Yale

Hyperbolic Embeddings and Hyperbolic GNNs

CPSC483: Deep Learning on Graph-Structured Data

Rex Ying

Readings

- Readings are updated on the website (syllabus page)
- **Lecture 18 readings:**
  - TransE: Translating Embeddings for Modeling Multi-relational Data
  - RotatE: Knowledge Graph Embedding by Relational Rotation in Complex Space
  - TuckER: Tensor Factorization for Knowledge Graph Completion
- **Lecture 19 readings:**
  - HGCN: Hyperbolic Graph Convolutional Neural Networks
  - Hyperbolic GNN survey

Content

- **Non-Euclidean Space**
- **Hyperbolic Embeddings**
- **Hyperbolic GNNs**

Content

- **Non-Euclidean Space**

- **Hyperbolic Embeddings**

- **Hyperbolic GNNs**

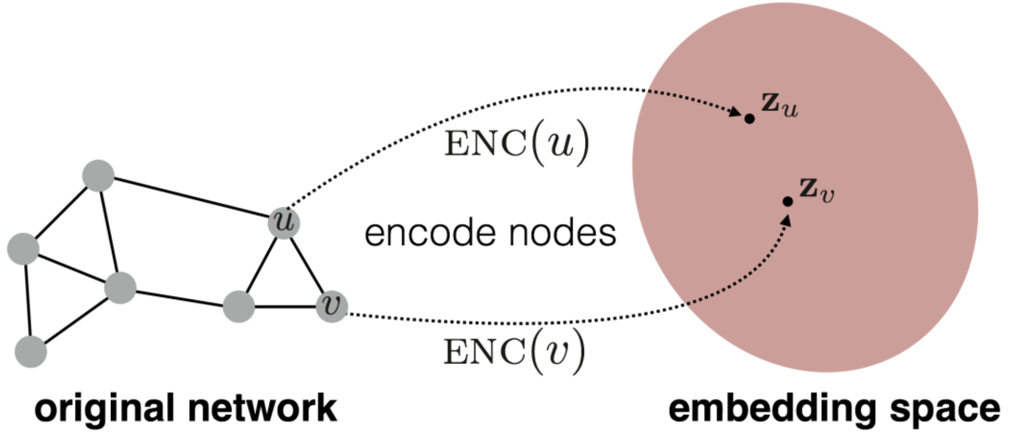
Recap: Graph Representation Learning

- **Step 1**: **Obtain node and edge features**, possibly augment them with structural properties of the input graphs

- **Step 2**: Use a parameterized **encoder** to map nodes to an embedding space

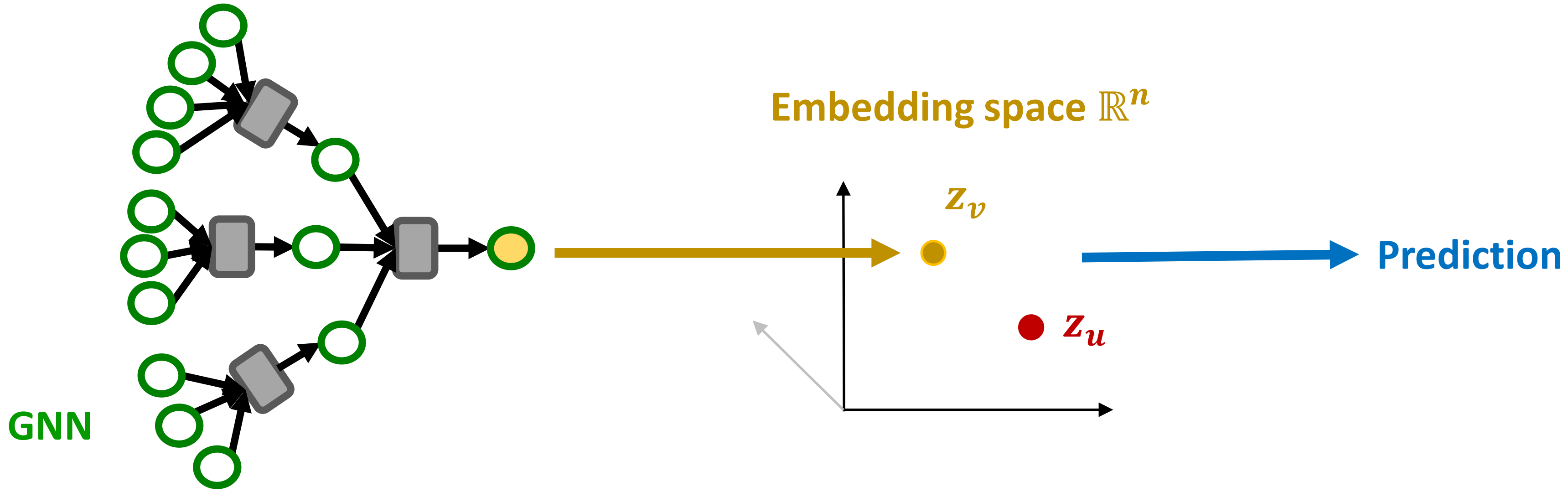
- **Step 3**: Make **predictions** on nodes/edges/graphs **based on embeddings**

- **Step 4**: Compute **loss** and **optimize** the parameters

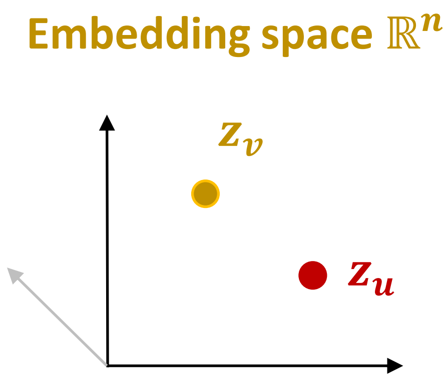


Architecture vs. Embedding Geometry

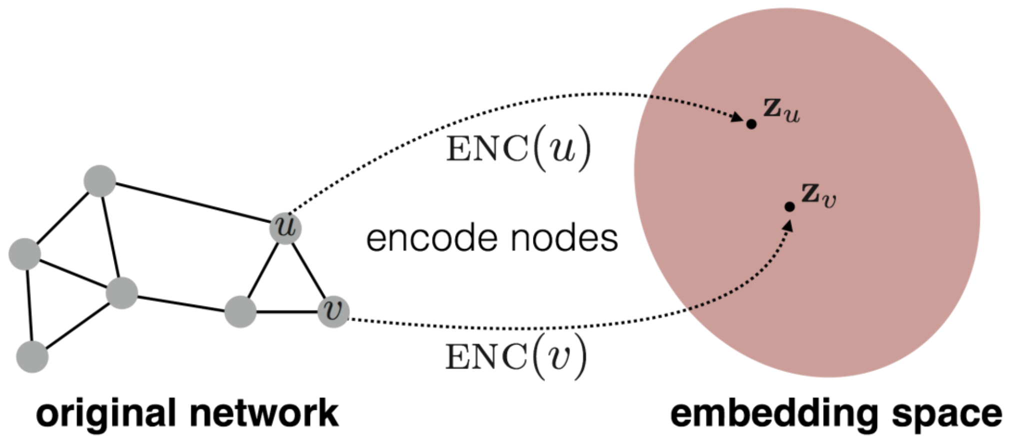
- **Architecture** and **embedding geometry** are both crucial to the expressive power of a neural network
- Embedding geometry is closely related to the objective function
- Better embedding geometry can benefit a variety of architectures



Question



Euclidean Space can not always capture complex graph structures

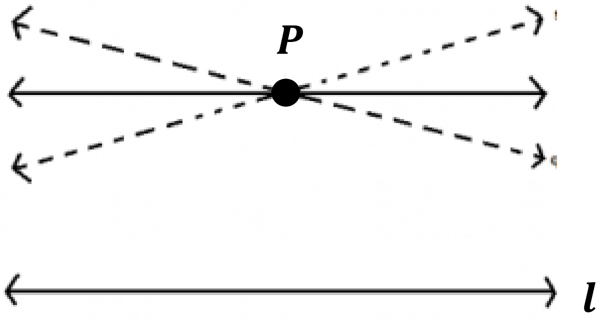


What embedding space geometry is optimal for data?

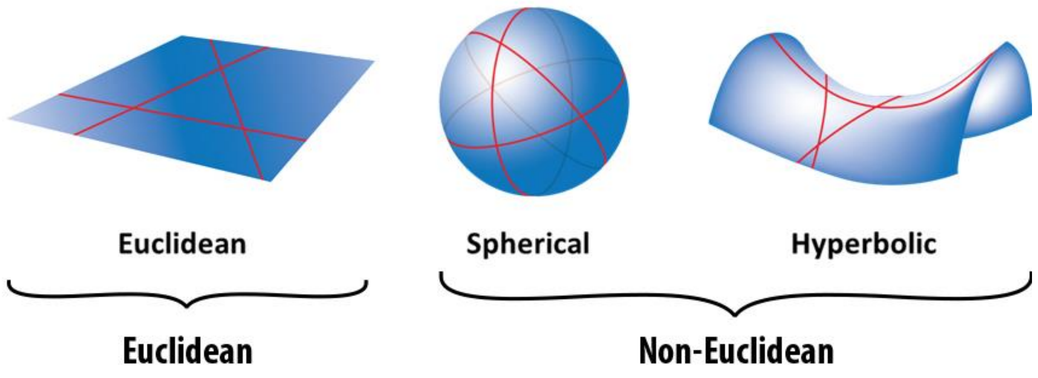
Consider non-euclidean space!

Non-Euclidean Space

- **Euclidean Space** (satisfy the **fifth parallel postulate of Euclidean Geometry**)
  - Given a line and a point not on it, **exactly one line parallel** to the given line can be drawn through the given point.
- **Non-Euclidean Space:**
  - **Hyperbolic**: negative curvature, infinitely many parallel lines (curve away from each other)
  - **Spherical**: positive curvature, no parallel lines

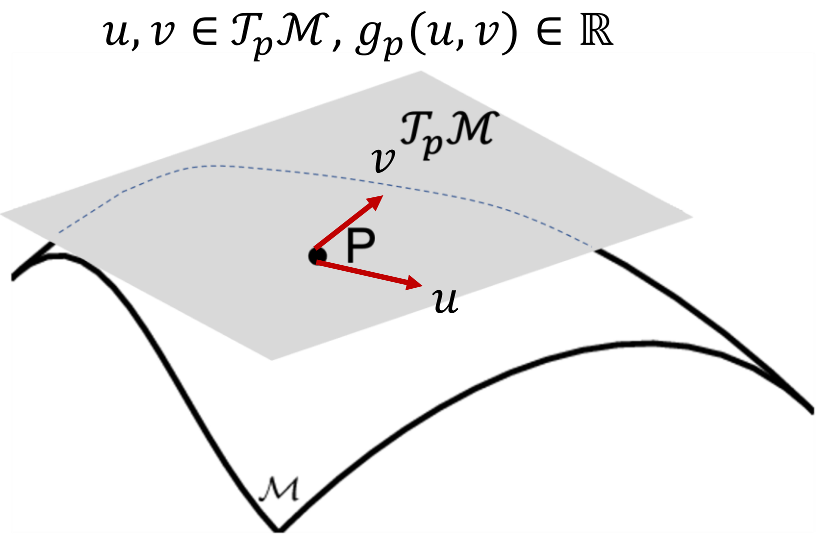


In Euclidean Space, only one line through P that is parallel to l



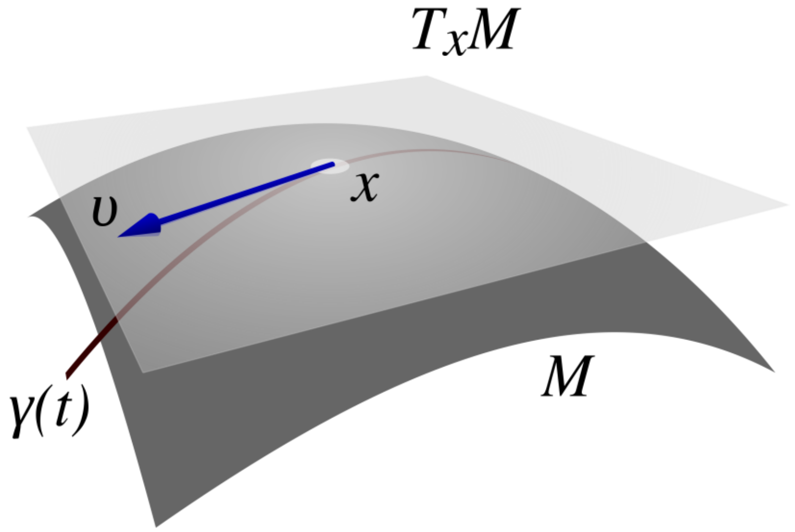
Riemannian Manifold

- **Manifold**: high-dimensional surface
- **Riemannian Manifold $\mathcal{M}$**
  - Equipped with
    - *Tangent space* $\mathcal{T}_p\mathcal{M}$: an $\mathbb{R}^d$ that approximates the manifold at any point $p \in \mathcal{M}$
    - *Inner product* g_p: $\mathcal{T}_p\mathcal{M} \times \mathcal{T}_p\mathcal{M} \rightarrow \mathbb{R}$
  - Both functions vary smoothly (differentiable) on the manifold



