O3	Occipital-Inf
FUSI	Fusiform
POST	Postcentral
P1	Parietal-Sup
P2	Parietal-Inf
SMG	SupraMarginal
AG	Angular
PQ	Precuneus
PCL	Paracentralobule
CAU	Caudate
PUT	Putamen
PAL	Pallidum
THAL	Thalamus
HES	Heschl
T1	Temporal-Sup
T1P	Temporal-Pole-Sup
T2	Temporal-Mid
T2P	Temporal-Pole-Mid
T3	Temporal-Inf



Hyperbolic space provides separatable room for different sub-structures.

Cacciola, Alberto, et al. "Coalescent embedding in the hyperbolic space unsupervisedly discloses the hidden geometry of the brain." BOOK OF ABSTRACTS. 2017.

Q&A

- Join our slack channel for Q&A

https://hyperbolicgnn.github.io/

4. Advanced Topics

Agenda

Part 1. INTRODUCTION (Min, 9:00 - 9:30)

1.1 An Overview of Graph Learning

1.2 Brief Introduction of Riemannian Geometry

1.3 Motivation for Hyperbolic Graph Learning

Coffee Break (9:30 – 10:00)

Part 2. HYPERBOLIC GRAPH NETWORKS (Menglin, 10:00 – 10:30)

2.1 Preliminary

2.2 Hyperbolic Shallow Models

2.2 Hyperbolic Neural Networks

2.3 Hyperbolic Graph Neural Networks

Part 3. APPLICATIONS (Menglin, 10:30 – 11:00)

3.1 HGNN for Recommendation Systems

3.2 HGNN for Knowledge Graph

3.3 HGNN for Bioinformatics

Part 4. ADVANCED TOPICS (Bo, 11:00 – 12:00)

4.1 Complex Structures

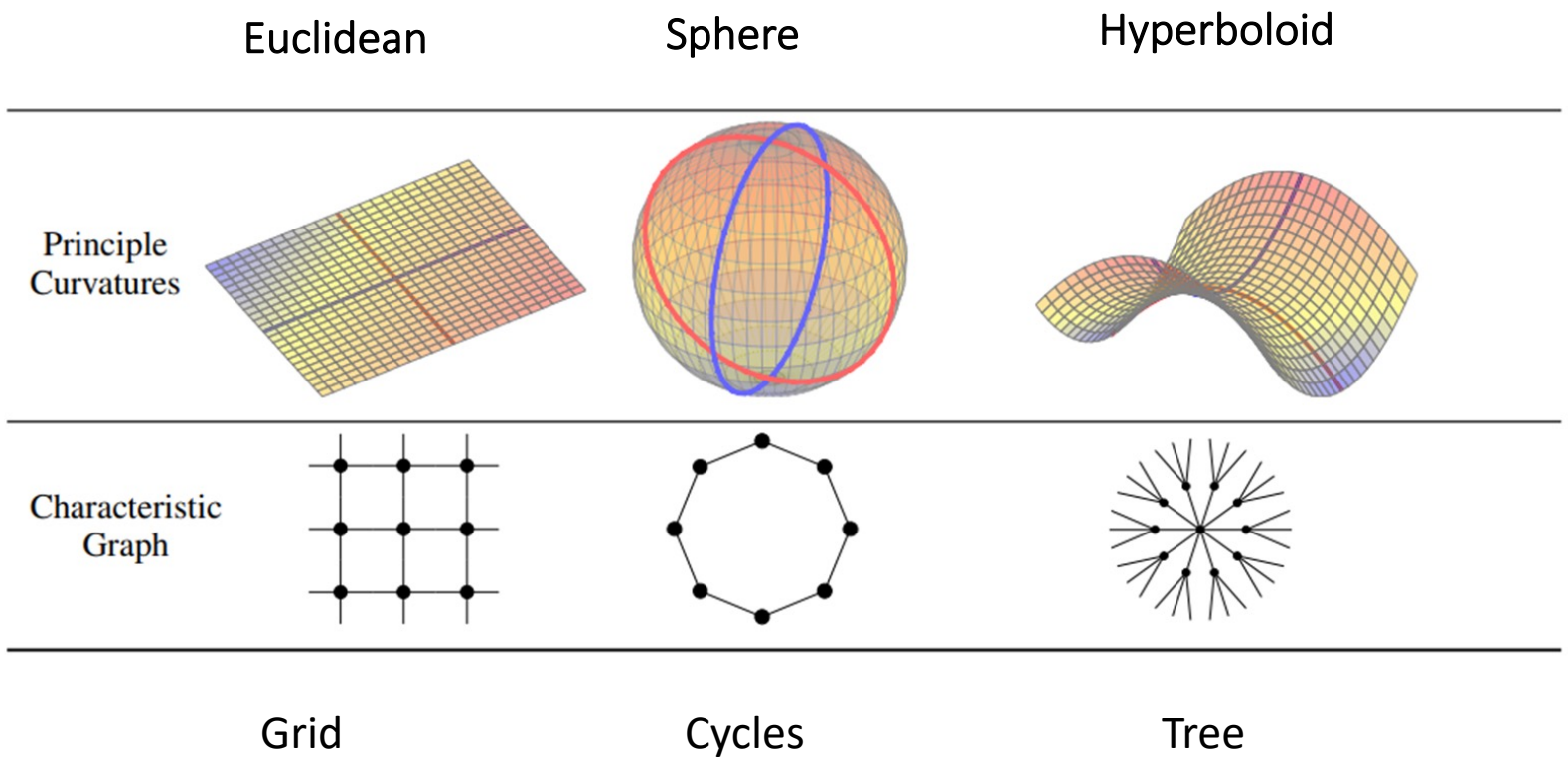
4.2 Curvature-aware Learning

4.3 Evolving Interactions

4.4 Trustworthy and Scalability

4.1 Complex Structures

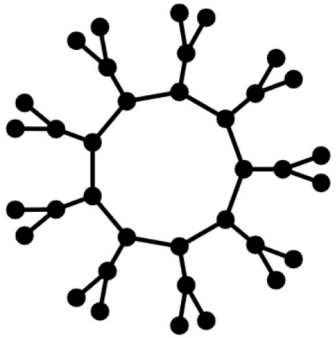
- Riemannian manifolds & graph structures



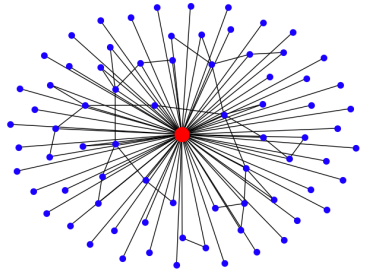
Weber, et al. "Curvature and representation learning: Identifying embedding spaces for relational data." *Neurips* (2018).

4.1 Complex Structures

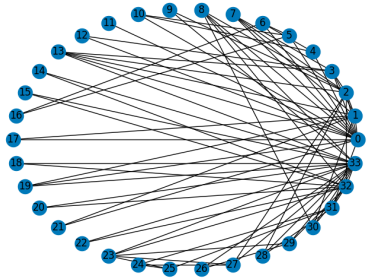
- Real-world graphs contain mixtures of cycles and trees



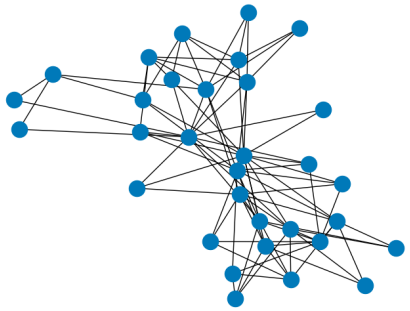
Cycle-Tree



Barabási-Albert network



Zachary's karate club



Knowledge Graphs

- How to measure the topological properties?
- Which embedding space to use for graphs with complex topologies?

Weber, et al. "Curvature and representation learning: Identifying embedding spaces for relational data." *Neurips* (2018).

Discrete Graph Curvature

- Continuous curvature: how much the surface deviates from being a plane

- Discrete curvature: how much the topology of a pair of neighborhoods deviates from a "flat" case (i.e., grid graph)
  - Gromov's $\delta-$ hyperbolicity
  - Graph Ricci curvature
  - Neighborhood growth rates

Source: Comparative analysis of two discretizations of Ricci curvature for complex networks

Gromov's δ – hyperbolicity

- Measure how tree-like a graph is

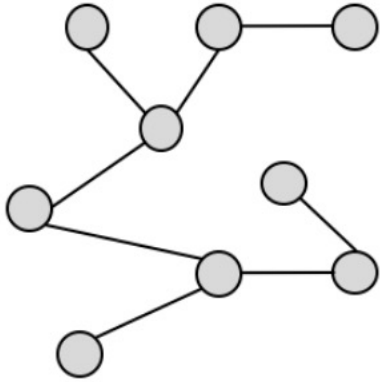
$$\delta_{worst}(G) = \max_{x,y,u,v \in V} \frac{S_1 - S_2}{2}$$

where $S_i = \{d(x,y) + d(u,v), d(x,u) + d(y,v), d(x,v) + d(y,u)\}$ and S_i is the ith largest element.

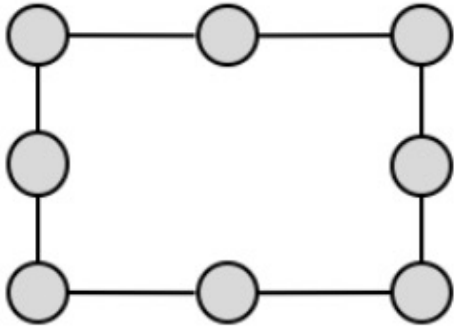
- The lower δ, the more hyperbolic is the graph dataset, and δ = 0 for trees

Gromov's δ – hyperbolicity

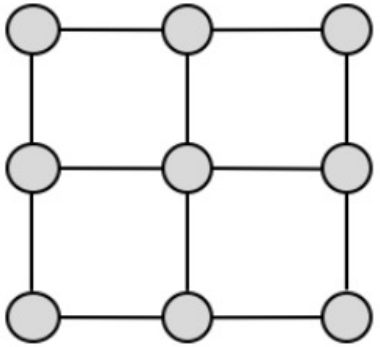
- Some example results



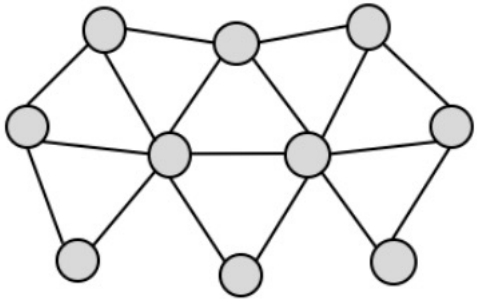
Trees (δ=0)



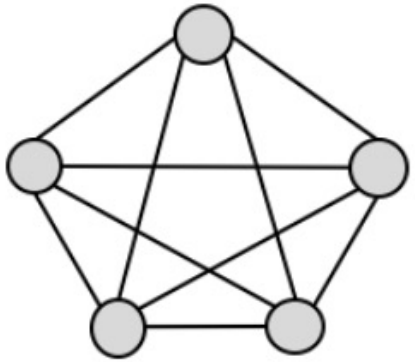
N-cycle (δ=N)



N×N grid (δ=N)



Chordal (δ=1)



Clique (δ=0)

Gromov's δ – hyperbolicity

- The time complexity is $O(n^4)$

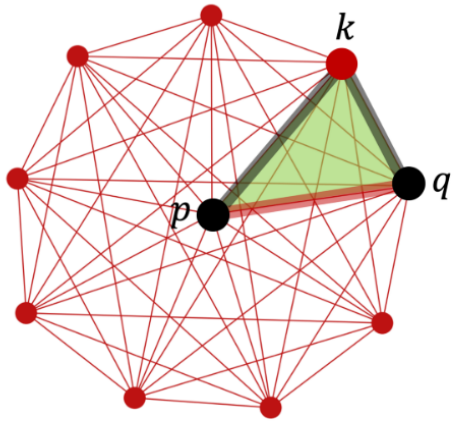
$$\delta_{worst}(G) = \max_{x,y,u,v \in V} \frac{S_1 - S_2}{2}$$

Enumerating all 4-point node sets!

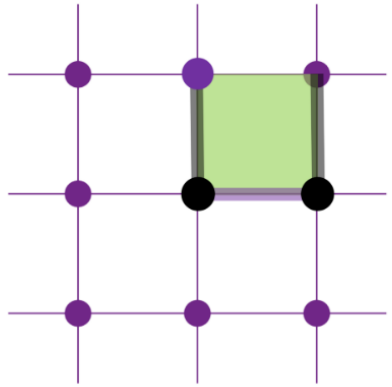
- Faster algorithm with complexity $O(n^{3.69})$ (H. Fournier et al. 2021)
  - SageMath: an implementation of exact hyperbolicity
    https://doc.sagemath.org/html/en/reference/graphs/sage/graphs/hyperbolicity.html
  - Sampling-based hyperbolicity

Graph Ricci Curvature

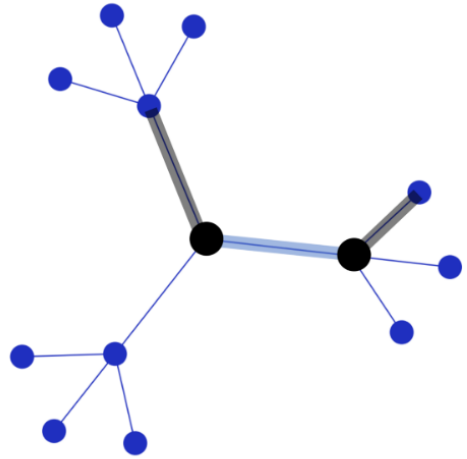
- Curvature on network: measures the changes of local connectivity



Clique (>0)



Grid (=0)



Tree (<0)

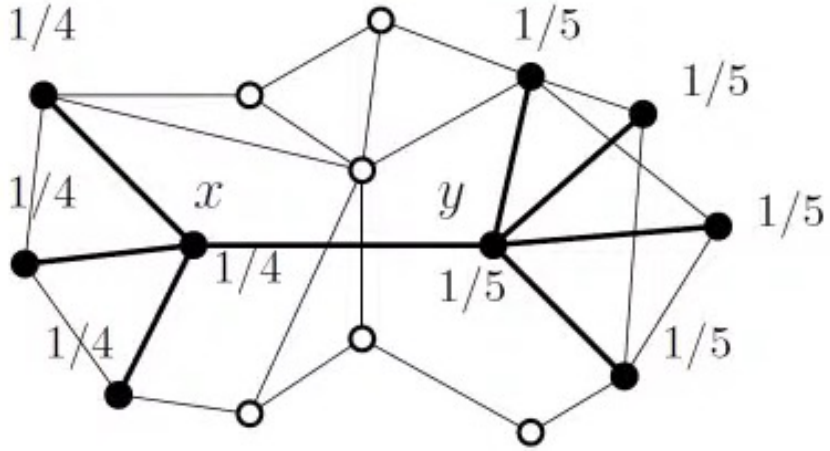
- Positively curved edge represents an edge within a cluster
- Negatively curved edge tent to be a bridge within clusters

Gosztolai., A. et. al. Unfolding the multiscale structure of networks with dynamical Ollivier-Ricci curvature. Nature Communications. 2021

