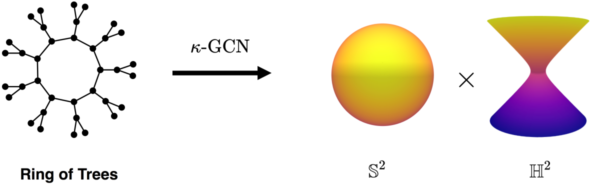
Constant Curvature Graph Convolutional Networks



Gregor Bachmann
ETH Zürich

Gary Bécigneul
MIT

Octavian Ganea
MIT

ETH zürich

Massachusetts Institute of Technology

Overview

Overview

- Embeddings of graphs into **hyperbolic** and **spherical** space and their products

Overview

- Embeddings of graphs into **hyperbolic** and **spherical** space and their products
- Extend gyrovector framework to spherical geometry and provide a **unifying formalism**

Overview

- Embeddings of graphs into **hyperbolic** and **spherical** space and their products
- Extend gyrovector framework to spherical geometry and provide a **unifying formalism**
- Introduce graph neural networks producing embeddings in product spaces

Overview

- Embeddings of graphs into **hyperbolic** and **spherical** space and their products
- Extend gyrovector framework to spherical geometry and provide a **unifying formalism**
- Introduce graph neural networks producing embeddings in product spaces
- Differentiable **transitions** in geometry during training in each component

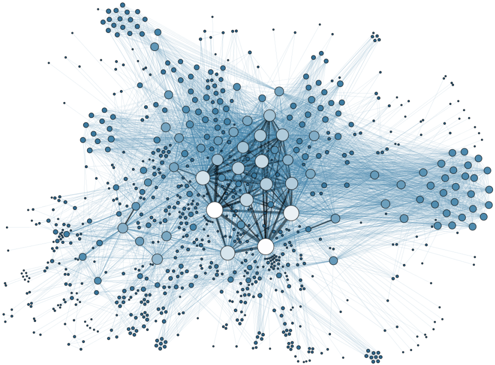
Graphs

Graphs

- Lots of data available in the form of **graphs** (social networks, railway tracks, phylogenetic trees etc.)

Graphs

- Lots of data available in the form of **graphs** (social networks, railway tracks, phylogenetic trees etc.)



Graphs

- Lots of data available in the form of **graphs** (social networks, railway tracks, phylogenetic trees etc.)

- Node set $\boldsymbol{V} = \{1, \ldots, n\}$ and adjacency matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$

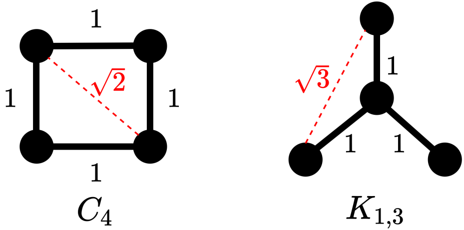
Where to Embed Graphs?

Where to Embed Graphs?

- Euclidean geometry **not** suitable for many graphs

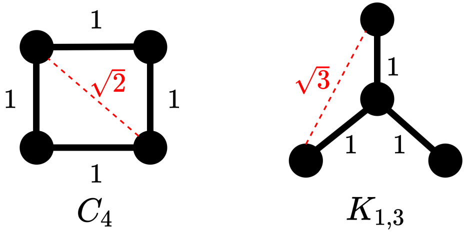
Where to Embed Graphs?

- Euclidean geometry **not** suitable for many graphs



Where to Embed Graphs?

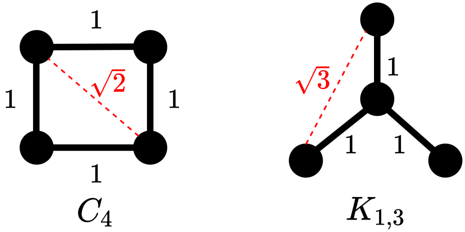
- Euclidean geometry **not** suitable for many graphs



- **Graph distance** $d_{\boldsymbol{G}}(i, j) =$ *"Shortest path from i to j"* not respected in Euclidean embedding

Where to Embed Graphs?