Tangent Space

- **Curve:** smooth path along manifold $\gamma: [0,1] \rightarrow \mathcal{M}$
- **Speed:** direction of change along the curve $\dot{\gamma}: [0,1] \rightarrow \mathcal{T}_x\mathcal{M}$
- **Tangent space $\mathcal{T}_x\mathcal{M}$:** space of **speed vectors $\boldsymbol{v}$** of all curves γ that **go through point $\boldsymbol{x}$** on the manifold $\mathcal{M}$

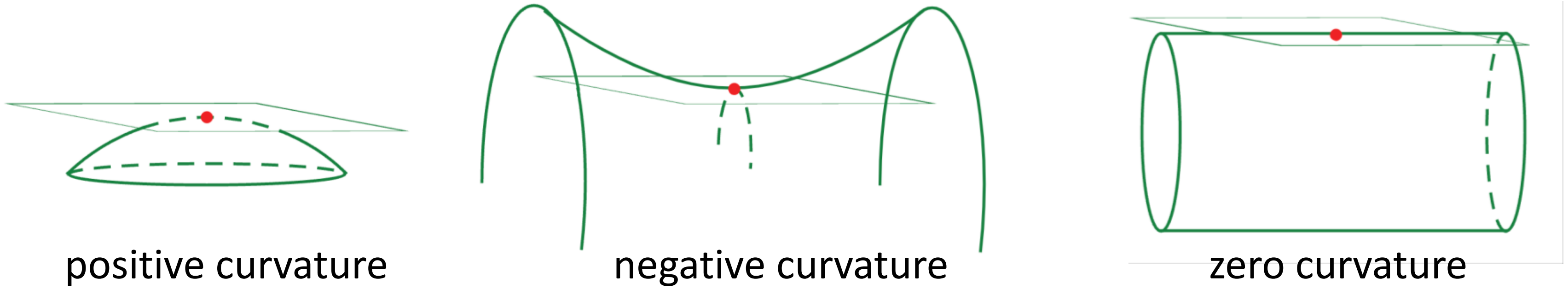


Curvature

- The **curvature** (sectional curvature) at a point measures how drastically a surface **bends away** from its tangent plane at this point

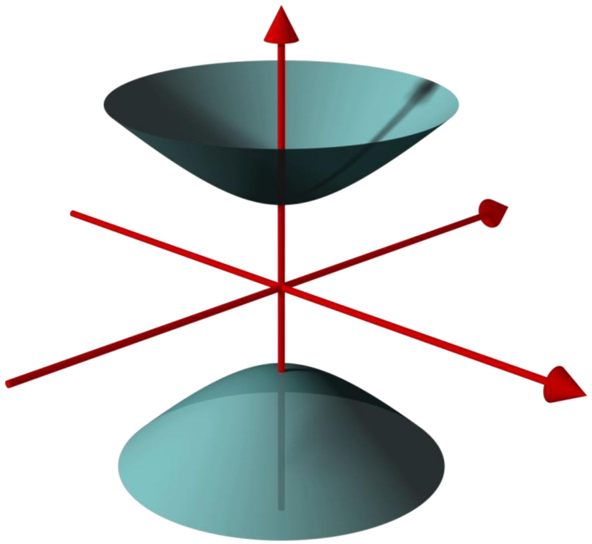
High-level Intuition:

- If the surface locally lives **entirely on one side** of the tangent space $\mathcal{T}_p\mathcal{M}$ ⇒ **Positive** curvature at point p
- If the tangent space $\mathcal{T}_p\mathcal{M}$ **cuts through** the surface ⇒ **Negative** curvature at point p
- If the surface has a line along which the **surface agrees with the tangent space** $\mathcal{T}_p\mathcal{M}$ ⇒ **Zero** curvature at point p

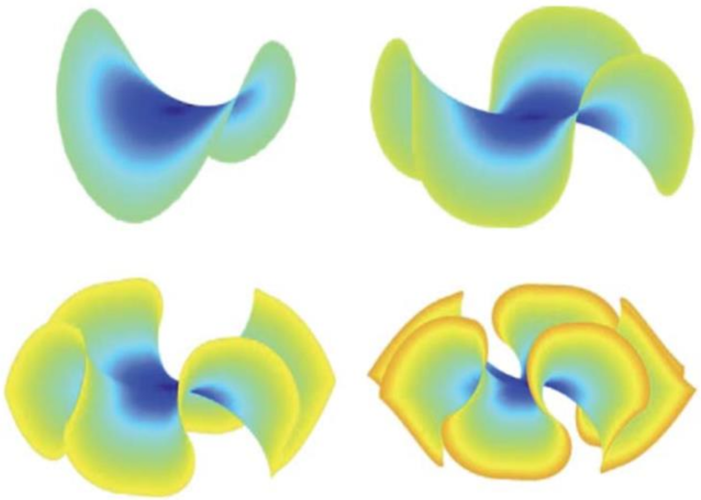


Hyperbolic Space

- **Hyperbolic space** is a Riemannian manifold with constant negative curvature $-1/K$, where $(K > 0)$
  - Becomes Euclidean when $K \to \infty$
- In **Euclidean space**, we can also find manifolds with constant negative curvature:



two sheet hyperboloid (source: Wikipedia)



[Periodic Amsler Surfaces](#)

Hyperbolic Space and Minkowski Space

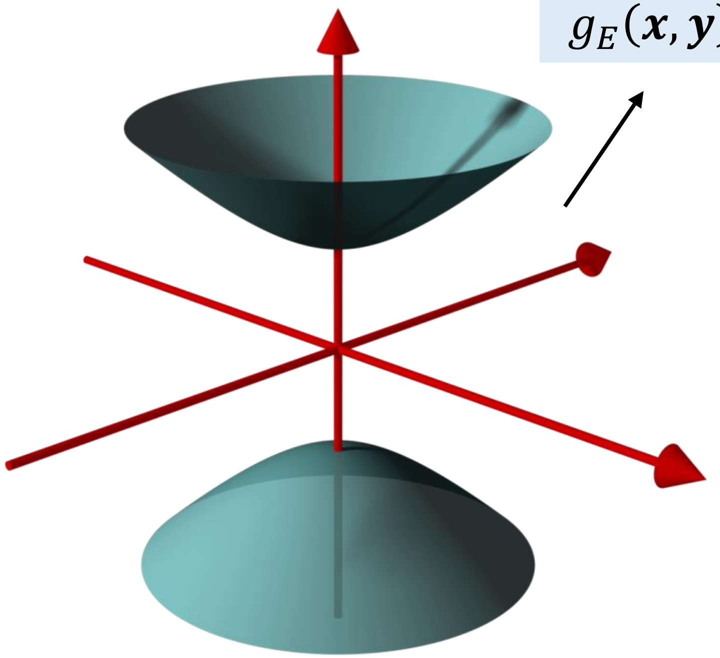
Hilbert's Theorem (1901): There exists no hyperbolic surface with constant negative curvature that can be embedded into $\mathbb{R}^3$.

- However, we can embed hyperbolic geometry into **Minkowski Space**
- The **Minkowski metric** in the Minkowski space is different from the Euclidean metric.
  - **Euclidean Metric:** $g_E(\boldsymbol{u}, \boldsymbol{v}) = u_1 v_1 + u_2 v_2 + \cdots + u_d v_d$
  - **Minkowski Metric:** $g_M(\boldsymbol{u}, \boldsymbol{v}) = \pm(u_1 v_1 - u_2 v_2 - \cdots - u_d v_d)$
  - Note: dimension 1 is treated differently in Minkowski Space.

Hyperboloid in Different Spaces

Euclidean Metric

$g_E(\boldsymbol{x}, \boldsymbol{y}) = x_1 y_1 + x_2 y_2 + x_3 y_3$



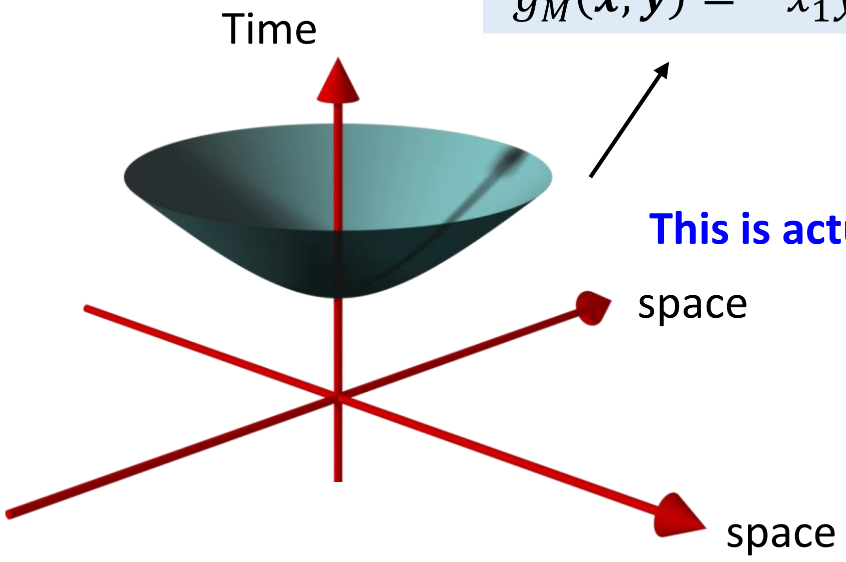
two sheet hyperboloid in **3D Euclidean space**

Distance in Euclidean: $d_E(\boldsymbol{x}, \boldsymbol{y})=\sqrt{2(1 - g_E(\boldsymbol{x}, \boldsymbol{y}))}$
(with normalized $\boldsymbol{x}$ and $\boldsymbol{y}$)

Minkowski Metric

$g_M(\boldsymbol{x}, \boldsymbol{y}) = -x_1 y_1 + x_2 y_2 + x_3 y_3$

This is actually hyperbolic!



2D Hyperboloid model in **3D Minkowski space**

Distance in Minkowski: $D_M^K(\boldsymbol{x}, \boldsymbol{y}) = \sqrt{K} \operatorname{arcosh}(-\frac{g_M(\boldsymbol{x}, \boldsymbol{y})}{K})$

Inner Product

- **Hyperboloid model** as a Riemannian manifold:
  - With Constant **Minkowski metric**:

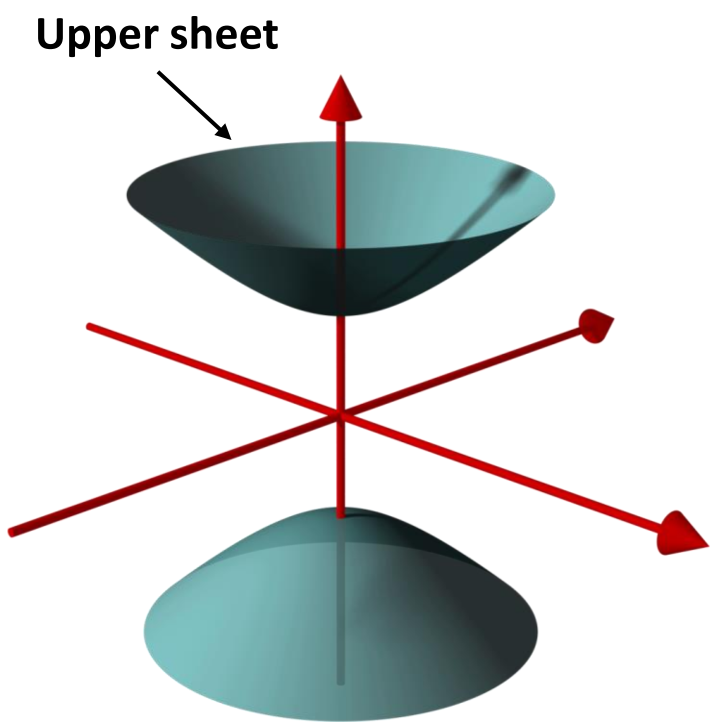
$$\langle .,. \rangle_{\mathcal{L}} : \mathbb{R}^{d+1} \times \mathbb{R}^{d+1} \to \mathbb{R}$$

$$\langle \boldsymbol{x}, \boldsymbol{y} \rangle_{\mathcal{L}} = \underbrace{\boxed{-x_0 y_0}}_{\text{Time-like}} + \underbrace{\boxed{x_1 y_1 + \ \ldots + x_d y_d}}_{\text{Space-like}}$$

  - **Hyperboloid model** $\mathbb{H}^{d,K} = \{\boldsymbol{x} \in \mathbb{R}^{d+1} : \langle \boldsymbol{x}, \boldsymbol{x} \rangle_{\mathcal{L}} = -K\}$, $-\frac{1}{K}$ is the curvature
  - Note: the points in hyperboloid model $\mathbb{H}^{d,K}$ are represented in $(d+1)$-dimensional Minkowski space.
  - The metric of hyperboloid model is different from the Euclidean metric!