Graph Ollivier-Ricci Curvature

- For an edge xy, consider the distance from x's neighbors to y's neighbors and compare it with the length of xy
- How to compute the distance of x's neighbors to y's neighbors?
  - **Optimal transport distance (Wasserstein distance)**

$$\kappa_{xy} = 1 - \frac{W(m_x, m_y)}{d(x, y)},$$



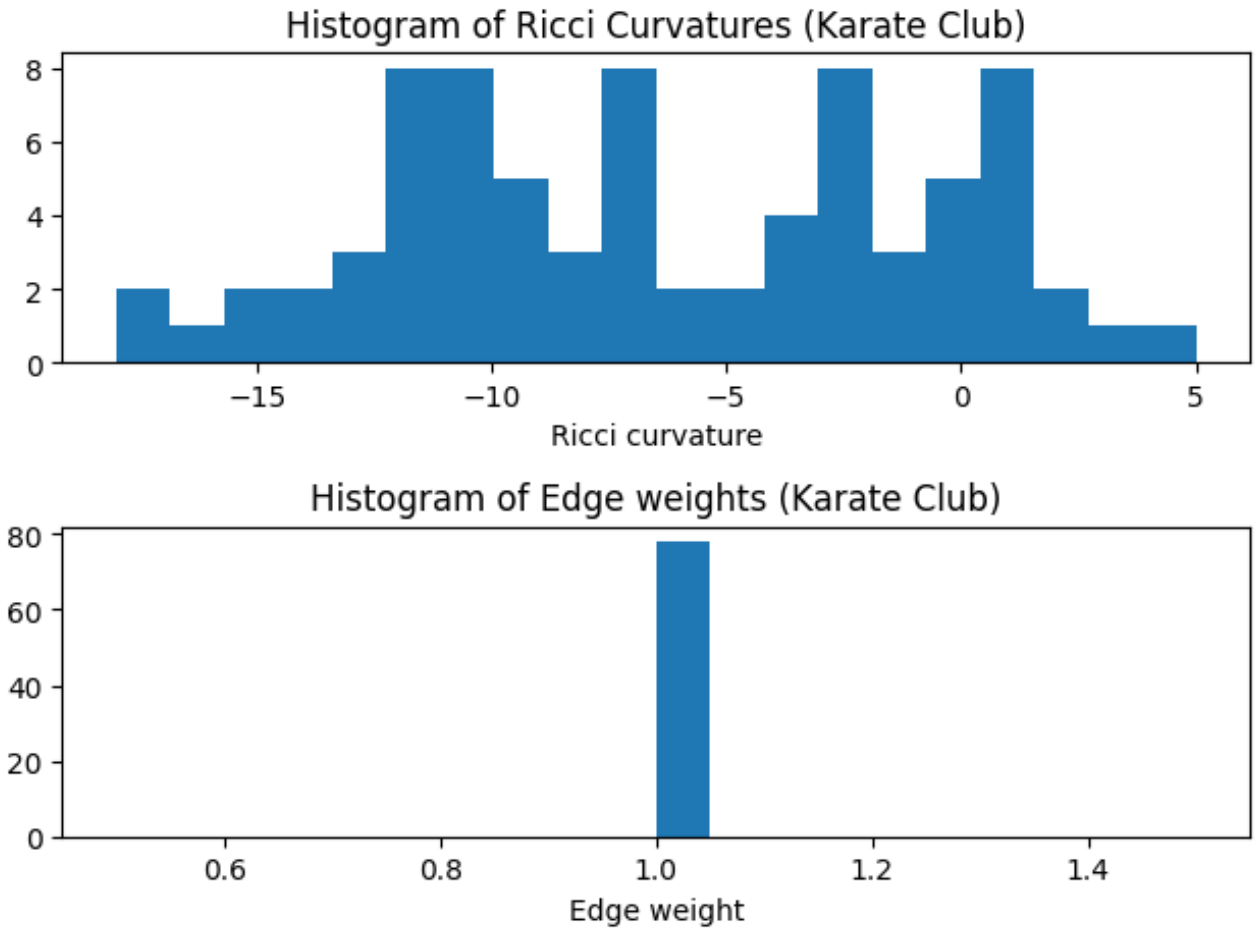
Graph Forman-Ricci Curvature

- Quantifies the degree of spread of the vertices
- Consider **triangles** as faces, and Δ_{uv} is the number of triangles that contain u, v.

$$F(u,v) = 4 - \deg(u) - \deg(v) + 3\Delta_{uv}$$

Graph Ricci Curvature

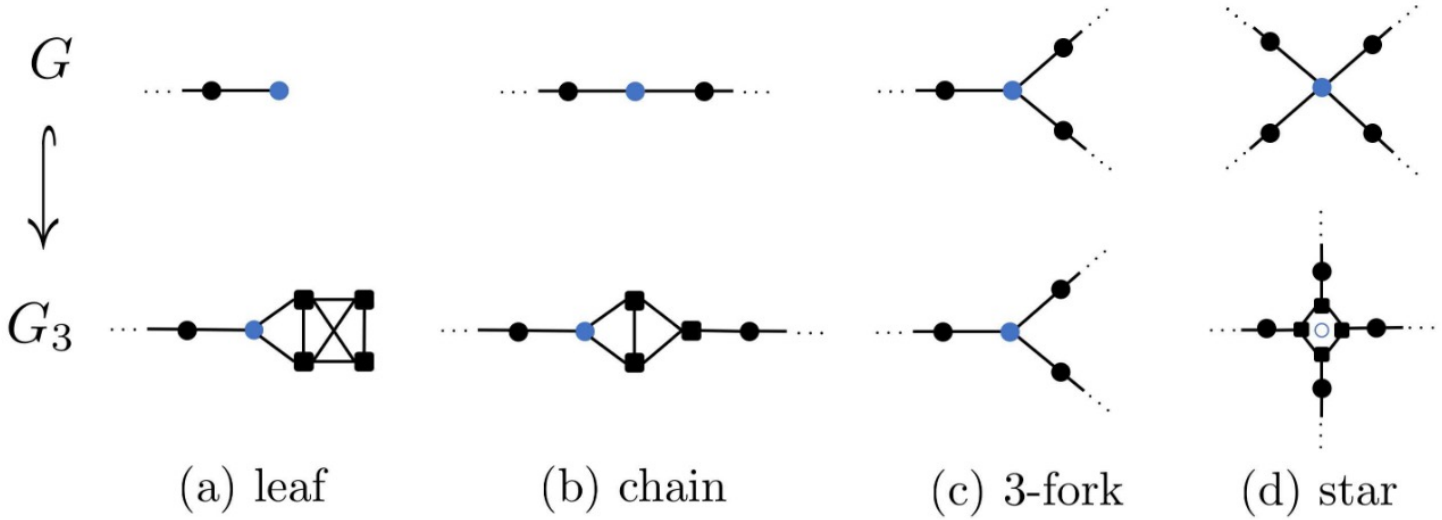
```python
import networkx as nx
from GraphRicciCurvature.OllivierRicci import OllivierRicci
from GraphRicciCurvature.FormanRicci import FormanRicci

print("\n- Import an example NetworkX karate club graph")
G = nx.karate_club_graph()

print("\n===== Compute the Ollivier-Ricci curvature of the giv
# compute the Ollivier-Ricci curvature of the given graph G
orc = OllivierRicci(G, alpha=0.5, verbose="INFO")
orc.compute_ricci_curvature()
print("Karate Club Graph: The Ollivier-Ricci curvature of edge
```



https://github.com/saibalmars/GraphRicciCurvature

Neighborhood growth rates (3-regular score)

- Exponentially expanding neighborhoods indicates tree-like properties
- Linear neighborhood growth rate indicates grid
- Sublinear neighborhood growth rate indicates cycles



Melanie Weber. Neighborhood Growth Determines Geometric Priors for Relational Representation Learning. AISTATS 2020.

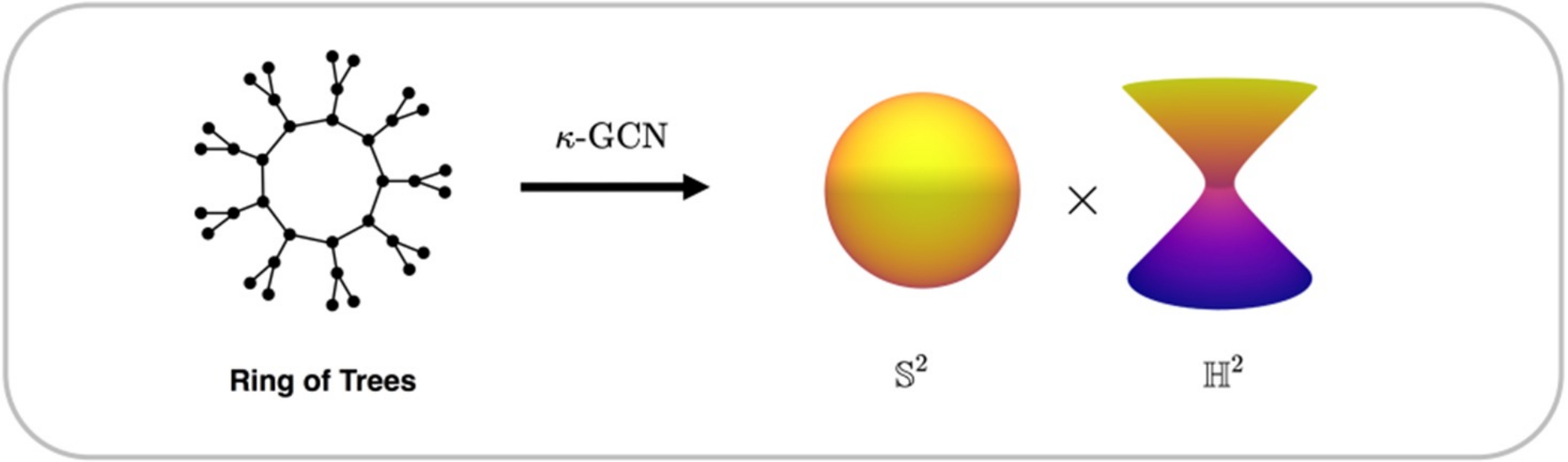
How to embed graphs with complex structures?

- **Interaction learning of mixture spaces**
  - Constant Curvature Graph Convolutional Networks. ICML 2020
  - Graph Geometry Interaction learning. NeurIPS 2020
  - A self-supervised mixed-curvature graph neural network. AAAI2022
  - …

- **Pseudo-Riemannian Embeddings**
  - Ultrahyperbolic representation learning. NeurIPS 2020.
  - Directed Graph Embeddings in Pseudo–Riemannian Manifolds. ICML 2021.
  - Pseudo-Riemannian Graph Convolutional Networks. NeurIPS 2022.
  - …

Constant Curvature Graph Convolution Networks

- Embed graphs into a product space of hyperbolic and spherical spaces

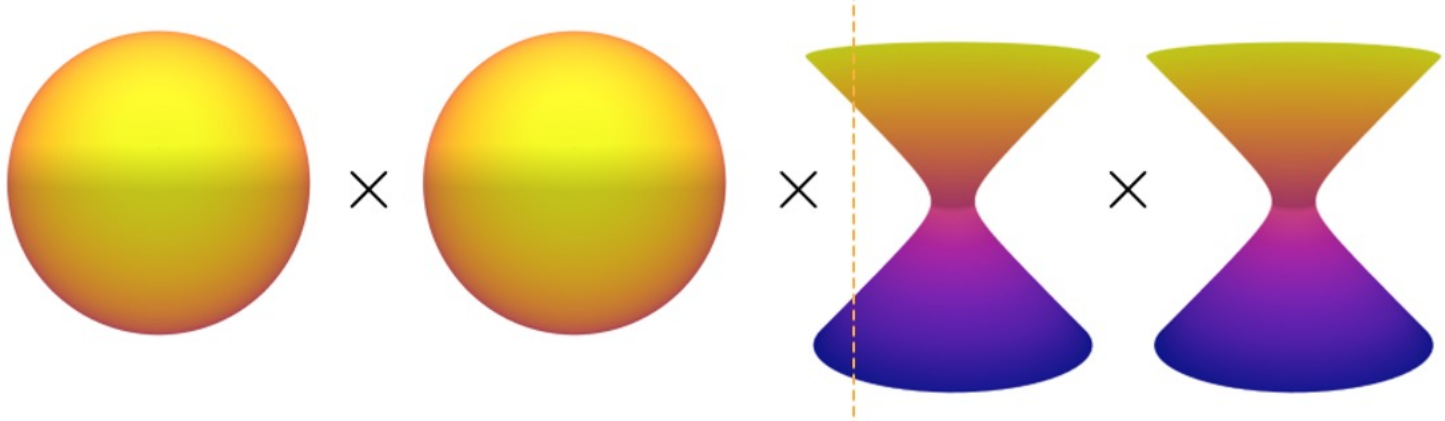


Bachmann , G. et.al. Constant Curvature Graph Convolution Networks. ICML 2020

Product Manifolds

- Product manifold is defined as the Cartesian product of many component manifolds $M = M_1 \times M_2 \times \ldots \times M_k$.

$$\mathcal{P} = \mathbb{E}^{d_{\mathbb{E}}}_{K_{\mathbb{E}}} \times \left(\bigtimes_{j=1}^{n_{\mathbb{H}}} \mathbb{H}^{d_j^{\mathbb{H}}}_{K_j^{\mathbb{H}}} \right) \times \left(\bigtimes_{k=1}^{n_{\mathbb{S}}} \mathbb{S}^{d_k^{\mathbb{S}}}_{K_k^{\mathbb{S}}} \right) = \mathbb{E} \times \left(\bigtimes_{j=1}^{n_{\mathbb{H}}} \mathbb{H}_j \right) \times \left(\bigtimes_{k=1}^{n_{\mathbb{S}}} \mathbb{S}_k \right),$$



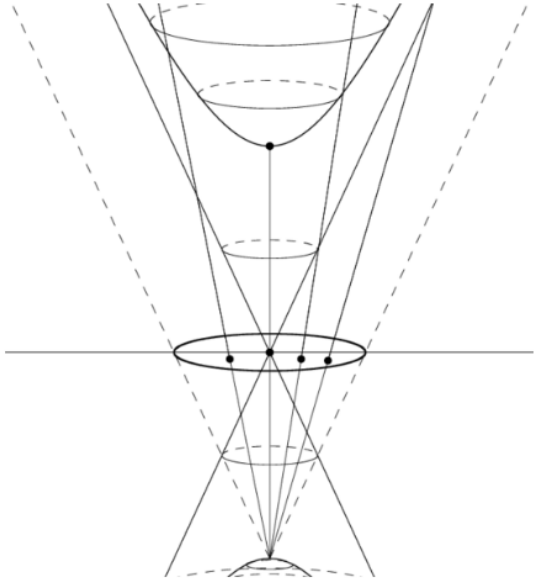
- The geodesics, distance, exponential, and logarithmic maps on P are the concatenation of the corresponding notions of the individual model spaces

$$\mathrm{Exp}_p(v) = (\mathrm{Exp}_{p_1}(v_1), \ldots, \mathrm{Exp}_{p_k}(v_k)), \qquad d^2_{\mathcal{P}}(x, y) = \sum_{i=1}^{k} d_i^2(x_i, y_i).$$

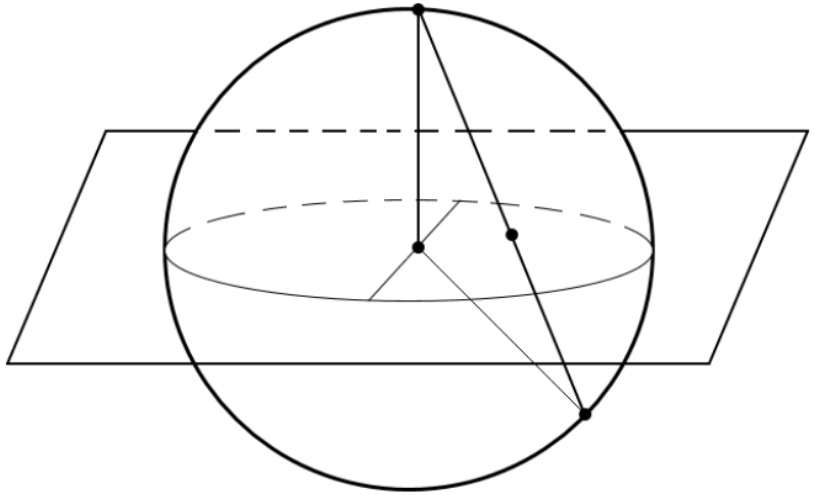
κ – sterographic model

- Unified formalism for any constant curvature manifolds $\kappa \in \mathbb{R}$

Hyperboloid & Poincaré Ball



Sphere & Stereographic Projection of Sphere



Bachmann , G. et.al. Constant Curvature Graph Convolution Networks. ICML 2020

κ – sterogrpah model

- Unified formalism (κ – sterogrpah model) for any constant curvature manifolds $\kappa \in \mathbb{R}$

$$\mathfrak{st}_\kappa^d = \{\boldsymbol{x} \in \mathbb{R}^d \mid -\kappa\|\boldsymbol{x}\|_2^2 < 1\}$$

