Hyperbolic Deep Learning for Foundation Models: A Tutorial

Neil He, Menglin Yang, Rex Ying
rex.ying@yale.edu

Yale University

Contributors

Tutors	 Neil He(Yale)  Menglin Yang(HKUSTGZ)  Rex Ying(Yale)
Contributor and coauthors	Hiren Madhu(Yale), Ngoc Bui(Yale), Ali Maatouk(Yale), Rishabh Anand(Yale), Melanie Weber(Harvard), Jiahong Liu(CUHK), Irwin King(CUHK)

Website

Slack Group

Outline

- Preliminary
  - Motivation
  - Hyperbolic Geometry
- Building Blocks
  - Basic Hyperbolic NN Operations
  - Hyperbolic NN Architectures
- Hyperbolic Foundation Models
  - Hyperbolic LLMs & Transformers
  - Hyperbolic Vision Foundation Models
  - Hyperbolic Multi-Modal Foundation Models

Our goals is to introduce:

1. Motivations for Hyperbolic Foundation Models
2. Hyperbolic Geometry Basics
3. Hyperbolic Basic Neural Operations
4. Current Methods in Hyperbolic Foundation Models
5. Future Directions

Part 1: Preliminary

Motivation	Geometry of Inputs to Foundation Models	
	Limitations of Euclidean Embeddings	
	Alternative Geometric Spaces	
Hyperbolic Geometry	Riemannian Manifold & Hyperbolic Space	Poincare Ball
		Lorentz Hyperboloid
	Tangent Spaces & Geodesics	Exponential Maps
		Logarithmic Maps
		Parallel Transport

Part 1: Preliminary – Goals:

1. Motivate Hyperbolic Geometry for Foundation Models
2. Introduce Basics of Hyperbolic Geometry

Part 2: Building Blocks

Hyperbolic Basic NN Operations	Linear Transformations
	Residual connection
	Normalization
	Activation
	Attention Mechanisms
Hyperbolic NN Model Architecture	MLP
	ResNet & CNN
	GNN

Part 2: Building Blocks – Goals:

1. Introduce Basics Hyperbolic Neural Network Operations (e.g. Linear Transformations, Attention Mechanisms)
2. Introduce Basic Hyperbolic Neural Networks Models

Part 3: Hyperbolic Foundation Models

Hyperbolic LLMs & Transformers	FNN, HNN++, HAN
	HypFormer
	HypLoRA
	HELM
Hyperbolic Vision Foundation Models	Hyp-ViT, HVT, LViT
	HCL, RHCL
Hyperbolic Multi-Modal Foundation Models	MERU, HypCoCLIP, L-CLIP
	H-BLIP-2

Part 3: Hyperbolic Foundation Models – Goals (70 Min):

1. Introduce Current Methods in Hyperbolic Foundation Models
2. Discuss Potential Feature Directions

Part 1: Background: Motivation & Theory

Token Relationship

- The sun rises above the river.
- The river flows through the forest.
- The forest is dense with tall trees.
- Trees sway gently in the wind.
- The wind carries the scent of flowers.
- Flowers bloom brightly under the sun.
- The sun sets over the mountains.
- The mountains echo with the sound of birds.
- Birds fly freely across the sky.
- The sky turns dark as stars appear.

How do we analyze token relationship?

- Word Transition: which words lead to each other in a piece of writing?
- Co-occurrence: which words tend to appear together in a Transformer input/output context?
- Pointwise Mutual Information: how many times more often two words co-occur than if they were independent?

Token Relationship Example: Word Transition

- "co-occurrence" of window size 1

	above	dense	flows	forest	is	rises	river	sun	tall	the	through	trees	with
above	0	0	0	0	0	0	0	0	0	1	0	0	0
dense	0	0	0	0	0	0	0	0	0	0	0	0	1
flows	0	0	0	0	0	0	0	0	0	0	1	0	0
forest	0	0	0	0	1	0	0	0	0	0	0	0	0
is	0	1	0	0	0	0	0	0	0	0	0	0	0
rises	1	0	0	0	0	0	0	0	0	0	0	0	0
river	0	0	1	0	0	0	0	0	0	0	0	0	0
sun	0	0	0	0	0	1	0	0	0	0	0	0	0
tall	0	0	0	0	0	0	0	0	0	0	0	1	0
the	0	0	0	2	0	0	2	1	0	0	0	0	0
through	0	0	0	0	0	0	0	0	0	1	0	0	0
trees	0	0	0	0	0	0	0	0	0	0	0	0	0
with	0	0	0	0	0	0	0	0	1	0	0	0	0

Token Relationship Example: Word Transition

	above	dense	flows	forest	is	rises	river	sun	tall	the	through	trees	with
above	0	0	0	0	0	0	0	0	0	1	0	0	0
dense	0	0	0	0	0	0	0	0	0	0	0	0	1
flows	0	0	0	0	0	0	0	0	0	0	1	0	0
forest	0	0	0	0	1	0	0	0	0	0	0	0	0
is	0	1	0	0	0	0	0	0	0	0	0	0	0
rises	1	0	0	0	0	0	0	0	0	0	0	0	0
river	0	0	1	0	0	0	0	0	0	0	0	0	0
sun	0	0	0	0	0	1	0	0	0	0	0	0	0
tall	0	0	0	0	0	0	0	0	0	0	0	1	0
the	0	0	0	2	0	0	2	1	0	0	0	0	0
through	0	0	0	0	0	0	0	0	0	1	0	0	0
trees	0	0	0	0	0	0	0	0	0	0	0	0	0
with	0	0	0	0	0	0	0	0	1	0	0	0	0

Word “the”: Token frequency is 5, out-degree is 5, in-degree is 2

Observations

- There is significant patterns in token relationships
- Tokens are not equal (in terms of frequencies)

Token Relationship Example: Word Transition

	above	dense	flows	forest	is	rises	river	sun	tall	the	through	trees	with
above	0	0	0	0	0	0	0	0	0	1	0	0	0
dense	0	0	0	0	0	0	0	0	0	0	0	0	1
flows	0	0	0	0	0	0	0	0	0	0	1	0	0
forest	0	0	0	0	1	0	0	0	0	0	0	0	0
is	0	1	0	0	0	0	0	0	0	0	0	0	0
rises	1	0	0	0	0	0	0	0	0	0	0	0	0
river	0	0	1	0	0	0	0	0	0	0	0	0	0
sun	0	0	0	0	0	1	0	0	0	0	0	0	0
tall	0	0	0	0	0	0	0	0	0	0	0	1	0
the	0	0	0	2	0	0	2	1	0	0	0	0	0
through	0	0	0	0	0	0	0	0	0	1	0	0	0
trees	0	0	0	0	0	0	0	0	0	0	0	0	0
with	0	0	0	0	0	0	0	0	1	0	0	0	0

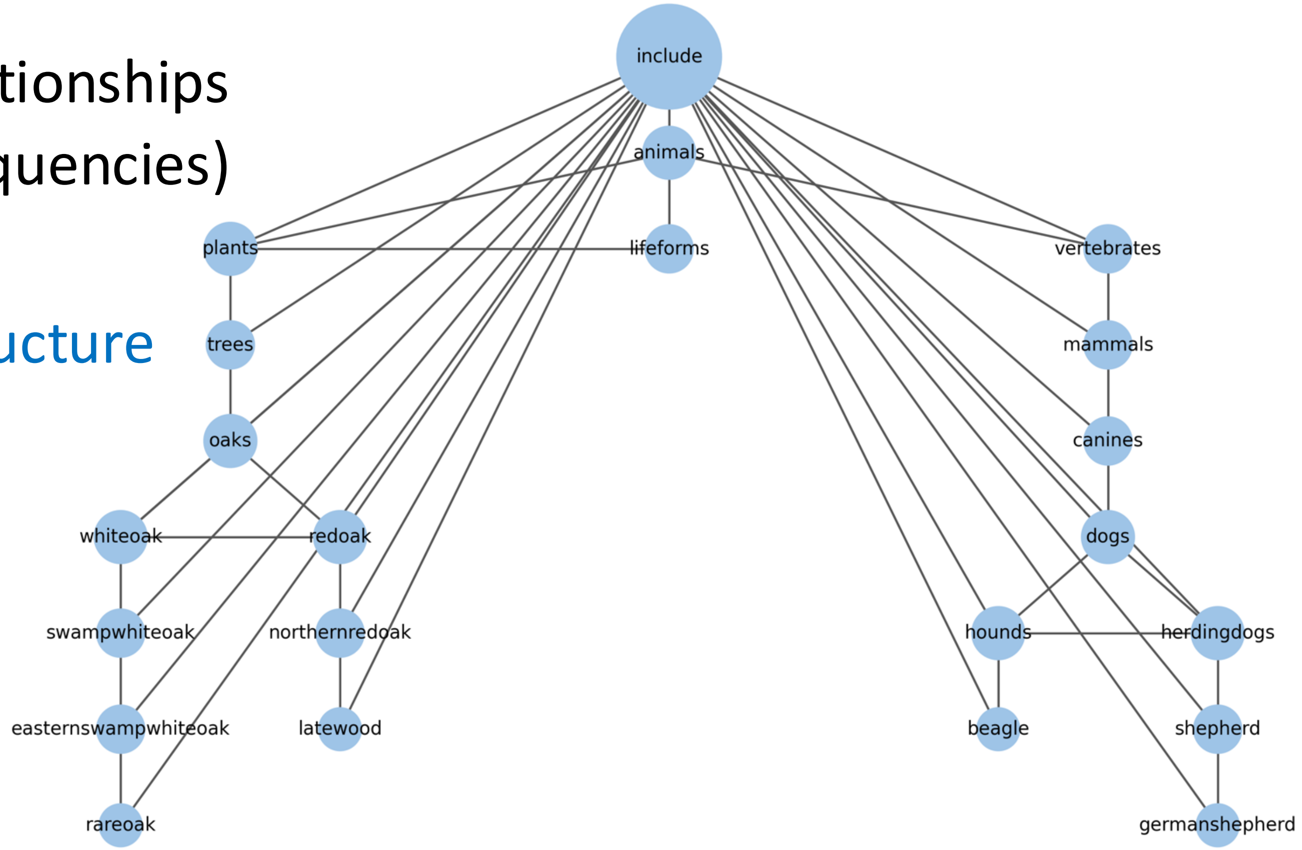
Most other token frequency, out/in degree are 1 or 0

Observations

- There is significant patterns in token relationships
- Tokens are not equal (in terms of frequencies)

Token Relationship Example: Word Transition

- Observations
- There is significant patterns in token relationships
  - Tokens are not equal (in terms of frequencies)
  - Co-occurrence(sentence-wise)
- Tokens have underlying (hierarchical) structure



"Lifeforms include animals and plants. Animals include vertebrates. Vertebrates include mammals. Mammals include canines. Canines include dogs. Dogs include herdingdogs and hounds. Herdingdogs include shepherd. Shepherd include German shepherd. Hounds include beagle. Plants include trees. Tress include oaks. Oaks include white oak and red oak. White oak include swamp white oak. Swamp white oak include Eastern swamp white oak. Eastern swamp white oak include rare oak. Red oak include Northern red oak. Northern red oak include late wood... "

Quantitate Analysis: Hyperbolicity

A four points interpretation:

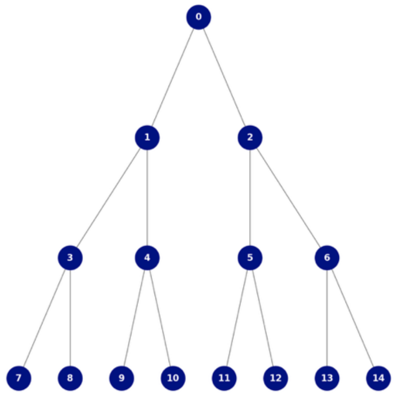
Define $(x, y)_w = d(w, x) + d(w, y) - d(x, y)$

$$\delta = \frac{1}{2}\sup\{\min\{(x, y)_w, (y, z)_w\} - (x, z)_w\}$$

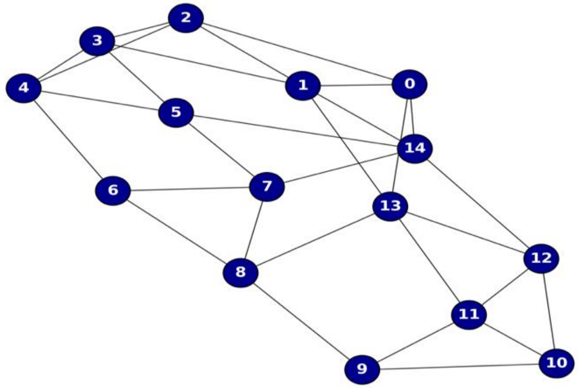
for any four points x, y, z, w

Hyperbolicity quantifies the distance of a graph from a tree-like structure

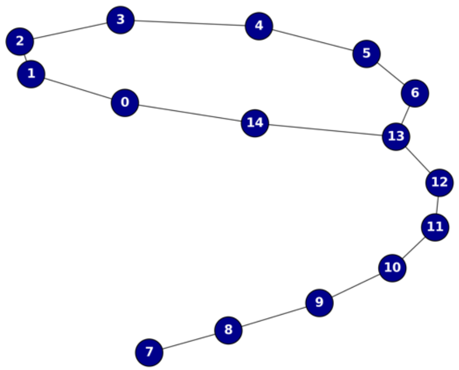
Quantitate Analysis: Hyperbolicity (2)



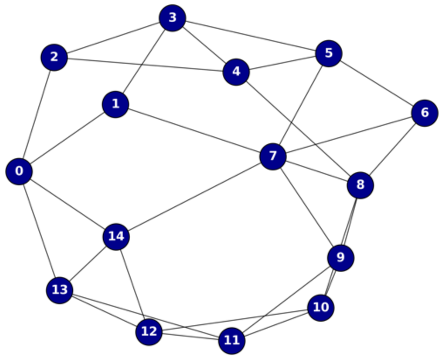
Hyperbolicity(∂)=0



Hyperbolicity(∂)=0.5



Hyperbolicity(∂)=0.25



Hyperbolicity(∂)=0.75

∂ = 0, tree-like structure, no cycles.