Hyperboloid Model

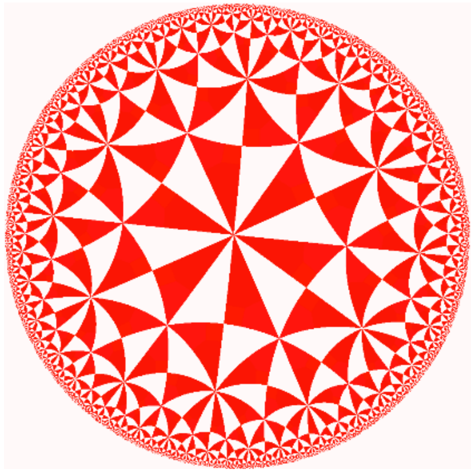
- **Hyperboloid Model** (Lorentz Model)
  - Upper sheet of 2-sheet hyperboloid
  - d-dimensional Hyperboloid can be represented in $(d + 1)$-dimensional Minkowski space
  - Subset of Euclidean space
  - Numerically more stable, simpler formula



two sheet hyperboloid

Poincaré Model

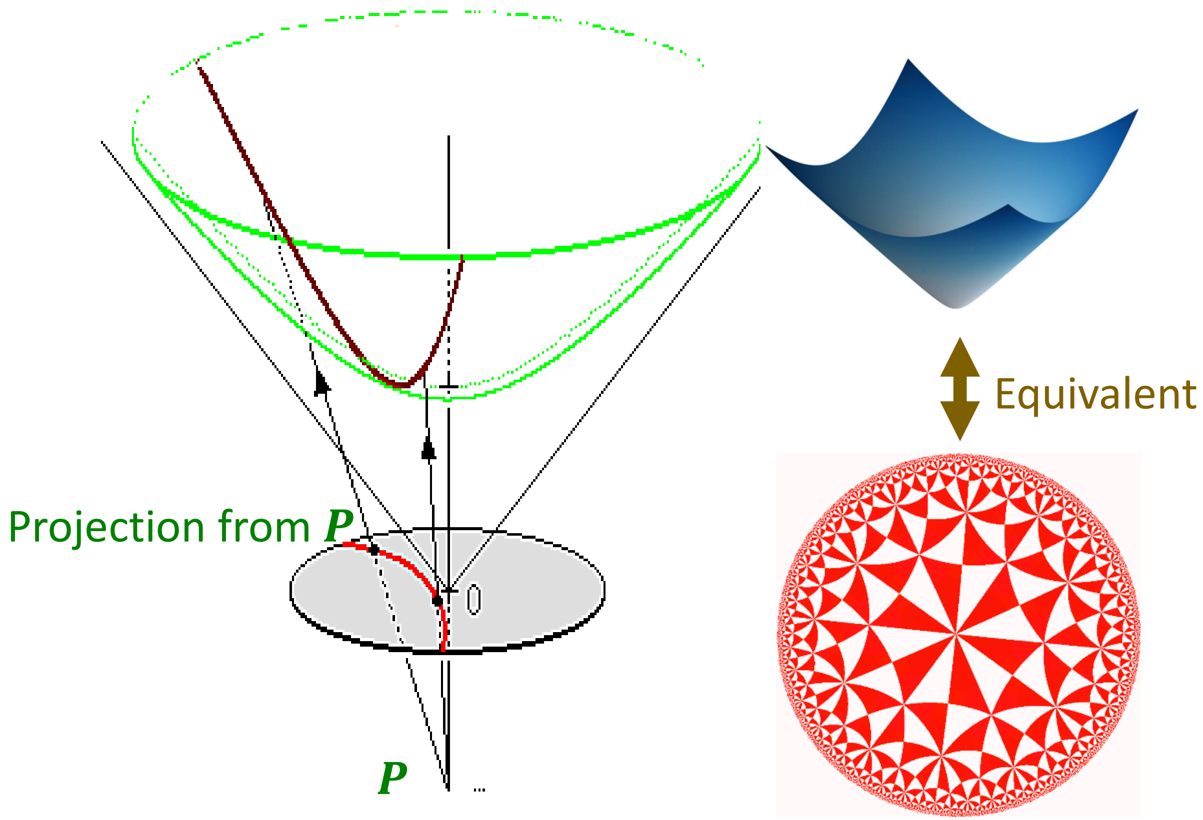
- **Poincaré Model**
  - Radius proportional to $\sqrt{K}$ ($-\frac{1}{K}$ is the curvature)
  - Open ball (exclude boundary)
  - Each triangle in the figure has the **same** area
  - **Exponentially many triangles** with the same area towards the boundary of Poincaré Ball



Poincaré: intuitive visualization

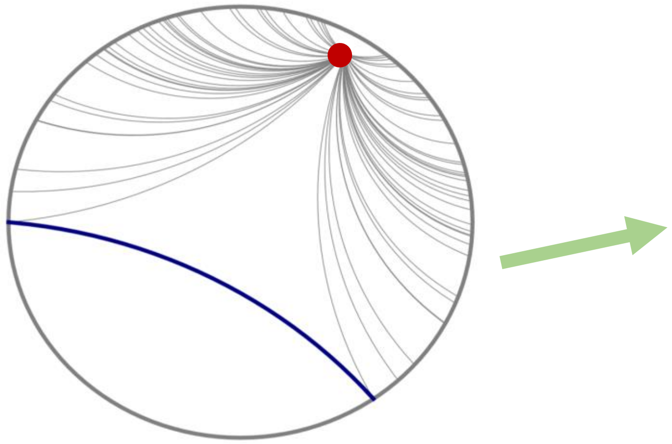
Equivalence

- d-dimensional Poincaré model and $(d+1)$-dimensional hyperboloid model are **equivalent**!
- 2d Poincaré model can be derived using a **projection** of 3d hyperboloid model through a specific point onto the unit circle of the $z = 0$ plane.



Geodesic

- **Geodesic**: shortest path in manifold
  - Analogous to straight lines in $\mathbb{R}^n$
  - Curved in hyperbolic space
- Geodesics visualization in Poincaré model: curved!



Set of geodesic lines from the red point to boundary of the Poincare ball that are parallel to the blue line

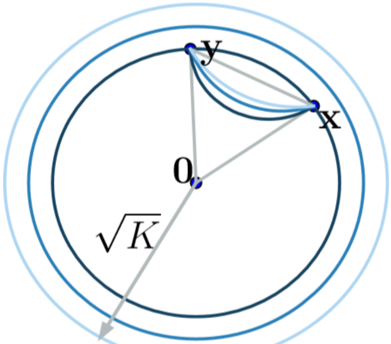
Geodesic Distance

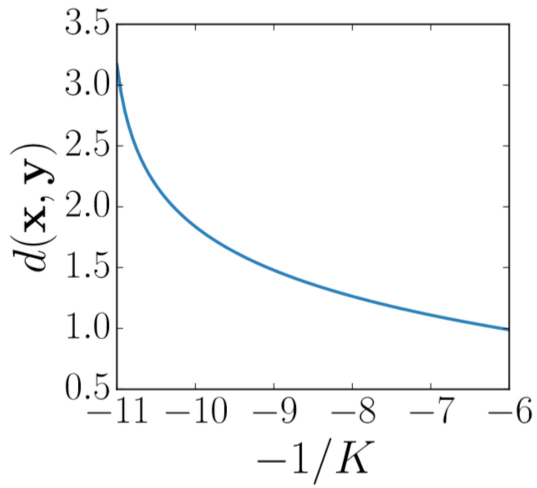
- **Geodesic distance** between $\boldsymbol{x}$ and $\boldsymbol{y}$ for $\mathbb{H}^{d,\mathrm{K}}$:

$$D_{\mathcal{L}}^{K}(\boldsymbol{x},\boldsymbol{y}) = \sqrt{K}\operatorname{arcosh}\left(-\frac{\langle \boldsymbol{x},\boldsymbol{y}\rangle_{\mathcal{L}}}{K}\right)$$

- The more negative the curvature:
  - the more geodesics bends inward
  - geodesic distance increases

$$\operatorname{arcosh}(x) = \ln(x + \sqrt{x^2 + 1})$$

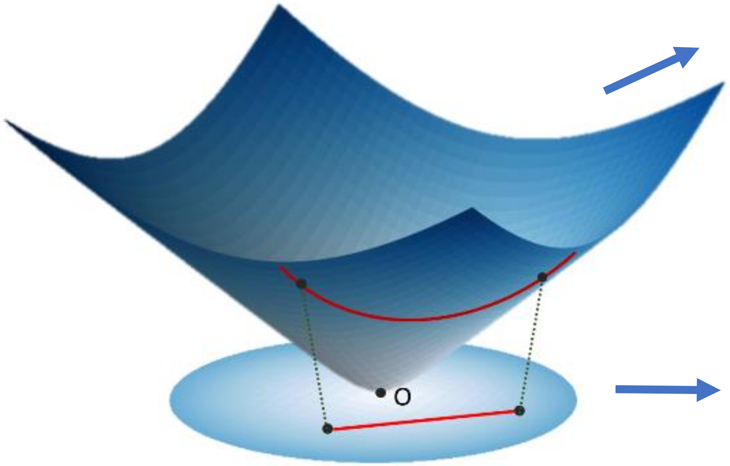




Dark blue: high curvature boundary and geodesics
Light blue: low curvature boundary and geodesics

Tangent Space

- Tangent space expression under **hyperboloid model** $\mathbb{H}^{\mathrm{d,K}}$ at point $\boldsymbol{x}$:
  - $\mathcal{T}_{\boldsymbol{x}}\mathbb{H}^{\mathrm{d,K}} = \{\boldsymbol{v} \in \mathbb{R}^{\mathrm{d+1}}: \langle \boldsymbol{v}, \boldsymbol{x} \rangle_{\mathcal{L}} = 0\}$
- A vector space (linear structure) with the same dimension as the hyperboloid model
- The best linear approximation to the manifold $\mathbb{H}^{\mathrm{d,K}}$ at point $\boldsymbol{x}$