- Euclidean geometry **not** suitable for many graphs



- **Graph distance** $d_{\boldsymbol{G}}(i,j) =$ *"Shortest path from i to j"* not respected in Euclidean embedding

- Arbitrary low distortion in **spherical** and **hyperbolic** space

Non-Euclidean Geometry

Non-Euclidean Geometry

- Focus on **constant sectional curvature** manifolds

Non-Euclidean Geometry

- Focus on **constant sectional curvature** manifolds
- Well-studied in the field of **Differential Geometry**

Non-Euclidean Geometry

- Focus on **constant sectional curvature** manifolds
- Well-studied in the field of **Differential Geometry**
- **Computationally attractive** expressions for distance, exponential map etc.

Hyperbolic Space as Poincaré Ball

Hyperbolic Space as Poincaré Ball

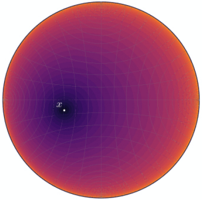
- $\mathbb{H}^n = \{\boldsymbol{x} : ||\boldsymbol{x}||_2 \leq \frac{1}{\sqrt{c}}\}$ with curvature $-c$ equipped with **Riemannian tensor** $g^c_{\boldsymbol{x}} = \frac{4}{(1-c||\boldsymbol{x}||^2)^2}\mathbb{1}$

Hyperbolic Space as Poincaré Ball

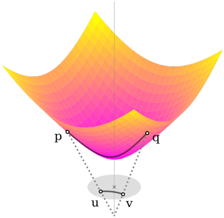
- $\mathbb{H}^n = \{\boldsymbol{x} : ||\boldsymbol{x}||_2 \leq \frac{1}{\sqrt{c}}\}$ with curvature $-c$ equipped with **Riemannian tensor** $g_{\boldsymbol{x}}^c = \frac{4}{(1-c||\boldsymbol{x}||^2)^2}\mathbb{1}$
- **Projection** of hyperboloid

Hyperbolic Space as Poincaré Ball

- $\mathbb{H}^n = \{\boldsymbol{x} : ||\boldsymbol{x}||_2 \leq \frac{1}{\sqrt{c}}\}$ with curvature $-c$ equipped with **Riemannian tensor** $g_{\boldsymbol{x}}^c = \frac{4}{(1-c||\boldsymbol{x}||^2)^2}\mathbb{1}$
- **Projection** of hyperboloid
- $d_{\mathbb{H}}^c(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{\sqrt{c}} \cosh^{-1}\left(1 + \frac{\frac{2}{c}||\boldsymbol{x}-\boldsymbol{y}||_2^2}{\left(\frac{1}{c}-||\boldsymbol{x}||_2^2\right)\left(\frac{1}{c}-||\boldsymbol{y}||_2^2\right)}\right)$



Heatmap of $d_{\mathbb{H}}^{\kappa}$



Projection of hyperboloid [4]

Gyrospace Structure

Gyrospace Structure

- Next best thing to a **vector space**

Gyrospace Structure

- Next best thing to a **vector space**
- Vector addition $\boldsymbol{x} + \boldsymbol{y} \mapsto \boldsymbol{x} \oplus_c \boldsymbol{y}$

Gyrospace Structure

- Next best thing to a **vector space**
- Vector addition $\boldsymbol{x} + \boldsymbol{y} \mapsto \boldsymbol{x} \oplus_c \boldsymbol{y}$
- Scalar multiplication $r\boldsymbol{x} \mapsto r \otimes_c \boldsymbol{x}$