	$\mathbb{R}^d$	$\mathfrak{st}_\kappa^d$
$\boldsymbol{x} \oplus_\kappa \boldsymbol{y}$	$\boldsymbol{x} + \boldsymbol{y}$	$\frac{(1-2\kappa \boldsymbol{x}^T \boldsymbol{y} - \kappa\|\|\boldsymbol{y}\|\|^2)\boldsymbol{x} + (1+\kappa\|\|\boldsymbol{x}\|\|^2)\boldsymbol{y}}{1-2\kappa \boldsymbol{x}^T \boldsymbol{y} + \kappa^2\|\|\boldsymbol{x}\|\|^2\|\|\boldsymbol{y}\|\|^2}$
$r \otimes_\kappa \boldsymbol{x}$	$r\boldsymbol{x}$	$\tan_\kappa\left(r \cdot \tan_\kappa^{-1}\|\|\boldsymbol{x}\|\|\right)\frac{\boldsymbol{x}}{\|\|\boldsymbol{x}\|\|}$
$\gamma_{\boldsymbol{x}\to\boldsymbol{y}}(t)$	$\boldsymbol{x} + t(\boldsymbol{y} - \boldsymbol{x})$	$\boldsymbol{x} \oplus_\kappa \left(t \otimes_\kappa \left(-\boldsymbol{x} \oplus_\kappa \boldsymbol{y}\right)\right)$

Bachmann , G. et.al. Constant Curvature Graph Convolution Networks. ICML 2020

Constant Curvature Graph Convolution Networks

- Extend the vanilla GCN to the constant curvature GCN

$$\boldsymbol{H}^{(t+1)} = \sigma\left(\hat{\boldsymbol{A}}\boldsymbol{H}^{(t)}\boldsymbol{W}^{(t)}\right) \quad \Rightarrow \quad \boldsymbol{H}^{(l+1)} = \sigma^{\otimes_\kappa}\left(\hat{\boldsymbol{A}} \boxtimes_\kappa \left(\boldsymbol{H}^{(l)} \otimes_\kappa \boldsymbol{W}^{(l)}\right)\right)$$

where $\sigma^{\otimes_\kappa}$ is the κ-stereographic version of σ

- Learn the curvature to adapt to the geometry of the data

$$\kappa\text{-GCN} \xrightarrow{\kappa \to 0} \text{GCN}.$$

Bachmann , G. et.al. Constant Curvature Graph Convolution Networks. ICML 2020

Constant Curvature Graph Convolution Networks

- Experiments on distortion task
  - Minimize the discrepancy between embedding distances and graph distances

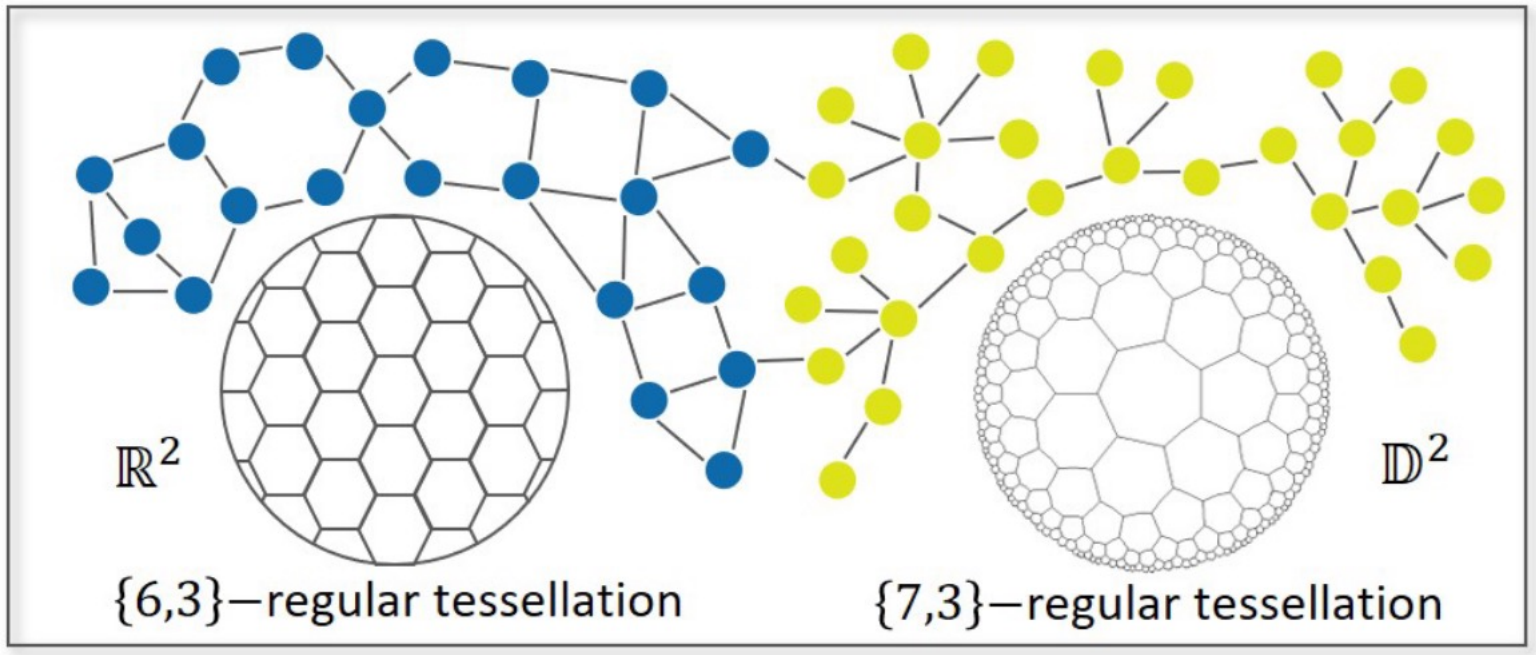
$$L(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n) = \frac{1}{n^2} \sum_{i,j} \left(\left(\frac{d_\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)}{d_{\boldsymbol{G}}(i,j)} \right)^2 - 1 \right)^2$$

  - Results on tree (negative curvature), spherical graph (positive curvature) and toroidal graph (product of positive curvature)
    - The best performing architecture is the one that matches the underlying geometry of the graph.

Model	Tree	Toroidal	Spherical
$\mathbb{E}^{10}$ (GCN)	0.0502	0.0603	0.0409
$\mathbb{H}^{10}$ (κ-GCN)	**0.0029**	0.272	0.267
$\mathbb{S}^{10}$ (κ-GCN)	0.473	0.0485	**0.0337**
$\mathbb{H}^5 \times \mathbb{H}^5$ (κ-GCN)	0.0048	0.112	0.152
$\mathbb{S}^5 \times \mathbb{S}^5$ (κ-GCN)	0.51	**0.0464**	0.0359

Bachmann , G. et.al. Constant Curvature Graph Convolution Networks. ICLM 2020

Graph Geometry Interaction Learning (GIL)



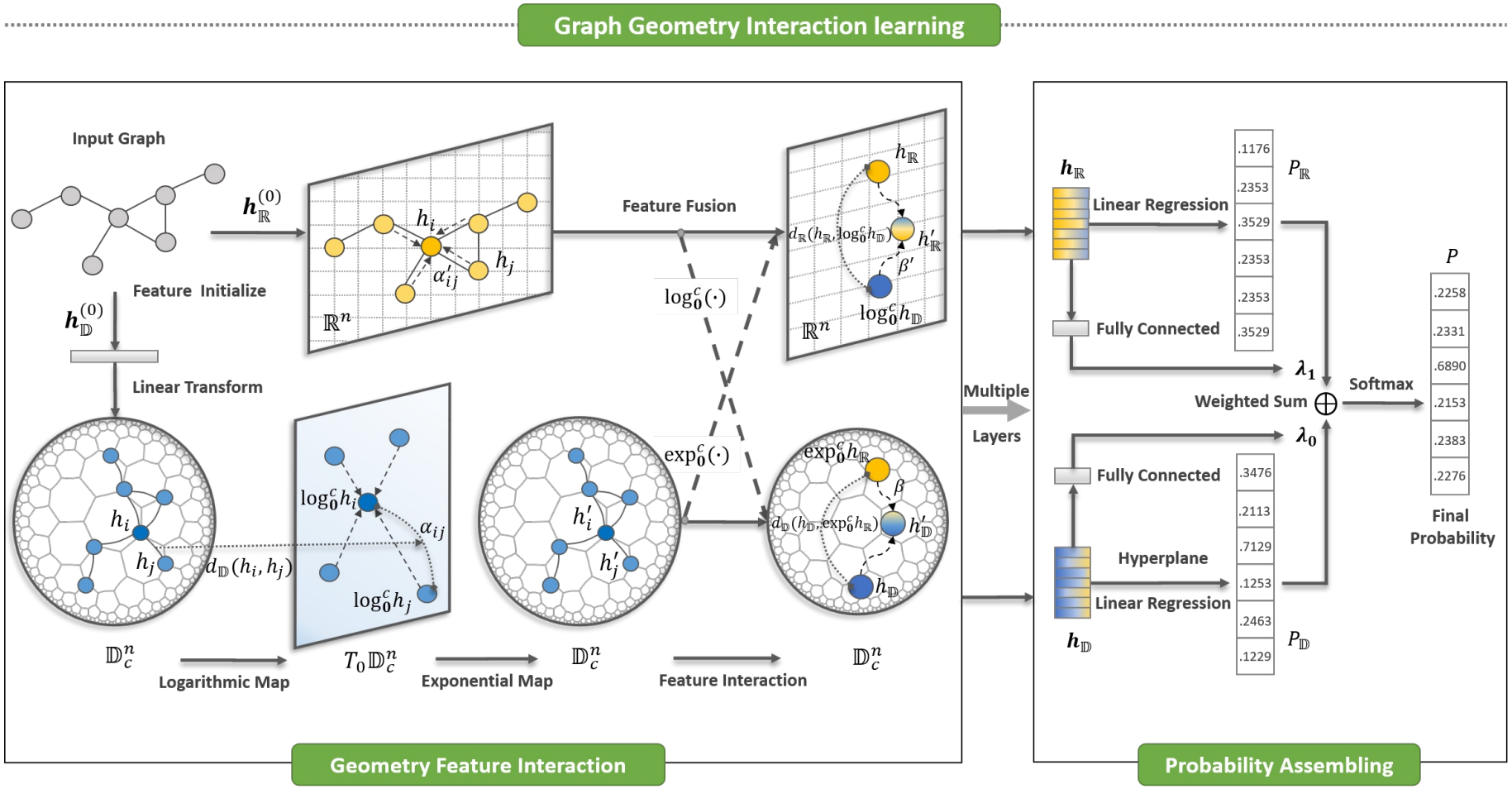
Example of a graph with both hierarchical and grid-like structures

Geometry Interaction Learning

- Euclidean space: grid data, powerful simplicity and efficiency
- Hyperbolic space: ability for hierarchical structure

Zhu S, et al. "Graph geometry interaction learning.“ NeuRIPS 2020

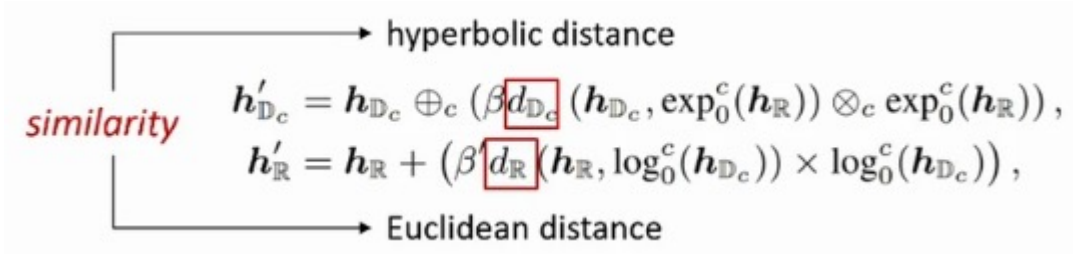
Graph Geometry Interaction Learning (GIL)

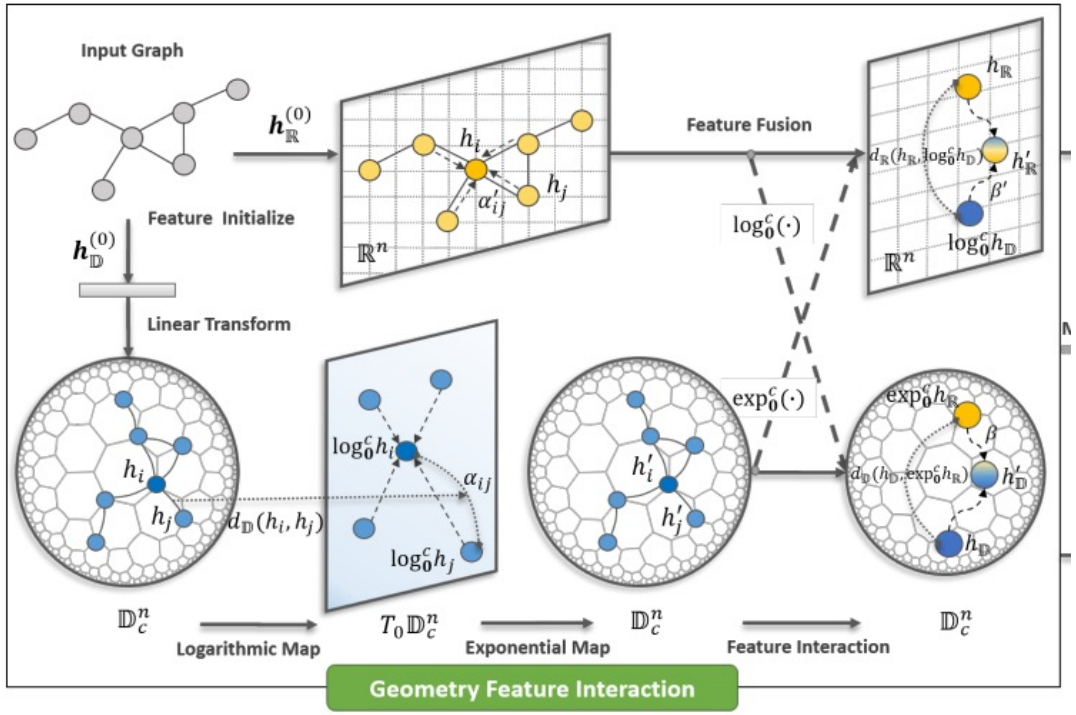


Zhu S, et al. "Graph geometry interaction learning.“ NeuRIPS 2020

Graph Geometry Interaction Learning(GIL)

- Geometry Message Propagation
  - Dual Feature propagation: propagates neighbor information simultaneously in a Euclidean space and a hyperbolic space via a distance-aware attention mechanism
  - Dual Feature Interaction: node features undergo an adaptive adjustment from the dual feature space based on distance similarity





Zhu S, et al. "Graph geometry interaction learning.“ NeuRIPS 2020

Graph Geometry Interaction Learning(GIL)

- Probability Assembling
  - One classifier

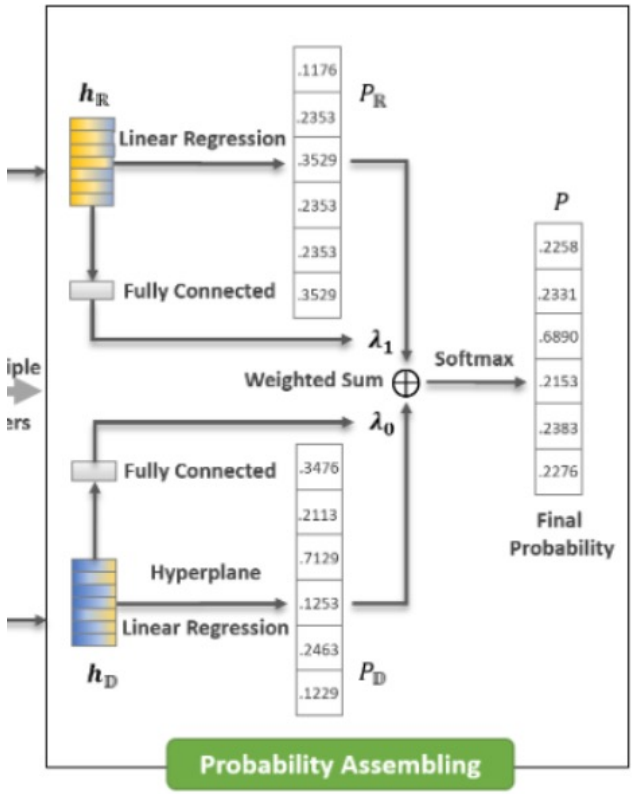
$$P = \text{Euc-Softmax}\left(f\left(\log_0^c(\boldsymbol{X}_{\mathbb{D}_c})||\boldsymbol{X}_{\mathbb{R}}\right)\right),$$
$$P = \text{Hyp-Softmax}\left(g\left(\boldsymbol{X}_{\mathbb{D}_c}||\exp_0^c(\boldsymbol{X}_{\mathbb{R}})\right)\right),$$