∂ = 0.25, one cycle, slight deviation from tree metric.

∂ = 0.5, moderate interconnectedness, more loops.

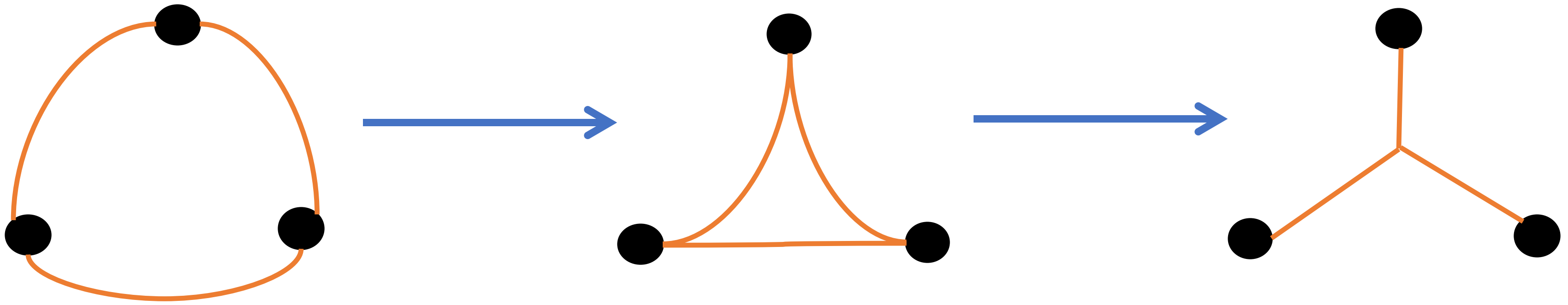
∂ = 0.75, dense structure, multiple loops, far from a tree.

Smaller hyperbolicity indicates fewer cycles, with certain nodes playing crucial roles.

Quantitate Analysis: Hyperbolicity (3)

Deviation from Tree metric: The above is can be seen as picking a base point *w* and see what kind of triangles can be drawn

- Turns out, the smaller the δ value, the *thinner are the allowed triangles*
- In a metric space, δ measure how thin are the thickest triangles



This is a measure of how much a metric space deviates from a tree metric: low hyperbolicity means thin and long triangles facing the same direction with increasingly more points distributed further from the origin

Hierarchies in LLM Token Distribution

- Hyperbolicity (0-1): measures how much data points are tree-like (hierarchical)
  - Lower values indicate more hierarchical distribution

Table 2. δ-Hyperbolicity of the token embedding in various LLMs across several datasets.

Model	arXiv	C4	Common Crawl	GitHub	StackExchange	Wikipedia
RoBERTa-Base (Liu et al., 2019b)	0.15 ± 0.06	0.18 ± 0.04	0.17 ± 0.04	0.12 ± 0.04	0.17 ± 0.07	0.07 ± 0.05
LLaMA3.1-8B (Grattafiori et al., 2024)	0.15 ± 0.05	0.16 ± 0.07	0.15 ± 0.06	0.12 ± 0.05	0.18 ± 0.06	0.10 ± 0.04
GPT-NeoX-20B (Black et al., 2022)	0.14 ± 0.03	0.17 ± 0.06	0.15 ± 0.05	0.11 ± 0.04	0.14 ± 0.04	0.09 ± 0.03
Gemma2-9B (Team et al., 2024)	0.17 ± 0.06	0.19 ± 0.04	0.20 ± 0.05	0.15 ± 0.05	0.18 ± 0.04	0.15 ± 0.03

Indicates hierarchical structure in token distribution

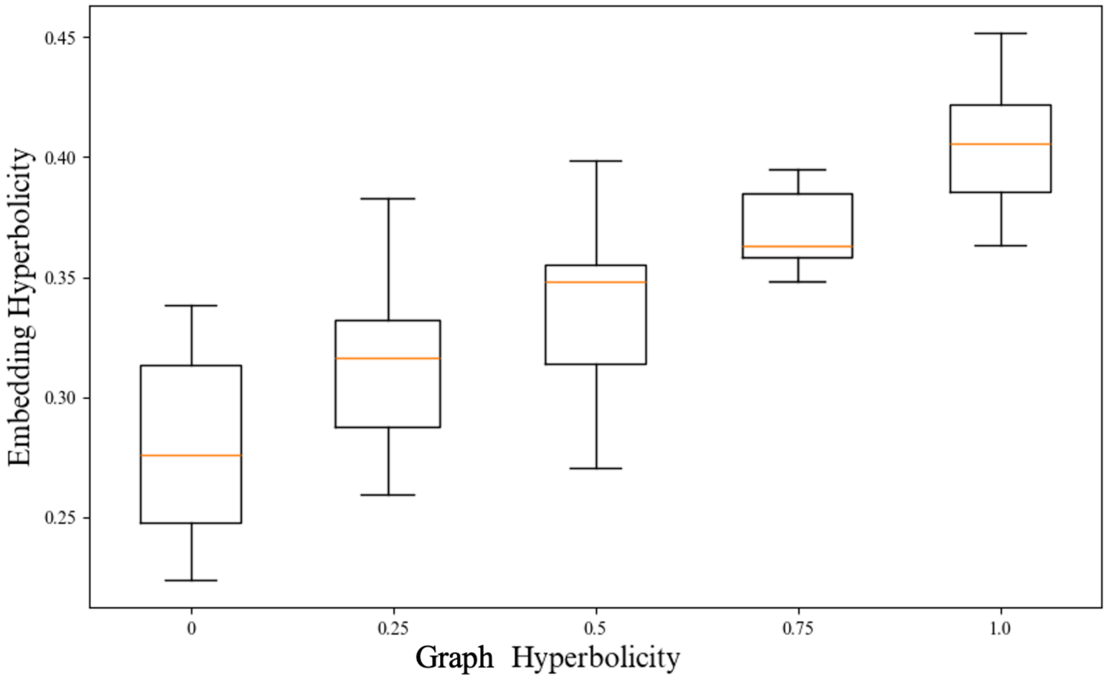
Reference values

Table 3. Hyperbolicity values δ for different metric spaces.

	Sphere Space	Dense Graph	PubMed Graph	Poincare Space	Tree Graph
δ	0.99 ± 0.01	0.62 ± 0.01	0.40 ± 0.04	0.14 ± 0.01	0.0

References: Neil He, Jiahong Liu, Buze Zhang, Ngoc Bui, Ali Maatouk, Menglin Yang, Irwin King, Melanie Weber, and Rex Ying. 2025. Position: Beyond Euclidean–Foundation Models Should Embrace Non-Euclidean Geometries. arXiv:2504.08896 (2025).

Embedding Hyperbolicity vs Graph Hyperbolicity



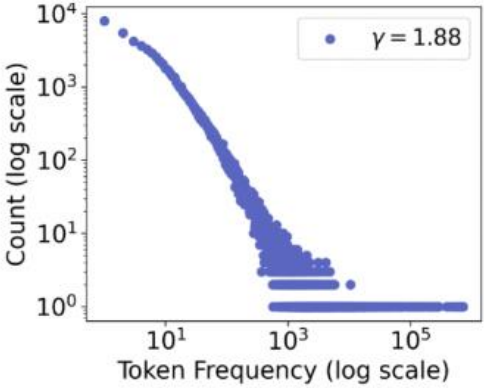
Positive correlation between graph hyperbolicity and embedding hyperbolicity

Compute token embedding hyperbolicity as a proxy for structure; lower values indicate a more tree-like shape.

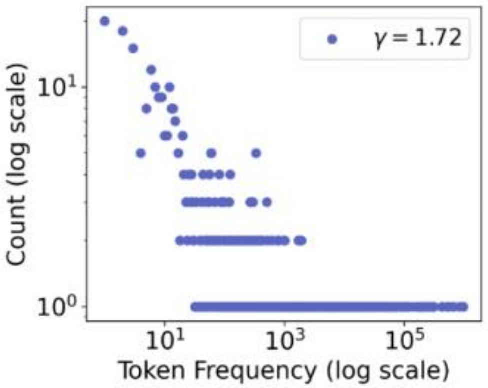
Scale-Free Property in Token Relationships

- Scale-free property across foundation models and modalities
  - Very few (exponentially) tokens appear very frequently/have large norm

Token Frequency (x-axis) v.s. Token count (y-axis)
"How many tokens appears *x* number of times"

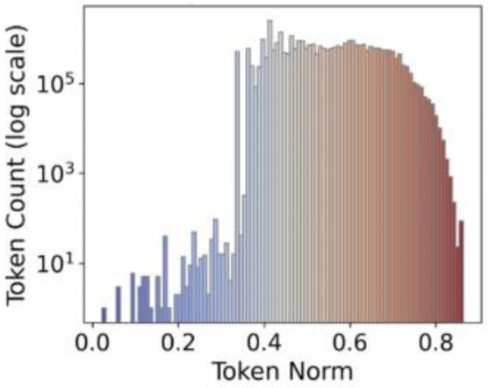


LLaMa3.1-8B

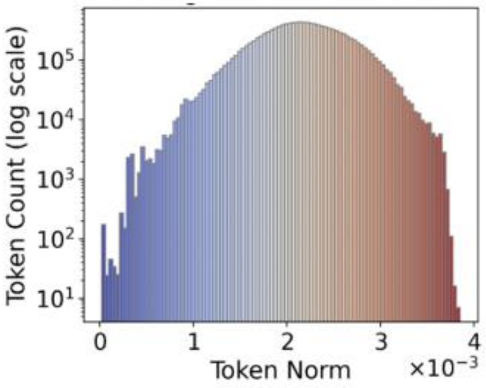


LLaMaGen

Token norm (x-axis) v.s. Token count (y-axis)
"How many time does a token with norm of value *x appear*"



LLaMa3.1-8B



LLaMaGen

Corpus: RedPajama (subset) (arXiv, C4, Common Crawl, GitHub, Wikipedia, and StackExchange); Mathematical Reasoning (GSM8K, MATH50K, MAWPS, SVAMP); Common Sense Reasoning (BoolQ, WinoGrande, OpenBookQA)

References: Neil He, Jiahong Liu, Buze Zhang, Ngoc Bui, Ali Maatouk, Menglin Yang, Irwin King, Melanie Weber, and Rex Ying. 2025. Position: Beyond Euclidean–Foundation Models Should Embrace Non-Euclidean Geometries. arXiv:2504.08896 (2025).

Embedding Norm vs Token Frequency

Table 7: Mean, Minimum, and Maximum Norm Values for Different Models and Groups

Model	Group	Norm (Mean (Min~Max))
LLaMA-7B	Group 1: *to, have, in, that, and, is, for*	0.95 (0.79~1.06)
	Group 2: *how, much, many, time, cost*	1.22 (1.12~1.30)
	Group 3: *animals, fruit, numbers, items, colors*	1.36 (1.32~1.43)
	Group 4: *dog, cow, apple, hours, dollars, minute, second, shoes, purple, bananas, puppies*	1.37 (1.31~1.44)
LLaMA-13B	Group 1: *to, have, in, that, and, is, for*	1.03 (0.83~1.26)
	Group 2: *how, much, many, time, cost*	1.43 (1.35~1.49)
	Group 3: *animals, fruit, numbers, items, colors*	1.50 (1.46~1.54)
	Group 4: *dog, cow, apple, hours, dollars, minute, second, shoes, purple, bananas, puppies*	1.50 (1.47~1.57)
Gemma-7B	Group 1: *to, have, in, that, and, is, for*	3.16 (3.06~3.30)
	Group 2: *how, much, many, time, cost*	3.56 (3.49~3.63)
	Group 3: *animals, fruit, numbers, items, colors*	3.84 (3.71~3.92)
	Group 4: *dog, cow, apple, hours, dollars, minute, second, shoes, purple, bananas, puppies*	4.03 (3.43~4.82)
LLaMA3-8B	Group 1: *to, have, in, that, and, is, for*	0.35 (0.33~0.40)
	Group 2: *how, much, many, time, cost*	0.46 (0.39~0.50)
	Group 3: *animals, fruit, numbers, items, colors*	0.53 (0.51~0.55)
	Group 4: *dog, cow, apple, hours, dollars, minute, second, shoes, purple, bananas, puppies*	0.59 (0.50~0.70)

References: Menglin Yang, Aosong Feng, Bo Xiong, Jihong Liu, Irwin King, and Rex Ying. 2024. Hyperbolic Fine-tuning for Large Language Models. ICML LLM Cognition Workshop (2024).

Embeddings Space Choices

- The **embedding space** is crucial for a model to faithfully represent such relationships between data points
  - Should Euclidean geometry remain the de facto choice for foundation models?