Gyrospace Structure

- Next best thing to a **vector space**
- Vector addition $\boldsymbol{x} + \boldsymbol{y} \mapsto \boldsymbol{x} \oplus_c \boldsymbol{y}$
- Scalar multiplication $r\boldsymbol{x} \mapsto r \otimes_c \boldsymbol{x}$
- Geodesic $\gamma_{\boldsymbol{x}\to\boldsymbol{y}}(t) = \boldsymbol{x} \oplus_c (t \otimes_c (-\boldsymbol{x} \oplus_c \boldsymbol{y}))$

Spherical Space as Stereographic Projection

Spherical Space as Stereographic Projection

- Stereographic projection of $\mathbb{S}^{d+1} \cong \mathbb{R}^d + g_{\boldsymbol{x}}^c$ where $g_{\boldsymbol{x}}^c = \frac{4}{(1+c||\boldsymbol{x}||^2)^2}\mathbb{1}$

Spherical Space as Stereographic Projection

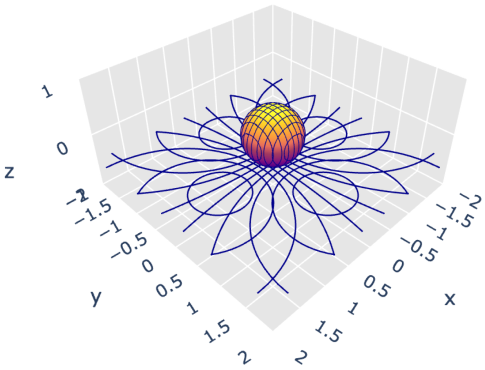
- Stereographic projection of $\mathbb{S}^{d+1} \cong \mathbb{R}^d + g_{\boldsymbol{x}}^c$ where $g_{\boldsymbol{x}}^c = \frac{4}{(1+c||\boldsymbol{x}||^2)^2}\mathbb{1}$

- $d_{\mathbb{S}}^c(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{\sqrt{c}} \cos^{-1}\left(1 + \frac{\frac{2}{c}||\boldsymbol{x}-\boldsymbol{y}||_2^2}{\left(\frac{1}{c}+||\boldsymbol{x}||_2^2\right)\left(\frac{1}{c}+||\boldsymbol{y}||_2^2\right)}\right)$

Spherical Space as Stereographic Projection

- Stereographic projection of $\mathbb{S}^{d+1} \cong \mathbb{R}^d + g_{\boldsymbol{x}}^c$ where $g_{\boldsymbol{x}}^c = \frac{4}{(1+c||\boldsymbol{x}||^2)^2}\mathbb{1}$

- $d_{\mathbb{S}}^c(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{\sqrt{c}} \cos^{-1}\left(1 + \frac{\frac{2}{c}||\boldsymbol{x}-\boldsymbol{y}||_2^2}{\left(\frac{1}{c}+||\boldsymbol{x}||_2^2\right)\left(\frac{1}{c}+||\boldsymbol{y}||_2^2\right)}\right)$



Our Contributions: 1) Unified Formalism

Our Contributions: 1) Unified Formalism

- κ-**stereographic model** for any $\kappa \in \mathbb{R}$:

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

Our Contributions: 1) Unified Formalism

- κ-**stereographic model** for any $\kappa \in \mathbb{R}$:

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

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

Our Contributions: 1) Unified Formalism

- **κ-stereographic model** for any $\kappa \in \mathbb{R}$:

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

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

- More unifying expressions for **distance**, **exponential map** etc. in our paper!

Our Contributions: 2) Matrix Multiplications

Our Contributions: 2) Matrix Multiplications

- Embeddings $\boldsymbol{X}$ where $\boldsymbol{X}_{i\bullet} \in \mathfrak{st}_{\kappa}^{d}$, $\boldsymbol{W} \in \mathbb{R}^{d \times k}$ and $\boldsymbol{A} \in \mathbb{R}^{n \times n}$