  - Two classifier

$$P = \boldsymbol{\lambda}_0 P_{\mathbb{D}_c} + \boldsymbol{\lambda}_1 P_{\mathbb{R}},$$
$$s.t.\ \boldsymbol{\lambda}_0 + \boldsymbol{\lambda}_1 = \mathbf{1},$$

Final probability of two parts are determined by the *node features*

$$\boldsymbol{\lambda}_0 = \text{sigmoid}\left(FC_0\left(\log_0^c(\boldsymbol{X}_{\mathbb{D}_c}^\top)\right)\right),\ \boldsymbol{\lambda}_1 = \text{sigmoid}\left(FC_1\left(\boldsymbol{X}_{\mathbb{R}}^\top\right)\right),$$
$$[\boldsymbol{\lambda}_0, \boldsymbol{\lambda}_1] = \frac{[\boldsymbol{\lambda}_0, \boldsymbol{\lambda}_1]}{\max\left(\|[\boldsymbol{\lambda}_0, \boldsymbol{\lambda}_1]\|_2, \epsilon\right)},$$

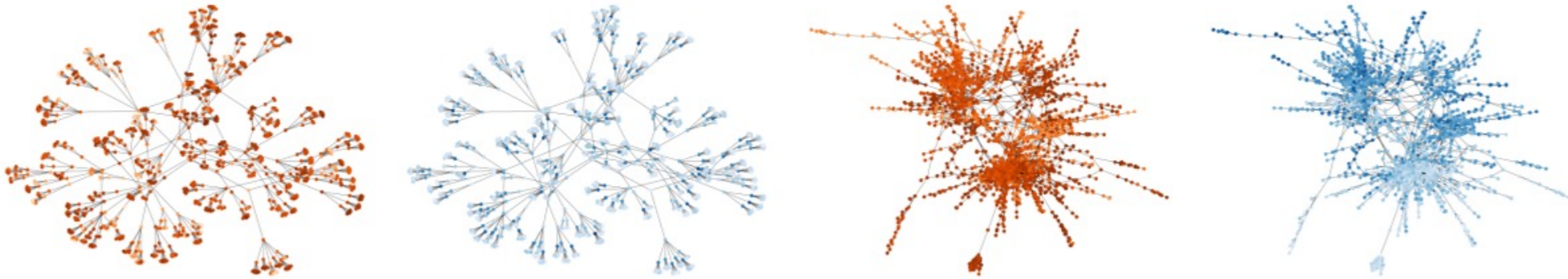


Zhu S, et al. "Graph geometry interaction learning.“ NeuRIPS 2020

Graph Geometry Interaction Learning(GIL)

Table 2: Link prediction (LP) in ROC AUC and node classification (NC) in Accuracy with the std.

Method	Disease		Airport		Pubmed		Citeseer		Cora	
	LP	NC	LP	NC	LP	NC	LP	NC	LP	NC
EUC	69.41±1.41	32.56±1.19	92.00±0.00	60.90±3.40	79.82±2.87	48.20±0.76	80.15±0.86	61.28±0.91	84.92±1.17	23.80±0.80
HYP	73.00±1.43	45.52±3.09	94.57±0.00	70.29±0.40	84.91±0.17	68.51±0.37	79.65±0.86	61.71±0.74	86.64±0.61	22.13±0.97
MLP	63.65±2.63	28.80±2.23	89.81±0.56	68.90±0.46	83.33±0.56	72.40±0.21	93.73±0.63	59.53±0.90	83.33±0.56	51.59±1.28
HNN	71.26±1.96	41.18±1.85	90.81±0.23	80.59±0.46	94.69±0.06	69.88±0.43	93.30±0.52	59.50±1.28	90.92±0.40	54.76±0.61
GCN	58.00±1.41	69.79±0.54	89.31±0.43	81.59±0.61	89.56±3.66	78.10±0.43	82.56±1.92	70.35±0.41	90.47±0.24	81.50±0.53
GAT	58.16±0.92	70.40±0.49	90.85±0.23	81.59±0.36	91.46±1.82	78.21±0.44	86.48±1.50	71.58±0.80	93.17±0.20	83.03±0.50
SAGE										77.90±2.50
SGC										81.32±0.50
HGNN										78.26±1.19
HGCN										78.03±0.98
HGAT										78.32±1.39
EucHy										81.38±0.62
GIL										**83.64±0.63**

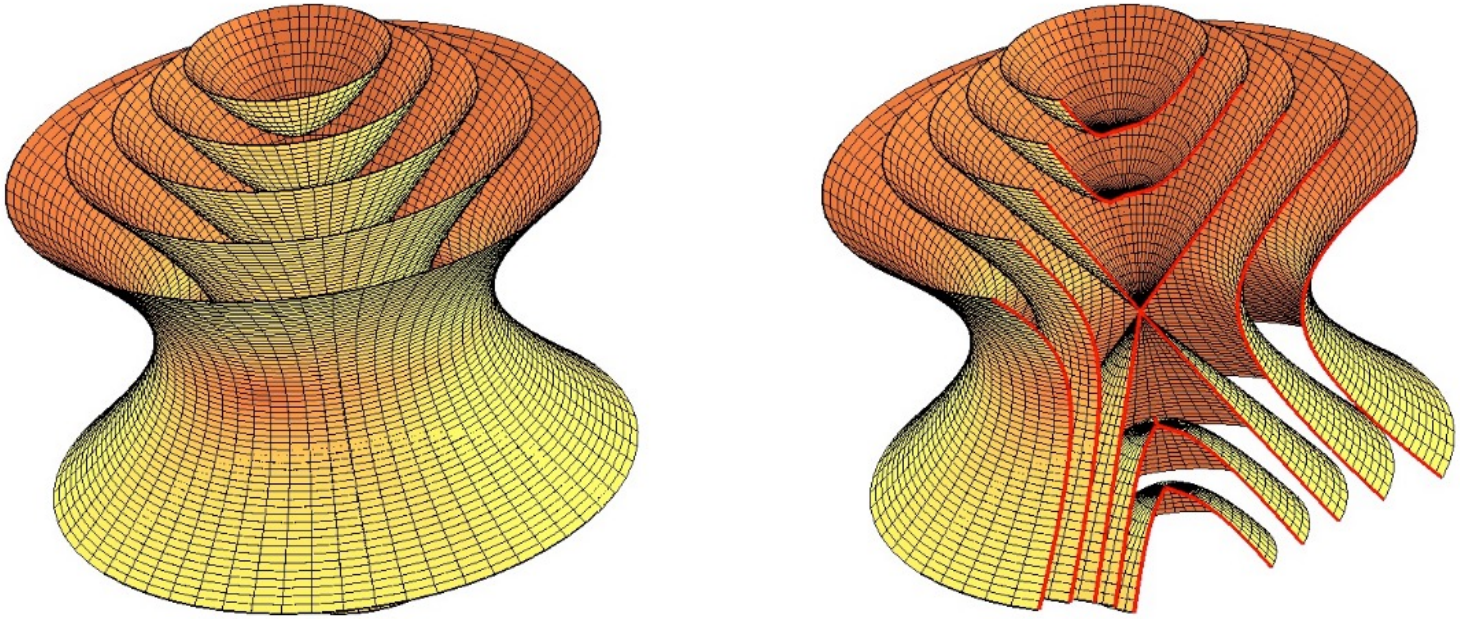


(a) Disease: Hyperbolic (b) Disease: Euclidean (c) Citeseer: Hyperbolic (d) Citeseer: Euclidean

Zhu S, et al. "Graph geometry interaction learning.“ NeuRIPS 2020

Pseudo-Riemannian Embedding

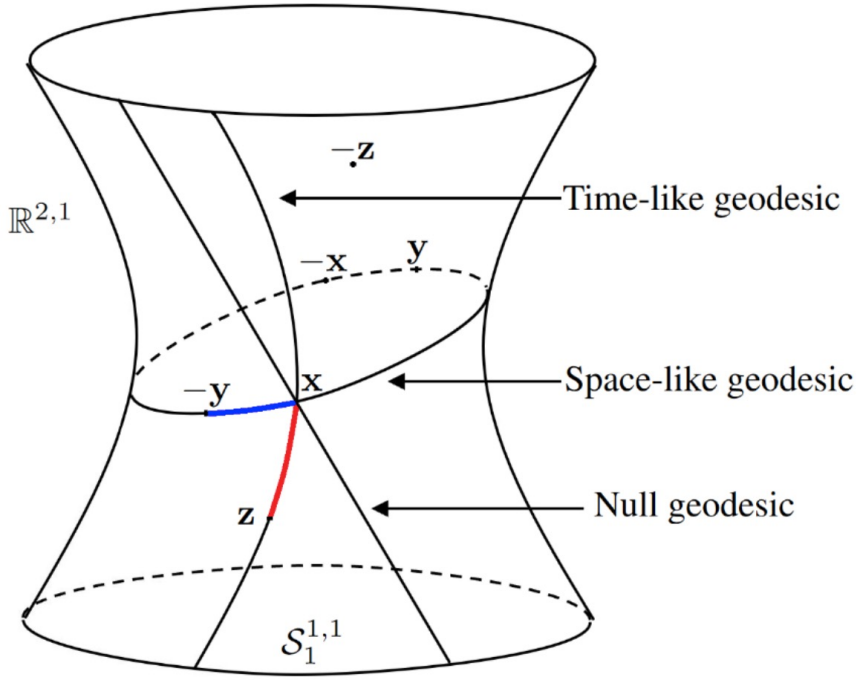
- **Pseudo-Riemannian manifolds** generalize Riemannian manifolds to cases where the inner product can be non-positive definite



Marc T. Law et al. Ultrahyperbolic Representation Learning. NeurIPS 2020.

Pseudo-Riemannian Embedding

- Pseudo-hyperboloids generalize spherical and hyperbolic manifolds to the class of pseudo-Riemannian manifolds



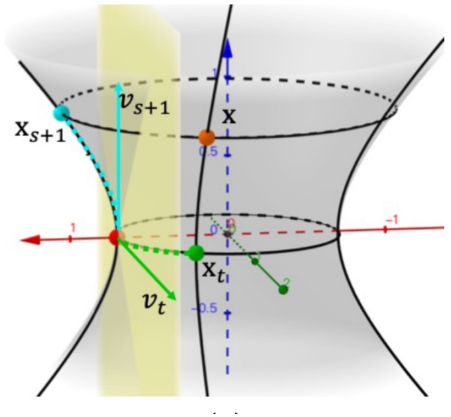
- Space-like geodesics fit **cycles**
- Time-like geodesics fit **trees**

Marc T. Law et al. Ultrahyperbolic Representation Learning. NeurIPS 2020.

Pseudo-Riemannian Embedding

- **Broken geodesics** occurs in Pseudo-Riemannian manifolds
  - Some points cannot be connected by a smooth geodesic
  - Logarithm map, distance and parallel transport are not available

- **Solution:**
  - Mapping operations into Riemannian spaces with **Diffeomorphism**



$$\psi : \mathcal{Q}_{\beta}^{s,t} \to \mathbb{S}_{-\beta}^{t} \times \mathbb{R}^{s}$$

Xiong, Bo et.al. Pseudo-Riemannian Graph Convolutional Networks. NeurIPS 2022.

Pseudo-Riemannian Graph Convolutional Networks

- **Diffeomorphic tangential operations**
  - Logmap & expmap with diffeomorphism

$$\log_{\mathbb{S}^t_{-\beta}\times\mathbb{R}^s}(\mathbf{x}') = \log_{\mathbb{S}^t_{-\beta}}(\mathbf{x}'_t) \,\|\, \log_{\mathbb{R}^s}(\mathbf{x}'_s), \quad \exp_{\mathbb{S}^t_{-\beta}\times\mathbb{R}^s}(\boldsymbol{\xi}) = \exp_{\mathbb{S}^t_{-\beta}}(\boldsymbol{\xi}_t) \,\|\, \exp_{\mathbb{R}^s}(\boldsymbol{\xi}_s)$$

Tangential operations. For function $f : \mathbb{R}^d \to \mathbb{R}^{d'}$, the pseudo-hyperboloid version $f^{\otimes} : \mathcal{Q}^{s,t}_{\beta} \to \mathcal{Q}^{s',t'}_{\beta}$ with $s+t=d$ and $s'+t'=d'$ can be defined by the means of $\widehat{\log}^{\beta}_{\mathbf{x}}(\cdot)$ and $\widehat{\exp}^{\beta}_{\mathbf{x}}(\cdot)$ as Eq. (5).

$$f^{\otimes}(\cdot) := \widehat{\exp}^{\beta}_{\mathbf{x}}\left(f\left(\widehat{\log}^{\beta}_{\mathbf{x}}(\cdot)\right)\right), \tag{5}$$

Xiong, Bo et.al. Pseudo-Riemannian Graph Convolutional Networks. NeurIPS 2022.

Pseudo-Riemannian Graph Convolutional Networks

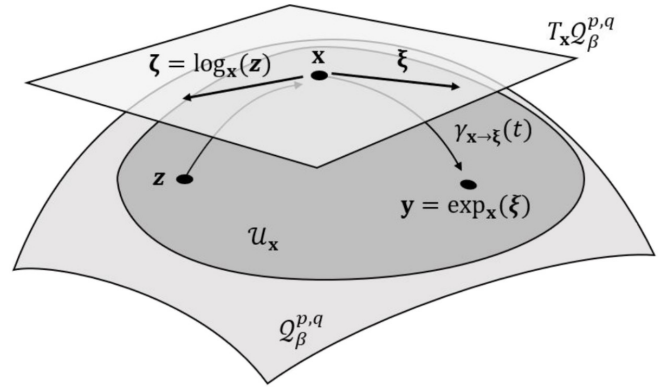
- **Diffeomorphic tangential operations**
  - Tangential transformation

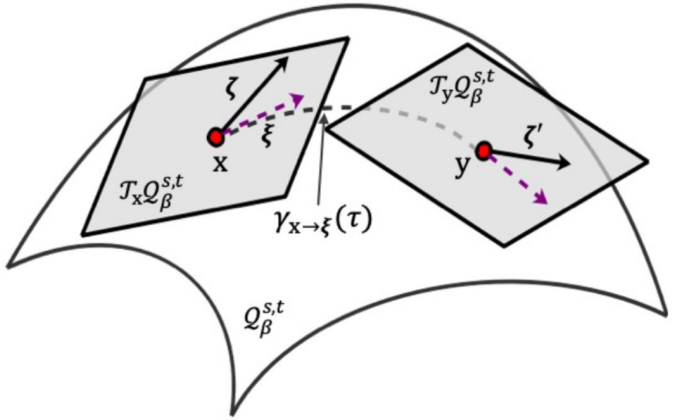
$$W^{\ell} \otimes^{\beta} \mathbf{h}^{\ell} := \widehat{\exp}_{\mathbf{o}}^{\beta}\left(W^{\ell}\widehat{\log}_{\mathbf{o}}^{\beta}\left(\mathbf{h}^{\ell}\right)\right)$$

  - Tangential aggregation

$$\mathbf{h}_i^{\ell+1} = \widehat{\exp}_{\mathbf{o}}^{\beta_{\ell+1}}\left(\sigma\left(\sum_{j\in\mathcal{N}(i)\cup\{i\}} \widehat{\log}_{\mathbf{o}}^{\beta_{\ell}}\left(W^{\ell} \otimes^{\beta_{\ell}} \mathbf{h}_j^{\ell} \oplus^{\beta_{\ell}} \mathbf{b}^{\ell}\right)\right)\right)$$

$$\widetilde{\mathbf{h}}^{\ell} \oplus^{\beta} \mathbf{b}^{\ell} := \begin{cases} \exp_{\widetilde{\mathbf{h}}^{\ell}}^{\beta}\left(P_{\mathbf{o}\to\widetilde{\mathbf{h}}^{\ell}}^{\beta}\left(\mathbf{b}^{\ell}\right)\right), & \text{if } \langle \mathbf{o}, \widetilde{\mathbf{h}}^{\ell}\rangle_t < |\beta| \\ -\exp_{-\widetilde{\mathbf{h}}^{\ell}}^{\beta}\left(P_{\mathbf{o}\to-\widetilde{\mathbf{h}}^{\ell}}^{\beta}\left(\mathbf{b}^{\ell}\right)\right), & \text{if } \langle \mathbf{o}, \widetilde{\mathbf{h}}^{\ell}\rangle_t \geq |\beta| \end{cases}$$





Xiong, Bo et.al. Pseudo-Riemannian Graph Convolutional Networks. NeurIPS 2022.