Hyperboloid model

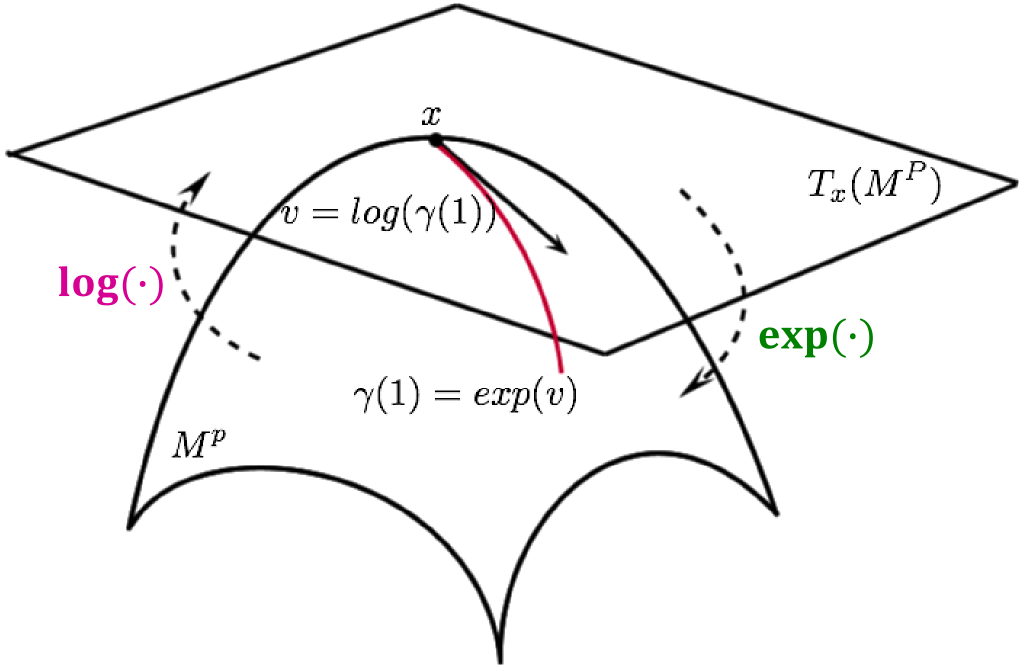
$\boldsymbol{d}$: hyperbolic space dimension

$\boldsymbol{K}$: negative inverse of curvature

Tangent space at north pole o

Mapping to and from Tangent Space

- **Exponential map**: $\mathcal{T}_{\boldsymbol{x}}\mathbb{H}^{\mathrm{d,K}} \rightarrow \mathbb{H}^{\mathrm{d,K}}$
  - from tangent space (Euclidean) to manifold
- **Logarithmic map**: $\mathbb{H}^{\mathrm{d,K}} \rightarrow \mathcal{T}_{\boldsymbol{x}}\mathbb{H}^{\mathrm{d,K}}$
  - from manifold to tangent space
  - inverse operation of exponential map

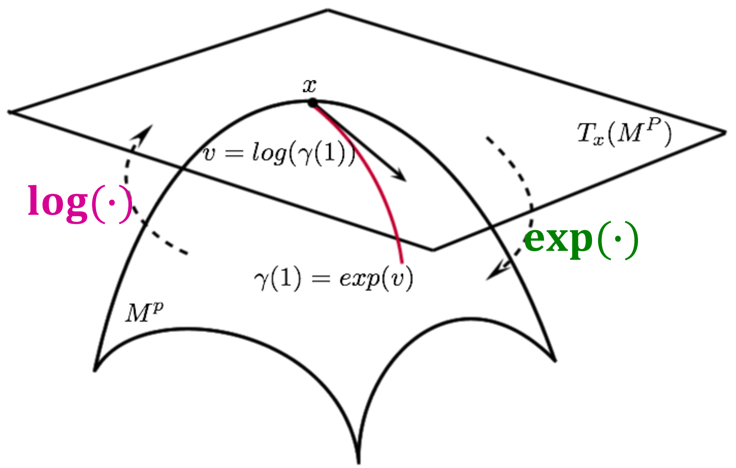


Exponential Map:

- For **hyperboloid model** $\mathbb{H}^{\mathrm{d,K}} = \{\boldsymbol{x} \in \mathbb{R}^{d+1}\colon \langle \boldsymbol{x}, \boldsymbol{x} \rangle_{\mathcal{L}} = -K\}$ at point $\boldsymbol{x}$
- **Exponential Map:**

$$\exp_{\boldsymbol{x}}^{K}(\boldsymbol{v}) = \cosh\left(\frac{\|\boldsymbol{v}\|_{\mathcal{L}}}{\sqrt{K}}\right)\boldsymbol{x} + \sqrt{K}\sinh\left(\frac{\|\boldsymbol{v}\|_{\mathcal{L}}}{\sqrt{K}}\right)\frac{\boldsymbol{v}}{\|\boldsymbol{v}\|_{\mathcal{L}}}$$

  - $\boldsymbol{v} \in \mathcal{T}_{\boldsymbol{x}}\mathbb{H}^{\mathrm{d,K}}$
  - $\cosh(x) = \frac{e^{x}+e^{-x}}{2}$, $\sinh(x) = \frac{e^{x}-e^{-x}}{2}$
  - $\|\boldsymbol{v}\|_{\mathcal{L}} = \langle \boldsymbol{v}, \boldsymbol{v} \rangle_{\mathcal{L}}$

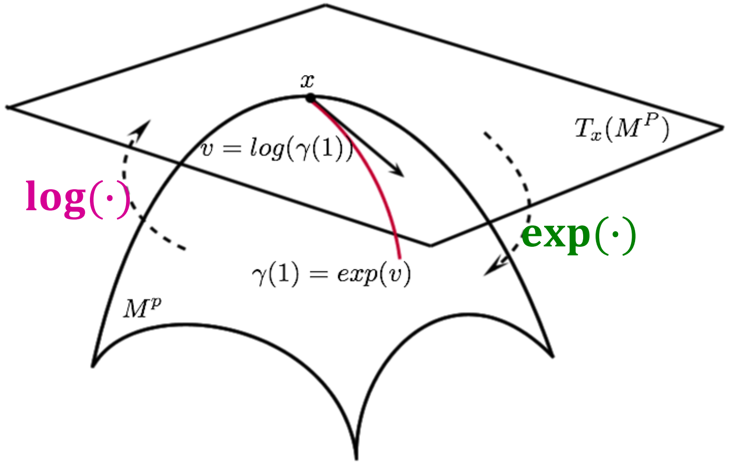


Logarithmic Map

- For **hyperboloid model** $\mathbb{H}^{\mathrm{d,K}} = \{\boldsymbol{x} \in \mathbb{R}^{d+1}: \langle \boldsymbol{x}, \boldsymbol{x} \rangle_{\mathcal{L}} = -K\}$ at point $\boldsymbol{x}$
- **Logarithmic map**:

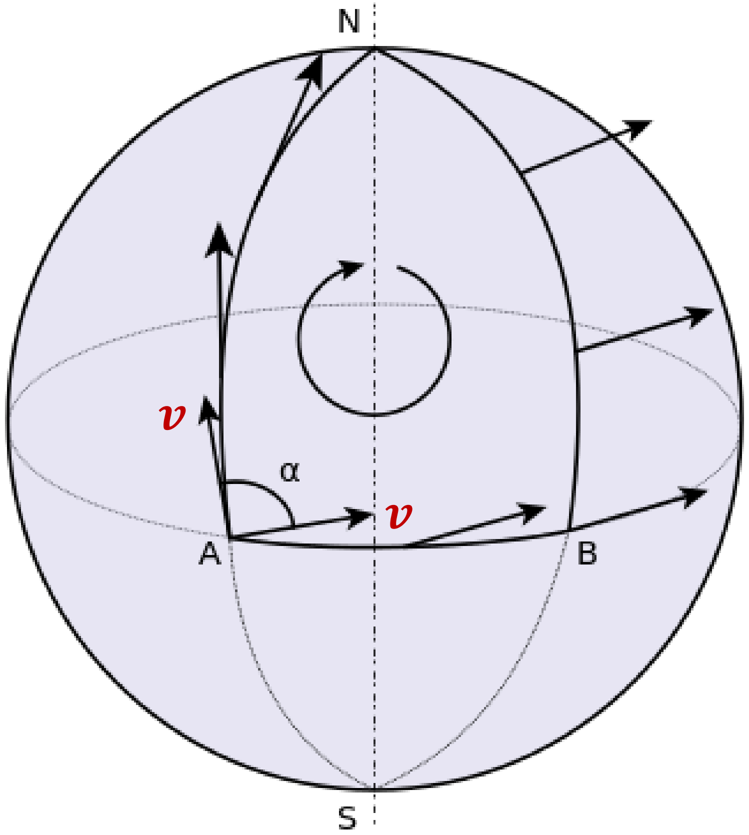
$$\log_{\boldsymbol{x}}^{K} \boldsymbol{y} = D_{\mathcal{L}}^{K}(\boldsymbol{x}, \boldsymbol{y}) \frac{\boldsymbol{y} + \frac{1}{K}\langle \boldsymbol{x}, \boldsymbol{y} \rangle_{\mathcal{L}} \boldsymbol{x}}{\left\| \boldsymbol{y} + \frac{1}{K}\langle \boldsymbol{x}, \boldsymbol{y} \rangle_{\mathcal{L}} \boldsymbol{x} \right\|_{\mathcal{L}}}$$

  - $\boldsymbol{y} \in \mathbb{H}^{\mathrm{d,K}}$
  - $D_{\mathcal{L}}^{K}(\boldsymbol{x}, \boldsymbol{y}) = \sqrt{K} \operatorname{arcosh}\left(-\frac{\langle \boldsymbol{x}, \boldsymbol{y} \rangle_{\mathcal{L}}}{K}\right)$ is geodesic distance



Parallel Transport (1)

- **Parallel Transport:** transport a vector along a smooth curve on the surface and keep parallel to itself locally.



Transport a tangent vector $\boldsymbol{v}$ along the surface with non-zero curvature. When travelling from A to N to B back to A, the direction of the vector $\boldsymbol{v}$ changes!

Parallel Transport (2)

- Parallel Transport $P_{\boldsymbol{x}\to\boldsymbol{y}}(\cdot)$ maps a vector $\boldsymbol{v} \in \mathcal{T}_{\boldsymbol{x}}\mathcal{M}$ to $P_{\boldsymbol{x}\to\boldsymbol{y}}(\boldsymbol{v}) \in \mathcal{T}_{\boldsymbol{y}}\mathcal{M}$
- If two points $\boldsymbol{x}$ and $\boldsymbol{y}$ on the hyperboloid $\mathbb{H}^{\mathrm{d,K}}$ are connected by a geodesic, then the parallel transport of tangent vector $\boldsymbol{v} \in \mathcal{T}_{\boldsymbol{x}}\mathbb{H}^{\mathrm{d,K}}$ to $\mathcal{T}_{\boldsymbol{y}}\mathbb{H}^{\mathrm{d,K}}$:

$$\boxed{P_{\boldsymbol{x}\to\boldsymbol{y}}(\boldsymbol{v}) = \boldsymbol{v} - \frac{\langle \log_{\boldsymbol{x}}^{K}(\boldsymbol{y}), \boldsymbol{v}\rangle_{\mathcal{L}}}{D_{\mathcal{L}}^{K}(\boldsymbol{x}, \boldsymbol{y})^2}\left(\log_{\boldsymbol{x}}^{\mathrm{K}} \boldsymbol{y} + \log_{\boldsymbol{y}}^{\mathrm{K}} \boldsymbol{x}\right)}$$

  - $\log_{\boldsymbol{x}}^{\mathrm{K}}$ is the **Logarithmic map** at point x.
  - $D_{\mathcal{L}}^{K}(\boldsymbol{x}, \boldsymbol{y}) = \sqrt{K}\operatorname{arcosh}\left(-\frac{\langle \boldsymbol{x}, \boldsymbol{y}\rangle_{\mathcal{L}}}{K}\right)$ is geodesic distance

Content

- Non-Euclidean Space
- **Hyperbolic Embeddings**
- Hyperbolic GNNs

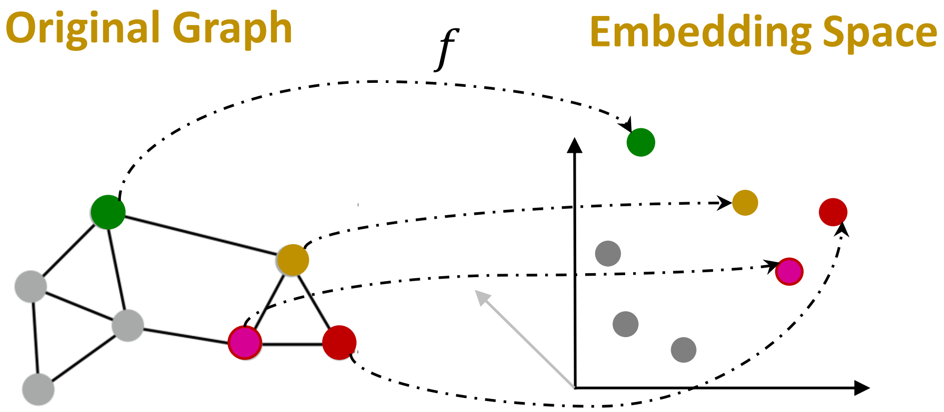
Optimal Embedding

Given a graph $G(V, E)$. Mapping $f: V \rightarrow W$, with distances d_V and d_W

How to measure the **quality of embedding**?