Our Contributions: 2) Matrix Multiplications

- Embeddings $\boldsymbol{X}$ where $\boldsymbol{X}_{i\bullet} \in \mathfrak{st}_{\kappa}^{d}$, $\boldsymbol{W} \in \mathbb{R}^{d \times k}$ and $\boldsymbol{A} \in \mathbb{R}^{n \times n}$
- **Right** matrix multiplication $\boldsymbol{XW}$ acts on columns $\boldsymbol{X}_{\bullet i}$

  Thus lift to tangent space at zero:

$$(\boldsymbol{X} \otimes_{\kappa} \boldsymbol{W})_{i\bullet} = exp_0^{\kappa}\left((\log_0^{\kappa}(\boldsymbol{X})\boldsymbol{W})_{i\bullet}\right)$$

Our Contributions: 2) Matrix Multiplications

- Embeddings $\boldsymbol{X}$ where $\boldsymbol{X}_{i\bullet} \in \mathfrak{st}_\kappa^d$, $\boldsymbol{W} \in \mathbb{R}^{d\times k}$ and $\boldsymbol{A} \in \mathbb{R}^{n\times n}$
- **Right** matrix multiplication $\boldsymbol{XW}$ acts on columns $\boldsymbol{X}_{\bullet i}$

  Thus lift to tangent space at zero:

$$(\boldsymbol{X} \otimes_\kappa \boldsymbol{W})_{i\bullet} = exp_0^\kappa\left((\log_0^\kappa(\boldsymbol{X})\boldsymbol{W})_{i\bullet}\right)$$

- Introduced in [2], we extended it to spherical spaces

Our Contributions: 2) Matrix Multiplications

Our Contributions: 2) Matrix Multiplications

- **Left** matrix multiplication $\boldsymbol{AX}$ acts on rows $\boldsymbol{X}_{i\bullet}$:

$$(\boldsymbol{AX})_{i\bullet} = \boldsymbol{A}_{i1}\boldsymbol{X}_{1\bullet} + \cdots + \boldsymbol{A}_{in}\boldsymbol{X}_{n\bullet}$$

Our Contributions: 2) Matrix Multiplications

- **Left** matrix multiplication $\boldsymbol{AX}$ acts on rows $\boldsymbol{X}_{i\bullet}$:

$$(\boldsymbol{AX})_{i\bullet} = \boldsymbol{A}_{i1}\boldsymbol{X}_{1\bullet} + \cdots + \boldsymbol{A}_{in}\boldsymbol{X}_{n\bullet}$$

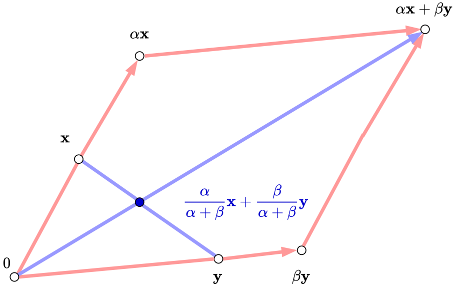
- **Idea:** Reduce problem of linear combination to definition of a non-euclidean **midpoint**

Our Contributions: 2) Matrix Multiplications

- **Left** matrix multiplication $\boldsymbol{AX}$ acts on rows $\boldsymbol{X}_{i\bullet}$:

$$(\boldsymbol{AX})_{i\bullet} = \boldsymbol{A}_{i1}\boldsymbol{X}_{1\bullet} + \cdots + \boldsymbol{A}_{in}\boldsymbol{X}_{n\bullet}$$

- **Idea:** Reduce problem of linear combination to definition of a non-euclidean **midpoint**



Our Contributions: 2) Matrix Multiplications

Our Contributions: 2) Matrix Multiplications

- Leverage **gyromidpoint** for hyperbolic space and extend it to $\mathfrak{st}_\kappa^d$:

$$m_\kappa(\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n; \boldsymbol{\alpha}) = \frac{1}{2} \otimes_\kappa \left(\sum_{i=1}^{n} \frac{\alpha_i \lambda_{\boldsymbol{x}_i}^\kappa}{\sum_{j=1}^{n} \alpha_j (\lambda_{\boldsymbol{x}_j}^\kappa - 1)} \boldsymbol{x}_i \right)$$