Pseudo-Riemannian Graph Convolutional Networks

- **Pseudo-hyperboloid**

Table 2: ROC AUC for Link Prediction (LP) and F1 score for Node Classifica

Dataset δ-hyperbolicity	Airport 1.0		Pubmed 3.5		CiteSeer 4.5			
Method	LP	NC	LP	NC	LP	NC		
GCN [2]	89.24±0.21	81.54±0.60	91.31±1.68	79.30±0.60	85.48±1.75	72.27±0.64	8	
GAT [3]	90.35±0.30	81.55±0.53	87.45±0.00	78.30±0.00	87.24±0.00	71.10±0.00	8	
SAGE [33]	89.86±0.52	82.79±0.17	90.70±0.07	77.30±0.09	90.71±0.20	69.20±0.10	8	
SGC [4]	89.80±0.34	80.69±0.23	90.54±0.07	78.60±0.30	89.61±0.23	71.60±0.03	8	
HGCN [27] ($\mathbb{H}^{16}$)	**96.03±0.26**	**90.57±0.36**	**96.08±0.21**	**80.50±1.23**	**96.31±0.41**	68.90±0.63	9	
κ-GCN [28] ($\mathbb{H}^{16}$)	**96.35±0.62**	**87.92±1.33**	96.60±0.32	77.96±0.36	95.34±0.16	**73.25±0.51**	9	
κ-GCN [28] ($\mathbb{S}^{16}$)	90.38±0.32	81.94±0.58	94.84±0.13	78.80±0.49	95.79±0.24	72.13±0.51	9	
κ-GCN [28] ($\mathbb{H}^8 \times \mathbb{S}^8$)	93.10±0.49	81.93±0.45	94.89±0.19	79.20±0.65	93.44±0.31	73.05±0.59	9	
$\mathcal{Q}$-GCN ($\mathcal{Q}^{15,1}$)	**96.30±0.22**	**89.72±0.52**	95.42±0.22	**80.50±0.26**	94.76±1.49	72.67±0.76	93.14±0.30	80.57±0.20
$\mathcal{Q}$-GCN ($\mathcal{Q}^{14,2}$)	94.37±0.44	84.40±0.35	**96.86±0.37**	**81.34±1.54**	94.78±0.17	**73.43±0.58**	93.41±0.57	81.62±0.21
$\mathcal{Q}$-GCN ($\mathcal{Q}^{13,3}$)	92.53±0.				54±0.16	**74**		
$\mathcal{Q}$-GCN ($\mathcal{Q}^{2,14}$)	90.03±0.				80±0.08	7		
$\mathcal{Q}$-GCN ($\mathcal{Q}^{1,15}$)	89.07±0.				**01±0.30**	**7**		
$\mathcal{Q}$-GCN ($\mathcal{Q}^{0,16}$)	89.01±0.				**21±0.38**	7		

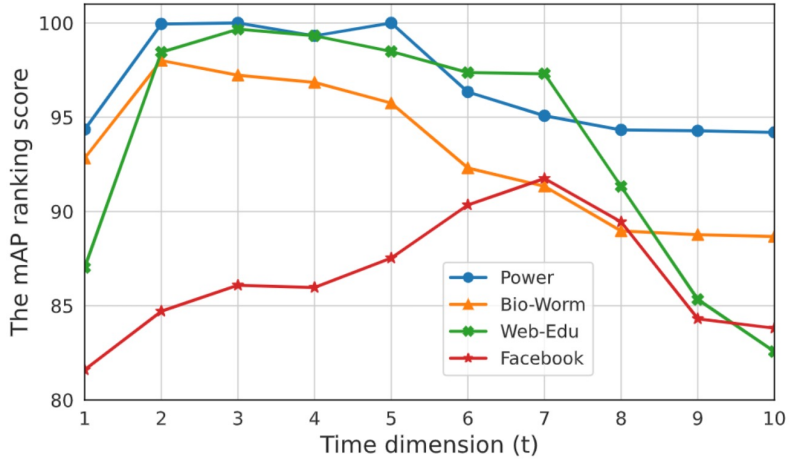
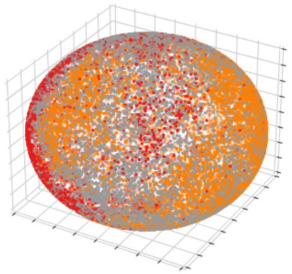
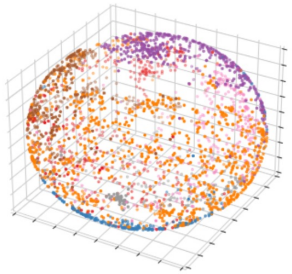


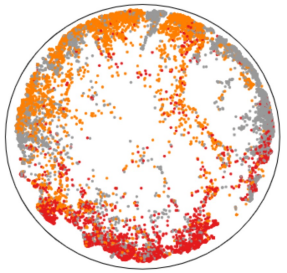
Figure 3: The mAP of graph reconstruction with varying number of time dimensions.



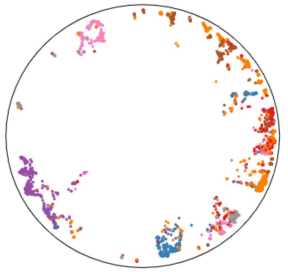
Pubmed



Cora



Pubmed

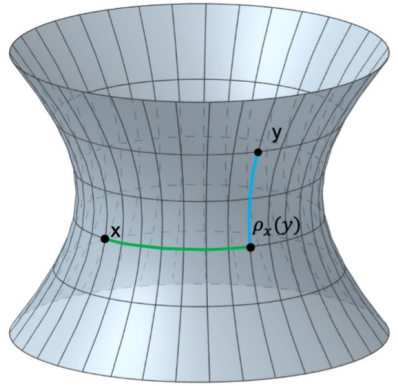


Cora

Xiong, Bo et.al. Pseudo-Riemannian Graph Convolutional Networks. NeurIPS 2022.

Pseudo-Riemannian Embeddings for Multi-Relational Graphs

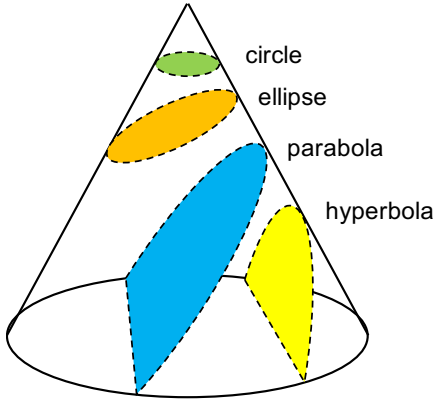
- Main ideas
  - Entities as points in the manifold
  - Relations as linear (pseudo-orthogonal) transformations



- Pseudo-orthogonal transformation

$$\mathbf{Q}^T \mathbf{J} \mathbf{Q} = \mathbf{J}, \tag{3}$$

where $\mathbf{J} = \begin{bmatrix} \mathbf{I}_p & \mathbf{0} \\ \mathbf{0} & -\mathbf{I}_q \end{bmatrix}$, $p + q = d$ and $\mathbf{I}_p$, $\mathbf{I}_q$ are identity matrices.



- Parameterization requires O(d^2) many parameters
- Constrained optimization

Xiong, Bo et.al. Ultrahyperbolic Knowledge Graph Embeddings. KDD 2022.

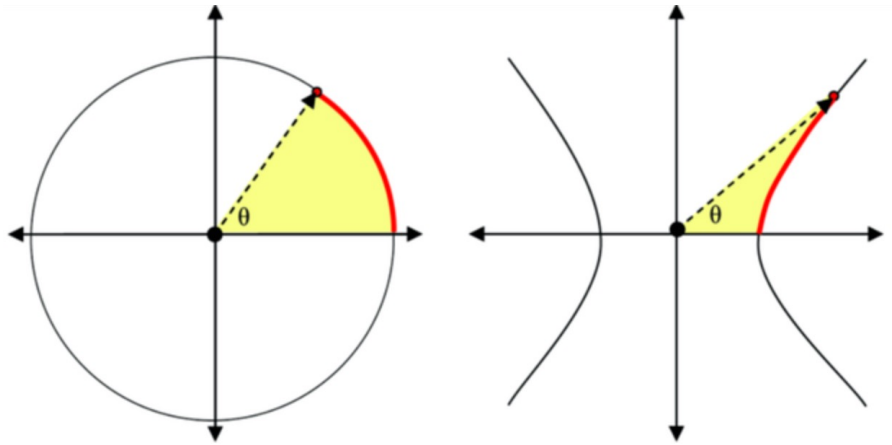
Pseudo-Riemannian Embeddings for Multi-Relational Graphs

- Pseudo-orthogonal transformation is decomposed into
  - circular rotation/reflection + hyperbolic rotation
  - by Hyperbolic Cosine-Sine decomposition [1]

$$\mathbf{Q} = \begin{bmatrix} \mathbf{U}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{U}_2 \end{bmatrix} \begin{bmatrix} \mathbf{C} & \mathbf{0} & \mathbf{S} \\ \mathbf{0} & I_{p-q} & \mathbf{0} \\ \mathbf{S} & \mathbf{0} & \mathbf{C} \end{bmatrix} \begin{bmatrix} \mathbf{V}_1^T & \mathbf{0} \\ \mathbf{0} & \mathbf{V}_2^T \end{bmatrix}, \quad (4)$$

where $\mathbf{C} = \mathrm{diag}\left(c_1, \ldots, c_q\right), \mathbf{S} = \mathrm{diag}\left(s_1, \ldots, s_q\right)$ *and* $\mathbf{C}^2 - \mathbf{S}^2 = \mathbf{I}_q$.

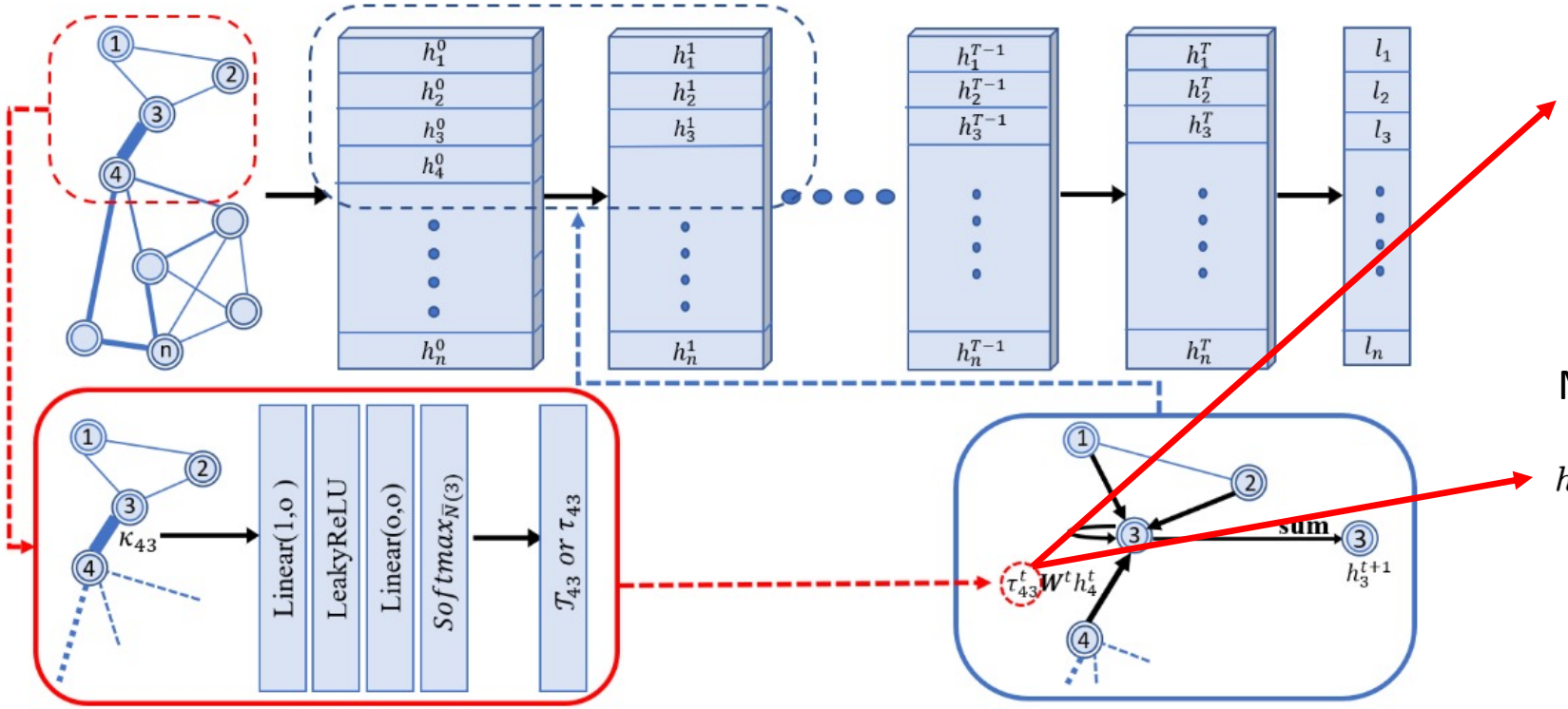
- Circular rotations/reflections encode non-hierarchy and relational patterns (e.g., symmetry)
- Hyperbolic rotations encode hierarchy



Xiong, Bo et.al. Ultrahyperbolic Knowledge Graph Embeddings. KDD 2022.

4.2 Curvature-Aware Learning

- Incorporate graph structural information to GCN
- MLP transforms curvature to reweigh messages



Single-Channel: CurvGN-1

$$\tau_{xy}^{t} = S^{t}(\mathrm{MLP}^{t}(\kappa_{xy}))$$

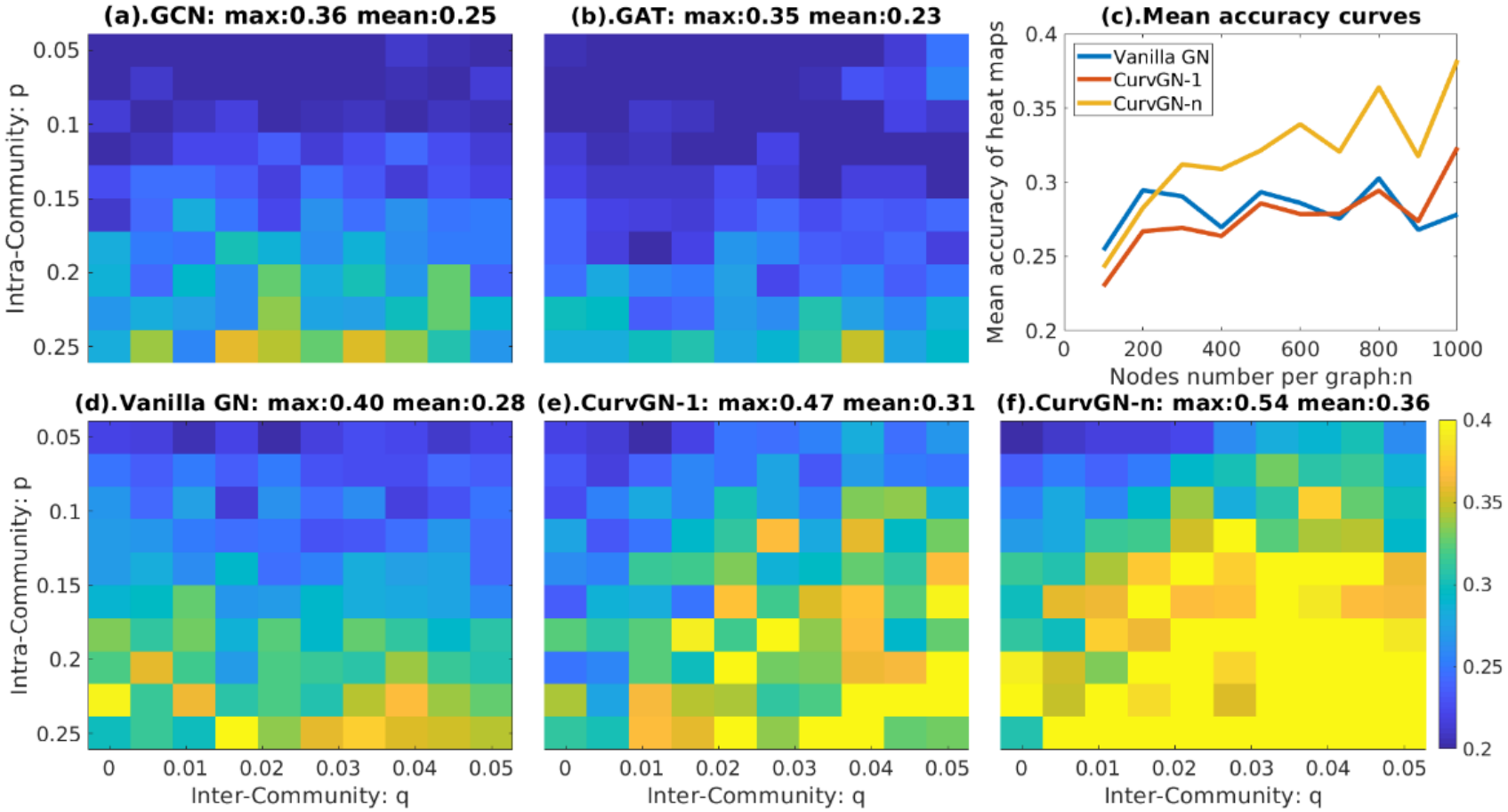
Multi-Channel: CurvGN-n

$$h_{x}^{t+1} = \sigma_{t}\left(\sum\nolimits_{y\in\overline{\mathcal{N}}(x)} \mathrm{diag}(\mathcal{T}_{xy}^{t})W^{t}h_{y}^{t}\right)$$