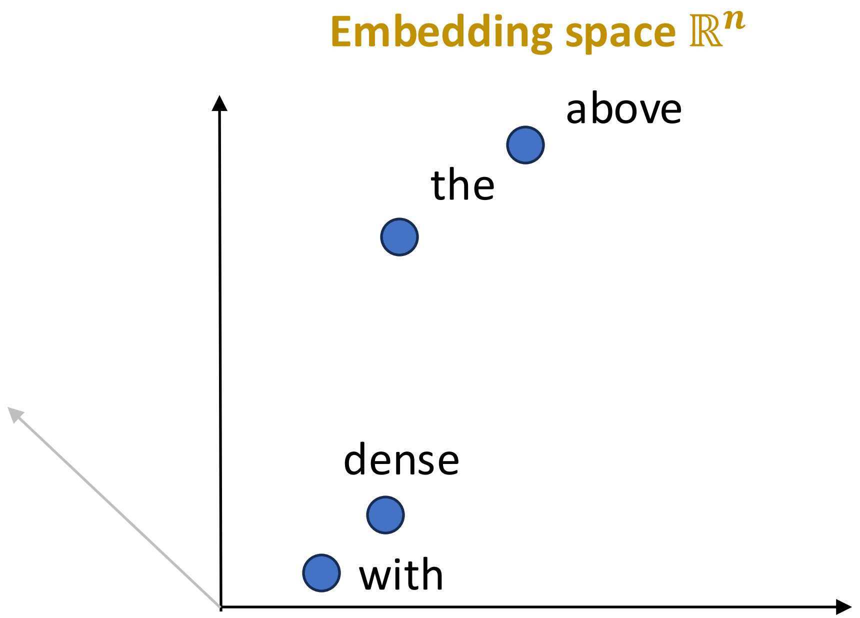
Embeddings Space Intuition

	above	dense	flows	forest	is	rises	river	sun	tall	the	through	trees	with
above	0	0	0	0	0	0	0	0	0	1	0	0	0
dense	0	0	0	0	0	0	0	0	0	0	0	0	1

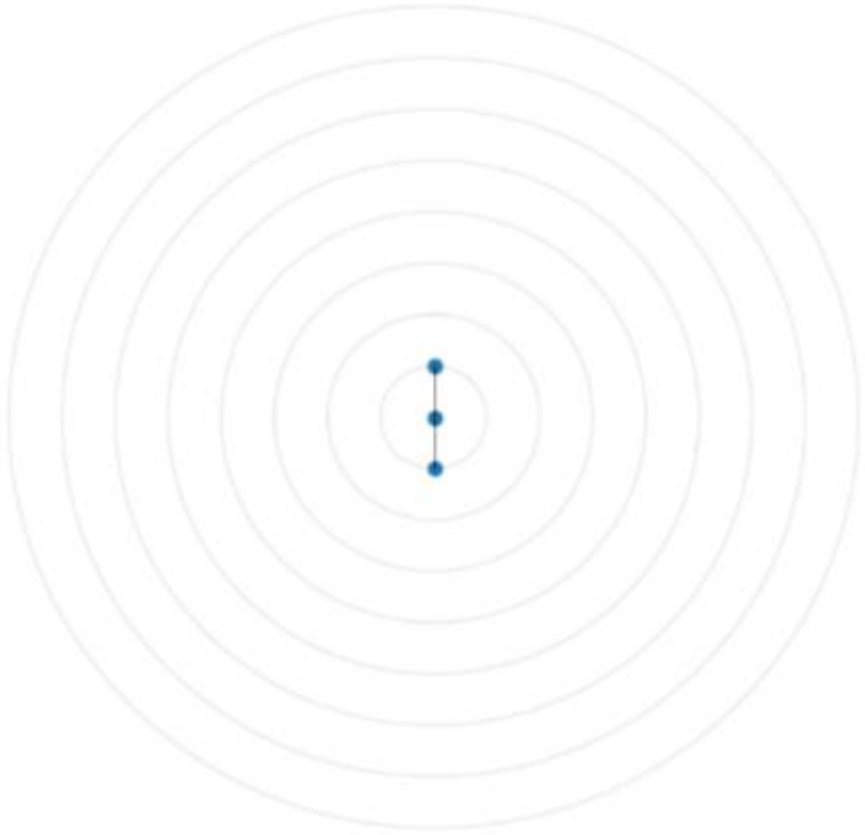
Attention score: computed through *inner product/cosine similarity*

Intuition: Co-occurring words should be embedded closer together!

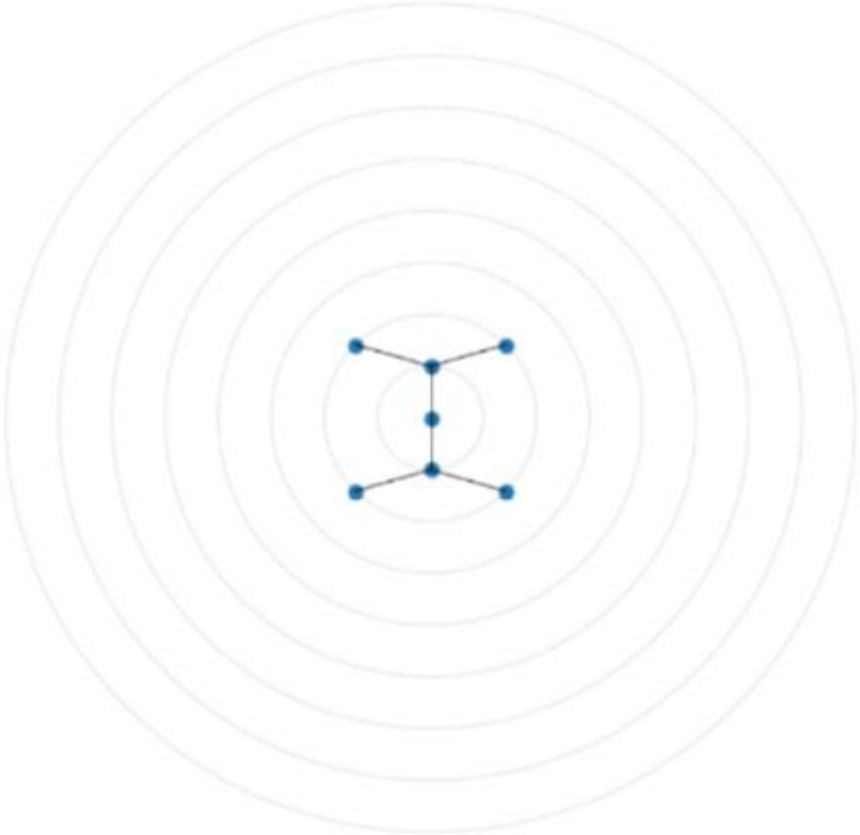
- Frequently co-occurring should attend more to each other !



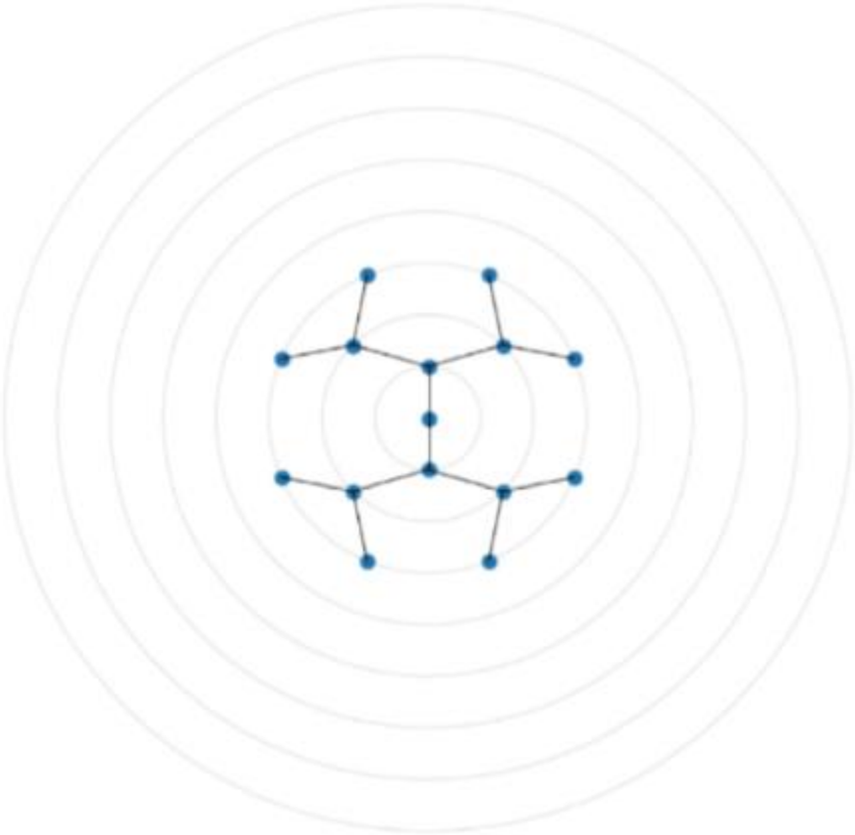
Example: Embedding Tree-structured Data



Example: Embedding Tree-structured Data

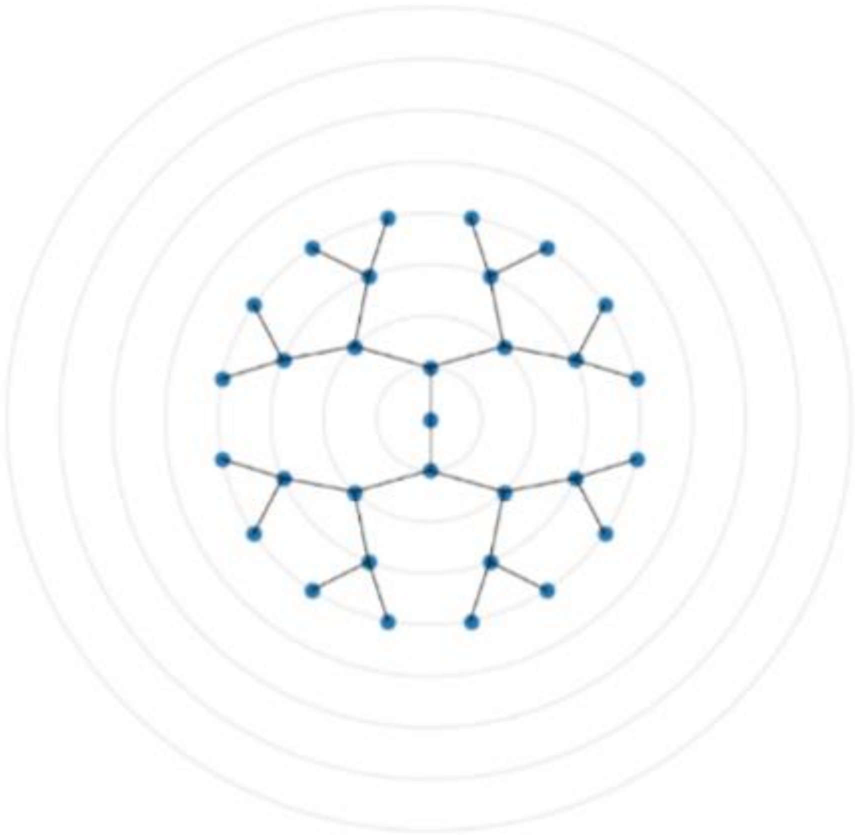


Example: Embedding Tree-structured Data

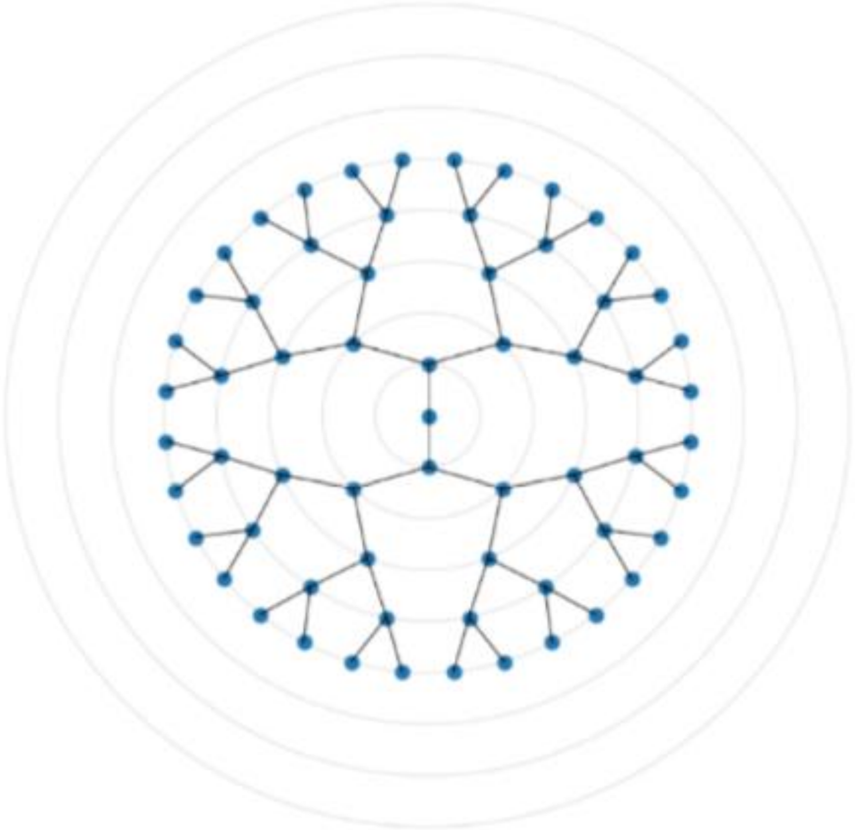


So far, so good
Nodes are close i.f.f. they are connected by an edge

Example: Embedding Tree-structured Data



Example: Embedding Tree-structured Data

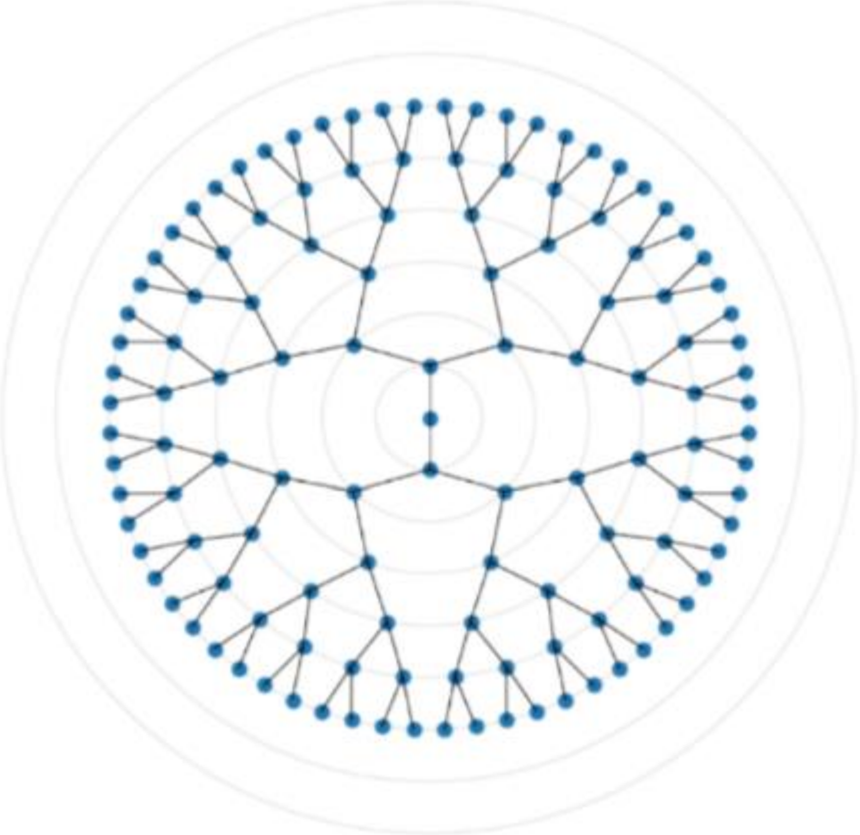


But the outermost nodes are becoming increasingly close to one another.