Our Contributions: 2) Matrix Multiplications

- Leverage **gyromidpoint** for hyperbolic space and extend it to $\mathfrak{st}_\kappa^d$:

$$m_\kappa(\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n; \boldsymbol{\alpha}) = \frac{1}{2} \otimes_\kappa \left(\sum_{i=1}^{n} \frac{\alpha_i \lambda_{\boldsymbol{x}_i}^\kappa}{\sum_{j=1}^n \alpha_j (\lambda_{\boldsymbol{x}_j}^\kappa - 1)} \boldsymbol{x}_i \right)$$

- Define left matrix multiplication row-wise:

$$(\boldsymbol{A} \boxtimes_\kappa \boldsymbol{X})_{i\bullet} := (\sum_j \boldsymbol{A}_{ij}) \otimes_\kappa m_\kappa(\boldsymbol{X}_{1\bullet}, \cdots, \boldsymbol{X}_{n\bullet}; \boldsymbol{A}_{i\bullet})$$

Our Contributions: 2) Matrix Multiplications

- Leverage **gyromidpoint** for hyperbolic space and extend it to $\mathfrak{st}_\kappa^d$:

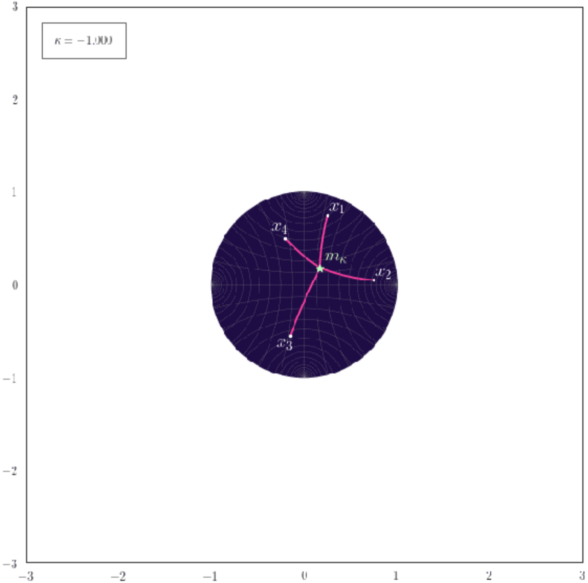
$$m_\kappa(\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n; \boldsymbol{\alpha}) = \frac{1}{2} \otimes_\kappa \left(\sum_{i=1}^{n} \frac{\alpha_i \lambda_{\boldsymbol{x}_i}^\kappa}{\sum_{j=1}^{n} \alpha_j (\lambda_{\boldsymbol{x}_j}^\kappa - 1)} \boldsymbol{x}_i \right)$$

- Define left matrix multiplication row-wise:

$$(\boldsymbol{A} \boxtimes_\kappa \boldsymbol{X})_{i\bullet} := (\sum_j \boldsymbol{A}_{ij}) \otimes_\kappa m_\kappa(\boldsymbol{X}_{1\bullet}, \cdots, \boldsymbol{X}_{n\bullet}; \boldsymbol{A}_{i\bullet})$$

- Same scaling behaviour: $d_\kappa(\boldsymbol{0}, r \otimes_\kappa \boldsymbol{x}) = r \cdot d_\kappa(\boldsymbol{0}, \boldsymbol{x})$

Gyromidpoint for Varying Curvature



Our Contributions: 3) Differentiable Interpolation

Our Contributions: 3) Differentiable Interpolation

- All quantities **recover** their Euclidean counterpart for $\kappa \to 0^{\pm}$

Our Contributions: 3) Differentiable Interpolation

- All quantities **recover** their Euclidean counterpart for $\kappa \to 0^{\pm}$
- We proved an even **stronger** result:

Our Contributions: 3) Differentiable Interpolation

- All quantities **recover** their Euclidean counterpart for $\kappa \to 0^{\pm}$
- We proved an even **stronger** result:

Differentiability of $\mathfrak{st}_{\kappa}^{d}$ w.r.t. κ around 0

The first order derivatives at 0^{-} and 0^{+} w.r.t. to κ of all the mentioned quantities **exist** and are **equal**.