Ze Ye et al. Curvature Graph Networks. ICLR 2020

Curvature Graph Networks (CurvGN)

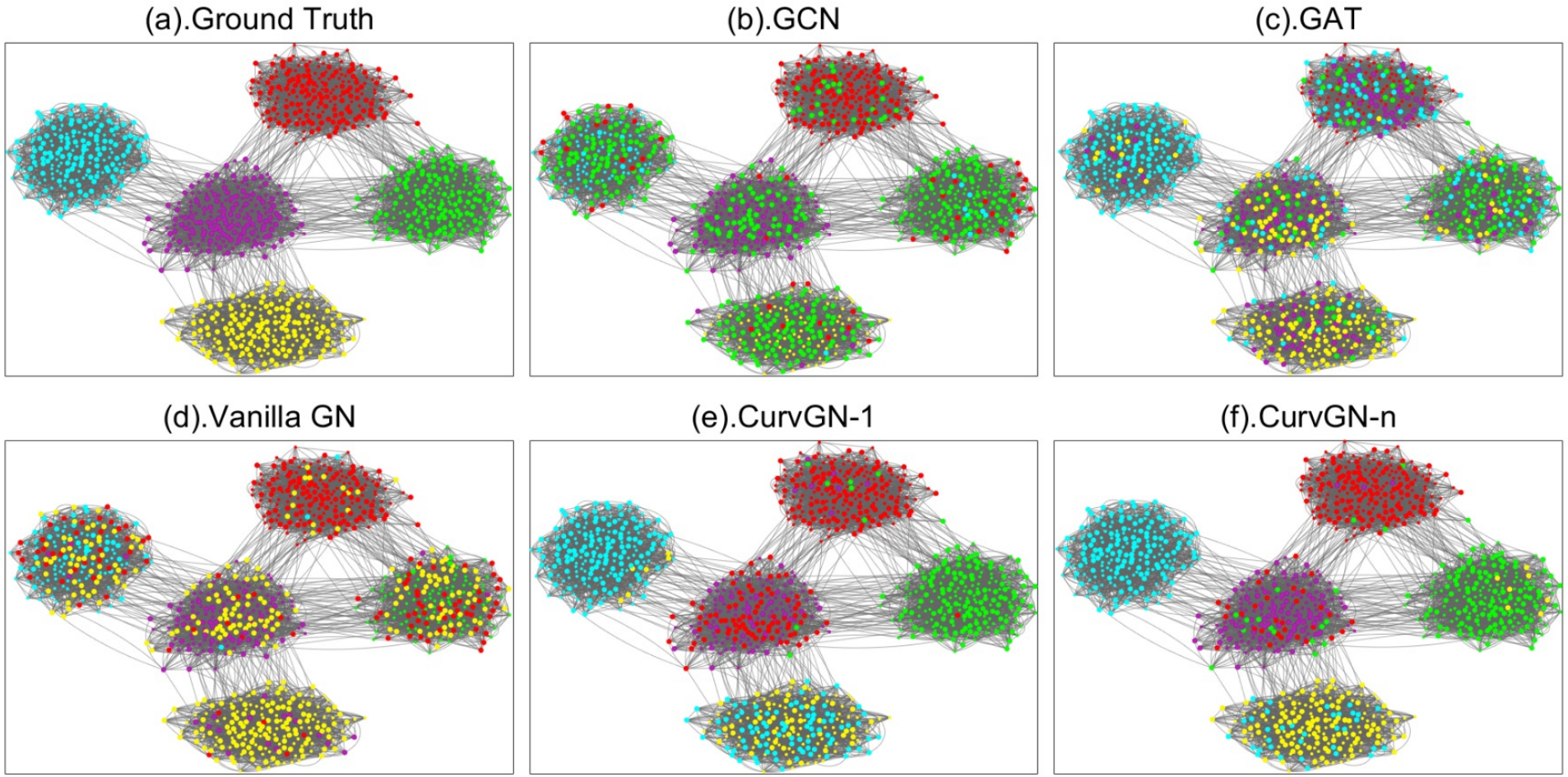
- Community detection



Ze Ye et al. Curvature Graph Networks. ICLR 2020

Curvature Graph Networks (CurvGN)

- Community detection



Ze Ye et al. Curvature Graph Networks. ICLR 2020

Curvature-aware graph embedding

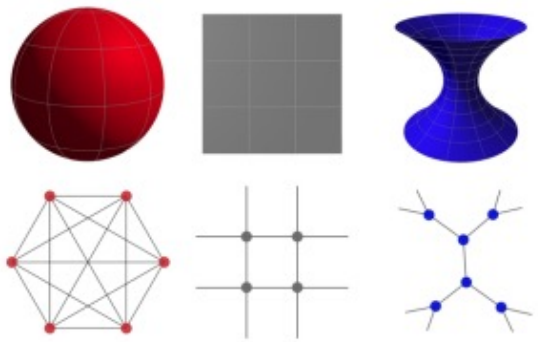
- Adapt an augmented Forman-Ricci curvature:

$$\mathrm{F}(i,j) = 4 - d_i - d_j + 3\gamma \sharp_{\Delta}(i,j), \quad \gamma > 0,$$

Degree of node i

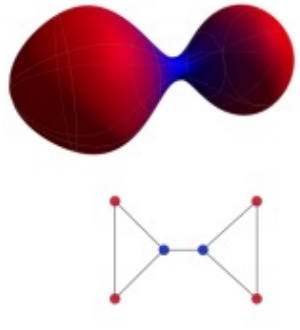
The number of triangles at the edge(i,j)

$$\mathrm{F}(i) = \frac{1}{d_i} \sum_{j:j\sim i} \mathrm{F}(i,j)$$



(a) Space forms (sphere, plane, hyperboloid)

(b) Product manifold



(c) Heterogeneous manifold

Heterogeneous manifolds for curvature-aware graph embedding. ICLR workshop 2022

Curvature-aware graph embedding

- Loss function: Minimize distance and curvature depending loss

$$\mathcal{L}(\{y_i\}) = \mathcal{L}_{\mathrm{d}}(\{y_i\}) + \tau \mathcal{L}_{\mathrm{c}}(\{y_i\}),$$

$$\mathcal{L}_d(\{y_i\}) = \sum_{i,j} \left| \frac{d^2_{g_h}(z_i, z_j) + (r_i - r_j)^2}{d^2_G(x_i, x_j)} - 1 \right|$$

Measures the relative squared distance Distortion.

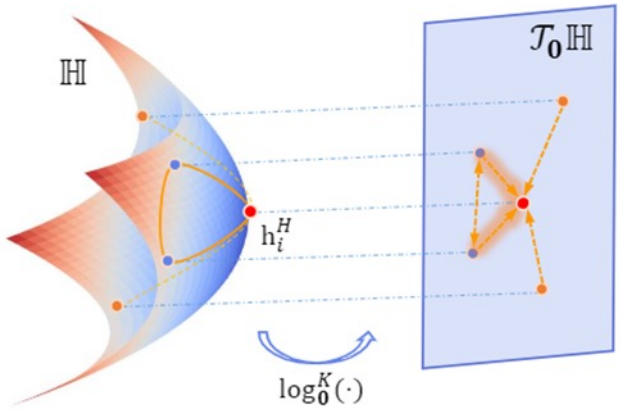
$$\mathcal{L}_c(\{y_i\}) = \sum_i \frac{(\mathrm{F}(x_i) - \mathrm{R}_h - \mathrm{R}_\alpha(r_i))^2}{(|\mathrm{F}(x_i)| + \epsilon)^2},$$

Measures how close the local geometry of the manifold around the embedded nodes resembles that of the graph.

Di Giovanni, F. et.al Heterogeneous manifolds for curvature-aware graph embedding. ICLR workshop 2022

Curvature-aware graph embedding

- Curvature-guided message aggregation



$$\tilde{\mathbf{h}}_i^{\ell,H} = \exp_{\mathbf{0}}^{K_{l-1}}\left(\sum_{j\in N(i)} \tilde{\kappa}_{i,j} \cdot \log_{\mathbf{0}}^{K_{l-1}}(\mathbf{h}_j^{\ell,H})\right)$$

$$\tilde{\kappa}_{i,j} = \mathrm{softmax}_{j\in\mathcal{N}(i)}\left(\mathrm{MLP}(\kappa_{i,j} - |L_i - L_j|)\right)$$

- Curvature-enhanced message exchange

$$l_{hc+} = \frac{1}{|E_\kappa|}\sum_{(i,j)\in E_\kappa} -\log\, p(\mathbf{x}_i^{\ell,\mathcal{H}}, \mathbf{x}_j^{\ell,\mathcal{H}}),$$

Yang, M. et.al. Hyperbolic Curvature Graph Neural Network. arXiv. 2022

Agenda

Part 1. INTRODUCTION (Min, 9:00 - 9:30)

1.1 An Overview of Graph Learning

1.2 Brief Introduction of Riemannian Geometry

1.3 Motivation for Hyperbolic Graph Learning

Coffee Break (9:30 – 10:000)

Part 2. HYPERBOLIC GRAPH NETWORKS (Menglin, 10:00 – 10:30)

2.1 Preliminary

2.2 Hyperbolic Shallow Models

2.2 Hyperbolic Neural Networks

2.3 Hyperbolic Graph Neural Networks

Part 3. APPLICATIONS (Menglin, 10:30 – 11:00)

3.1 HGNN for Recommendation Systems

3.2 HGNN for Knowledge Graph

3.3 HGNN for Bioinformatics

Part 4. ADVANCED TOPICS (Bo, 11:00 – 12:00)

4.1 Complex Structures

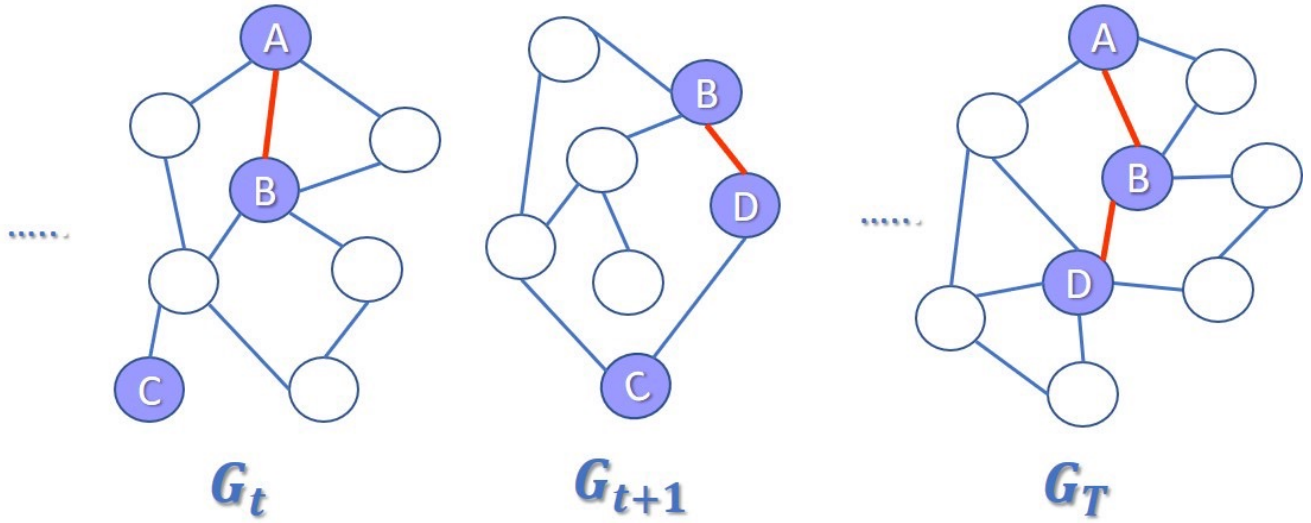
4.2 Curvature-aware Learning

4.3 Evolving Interactions

4.4 Trustworthy and Scalability

4.3 Evolving Interactions

- Discrete-time temporal network embedding via implicit hierarchical learning in hyperbolic space. KDD2021
- Hyperbolic Variational Graph Neural Network for Modeling Dynamic Graphs. AAAI2021
- A Self-supervised Riemannian GNN with Time Varying Curvature for Temporal Graph Learning. CIKM 2022
- …



https://towardsdatascience.com/tdgraphembed-temporal-dynamic-graph-level-embedding-1cc611bc7db0

Hyperbolic Temporal Network Embeddings

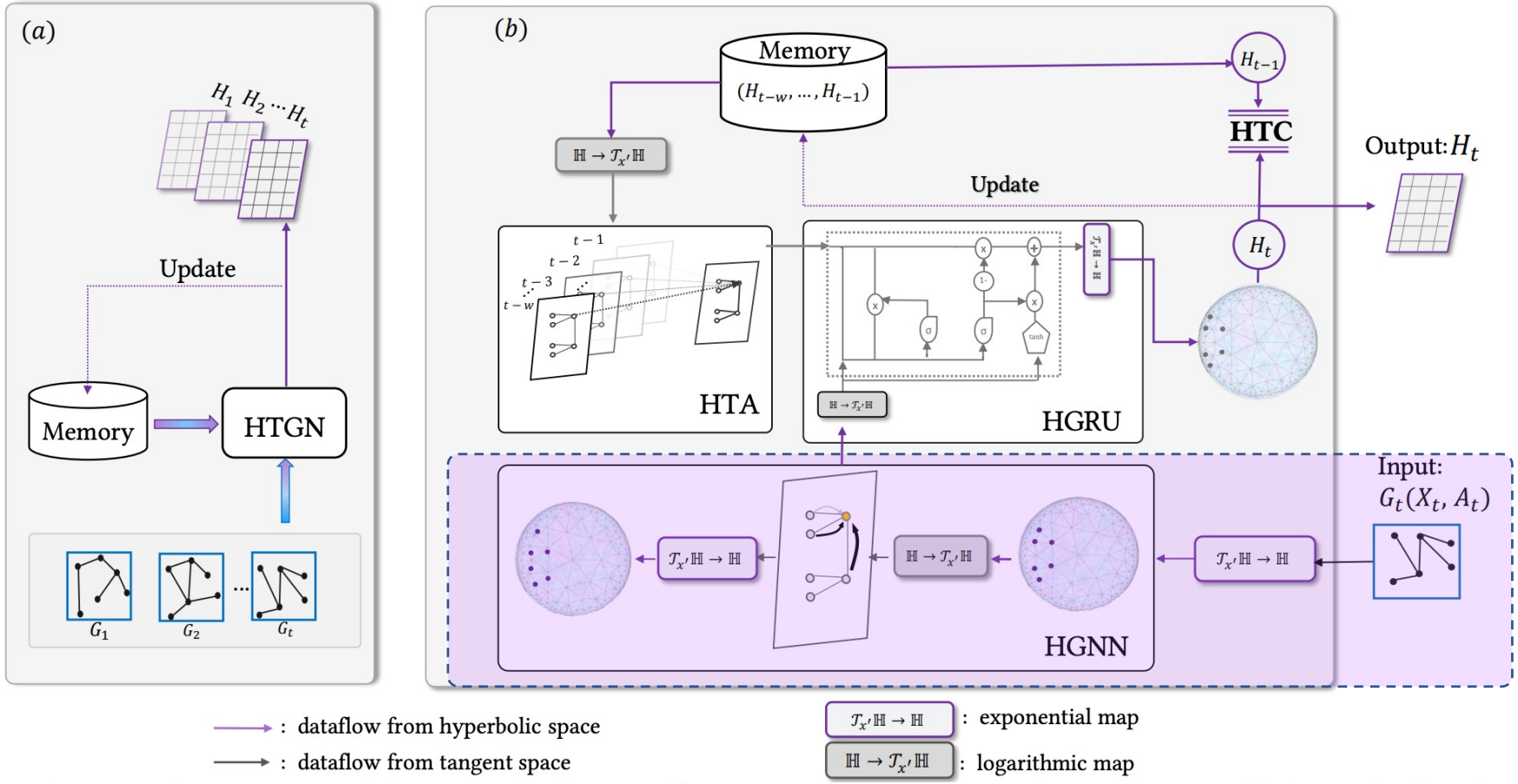


Figure 3: Schematic of HTGN. Figure (a) is a sketch of HTGN illustrating the recurrent paradigm and Figure (b) shows the data flow of an HTGN unit.