….

Even though they are not connected by an edge in the graph.

Example: Embedding Tree-structured Data

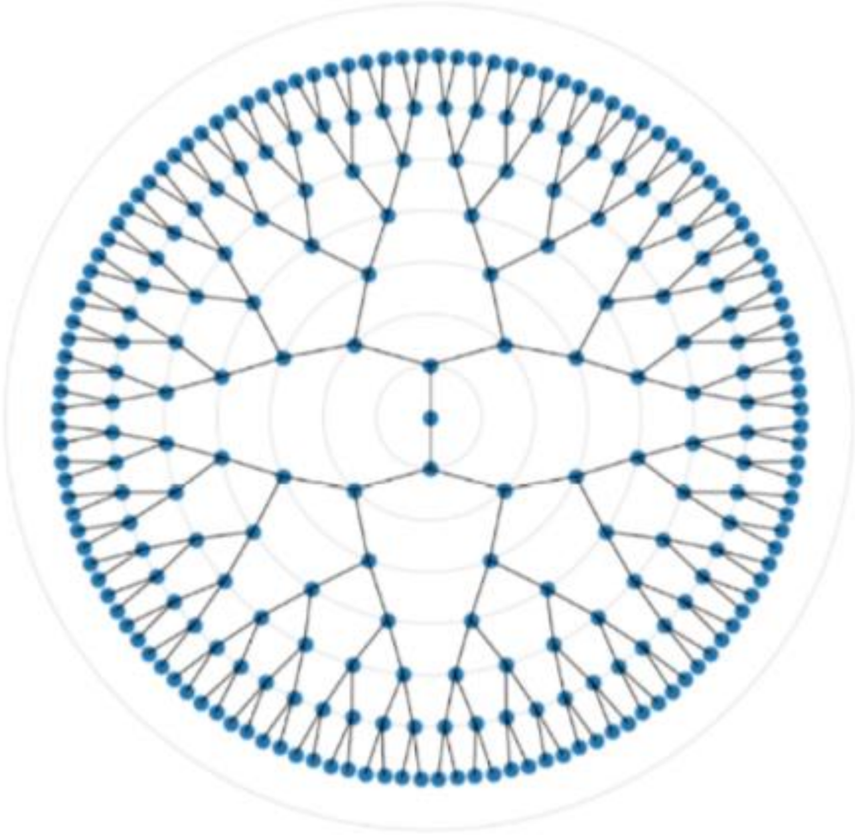


But the outermost nodes are becoming increasingly close to one another.

….

Even though they are not connected by an edge in the graph.

Example: Embedding Tree-structured Data



Things only get worse!
We have lost our property:

“close i.f.f share edge”

Issues with Euclidean Embeddings: Distortion

- Euclidean space leads to ***significant distortion*** regardless of the embedding dimensions

Theorem

(Informal; Lee et al., (2007)) There is a lower bound in the minimal distortion of embedding hierarchical structures (e.g. token relationships) into Euclidean space ($\mathbb{R}^n$).

"There is a ***performance bottleneck*** on how well Euclidean foundation models can represent complex token relationships"

Issues with Euclidean Embeddings: Dimension Dilemma

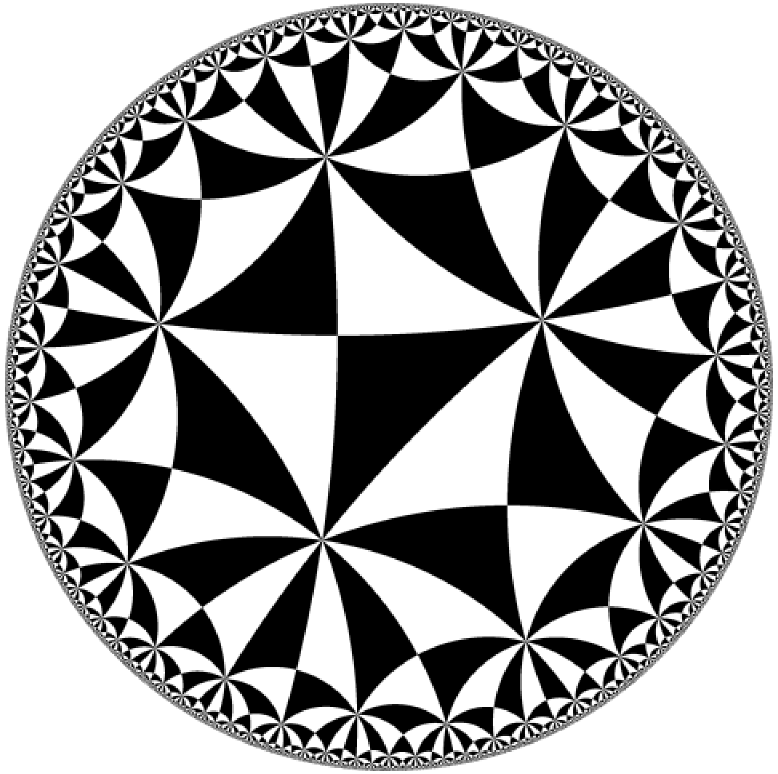
- Euclidean space face the dilemma of **dimension-distortion tradeoffs**
  - High dimensionality is often required to embed complex token relations in Euclidean space with (relatively) low distortion

Theorem

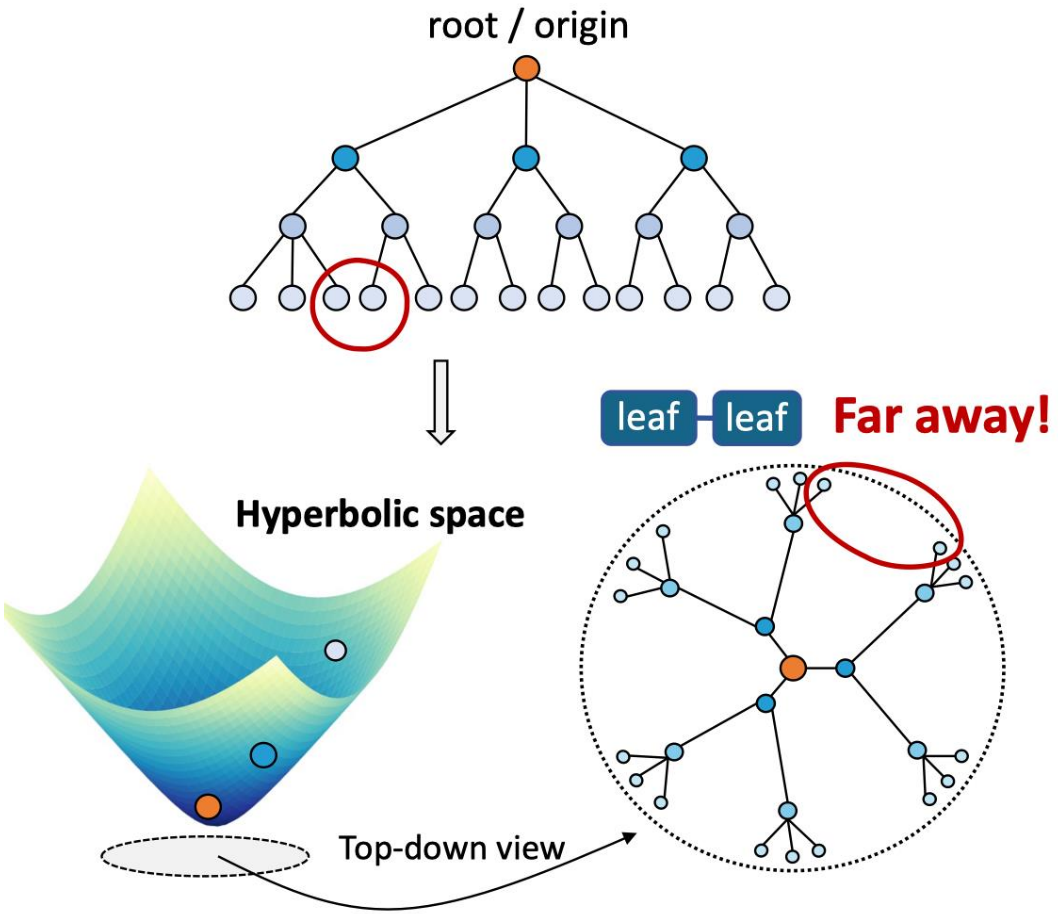
(Informal; Matoušek (2002)) The dimension required when embedding unweighted graphs (in the form of token relationships/self-attention) grows **near-quadratically** *w.r.t to distortion.*

"Euclidean foundation models have ***limited scalability***"

Potential Solution: Hyperbolic Embedding Space



The volume of a ball in the hyperbolic space grows **exponentially** with its radius

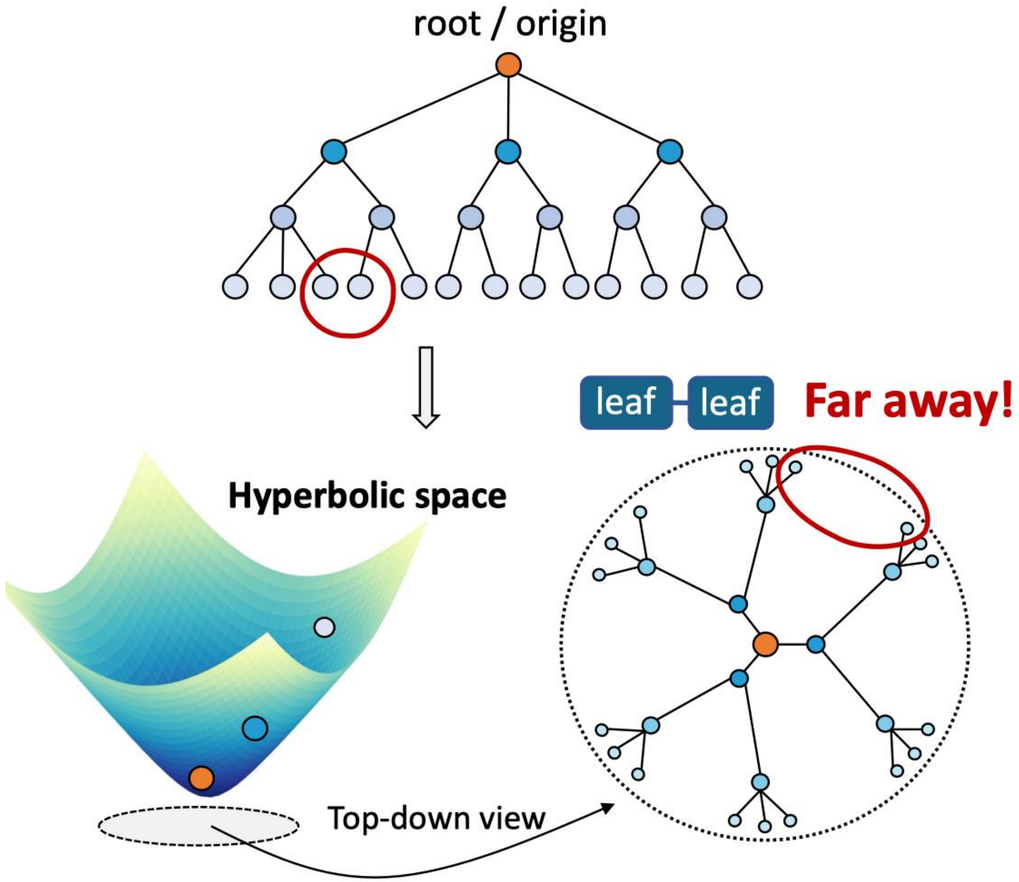


Hyperbolic Geometry for Foundation Models

We need an embedding space that can ***better represent token relationship***!