Our Contributions: 3) Differentiable Interpolation

- All quantities **recover** their Euclidean counterpart for $\kappa \to 0^{\pm}$
- We proved an even **stronger** result:

Differentiability of $\mathfrak{st}_\kappa^d$ w.r.t. κ around 0

The first order derivatives at 0^- and 0^+ w.r.t. to κ of all the mentioned quantities **exist** and are **equal**.

- Enables **learning the curvature** κ with gradient descent with a differentiable change of sign

Our Contributions: 4) Constant Curvature GCN

Our Contributions: 4) Constant Curvature GCN

- Given graph $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{A}, \boldsymbol{X})$ where $\boldsymbol{V} = \{1, \ldots, n\}$, adjacency $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ and node-level features $\boldsymbol{X} \in \mathbb{R}^{n \times d}$

Our Contributions: 4) Constant Curvature GCN

- Given graph $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{A}, \boldsymbol{X})$ where $\boldsymbol{V} = \{1, \ldots, n\}$, adjacency $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ and node-level features $\boldsymbol{X} \in \mathbb{R}^{n \times d}$

- **Graph neural networks** are a very popular class of models for inference on graphs

Our Contributions: 4) Constant Curvature GCN

- Given graph $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{A}, \boldsymbol{X})$ where $\boldsymbol{V} = \{1, \ldots, n\}$, adjacency $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ and node-level features $\boldsymbol{X} \in \mathbb{R}^{n \times d}$

- **Graph neural networks** are a very popular class of models for inference on graphs

- We extend the vanilla **GCN** [3]:

$$\boldsymbol{H}^{(t+1)} = \sigma\left(\hat{\boldsymbol{A}}\boldsymbol{H}^{(t)}\boldsymbol{W}^{(t)}\right)$$

  for some non-linearity σ, $\hat{\boldsymbol{A}} = \tilde{\boldsymbol{D}}^{-\frac{1}{2}}\left(\boldsymbol{A} + \mathbb{1}\right)\tilde{\boldsymbol{D}}^{-\frac{1}{2}}$, $\tilde{\boldsymbol{D}}_{ii} = \sum_k \tilde{\boldsymbol{A}}_{ik}$ and trainable parameters $\boldsymbol{W}^{(l)}$

Our Contributions: 4) Constant Curvature GCN

Our Contributions: 4) Constant Curvature GCN

- Turn it **non-euclidean**:

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

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

Our Contributions: 4) Constant Curvature GCN

- Turn it **non-euclidean**:

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

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

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

Our Contributions: 4) Constant Curvature GCN

- Turn it **non-euclidean**:

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

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

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

- Allows for **differentiable transitions** in the geometry during training

Our Contributions: 5) Product GCN

Our Contributions: 5) Product GCN

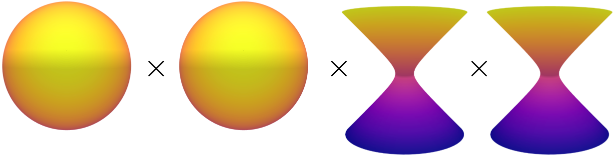
- We can take it one step further: Embed in **product space**

$$\mathfrak{st}^d_{\kappa_1} \times \cdots \times \mathfrak{st}^d_{\kappa_m}$$

Our Contributions: 5) Product GCN

- We can take it one step further: Embed in **product space**

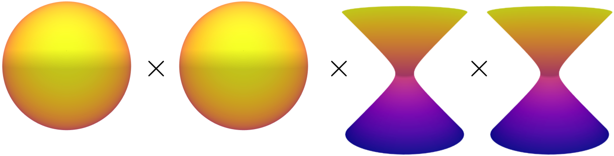
$$\mathfrak{st}^d_{\kappa_1} \times \cdots \times \mathfrak{st}^d_{\kappa_m}$$