High-level Intuition:

- Consider node $i \in V$, the embeddings of neighbor node in $\mathcal{N}(i)$ should be close to $f(i)$ in the embedding space W
- Distances between embedding vectors $f(i)$ and $f(j)$ in the embedding space W should be close to the distance in original graph G
  - Recall Position-aware GNNs (lecture 10)

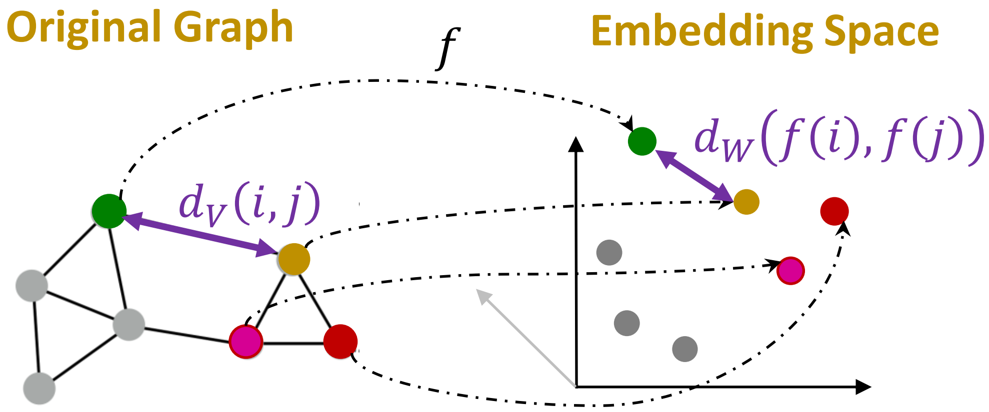


Distance Distortion

- **Distance Distortion:**

$$D(f) = \frac{1}{C_n^2} \left(\sum_{i,j \in V, i \neq j} \frac{\left| d_W\big(f(i), f(j)\big) - d_V(i,j) \right|}{d_V(i,j)}\right)$$

- $C_n^2 = \frac{\mathrm{n(n-1)}}{2}$
- The lower distortion, the better embedding
- The best distortion is $D(f) = 0$, preserving the distances between node pairs exactly

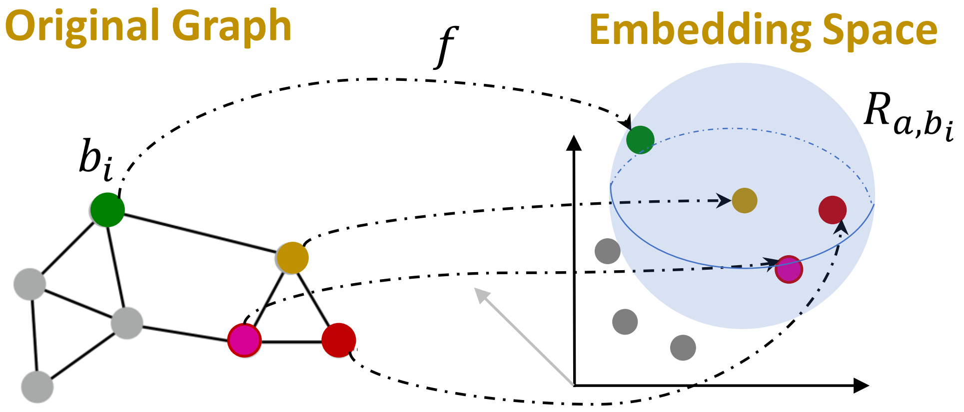


Mean Average Precision

- **Mean Average Precision (mAP)**

$$mAP(f) = \frac{1}{|V|}\sum_{a\in V}\frac{1}{\deg(a)}\sum_{b_i\in\mathcal{N}_a}\frac{|\mathcal{N}_a \cap R_{a,b_i}|}{|R_{a,b_i}|}$$

  - V is the node set, $\deg(a)$ denotes the degree of node a
  - $\mathcal{N}_a$ denotes the 1-hop neighbor nodes of a
  - R_{a,b_i} is the set of nodes whose embeddings fall into the smallest ball centered at the embedding of a, that can retrieve b_i
  - Used here at page 3
- The larger MAP, the better embedding.
- $MAP(f) \leq 1$
- Note: we do not consider node features here



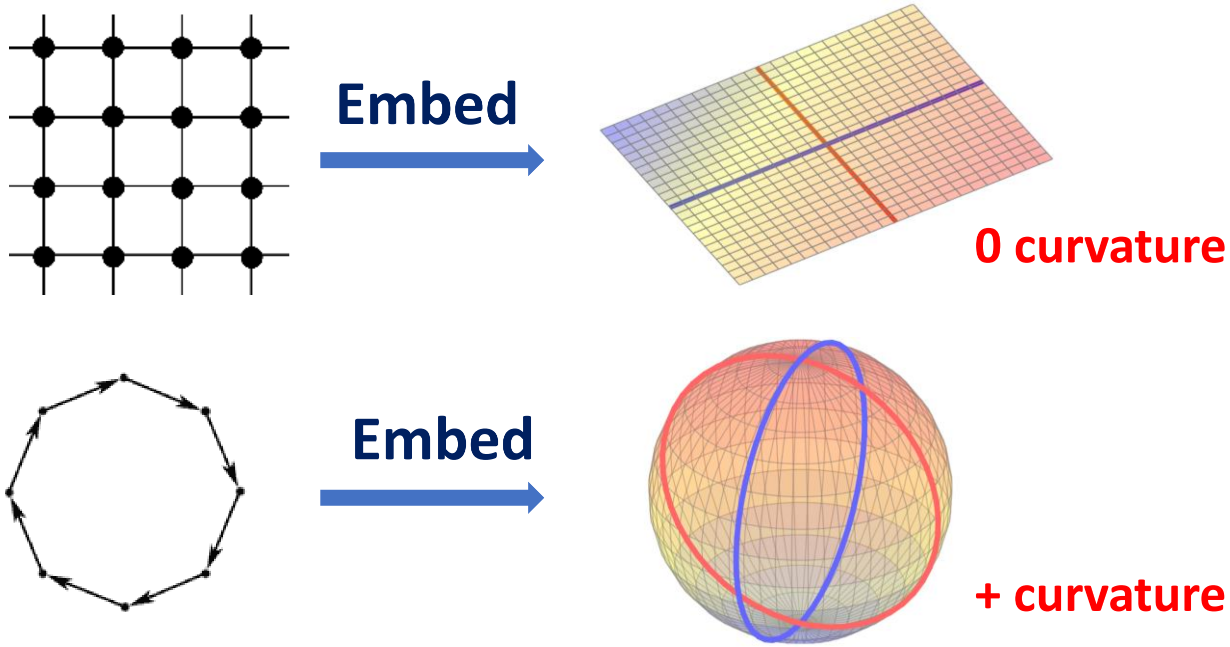
Distortion vs. mAP

- **mAP** is a local measurement which does not depend on an explicit distance
  - It combines the effect of **precision** and **recall** when performing link prediction task

- **Distortion** is a global metric and helps to preserves the explicit value of distances
  - It can be useful in applications where we need to approximate more complex distances than link prediction (which can be viewed as a binary version of distance)
  - Examples: graph / sequence edit distance, shortest path distance, transportation distance (e.g., Google Map) …

Graph with Grids and Cycles

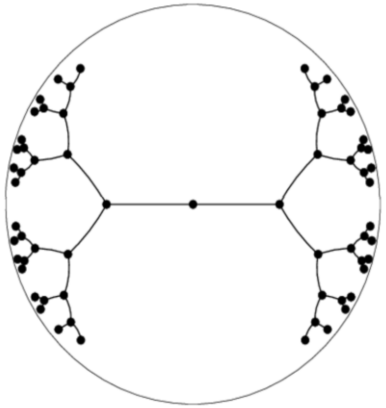
- **Euclidean** space preserves neighbor nodes and distances for **grid-like** graphs
- **Spherical** space preserves neighbor nodes and distances for **cycle-like** graphs
- $MAP(f) = 1, D(f) = 0$



Graph with Grids and Cycles

- **Euclidean** space preserves neighbor nodes and distances for **grid-like** graphs
- **Spherical** space preserves neighbor nodes and distances for **cycle-like** graphs
- $MAP(f) = 1, D(f) = 0$

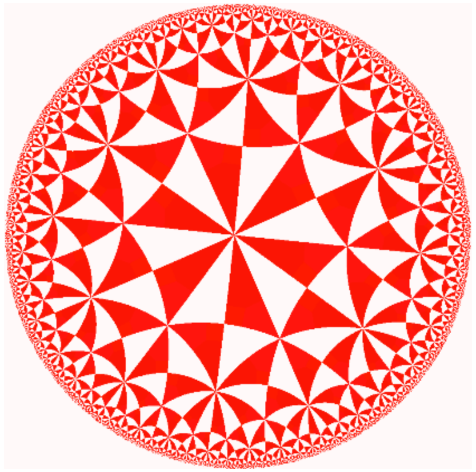
What about Tree-like graphs?



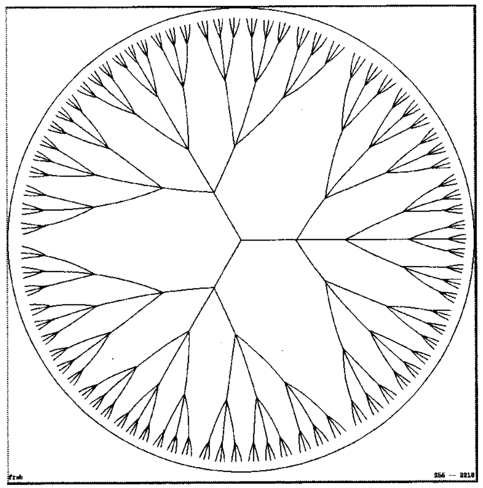
Hyperbolic Geometry!

Exponential volume growth

- The **volume** of d-dimensional Euclidean Ball with radius r:
$$V_d^E(r) \propto r^d$$
- In a tree, the number of nodes grows exponentially with the tree depth
- The volume of a **Poincaré model** in the hyperbolic space grows exponentially with its radius!
$$V_2^H(r) \propto e^r$$



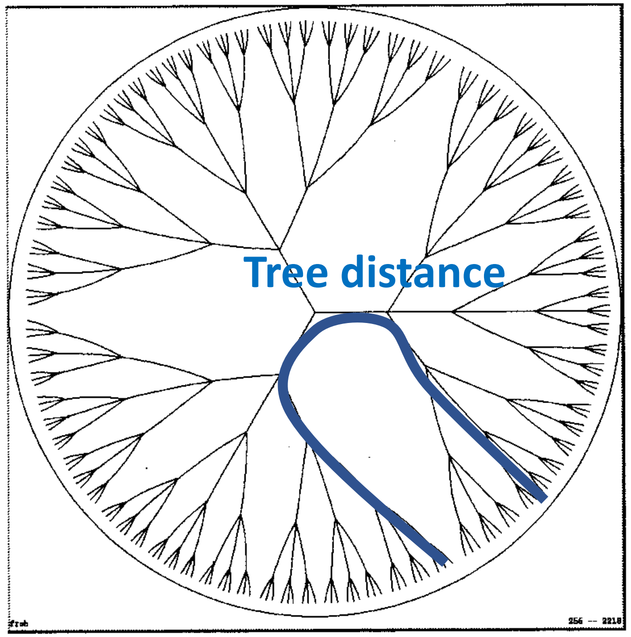
Poincaré: intuitive visualization

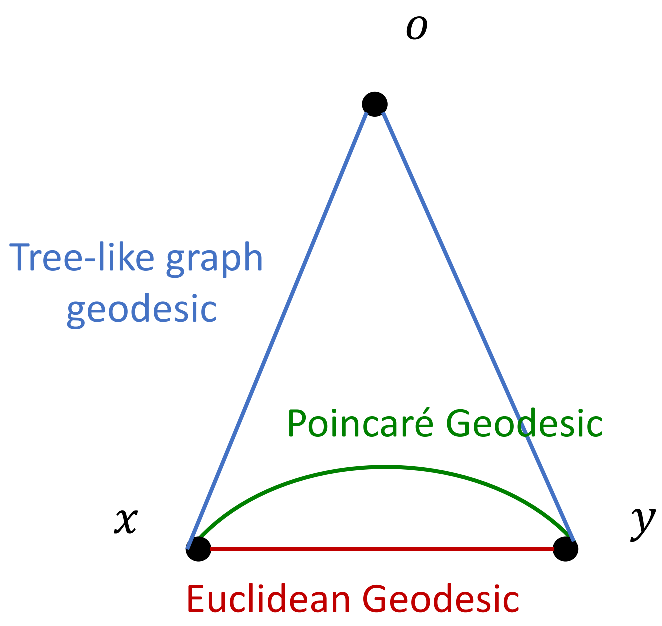


a hierarchical tree

Lower Distance Distortion

- **In Poincaré model, geodesic bends inwards**
- similar to trees: shortest path go through the LCA (lowest common ancestor)