Yang. M. et.al. Discrete-time temporal network embedding via implicit hierarchical learning in hyperbolic space KDD2021

Hyperbolic Temporal Network Embeddings

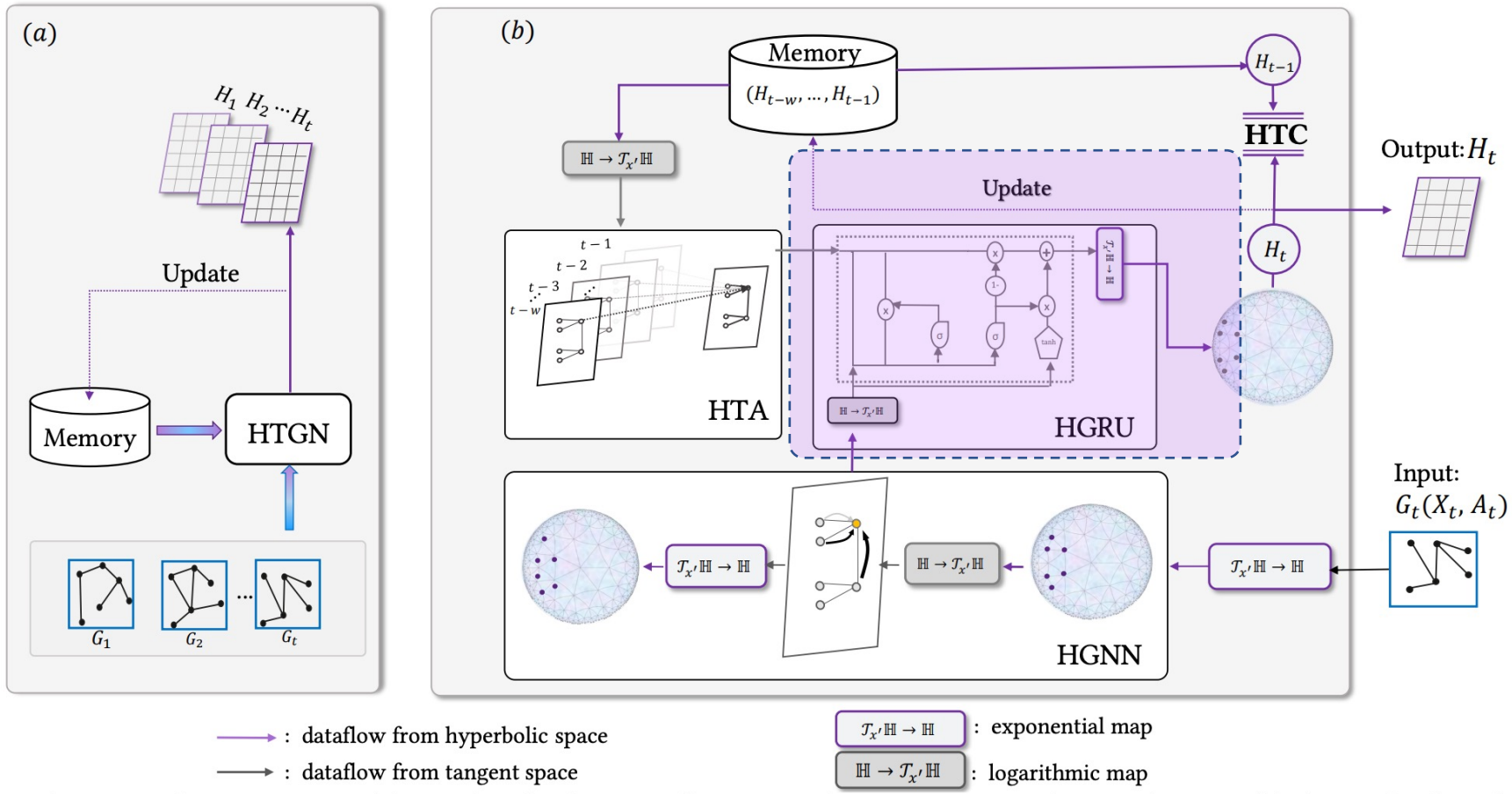


Figure 3: Schematic of HTGN. Figure (a) is a sketch of HTGN illustrating the recurrent paradigm and Figure (b) shows the data flow of an HTGN unit.

Yang. M. et.al. Discrete-time temporal network embedding via implicit hierarchical learning in hyperbolic space KDD2021

Hyperbolic Temporal Network Embeddings

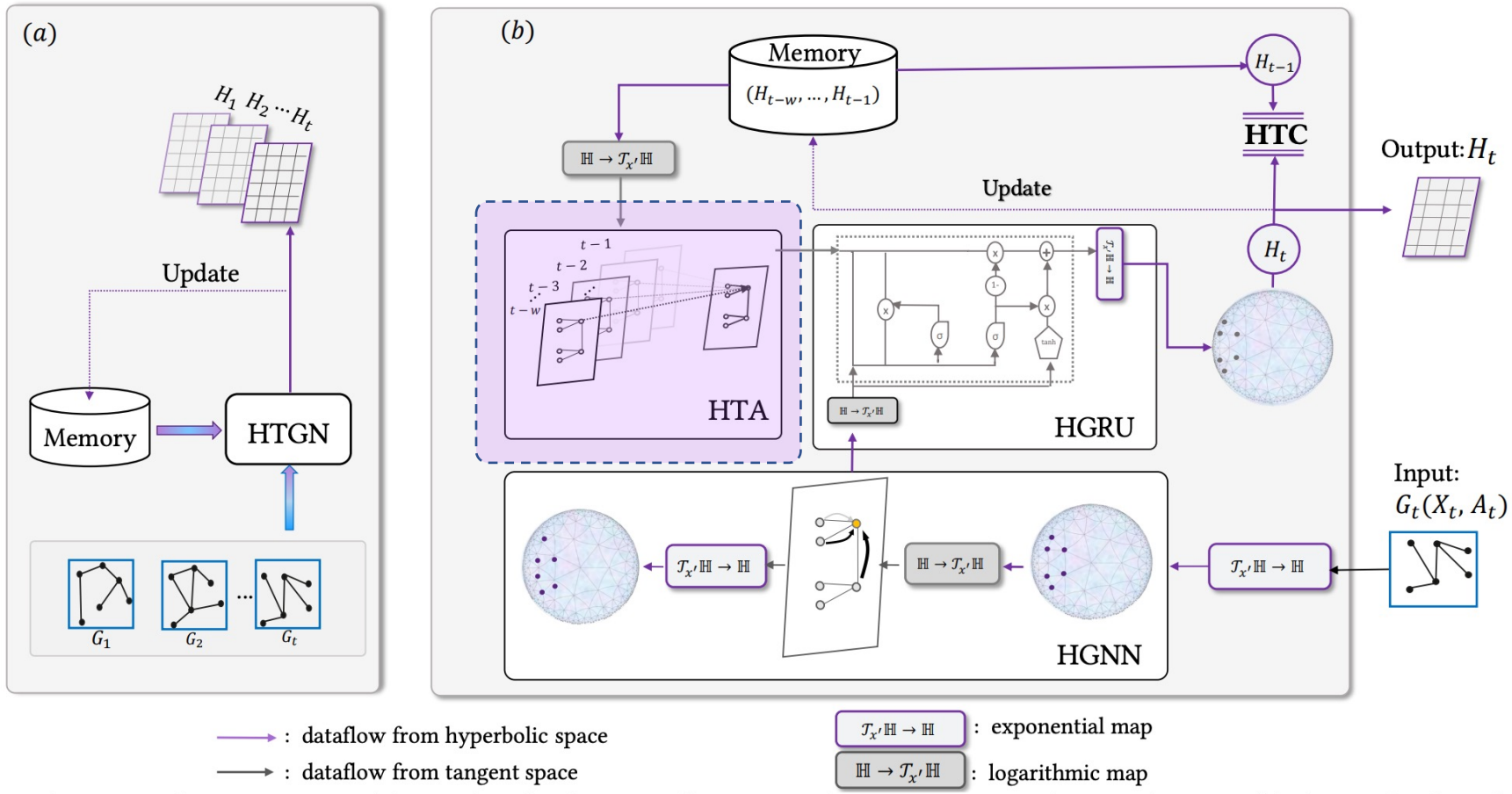


Figure 3: Schematic of HTGN. Figure (a) is a sketch of HTGN illustrating the recurrent paradigm and Figure (b) shows the data flow of an HTGN unit.

Yang. M. et.al. Discrete-time temporal network embedding via implicit hierarchical learning in hyperbolic space KDD2021

Hyperbolic Temporal Network Embeddings

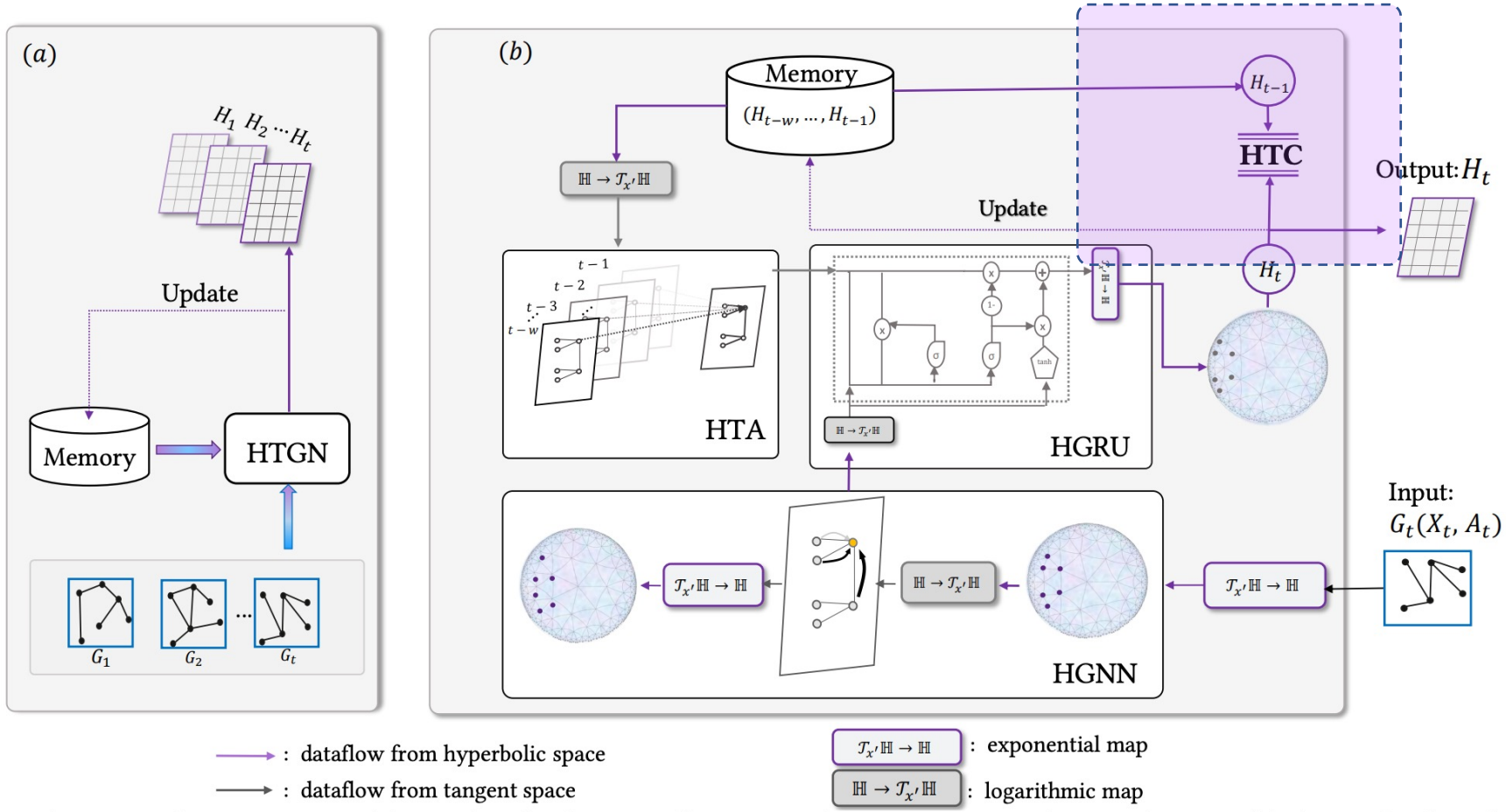


Figure 3: Schematic of HTGN. Figure (a) is a sketch of HTGN illustrating the recurrent paradigm and Figure (b) shows the data flow of an HTGN unit.

Yang. M. et.al. Discrete-time temporal network embedding via implicit hierarchical learning in hyperbolic space KDD 2021

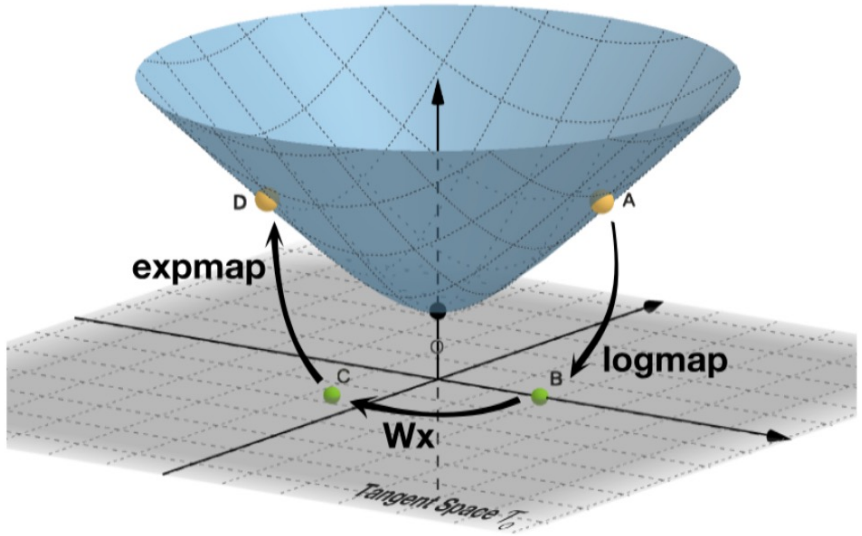
Hyperbolic Temporal Network Embeddings

Table 3: AUC (left) and AP (right) scores of dynamic link prediction on real-world dynamic graphs.