Our Contributions: 5) Product GCN

- We can take it one step further: Embed in **product space**

$$\mathfrak{st}^d_{\kappa_1} \times \cdots \times \mathfrak{st}^d_{\kappa_m}$$

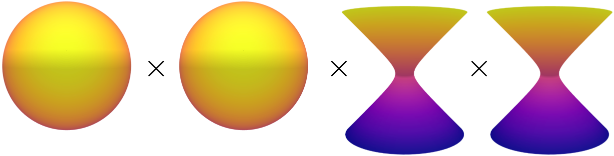


- Again we find a **gyrovector space** structure

Our Contributions: 5) Product GCN

- We can take it one step further: Embed in **product space**

$$\mathfrak{st}^d_{\kappa_1} \times \cdots \times \mathfrak{st}^d_{\kappa_m}$$



- Again we find a **gyrovector space** structure

- The **operations** extend component-wise while still preserving the desired properties

Experiments: Distortion Task

Experiments: Distortion Task

- **Minimize** the discrepancy between embedding distances and graph distances

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

Experiments: Distortion Task

- **Minimize** the discrepancy between embedding distances and graph distances

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

- Train κ-GCN on three syntethic datasets, **tree** (negative curvature), **spherical** graph (positive curvature) and **toroidal** graph (product of positive curvature)

Experiments: Distortion Task

- **Minimize** the discrepancy between embedding distances and graph distances

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

- Train κ-GCN on three syntethic datasets, **tree** (negative curvature), **spherical** graph (positive curvature) and **toroidal** graph (product of positive curvature)

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

Experiments: Node Classification

- Evaluate on four **real-world** datasets

Experiments: Node Classification

- Evaluate on four **real-world** datasets
- Report mean **accuracy** across 5 splits and 5 runs each

Experiments: Node Classification

- Evaluate on four **real-world** datasets
- Report mean **accuracy** across 5 splits and 5 runs each

MODEL	CITESEER	CORA	PUBMED	AIRPORT
$\mathbb{E}^{16}$ [3]	72.9 ± 0.54	81.4 ± 0.4	79.2 ± 0.39	81.4 ± 0.29
$\mathbb{H}^{16}$ [1]	71 ± 0.49	80.3 ± 0.46	$\mathbf{79.8 \pm 0.43}$	$\mathbf{84.4 \pm 0.41}$
$\mathbb{H}^{16}$ (κ-GCN)	$\mathbf{73.2 \pm 0.51}$	81.2 ± 0.5	78.5 ± 0.36	81.9 ± 0.33
$\mathbb{S}^{16}$ (κ-GCN)	72.1 ± 0.45	$\mathbf{81.9 \pm 0.45}$	78.8 ± 0.49	80.9 ± 0.58
PROD-GCN	71.1 ± 0.59	80.8 ± 0.41	78.1 ± 0.6	81.7 ± 0.44

THANK YOU!

Check out our website **hyperbolicdeeplearning.com**

$\mathbb{H}\mathcal{D}\mathcal{L}$

HYPERBOLIC DEEP LEARNING

References

[1] Chami, I., Ying, R., Ré, C., and Leskovec, J. (2019). Hyperbolic graph convolutional neural networks. *Advances in Neural Information processing systems*.

[2] Ganea, O., Bécigneul, G., and Hofmann, T. (2018). Hyperbolic neural networks. In *Advances in Neural Information Processing Systems*, pages 5345–5355.

[3] Kipf, T. N. and Welling, M. (2017). Semi-supervised classification with graph convolutional networks. *International Conference on Learning Representations*.

[4] Nickel, M. and Kiela, D. (2018). Learning continuous hierarchies in the lorentz model of hyperbolic geometry. In *International Conference on Machine Learning*.