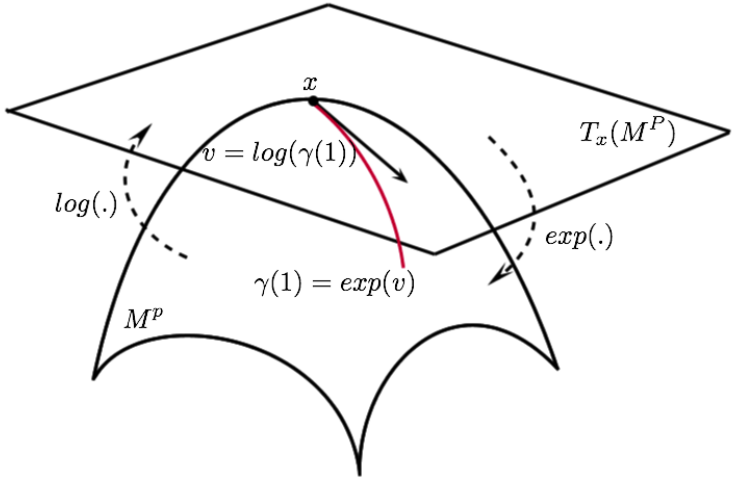
Content

- **Non-Euclidean Space**
- **Hyperbolic Embeddings**
- **Hyperbolic GNNs**

Challenges in Hyperbolic GNN

Challenges:

- Input node features are usually **Euclidean**
- Perform **hyperbolic aggregation** for message passing
- Choose hyperbolic spaces with the **right amount of curvature** at every layer of the GNN

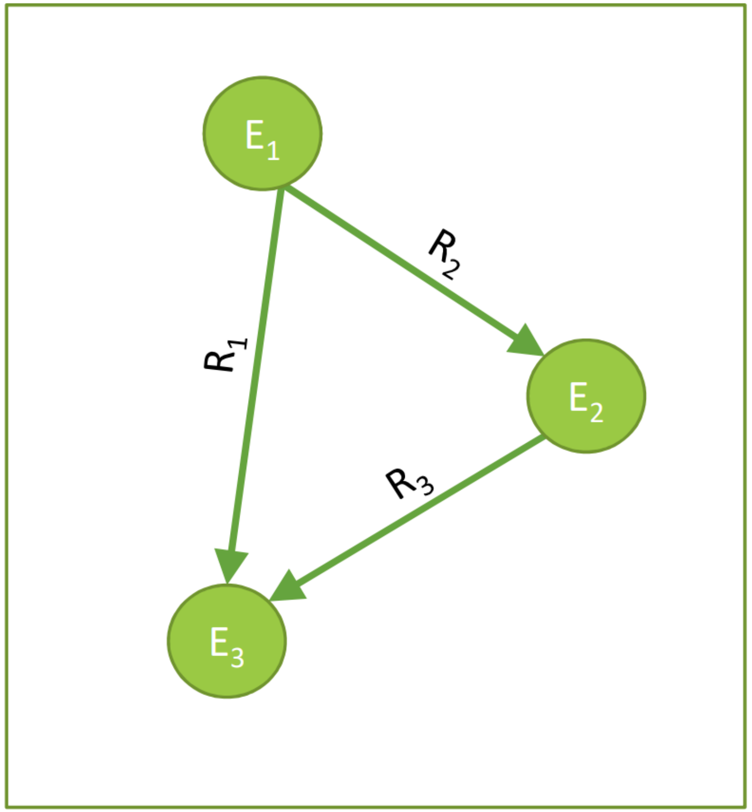


Recap: Knowledge Graph

Knowledge in graph:

- A set of triplets <head entity, relationship, tail entity>
- Capture entities, types, and relationships

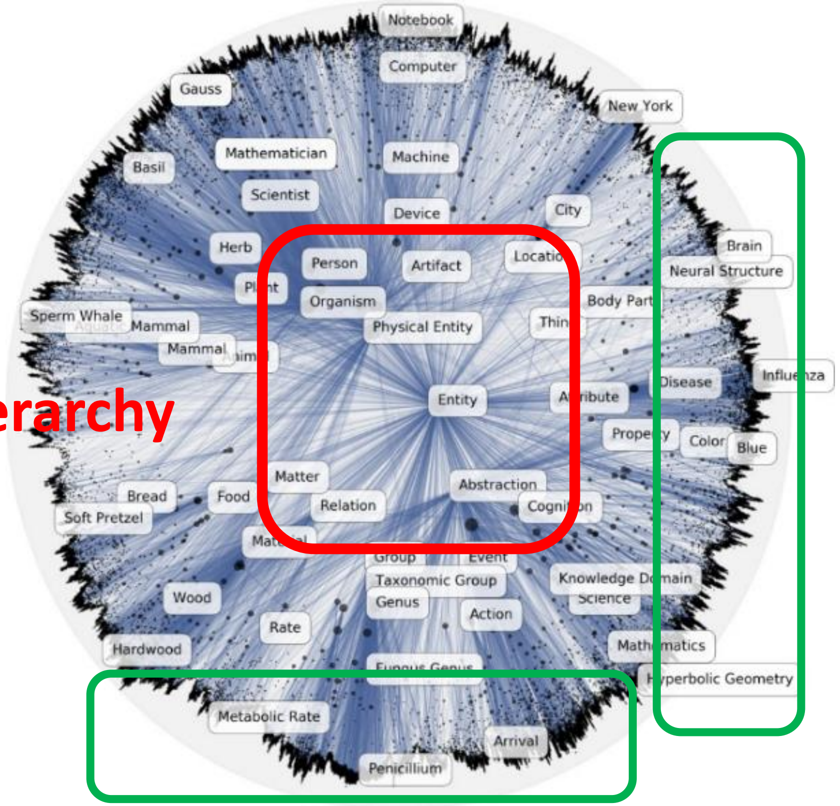
- Nodes are **entities**
- Nodes are labeled with their **types**
- Edges between two nodes capture **relationships** between entities
- **KG is an example of a heterogeneous graph**
  - **Recap:** Heterogenous graph is a graph with multiple node types and edge types



E: entity
R: relation

Tasks

- Graph representation learning on **hierarchical** graphs
  - Link Prediction
  - Node Classification



Nodes in high hierarchy

Nodes in low hierarchy

Poincaré embedding (Nickel et. al.)

Problem Setting

- Given a Graph $G = (V, E)$, V is vertex set, E is edge set,
- $x_i^{0,E}$ indicates the **initial (first layer) feature** of node i in a Euclidean Space
- We use ^{E} to indicate the features in Euclidean Space, ^{H} to denote hyperbolic space, ^{l} to denote the l-th layer feature
- **Goal:** learn a mapping f which maps nodes to d-dimension embedding vectors

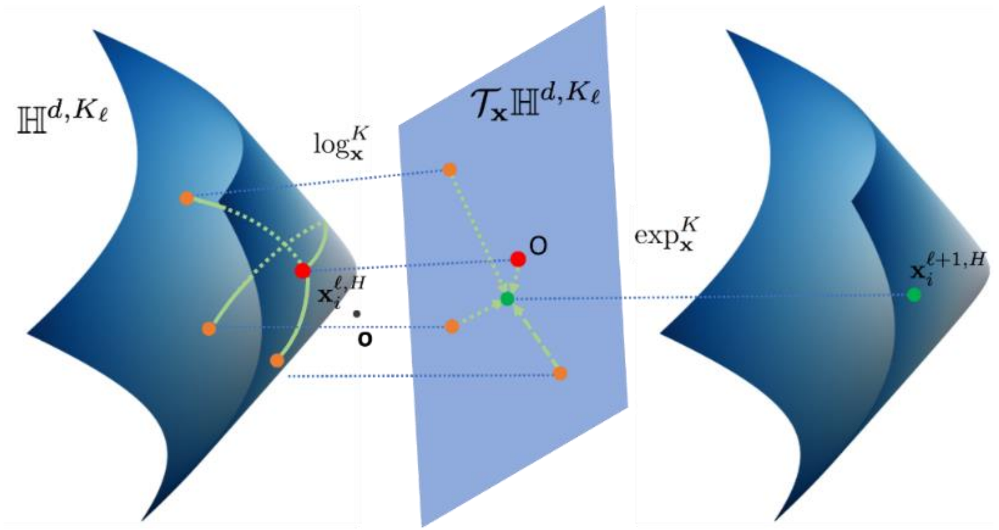
$$f: \left(V, E, \left(x_i^{0,E}\right)_{i \in V}\right) \to Z \in \mathbb{R}^{|V| \times d}$$

Overview: Hyperbolic GNN (HGCN)

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

K_l: curvature at layer l

At every layer:

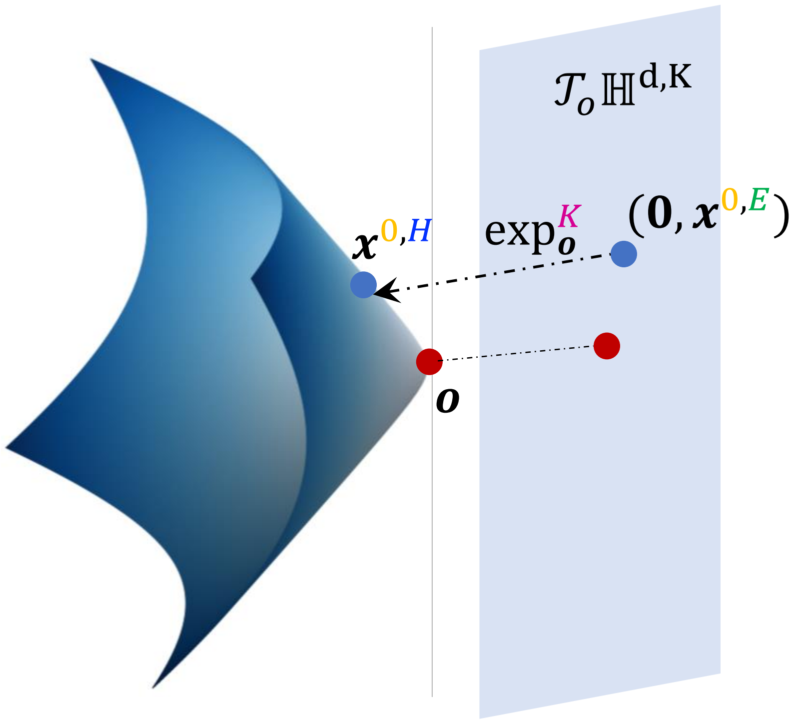


Message **Aggregation** **Update**

Hyperbolic GNN: Transformation

- $\boldsymbol{x}^{0,E} \in \mathbb{R}^d$ denotes input Euclidean features
- We use $\boldsymbol{o} = \{\sqrt{\boldsymbol{K}}, \mathbf{0}, \dots, \mathbf{0}\} \in \mathbb{H}^{\mathrm{d,K}}$ (the north pole in $\mathbb{H}^{\mathrm{d,K}}$) as a reference point to perform exponential mapping
  - $\mathcal{T}_{\boldsymbol{o}}\mathbb{H}^{\mathrm{d,K}} = \{\boldsymbol{v} \in \mathbb{R}^{\mathrm{d+1}}: \langle \boldsymbol{v}, \boldsymbol{o} \rangle_{\mathcal{L}} = 0\}$
  - We have $\langle \boldsymbol{o}, (0, \boldsymbol{x}^{0,E}) \rangle = 0$

$(0, \boldsymbol{x}^{0,E})$ can be interpreted as a point in $\mathcal{T}_o\mathbb{H}^{\mathrm{d,K}}$!