- The distance between low-level tokens on different branches should be maximized and far away
- The distance between a high-level token and a low-level token should be minimized and close

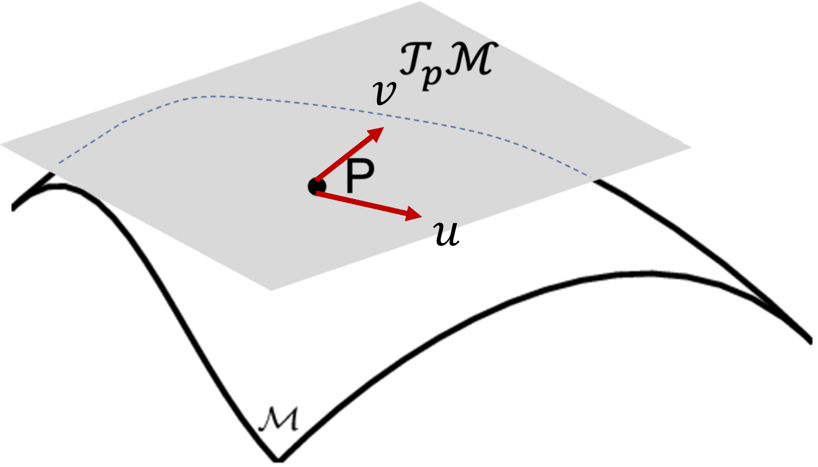
- Solution: any tree (i.e. hierarchical distribution) can be embedded into ***hyperbolic space*** with ***arbitrarily low distortion***!!



Riemannian Manifold

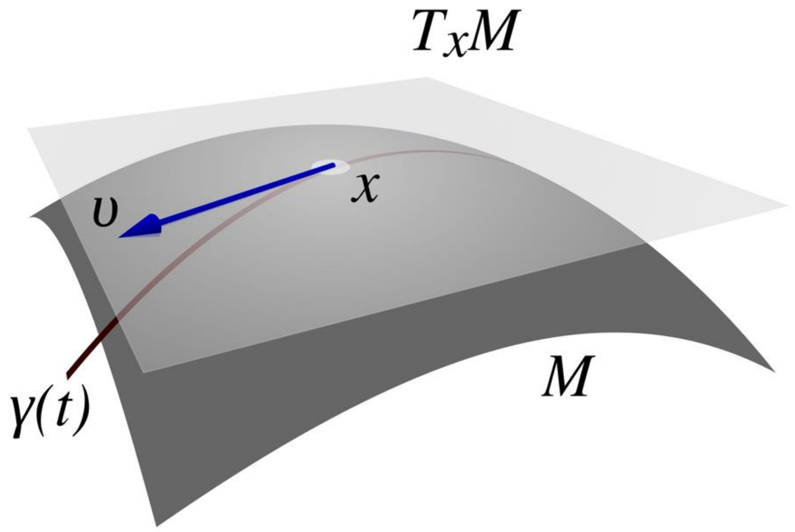
- **Manifold**: high-dimensional surface
- **Riemannian Manifold $\mathcal{M}$**
  - Equipped with
    - *Tangent space $\mathcal{T}_p\mathcal{M}$*: an $\mathbb{R}^d$ that approximates the manifold at any point $p \in \mathcal{M}$
    - *Inner product g_p*: $\mathcal{T}_p\mathcal{M} \times \mathcal{T}_p\mathcal{M}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

$u, v \in \mathcal{T}_p\mathcal{M},\ g_p(u, v) \in \mathbb{R}$



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 $\mathcal{M}$

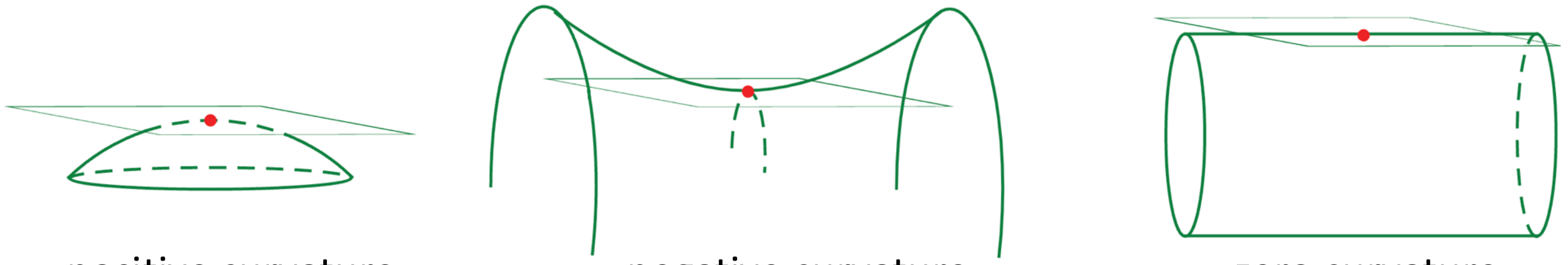


Curvature

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

High-level Intuition:

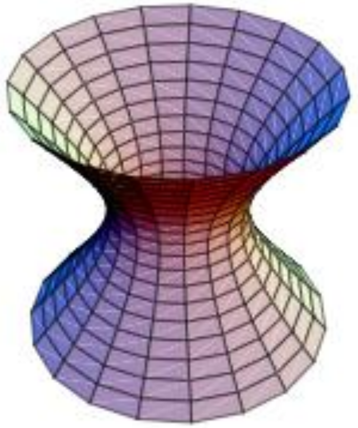
- If the surface locally lives **entirely on one side** of the tangent space $\mathcal{T}_p\mathcal{M}$ ⇒ **Positive** curvature at point p
- If the tangent space $\mathcal{T}_p\mathcal{M}$ **cuts through** the surface ⇒ **Negative** curvature at point p
- If the surface has a line along which the **surface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



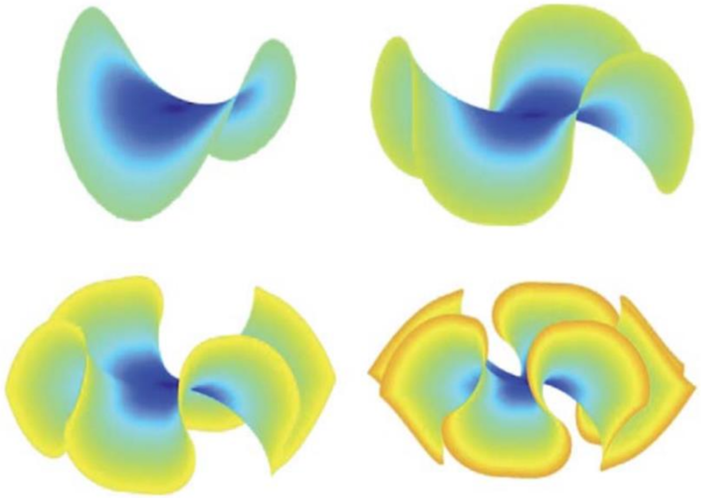
positive curvature negative curvature zero curvature

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:



One-sheet hyperboloid

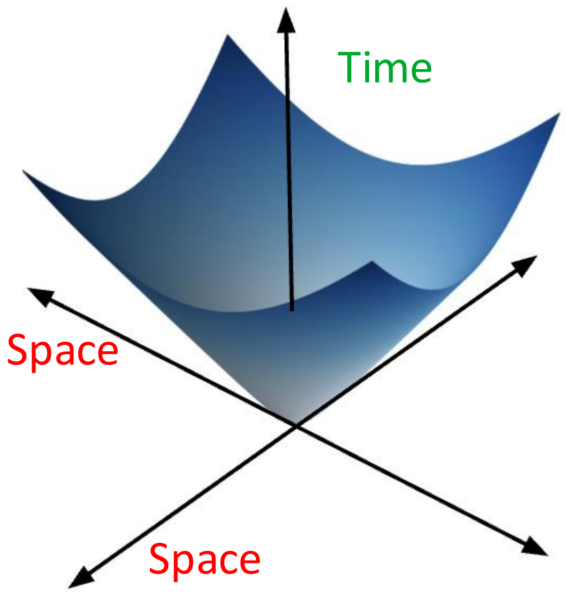


Periodic Amsler Surfaces

Hyperbolic Space and Minkowski Space

- Hyperbolic space can be naturally embedded into a **Minkowski Space**

- The **Minkowski metric** in the Minkowski space is different from the Euclidean metric.
  - **Euclidean Metric:** $g_E(\boldsymbol{u}, \boldsymbol{v}) = u_0 v_0 + u_1 v_1 + \cdots + u_d v_d$
  - **Minkowski Metric:** $g_M(\boldsymbol{u}, \boldsymbol{v}) = \pm(u_0 v_0 - u_1 v_1 - \cdots - u_d v_d)$
    - Without loss of generality we can take the $+$ sign
  - Note: dimension 1 is treated differently in Minkowski Space.



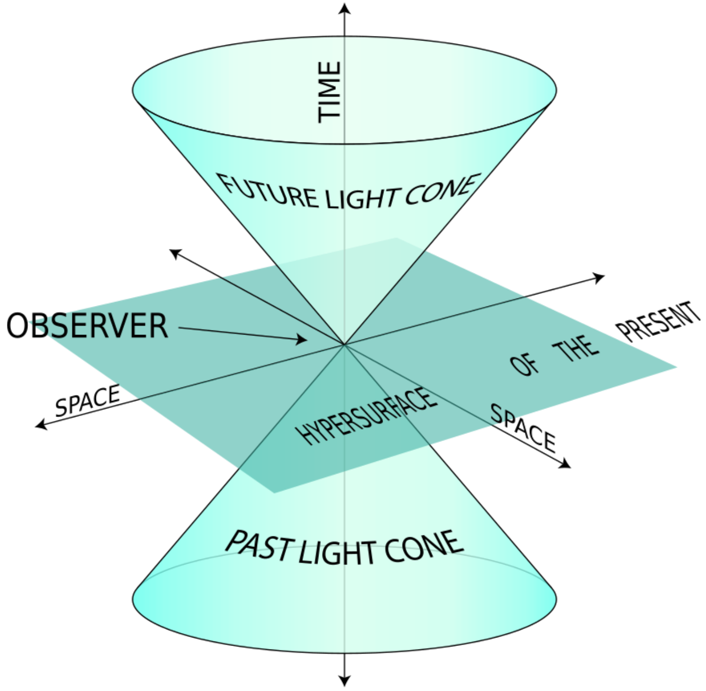
Inner Product

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

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

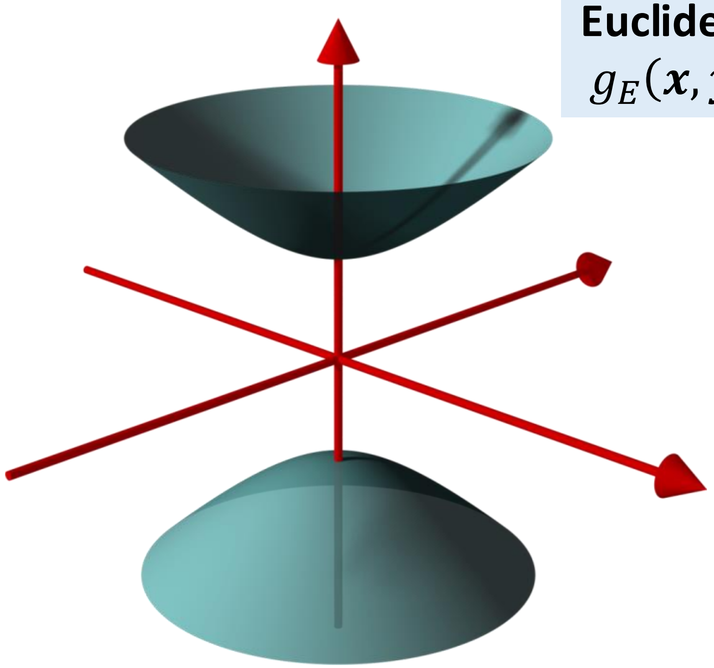
$$\langle \boldsymbol{x}, \boldsymbol{y} \rangle_{\mathcal{L}} = \boxed{-x_0 y_0} + \boxed{x_1 y_1 + \ldots + x_d y_d}$$

Time-like | 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 in Different Spaces



Euclidean Metric
$g_E(\boldsymbol{x}, \boldsymbol{y}) = x_1y_1 + x_2y_2 + x_3y_3$

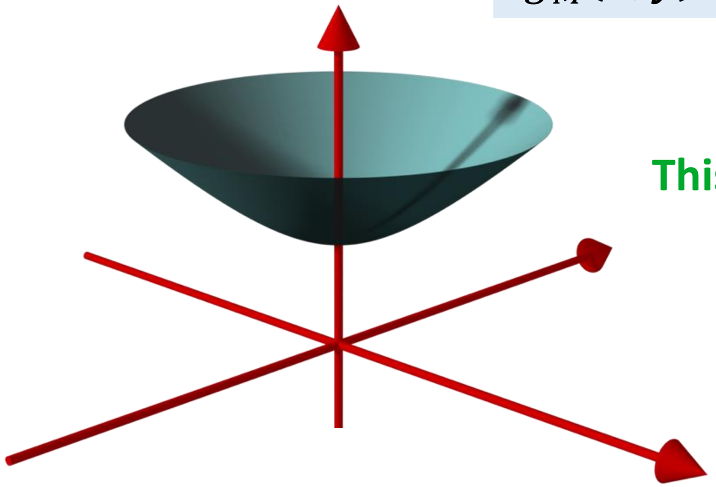
Two sheet hyperboloid in **3D Euclidean space**

Geodesic distance in Euclidean hyperboloid:

$d_E(\boldsymbol{x}, \boldsymbol{y}) = \sqrt{2(1 - g_E(\boldsymbol{x}, \boldsymbol{y}))}$

(with normalized $\boldsymbol{x}$ and $\boldsymbol{y}$)

Minkowski Metric
$g_M(\boldsymbol{x}, \boldsymbol{y}) = -x_1y_1 + x_2y_2 + x_3y_3$