	AUC						AP					
Dataset	DISEASE	HepPh	FB	AS733	Enron	COLAB	DISEASE	HepPh	FB	AS733	Enron	COLAB
GAE	72.55 ± 1.20	69.44 ± 0.56	63.07 ± 0.93	93.21 ± 1.53	92.50 ± 0.68	84.57 ± 0.64	60.55 ± 1.01	73.61 ± 0.58	65.35 ± 0.90	94.75 ± 0.90	*93.48 ± 0.64*	87.69 ± 0.44
VGAE	83.08 ± 1.27	72.39 ± 0.11	67.16 ± 0.53	*95.76 ± 0.91*	91.93 ± 0.34	85.16 ± 0.74	78.34 ± 1.25	75.78 ± 0.06	69.73 ± 0.17	96.42 ± 0.55	93.45 ± 0.49	88.70 ± 0.35
EvolveGCN	73.55 ± 4.23	76.82 ± 1.46	76.85 ± 0.85	92.47 ± 0.04	90.12 ± 0.69	83.88 ± 0.53	73.25 ± 3.44	81.18 ± 0.89	80.87 ± 0.64	95.28 ± 0.01	92.71 ± 0.34	87.53 ± 0.22
GRUGCN	79.25 ± 1.69	*82.86 ± 0.53*	*79.38 ± 1.02*	94.96 ± 0.35	92.47 ± 0.36	84.60 ± 0.92	65.26 ± 1.94	*85.87 ± 0.23*	*82.77 ± 0.75*	96.64 ± 0.22	93.38 ± 0.24	87.87 ± 0.58
DySAT	73.74 ± 2.28	81.02 ± 0.25	76.88 ± 0.08	95.06 ± 0.21	93.06 ± 0.97	*87.25 ± 1.70*	63.81 ± 1.86	84.47 ± 0.23	80.39 ± 0.14	*96.72 ± 0.12*	93.06 ± 1.05	*90.40 ± 1.47*
VGRNN	*86.44 ± 3.12*	77.65 ± 0.99	78.11 ± 1.11	95.17 ± 0.62	*93.10 ± 0.57*	85.95 ± 0.49	*82.00 ± 3.83*	80.95 ± 0.94	80.40 ± 0.74	96.69 ± 0.31	93.29 ± 0.69	87.77 ± 0.79
HTGN (Ours)	89.65 ± 0.70	91.13 ± 0.14	83.70 ± 0.33	98.75 ± 0.03	94.17 ± 0.17	89.26 ± 0.17	84.63 ± 0.65	89.52 ± 0.28	83.80 ± 0.43	98.41 ± 0.03	94.31 ± 0.26	91.91 ± 0.07
Gain (%)	+3.71	+9.98	+5.44	+3.12	+1.15	+2.30	+3.21	+4.25	+1.24	+1.75	+0.89	+1.67

4.4 Trustworthy and Scalability

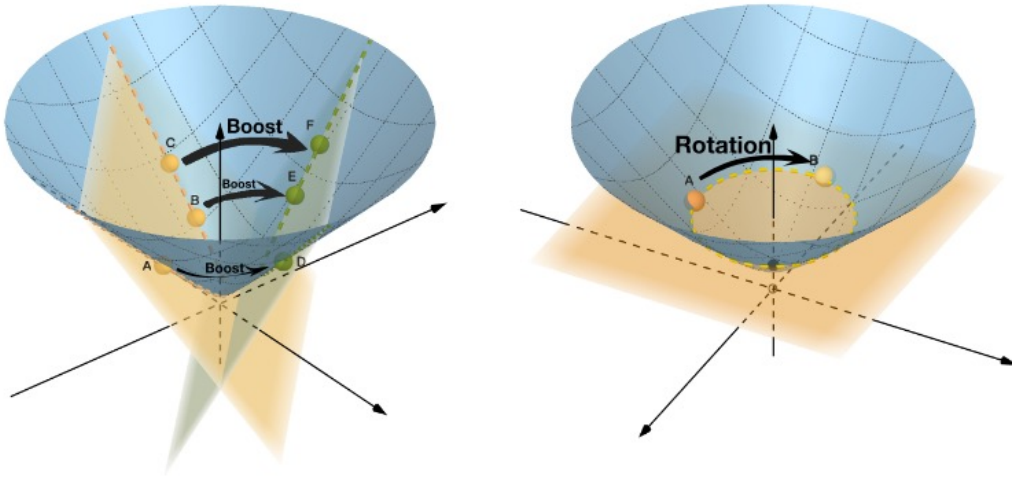
- HGNN operations are frequently mapped to the tangent space, resulting in information loss
- Vanishing gradients & numerical stability
- Scalability issues of HGNN



(a) Linear layer formalized in tangent space

Fully Hyperbolic Neural Networks

- A fully hyperbolic framework based on the Lorentz model
  - Lorentz transformations as linear transformation layers
  - Lorentz transformations = Lorentz Boost + Lorentz rotation
  - Tangent linear layer does not include the boost operations



(b) Lorentz boost (c) Lorentz rotation

Chen. W. et.al. Fully Hyperbolic Neural Networks. ACL 2022

Fully Hyperbolic Neural Networks

- Improving knowledge graph embeddings and machine translation

Model	WN18RR #Dims	WN18RR MRR	WN18RR H@10	WN18RR H@3	WN18RR H@1	FB15k-237 #Dims	FB15k-237 MRR	FB15k-237 H@10	FB15k-237 H@3	FB15k-237 H@1
TransE (Bordes et al., 2013)	180	22.7	50.6	38.6	3.5	200	28.0	48.0	32.1	17.7
DistMult (Yang et al., 2015)	270	41.5	48.5	43.0	38.1	200	19.3	35.3	20.8	11.5
ComplEx (Trouillon et al., 2017)	230	43.2	50.0	45.2	39.6	200	25.7	44.3	29.3	16.5
ConvE (Dettmers et al., 2018)	120	43.5	50.0	44.6	40.1	200	30.4	49.0	33.5	21.3
RotatE (Sun et al., 2019)	1,000	47.3	55.3	48.8	43.2	1,024	30.1	48.5	33.1	21.0
TuckER (Balazevic et al., 2019b)	200	46.1	53.5	47.8	42.3	200	34.7	53.3	38.4	25.4
MuRP (Balazevic et al., 2019a)	32	46.5	54.4	48.4	42.0	32	32.3	50.1	35.3	23.5
RotH (Chami et al., 2020a)	32	47.2	55.3	49.0	42.8	32	31.4	49.7	34.6	22.3
AttH (Chami et al., 2020a)	32	46.6	55.1	48.4	41.9					
HyboNet	32	48.9	55.3	50.3	45.5					
MuRP (Balazevic et al., 2019a)	β	48.1	56.6	49.5	44.0					
RotH (Chami et al., 2020a)	β	49.6	**58.6**	51.4	44.9					
AttH (Chami et al., 2020a)	β	48.6	57.3	49.9	44.3					
HyboNet	β	**51.3**	56.9	**52.7**	**48.2**					

Model	IWSLT'14 d=64	WMT'14 d=64	WMT'14 d=128	WMT'14 d=256
ConvSeq2Seq	23.6	14.9	20.0	21.8
Transformer	23.0	17.0	21.7	25.1
HyperNN++	22.0	17.0	19.4	21.8
HAtt	23.7	18.8	22.5	25.5
HyboNet	**25.9**	**19.7**	**23.3**	**26.2**

Chen. W. et.al. Fully Hyperbolic Neural Networks. ACL 2022

Hyperbolic Neural Networks using Taylor Series Approximations

- Motivation: Poincaré ball formulation primarily relies on the hyperbolic trigonometric functions: **tanh and cosh**
- Idea: Approximate these functions using **polynomial Taylor series expansion (PTSE)**

$$\tanh(x) \approx PTSE(\tanh(x)) = \sum_{i=1}^{n} \frac{(-1)^{i-1} 2^{2i} (2^{2i} - 1) B_n x^{2i-1}}{(2i)!}$$

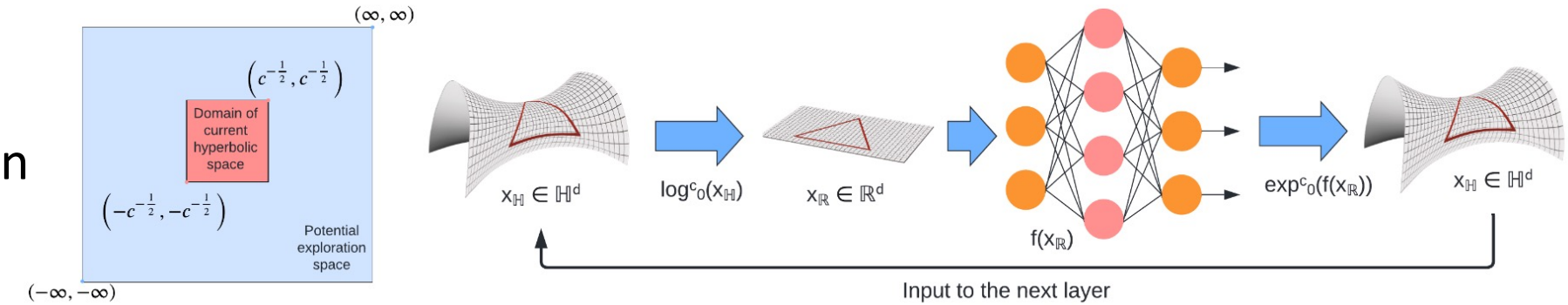
$$\operatorname{artanh}(x) \approx PTSE(\operatorname{artanh}(x)) = \sum_{i=1}^{n} \frac{x^{2i-1}}{2i-1}$$

$$\operatorname{arcosh}(x) \approx PTSE(\operatorname{arcosh}(x)) = \sum_{i=1}^{n} \left(\frac{(-1)^{i+1}(2x-1)^i}{i} - \left(\frac{\prod_{j=1}^{i} \frac{2j-1}{2j}}{(2i)x^{2i}} \right) \right) \forall |x| \geq 1$$

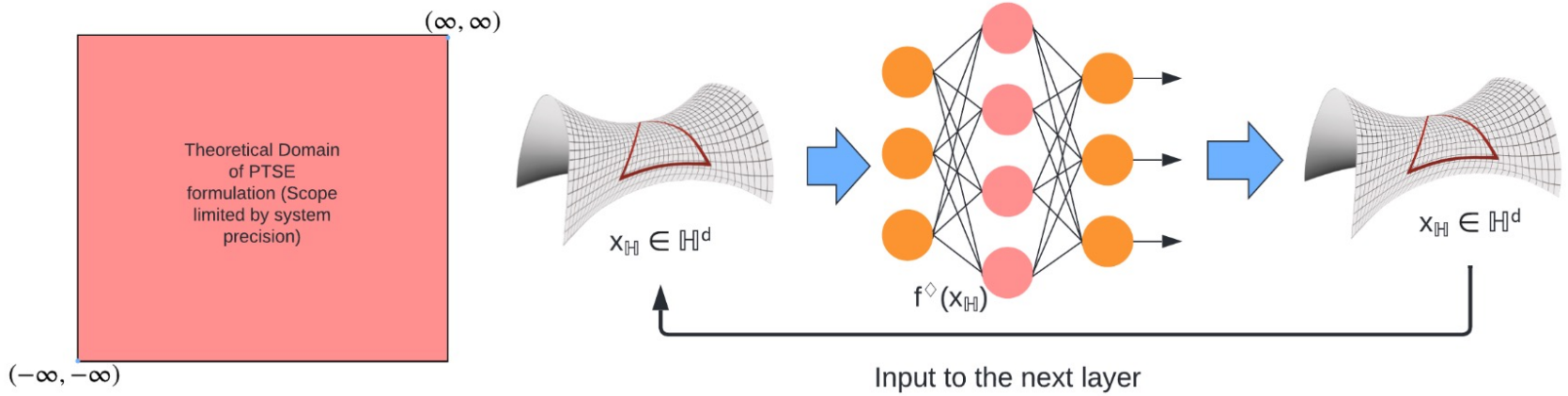
Choudhary. N. et.al . Towards Scalable Hyperbolic Neural Networks using Taylor Series Approximations. arXiv 2022.

Hyperbolic Neural Networks using Taylor Series Approximations

Tangent formulation



PTSE formulation



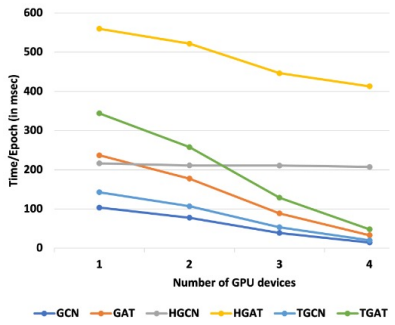
(a) PTSE formulation extends the exploration space to $\mathbb{R}$.

(b) PTSE formulations can directly be applied to embeddings in the hyperbolic space, thus avoiding a frequent back and forth mapping.

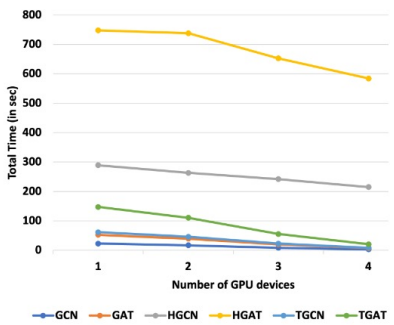
Choudhary. N. et.al . Towards Scalable Hyperbolic Neural Networks using Taylor Series Approximations. arXiv 2022.

Hyperbolic Neural Networks using Taylor Series Approximations

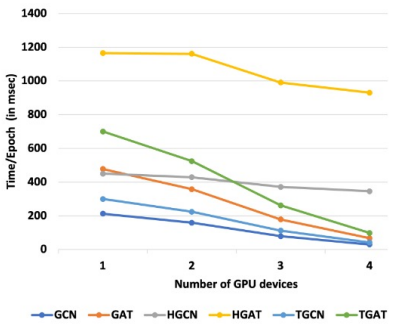
Models	Node Classification Accuracy (%) Cora	Pubmed	Citeseer	Link Prediction ROC in (%) Cora	Pubmed	Citeseer
GCN	80.1	78.5	71.4	90.2	92.6	91.3
GAT	82.7	79.0	**71.6**	89.6	92.4	**93.6**
HGCN	77.9	78.9	69.6	91.4	**95.0**	92.8
HGAT	79.6	79.2	68.1	90.8	93.9	92.2
TGCN	77.6	80.8	68.8	**91.6**	94.2	92.3
TGAT	**81.7**	**81.3**	69.9	91.2	94.4	92.7



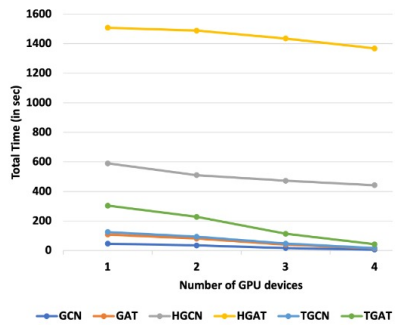
(a) Time/Epoch (vs) # GPU for Node classification.



(b) Total time (vs) # GPU for Node classification.



(c) Time/Epoch (vs) # GPU for Link prediction.



(d) Total time (vs) # GPU for Link prediction.

Choudhary. N. et.al . Towards Scalable Hyperbolic Neural Networks using Taylor Series Approximations. arXiv 2022.

Summary & Opportunities

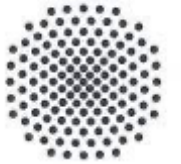
- **Summary**
  - Core components of hyperbolic graph neural networks
  - Applications in recommendation systems, knowledge graphs & healthcare
  - Challenges and methods beyond hyperbolic spaces
- **Open challenges & Opportunities**
  - Scalability and trustworthy
  - Curvature & geometry help over-squashing and bottlenecks
  - Generalization capability in long-tailed settings
  - Foundation models (LLMs) in hyperbolic space
  - …

Resource

- Survey
  - Hyperbolic Graph Neural Networks: A Review of Methods and Application
  - Hyperbolic Deep Neural Networks: A Survey
- Tools
  - Geoopt: https://github.com/geoopt/geoopt
  - PyG: https://github.com/pyg-team/pytorch_geometric
  - https://github.com/alibaba/Curvature-Learning-Framework
- Related Tutorial
  - https://www.youtube.com/watch?v=MdPk3qD4Wig&ab_channel=HazyResearch
  - https://nurendra.me/hyperbolic-networks-tutorial/www-2022/
  - https://hyperbolicgraphlearning.github.io/
  - https://sites.google.com/view/hyperbolic-tutorial-eccv22/homepage
- GitHub Repos
  - https://github.com/marlin-codes/Awesome-Hyperbolic-Learning
  - https://github.com/xiaoiker/Awesome-Hyperbolic-NeuralNetworks

Acknowledgement

imprs-is



Universität Stuttgart

Slack channel

- Join our slack channel for Q&A

https://hyperbolicgnn.github.io/

Thanks !