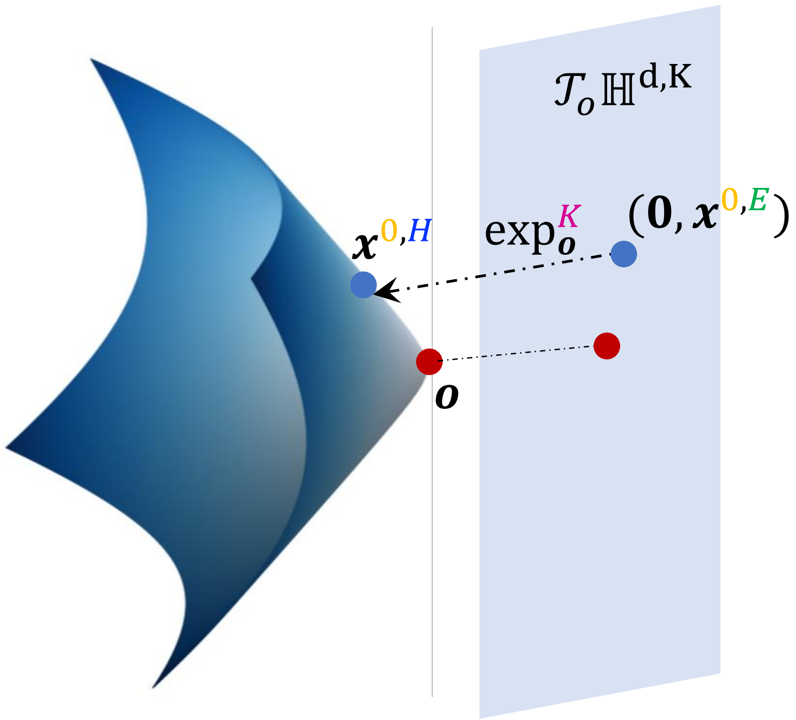
Hyperbolic GNN: Transformation

- **Input Transformation**

$$\boldsymbol{x}^{0,H} := \exp_{\boldsymbol{o}}^{K}\left((0, \boldsymbol{x}^{0,E})\right)$$

$$= \left(\sqrt{K}\cosh\left(\frac{\left\|\boldsymbol{x}^{0,E}\right\|_2}{\sqrt{K}}\right), \sqrt{K}\sinh\left(\frac{\left\|\boldsymbol{x}^{0,E}\right\|_2}{\sqrt{K}}\right)\frac{\boldsymbol{x}^{0,E}}{\left\|\boldsymbol{x}^{0,E}\right\|_2}\right)$$

- $(0, \boldsymbol{x}^{0,E})$ is a point in $\mathcal{T}_o\mathbb{H}^{\mathrm{d,K}}$
- $\exp_{\boldsymbol{o}}^{K}$ maps the point to $\mathbb{H}^{\mathrm{d,K}}$



Hyperbolic GNN: Message

- **Message**:

$$h_i^{l,H} = \left(W^l \otimes^{K_{l-1}} x_i^{l-1,H}\right) \oplus^{K_{l-1}} b^l$$

- Hyperbolic linear: $W^l \otimes^K x^H := \exp_o^K(W^l \log_o^K(x^H))$
  - $\log_o^K$ maps hyperbolic points x^H to tangent space $\mathcal{T}_o \mathbb{H}^{d_1,\mathrm{K}}$
  - do linear transformation in $\mathcal{T}_o \mathbb{H}^{\mathrm{d,K}}$ with transformation matrix $W^l \in \mathbb{R}^{d_1 \times d_2}$
  - $\exp_o^K$ maps points back to the hyperboloid $\mathbb{H}^{d_2,\mathrm{K}}$
- Mobius addition: $x^H \oplus^K b := \exp_{x^H}^K\left(\boxed{P_{o \to x^H}^K(b)}\right)$

In tangent space $\mathcal{T}_{x^H} \mathbb{H}^{\mathrm{d,K}}$

Recap: Parallel Transport

$$P_{x \to y}(v) = v - \frac{\langle \log_x^K(y), v \rangle_{\mathcal{L}}}{D_{\mathcal{L}}^K(x, y)^2}(\log_x^{\mathrm{K}} y + \log_y^{\mathrm{K}} x)$$

Hyperbolic GNN: Aggregation

- Given hyperbolic messages $\boldsymbol{h}_i^{l,H}, \boldsymbol{h}_j^{l,H}$ of node i and node j
- Map $\boldsymbol{h}_i^{l,H}, \boldsymbol{h}_j^{l,H}$ to the tangent space of the origin $\mathcal{T}_o\mathbb{H}^{\mathrm{d,K}}$ and calculate **attention weight $\boldsymbol{w}_{ij}^{l}$** (node j to node i)

$$w_{ij}^{l} = \mathrm{Softmax}_{j\in\mathcal{N}(i)}(\mathrm{MLP}(\log_{\boldsymbol{o}}^{K}(\boldsymbol{h}_i^{l,H})||\log_{\boldsymbol{o}}^{K}(\boldsymbol{h}_j^{l,H})))$$

concatenation

- **Aggregation**:

$$\boldsymbol{y}_i^{l,H} = AGG^{K}(\boldsymbol{h}^{l,H})_i := \exp_{\boldsymbol{h}_i^{l,H}}^{K}(\sum_{j\in\mathcal{N}(i)} \boxed{w_{ij}^{l}\log_{\boldsymbol{h}_i^{l,H}}^{K}(\boldsymbol{h}_j^{l,H})})$$

- Note: curvature K is layer-wise and trainable!

Note: do aggregation in Tangent space $\mathcal{T}_{h_i^{l,H}}\mathbb{H}^{\mathrm{d,K}}$ when considering node i

Hyperbolic GNN: Update

- **Update**:

$$x_i^{l,H} = \mathrm{Update}^{K_{l-1},K_l}\left(\boldsymbol{y}_i^{l,H}\right) \coloneqq \exp_{\boldsymbol{o}}^{K_l}\left(\sigma\left(\log_{\boldsymbol{o}}^{K_{l-1}}\left(\boldsymbol{y}_i^{l,H}\right)\right)\right)$$

- σ is a non-linear activation
- Apply activation in $\mathcal{T}_o\mathbb{H}^{\mathrm{d},\mathrm{K}_{l-1}}$ and then map back to $\mathbb{H}^{\mathrm{d},\mathrm{K}_l}$
- Tangent space of origin $\mathcal{T}_o\mathbb{H}^{\mathrm{d,K}}$ is shared across hyperboloids $\mathbb{H}^{\mathrm{d},K}$ with any curvature K
- $\mathrm{Update}^{K_{l-1},K_l}(\cdot)$ enables HGNN to **smoothly vary curvature** at each layer from K_{l-1} to K_l

Hyperbolic GNN: Predict

- For **link prediction**, HGCN uses Fermi-Dirac decoder:

$$p\left((i,j) \in E \mid x_i^{L,H}, x_j^{L,H}\right) = \left[e^{\left(d_L^{K_L}\left(x_i^{L,H}, x_j^{L,H}\right)^2 - r\right)/t} + 1\right]^{-1}$$

- $p\left((i,j) \in E \mid x_i^{L,H}, x_j^{L,H}\right) \in (0,1]$
- $d_L^{K_L}(\cdot,\cdot)$ is the hyperbolic distance in $\mathbb{H}^{\mathrm{d},\mathrm{K}_L}$
- r and t are hyper-parameters
- For **node classification**, use **exponential map** to map hyperbolic embeddings into Euclidean tangent space at O, and perform multi-class classification with standard softmax and cross entropy

δ-Hyperbolicity

Gromov's δ-Hyperbolicity

An undirected graph $G = (V, E)$ can be viewed as a metric space V with the graph distance d_G. Given $u, v, w, t \in V$ satisfying

$$d(u,v) + d(w,t) \geq d(u,t) + d(w,v) \geq d(u,w) + d(v,t),$$

Four-points condition

we denote

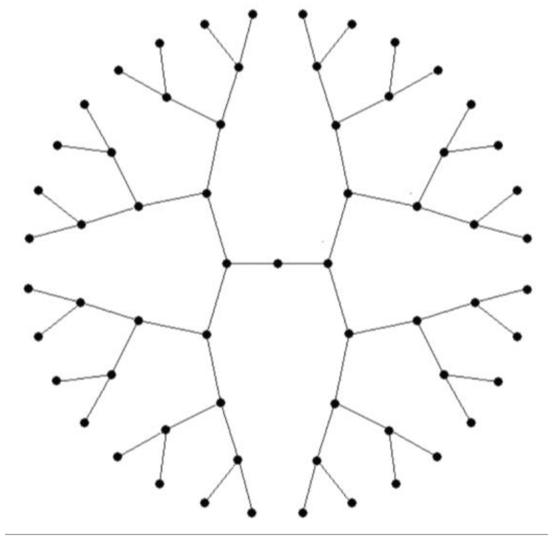
$$\delta(u,v,w,t) = \frac{d(u,v) + d(w,t) - d(u,t) - d(w,v)}{2}$$

The δ-Hyperbolicity of the graph is defined as

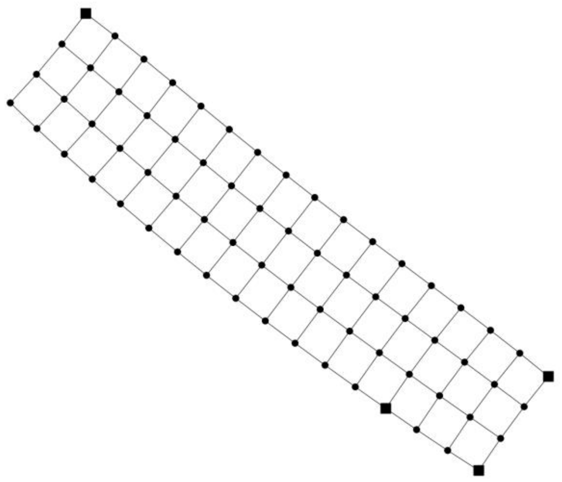
$$\delta(G, d_G) = \sup_{u,v,w,t \in V} \delta(u,v,w,t)$$

δ-Hyperbolicity

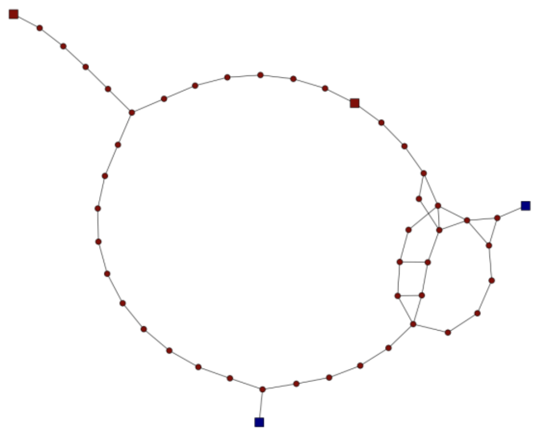
- The **lower** δ, the **more hyperbolic** is the graph
- $\delta = 0$ for trees.