This is hyperbolic

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

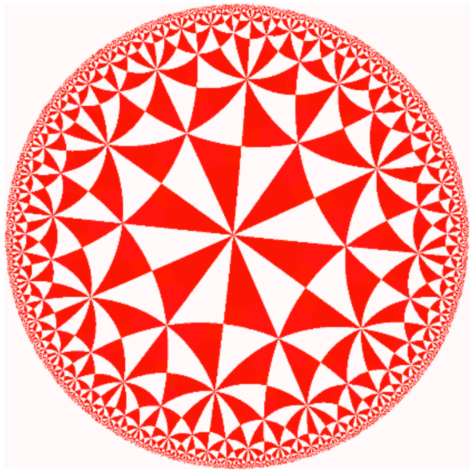
Geodesic distance in Minkowski hyperboloid:

$D_M^K(\boldsymbol{x}, \boldsymbol{y}) = \sqrt{K}\text{arcosh}\left(-\frac{g_M(\boldsymbol{x},\boldsymbol{y})}{K}\right)$

Performing deep learning operations in hyperbolic space is non-trivial

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

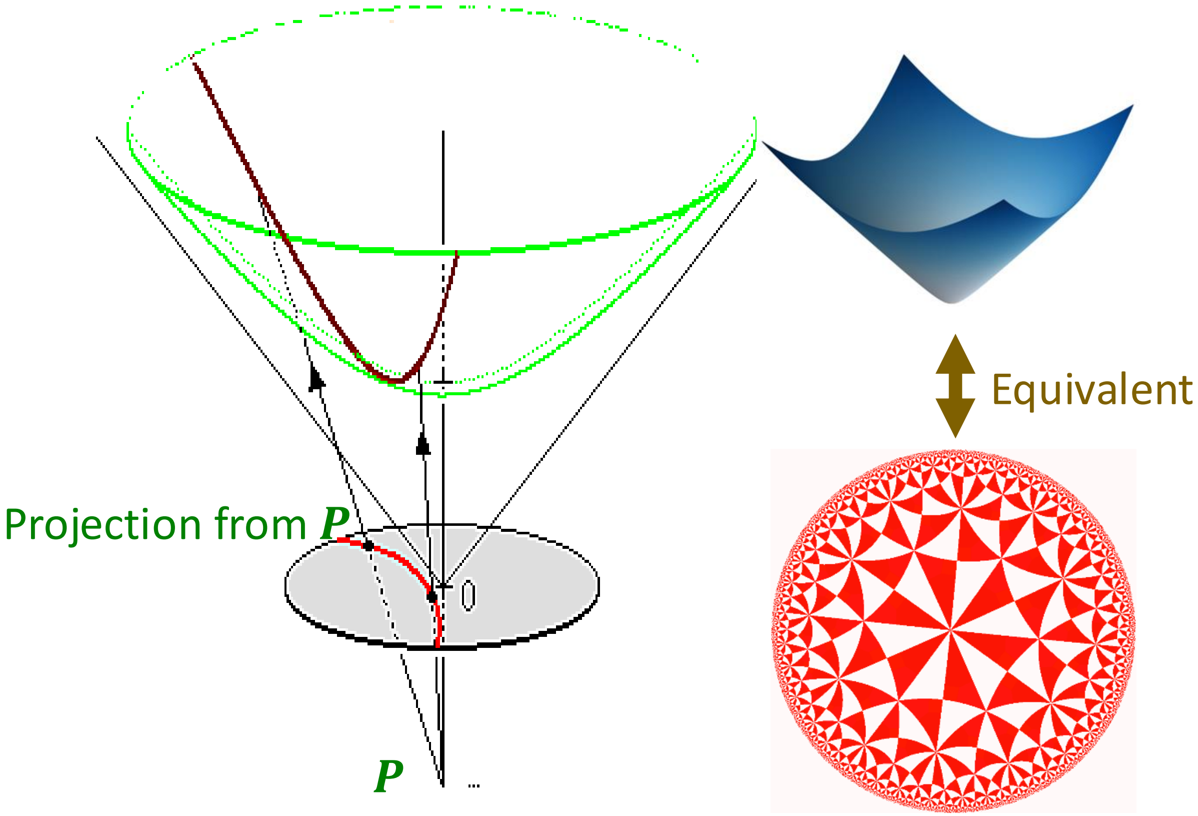


Poincaré: intuitive visualization

Other models exist as well, e.g. Klein model

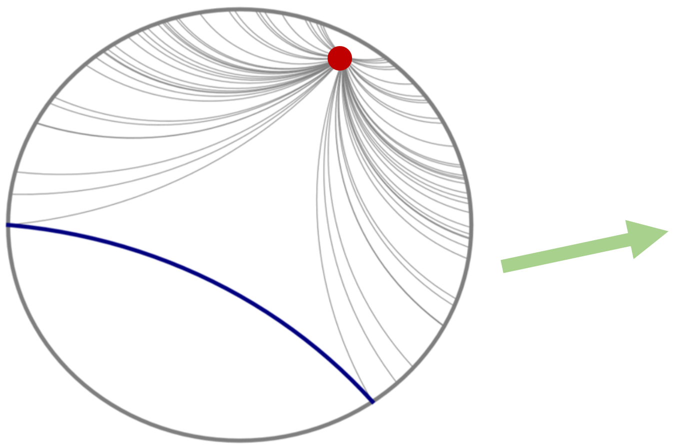
Equivalence

- 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

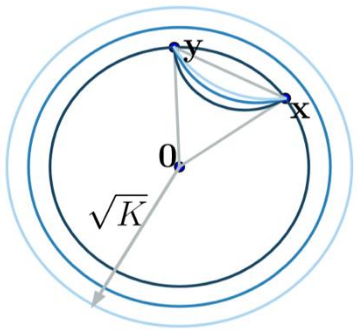
Geodesic Distance

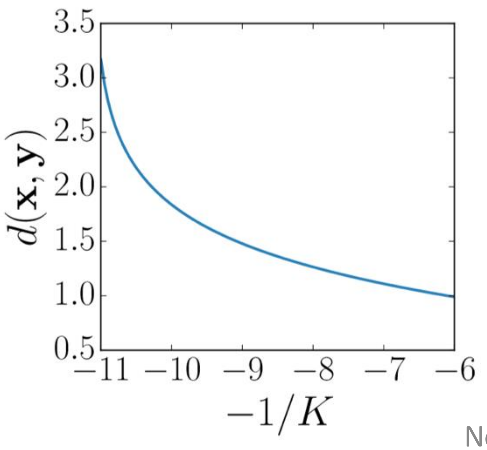
- **Geodesic distance** between $\boldsymbol{x}$ and $\boldsymbol{y}$ for $\mathbb{H}^{d,\mathrm{K}}$:

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

- Negative Lorentz Distance: $D_{\mathcal{L}}^{K}(\boldsymbol{x},\boldsymbol{y}) = \frac{1}{K} - 2\langle \boldsymbol{x},\boldsymbol{y}\rangle_{\mathcal{L}}$
- The more negative the curvature:
  - the more geodesics bends inward
  - geodesic distance increases

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

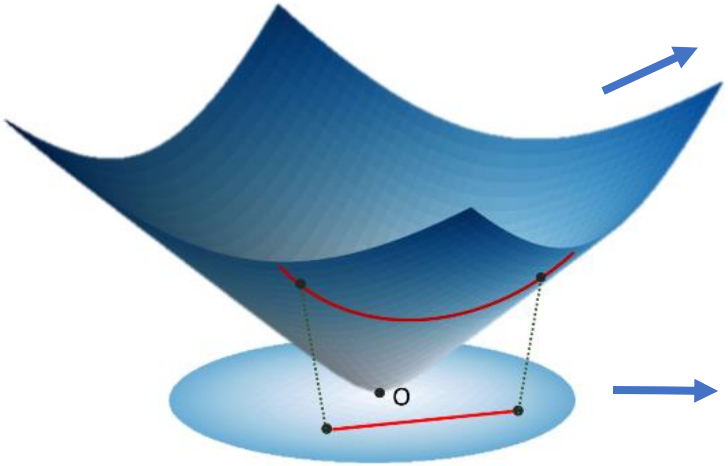




Dark blue: high curvature boundary and geodesics
Light blue: low curvature boundary and geodesics

Tangent Space

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



Hyperboloid model

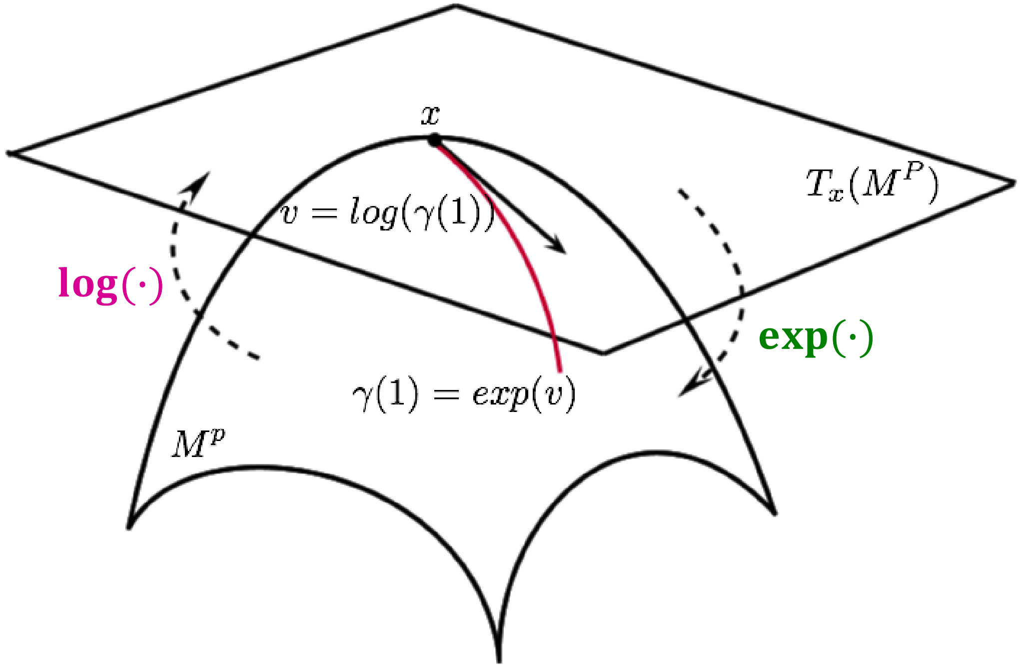
$\boldsymbol{d}$: hyperbolic space dimension

$\boldsymbol{K}$: 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

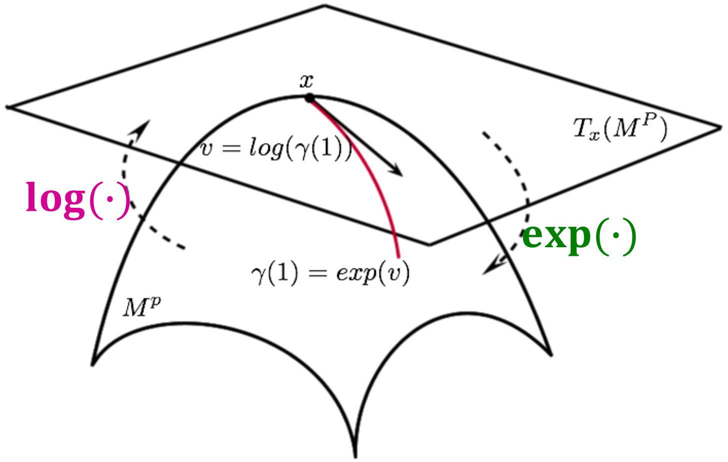


Exponential Map:

- 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:**

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

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

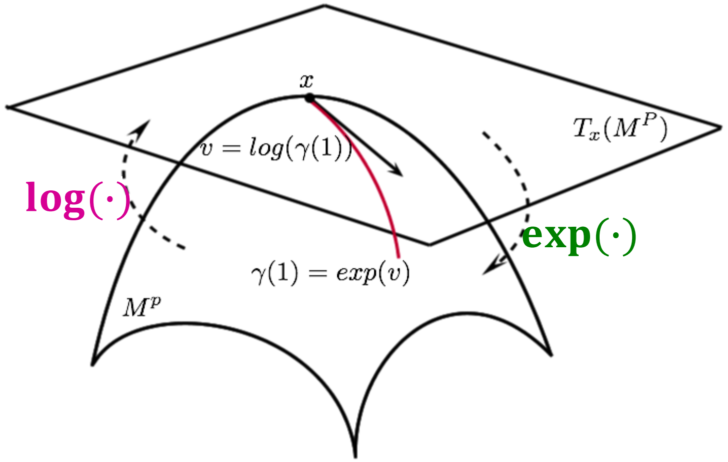


Logarithmic Map

- 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**:

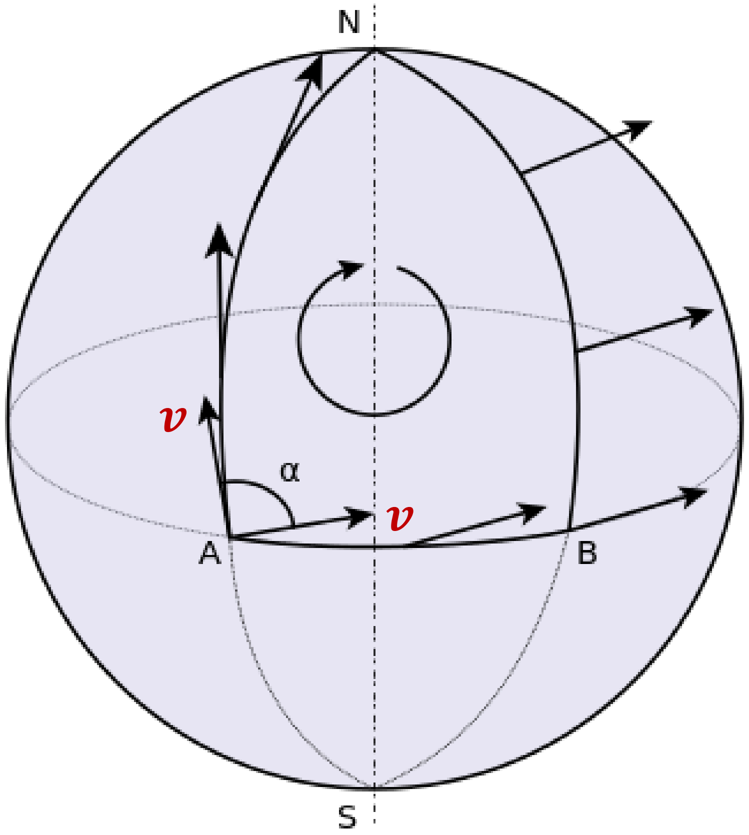
$$\log_{\boldsymbol{x}}^{K} \boldsymbol{y} = D_{\mathcal{L}}^{K}(\boldsymbol{x}, \boldsymbol{y}) \frac{\boldsymbol{y} + \frac{1}{K}\langle \boldsymbol{x}, \boldsymbol{y} \rangle_{\mathcal{L}} \boldsymbol{x}}{\left\| \boldsymbol{y} + \frac{1}{K}\langle \boldsymbol{x}, \boldsymbol{y} \rangle_{\mathcal{L}}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

  - $\boldsymbol{y} \in \mathbb{H}^{\mathrm{d,K}}$
  - $D_{\mathcal{L}}^{K}(\boldsymbol{x}, \boldsymbol{y}) = \sqrt{K} \operatorname{arcosh}(-\frac{\langle \boldsymbol{x}, \boldsymbol{y} \rangle_{\mathcal{L}}}{K})$ 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 $\mathcal{T}_{\boldsymbol{y}}\mathbb{H}^{\mathrm{d,K}}$:

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

  - $\log_{\boldsymbol{x}}^{\mathrm{K}}$ 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

Euclidean Embedding: Common Misunderstanding

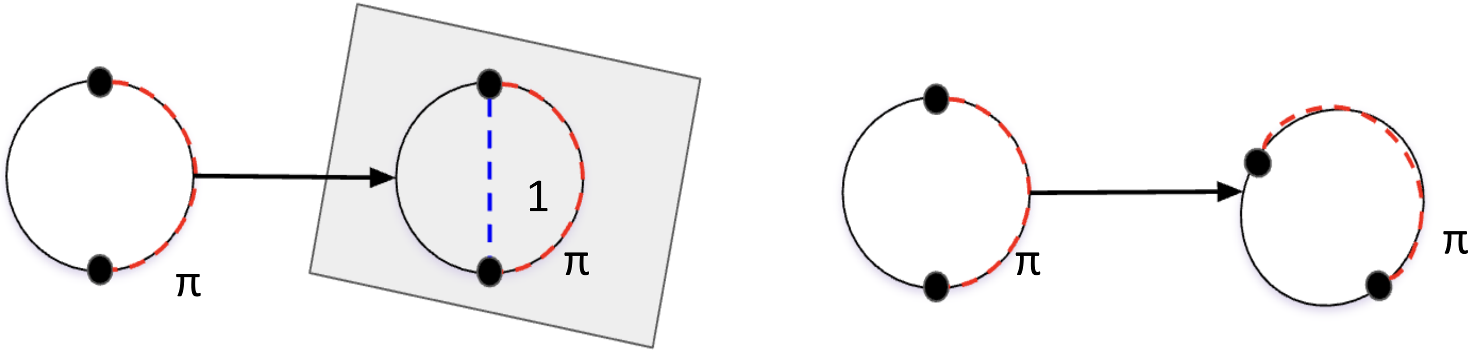
- Nash Embedding Theorem (and similar): roughly, any n-dimensional Riemannian manifold can be embedded in R^{2n}
  - This is an embedding of ***manifolds*** instead of ***metric spaces***, i.e. distance is still globally distorted

Isometric Embedding of Manifolds

- Shortest path between points are not necessarily the same globally
- e.g. Embedding sphere in Euclidean space

Isometric Embedding of Metric Spaces

- Distance between any two points (global behavior) is preserved in the new space
- e.g. Rotation

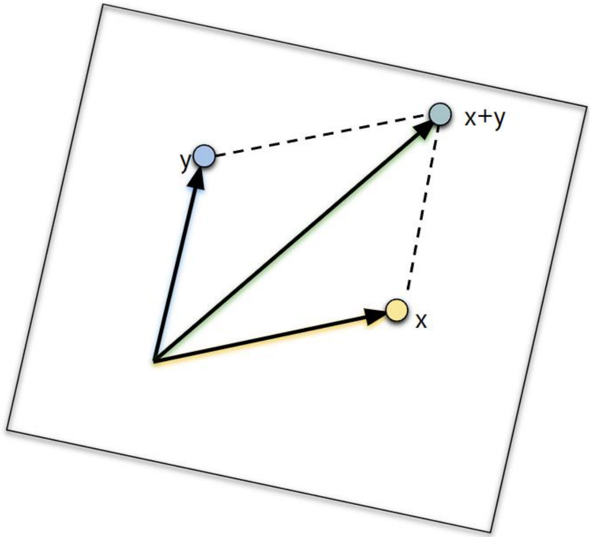


End of Part 1

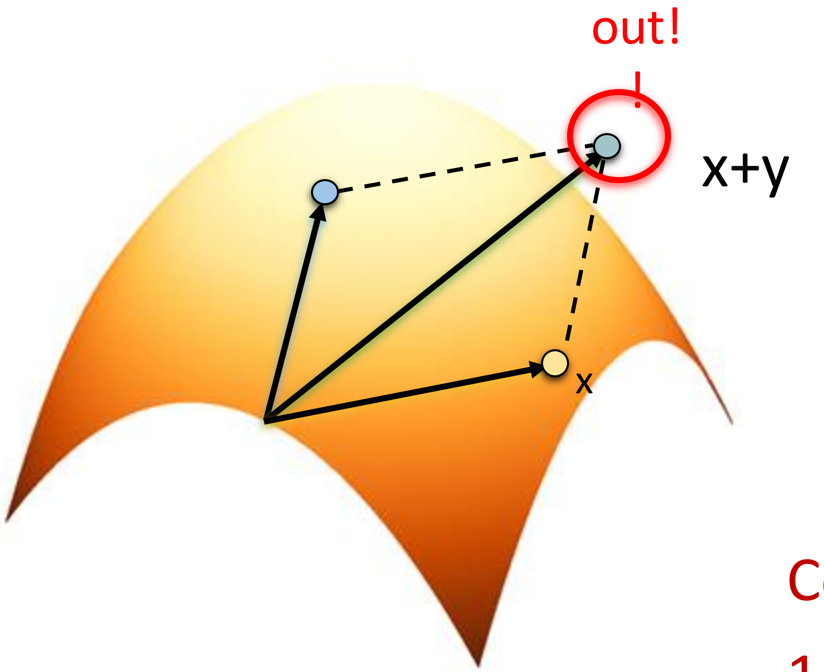
Part 2: Building Blocks for Hyperbolic Operations: Hyperbolic Neural Operations

Hyperbolic Operations: Difficulties

Addition in Euclidean Space



Addition in Hyperbolic Space?



Considerations:

1. Satisfy manifold constraints
2. Satisfy neural operation properties

Categorization of Hyperbolic Operations

In general, there are *two types* of hyperbolic operations:

- *Tangent-space-based operations*, which we will denote $f^{T,K}$
  - K is the curvature of the embedding space
  - T indicates the operation is implemented through the tangent-space-based method
- *Fully hyperbolic operations*, which we will denote $f^{F,K}$
  - K is still curvature
  - F indicates a fully hyperbolic operation

Strategy 1: Tangent-Space Based Operations (1)

Recall: The tangent space is an Euclidean space

- Intuition: we know how to perform Euclidean operations!

General Recipe: Use a Euclidean function $f: \mathbb{R}^{d+1} \to \mathbb{R}^{d+1}$ on the tangent space

- e.g. Linear transformer: $f(x) = Wx + b$, non-linear activation: $f(x) = ReLU(x)$

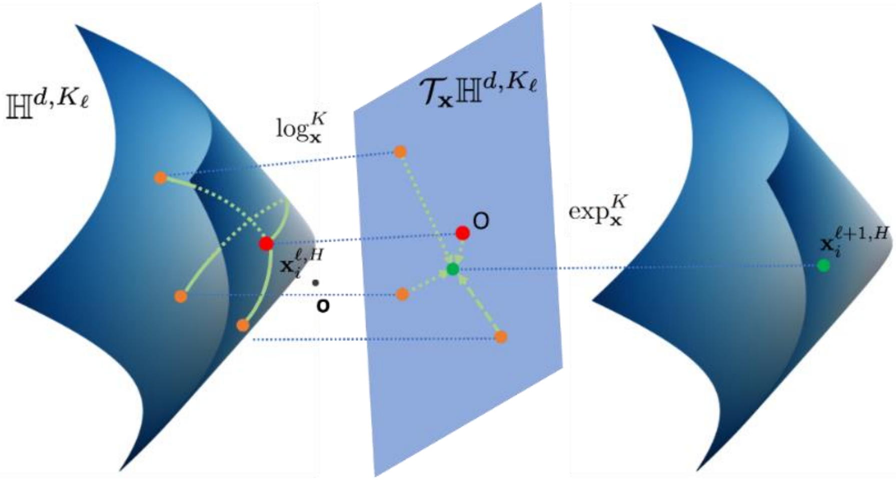


Image Source: Chami, Ines, et al. "Hyperbolic graph convolutional neural networks." Advances in neural information processing systems 32 (2019).

Strategy 1: Tangent-Space Based Operations (2)

Map input to tangent space of the origin, so f is a valid operation

↓

Perform Euclidean operation

↓

Lift the output back to $\mathbb{H}^{d,K}$

↓

$$f^{T,K}(x) = \exp_{\boldsymbol{o}}^{K}(f(\log_{\boldsymbol{o}}^{K}(x)))$$

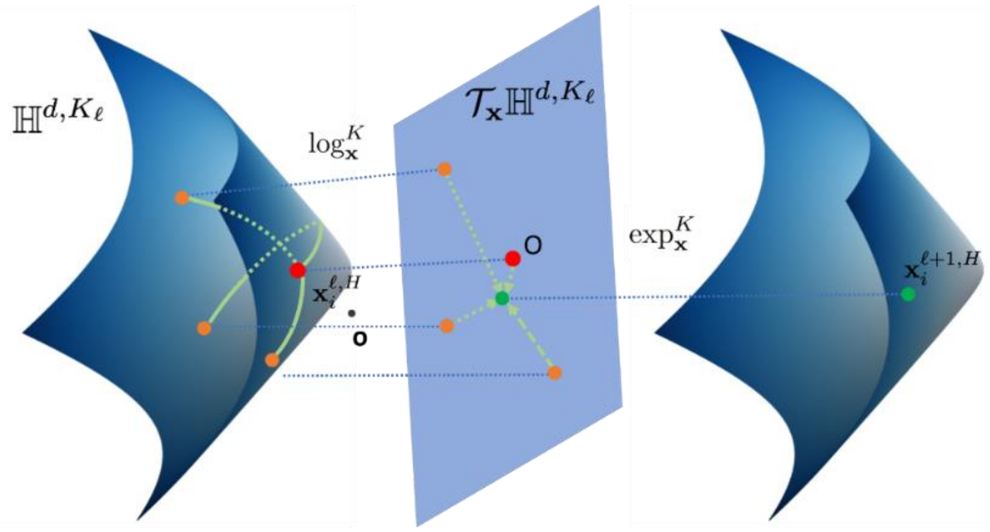


Image Source: Chami, Ines, et al. "Hyperbolic graph convolutional neural networks." Advances in neural information processing systems 32 (2019).

Strategy 1: Cons

Computational Inefficiency: the repeated mappings to and from the tangent space cause significant computational overhead

Numerical Instability: the mappings could cause numerical stability issues; e.g. in logarithmic map:

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

If the points are close together, we risk dividing by or calling $arccosin$ on 0.

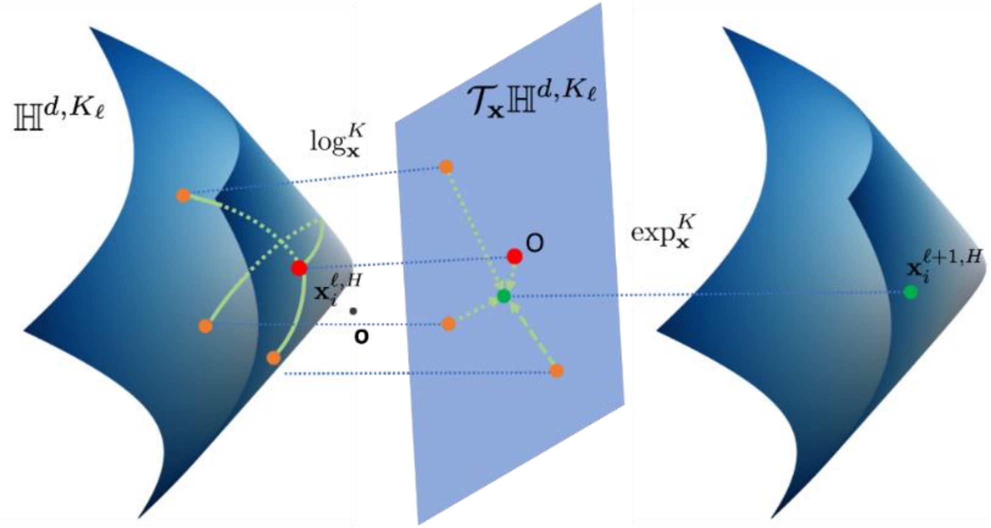


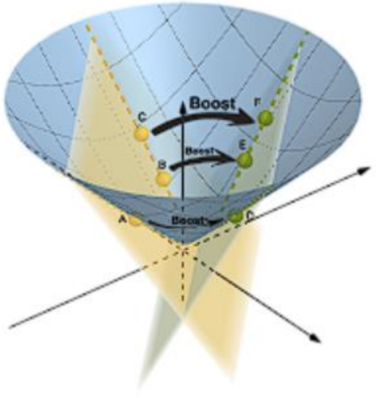
Image Source: Chami, Ines, et al. "Hyperbolic graph convolutional neural networks." Advances in neural information processing systems 32 (2019).