$\boldsymbol{\delta = 0}$



$\boldsymbol{\delta = 3.0}$



$\boldsymbol{\delta = 4.5}$

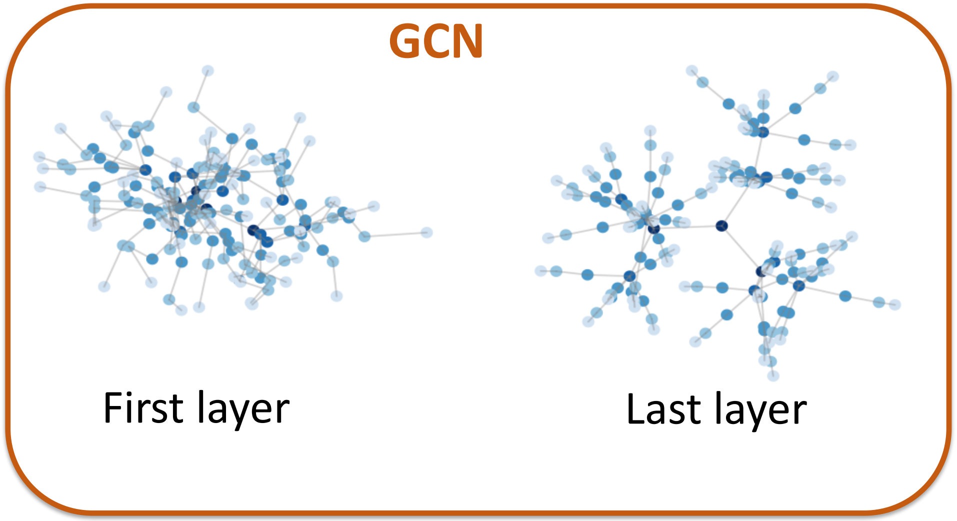
Experimental Results

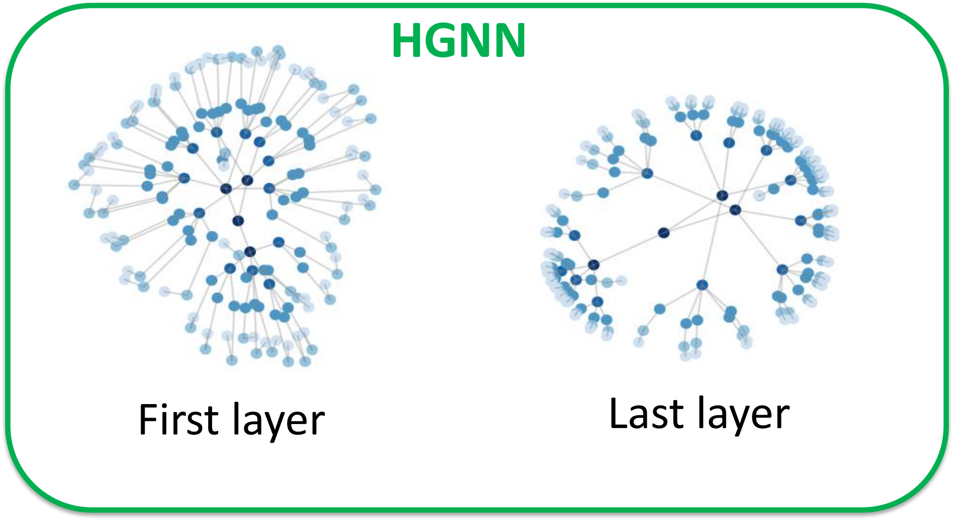
	Dataset	Disease		Disease-M		Human PPI		Airport		PubMed		Cora	
	Hyperbolicity δ	$\delta = 0$		$\delta = 0$		$\delta = 1$		$\delta = 1$		$\delta = 3.5$		$\delta = 11$	
	Method	LP	NC	LP	NC	LP	NC	LP	NC	LP	NC	LP	NC
Shallow	Euc	59.8 ± 2.0	32.5 ± 1.1	-	-	-	-	92.0 ± 0.0	60.9 ± 3.4	83.3 ± 0.1	48.2 ± 0.7	82.5 ± 0.3	23.8 ± 0.7
Shallow	Hyp [29]	63.5 ± 0.6	45.5 ± 3.3	-	-	-	-	94.5 ± 0.0	70.2 ± 0.1	87.5 ± 0.1	68.5 ± 0.3	87.6 ± 0.2	22.0 ± 1.5
Shallow	Euc-Mixed	49.6 ± 1.1	35.2 ± 3.4	-	-	-	-	91.5 ± 0.1	68.3 ± 2.3	86.0 ± 1.3	63.0 ± 0.3	84.4 ± 0.2	46.1 ± 0.4
Shallow	Hyp-Mixed	55.1 ± 1.3	56.9 ± 1.5	-	-	-	-	93.3 ± 0.0	69.6 ± 0.1	83.8 ± 0.3	73.9 ± 0.2	85.6 ± 0.5	45.9 ± 0.3
NN	MLP	72.6 ± 0.6	28.8 ± 2.5	55.3 ± 0.5	55.9 ± 0.3	67.8 ± 0.2	55.3 ± 0.4	89.8 ± 0.5	68.6 ± 0.6	84.1 ± 0.9	72.4 ± 0.2	83.1 ± 0.5	51.5 ± 1.0
NN	HNN[10]	75.1 ± 0.3	41.0 ± 1.8	60.9 ± 0.4	56.2 ± 0.3	72.9 ± 0.3	59.3 ± 0.4	90.8 ± 0.2	80.5 ± 0.5	94.9 ± 0.1	69.8 ± 0.4	89.0 ± 0.1	54.6 ± 0.4
GNN	GCN[21]	64.7 ± 0.5	69.7 ± 0.4	66.0 ± 0.8	59.4 ± 3.4	77.0 ± 0.5	69.7 ± 0.3	89.3 ± 0.4	81.4 ± 0.6	91.1 ± 0.5	78.1 ± 0.2	90.4 ± 0.2	81.3 ± 0.3
GNN	GAT [41]	69.8 ± 0.3	70.4 ± 0.4	69.5 ± 0.4	62.5 ± 0.7	76.8 ± 0.4	70.5 ± 0.4	90.5 ± 0.3	81.5 ± 0.3	91.2 ± 0.1	79.0 ± 0.3	$\mathbf{93.7} \pm 0.1$	$\mathbf{83.0} \pm 0.7$
GNN	SAGE [15]	65.9 ± 0.3	69.1 ± 0.6	67.4 ± 0.5	61.3 ± 0.4	78.1 ± 0.6	69.1 ± 0.3	90.4 ± 0.5	82.1 ± 0.5	86.2 ± 1.0	77.4 ± 2.2	85.5 ± 0.6	77.9 ± 2.4
GNN	SGC [44]	65.1 ± 0.2	69.5 ± 0.2	66.2 ± 0.2	60.5 ± 0.3	76.1 ± 0.2	71.3 ± 0.1	89.8 ± 0.3	80.6 ± 0.1	94.1 ± 0.0	78.9 ± 0.0	91.5 ± 0.1	81.0 ± 0.1
Ours	HGCN	$\mathbf{90.8} \pm 0.3$	$\mathbf{74.5} \pm 0.9$	$\mathbf{78.1} \pm 0.4$	$\mathbf{72.2} \pm 0.5$	$\mathbf{84.5} \pm 0.4$	$\mathbf{74.6} \pm 0.3$	$\mathbf{96.4} \pm 0.1$	$\mathbf{90.6} \pm 0.2$	$\mathbf{96.3} \pm 0.0$	$\mathbf{80.3} \pm 0.3$	92.9 ± 0.1	79.9 ± 0.2
Ours	(%) Err Red	-63.1%	-13.8%	-28.2%	-25.9%	-29.2%	-11.5%	-60.9%	-47.5%	-27.5%	-6.2%	+12.7%	+18.2%

- LP denotes link prediction
- NC denotes node classification

Embedding Visualization

- Visualization on the Poincaré disk for link prediction on *DISEASE* ($\delta = 0$)
- Color indicates the **depth/hierarchy** of the node in a tree
  - Darker color ⇒ deeper in a tree ⇒lower hierarchy
- GCN hardly captures hierarchy, while HGNN preserves node hierarchies

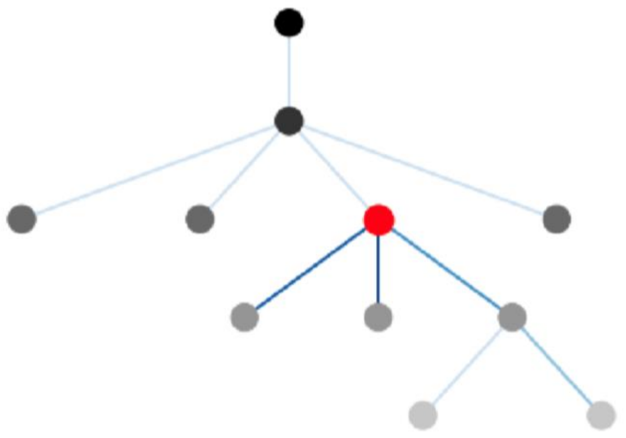




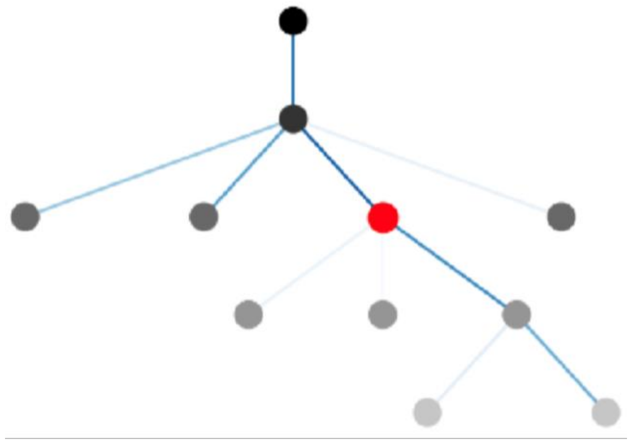
Lower distortion

Attention Visualization

- **Attention weights** in 2-hop neighbor of a center node on *DISEASE* $(\delta = 0)$
- **Darkness** of the color denotes their **hierarchy**. **Intensity** of the edges denotes the **attention weights**
- In HGNN, the center node pays more attention to its **(grand) parent**, who is with a **higher hierarchy**.



Euclidean GAT

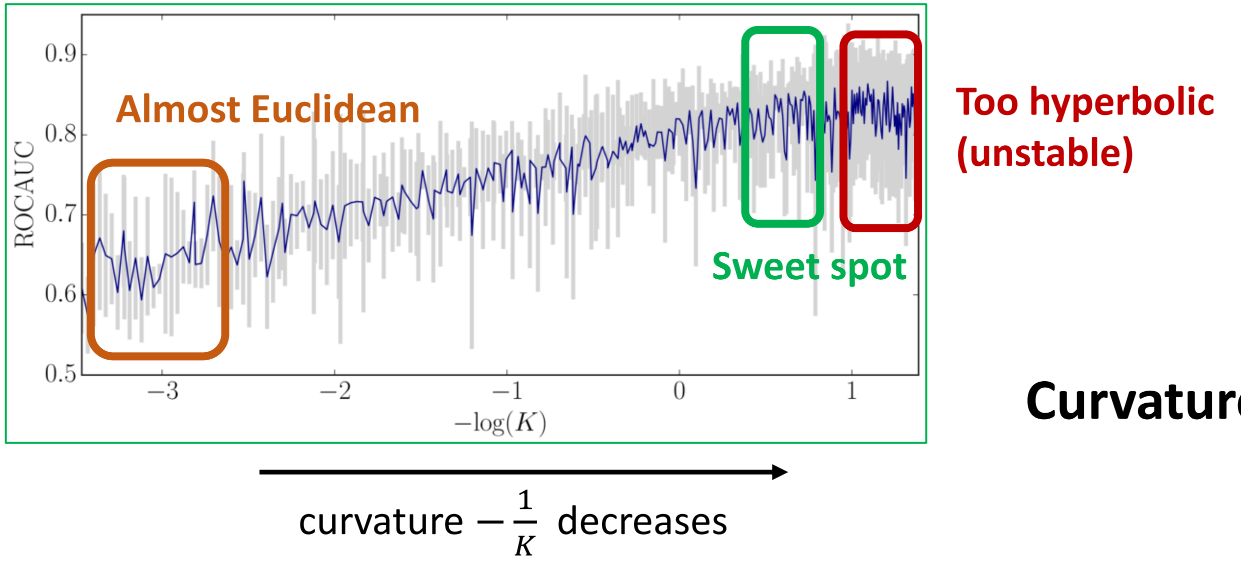


HGNN

● is the node where we compute attention

Performance V.S. Curvature

- Adjusting and training the curvature leads to improve the performance
- **Decreasing** the curvature **improves** link prediction performance on *DISEASE* $(\delta = 0)$



Curvature is important!

Summary of Hyperbolic Embedding

- **Hyperbolic embeddings** use hyperbolic geometry with constant negative curvature to preserve graph distances and complex relationships, particularly for **hierarchical and tree-like graphs.**
- **HGCN:** Graph convolutional network in **hyperbolic space**
  - maps Euclidean input features to hyperbolic embedding space, performs message aggregation in the tangent space and maps back to the hyperbolic space
- Experiments show decreasing the curvature of embedding space improves the performance over graphs with lower **δ-Hyperbolicity.**