Strategy 1: Cons: Lorentz Rotation & Lorentz Boost

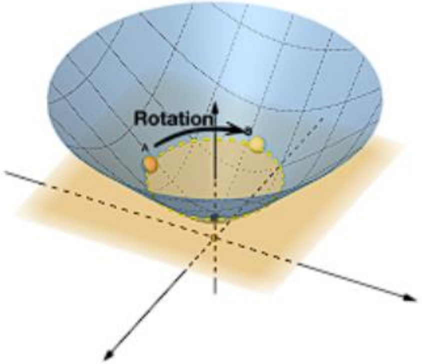
Expressiveness Issues: transformations implemented through $f^{T,K}$ might not cover all types of operations

- Lorentz linear transformation consists of a ***Lorentz Boost*** and a ***Lorentz Rotation***, but tangent-space-based operations do not cover all cases

Constant velocity transformation without rotating the spatial axis



Lorentz Boost



Lorentz Rotation

Rotating the spatial axis by applying a rotation matrix on the space-like dimension

Image Source: Weize Chen, Xu Han, Yankai Lin, Hexu Zhao, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2021. Fully Hyperbolic Neural Networks. arXiv:2105.14686 (2021).

Strategy 2: Fully Hyperbolic Operations

Solution: operate directly on the manifold *"Fully Hyperbolic"*

Two strategies: *Pseudo Lorentz Rotation* v.s. *Pseudo Lorentz Boost*

Pseudo Lorentz Boost : Use a Euclidean function $f: \mathbb{R}^{d+1} \rightarrow \mathbb{R}^{d}$

- e.g. Linear transformer: $f(x) = Wx + b$

Perform f on $x \in \mathbb{H}^{d,K}$

Compute the associating time-like dimension

$$f^{F,K}(x) = \left(\underbrace{\sqrt{||Wx_{time,space}||^2 - 1/K}}_{time-like\ dim}, \underbrace{Wx_{time,space}}_{space-like\ dim} \right)$$

Computes output with ***both*** time and space dimensions of the inputs

Impose Lorentzian constraints

Reference: Menglin Yang, Harshit Verma, Delvin Ce Zhang, Jiahong Liu, Irwin King, and Rex Ying. 2024. Hypformer: Exploring efficient transformer fully in hyperbolic space. In KDD. 3770–3781.
Weize Chen, Xu Han, Yankai Lin, Hexu Zhao, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2021. Fully Hyperbolic Neural Networks. arXiv:2105.14686 (2021).

Strategy 2: Fully Hyperbolic Operations Cont'd

Solution: operate directly on the manifold ***"Fully Hyperbolic"***

Two strategies: ***Pseudo Lorentz Rotation*** v.s. *Pseudo Lorentz Boost*

Pseudo Lorentz Rotation: Use a Euclidean function $f: \mathbb{R}^d \rightarrow \mathbb{R}^d$

- e.g. Linear transformation: $f(x) = ReLU(x)$

Perform f on the *space-like dimension* of $x \in \mathbb{H}^{d,K}$

Transformation on ***only*** the space dimension

Compute the associating time-like dimension

Impose Lorentzian constraints

$$f^{F,K}(x) = \left(\underbrace{\sqrt{||f(x_{space})||^2 - 1/K}}_{time-like\ dim}, \underbrace{f(x_{space})}_{space-like\ dim} \right)$$

References: Menglin Yang, Harshit Verma, Delvin Ce Zhang, Jiahong Liu, Irwin King, and Rex Ying. 2024. Hypformer: Exploring efficient transformer fully in hyperbolic space. In KDD. 3770–3781.

Strategy 2: Fully Hyperbolic Operations Cont'd

Example: Tangent-space-based Linear Transformation $f^{T,K}$ is a Pseudo Lorentz Rotation!

- $f^{T,K}(x) = \exp_{\boldsymbol{o}}^{K}(f(\log_{\boldsymbol{o}}^{K}(x)))$
- $f(x) = Wx + b$

$$\begin{pmatrix} * & 0 \\ 0 & f(\cdot) \end{pmatrix} \log_{\boldsymbol{o}}^{K} \begin{pmatrix} x_{time} \\ x_{space} \end{pmatrix}$$

First coordinate of tangent vectors(of the origin) is ***0***, so the upper left entry does not affect the output

$$f^{T,K}(x) = \begin{pmatrix} \frac{\cosh(\beta)}{-Kx_{time}} & 0 \\ 0 & \frac{\sinh(\beta)W}{\sqrt{-K}||Wx_{space}||} \end{pmatrix} \begin{pmatrix} x_{time} \\ x_{space} \end{pmatrix};$$

$$\beta = \frac{\sqrt{-K}\operatorname{arccosh}\left(\sqrt{-Kx_{time}}\right)W}{\sqrt{-Kx^{\wedge}2_{time}}} ||Wx_{space}||$$

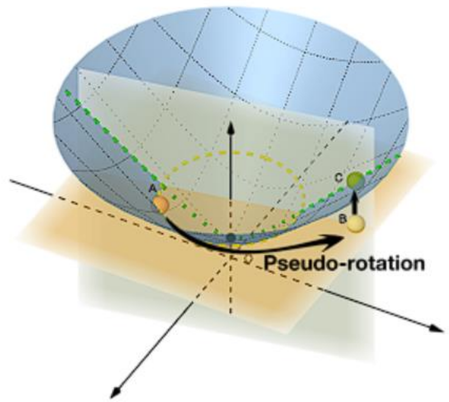


Image Source and Reference: Chami, Ines, et al. "Hyperbolic graph convolutional neural networks." Advances in neural information processing systems 32 (2019).

Strategy 2: Fully Hyperbolic Operations Cont'd

Pseudo Lorentz Rotation v.s. Pseudo Lorentz Boost: Comparison

Pseudo Lorentz Rotation: transformation on without time and space interaction

$$\begin{pmatrix} \frac{\sqrt{||f(x_{space})||^2 - 1/K}}{x_{time}} & 0 \\ 0 & f(\cdot) \end{pmatrix} \begin{pmatrix} x_{time} \\ x_{space} \end{pmatrix}$$

Off-diagonal values are zero

Pseudo Lorentz Boost: transformation on both time and space-like dimension

$$\begin{pmatrix} \sqrt{||Wx||^2 - 1/K)}\boldsymbol{e_0}, & W_{0,:} \\ \sqrt{||Wx||^2 - 1/K)}\boldsymbol{e_{1:d'}} & W_{1:,:} \end{pmatrix} \begin{pmatrix} x_{time} \\ x_{space} \end{pmatrix}$$

Non-zero off-diagonal terms

References: Menglin Yang, Harshit Verma, Delvin Ce Zhang, Jiahong Liu, Irwin King, and Rex Ying. 2024. Hypformer: Exploring efficient transformer fully in hyperbolic space. In KDD. 3770–3781

Refining Hyperbolic Operations

Intuition: take advantages of the *freedom in curvature – vary the curvature* through hyperbolic operations/layers

- For *tangent-space-based* operations: $f^{T}_{K,K'}(x) = \sqrt{\frac{K}{K'}} f^{T,K}(x)$
- For *fully hyperbolic* operations: $f^{F}_{K,K'}(x) = \sqrt{\frac{K}{K'}} f^{F,K}(x)$

Recalibrate coefficient for curvature changes:

$$\sqrt{\frac{K}{K'}} x = \exp_{\boldsymbol{o}}^{K'}\left(\log_{\boldsymbol{o}}^{K}(x)\right)$$

- Tangent space at the origin is the same across different curvature spaces!

Reference: Chami, Ines, et al. "Hyperbolic graph convolutional neural networks." Advances in neural information processing systems 32 (2019).
Menglin Yang, Harshit Verma, Delvin Ce Zhang, Jiahong Liu, Irwin King, and Rex Ying. 2024. Hypformer: Exploring efficient transformer fully in hyperbolic space. In KDD. 3770–3781.

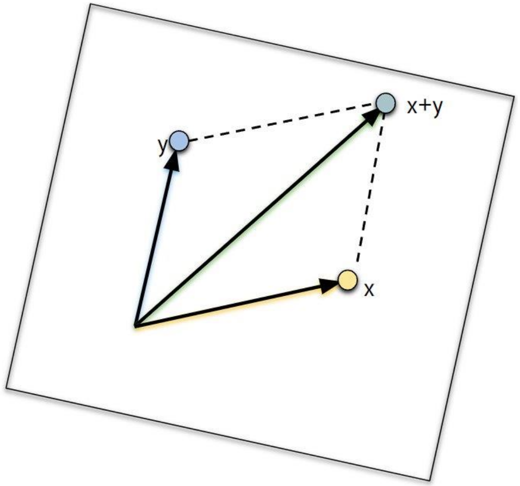
Hyperbolic Residual Connection & Addition

Recall: Addition is difficult in hyperbolic space!

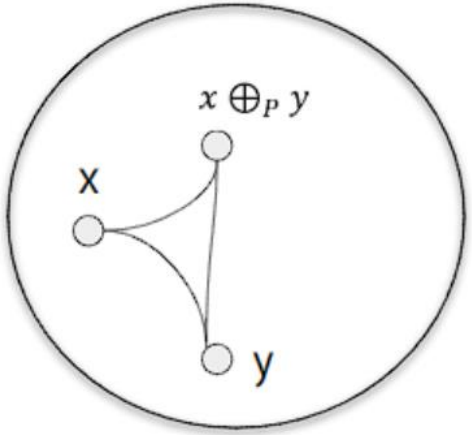
Tangent-space based method: *Möbius Addition* based on *parallel transport*:

$$x \oplus_P y = \exp_{\boldsymbol{x}}^K(P_{\boldsymbol{o}\to\boldsymbol{x}}(\log_{\boldsymbol{o}}^K(y)))$$

Vector Space formulation



Gyrovector Space formulation



Hyperbolic Residual Connection & Addition

Recall: Addition is difficult in hyperbolic space!

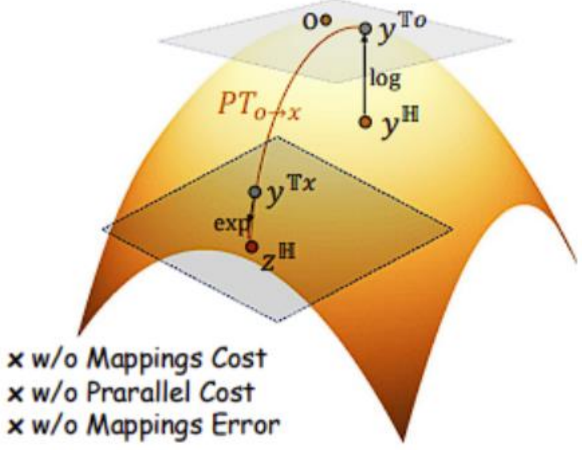
Fully hyperbolic method: generalized Lorent weighted sum

$$x \oplus_L y = \alpha \boldsymbol{x} + \beta \boldsymbol{y}$$

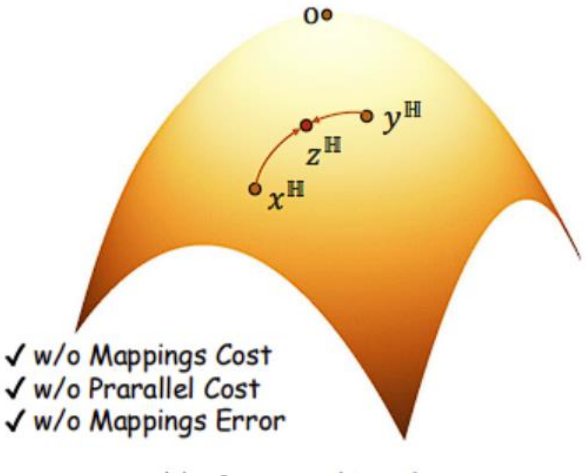
$$\alpha = \frac{w_x}{\sqrt{-K}\,\left\|\,|w_x \boldsymbol{x} + w_y \boldsymbol{y}|\,\right\|_{\mathcal{L}}}$$

$$\beta = \frac{w_y}{\sqrt{-K}\,\left\|\,|w_x \boldsymbol{x} + w_y \boldsymbol{y}|\,\right\|_{\mathcal{L}}}$$

$$w_x, w_y > 0$$



$x \oplus_P y$



$x \oplus_L y$

More ***efficient***, ***stable***, and ***expressive!***

Image Source: Neil He, Menglin Yang, and Rex Ying. 2025. Lorentzian Residual Neural Networks. In KDD.

Euclidean Self-Attention

Self-attention is a vital component in Euclidean Transformer-based foundation models:

- LLMs – text data
- ViTs – visual data
- CLIP models – multi-modal data

The key is to compute a ***weighted sum*** of value vector $\{V_j\}$ using weights based on similarity scores of keys $\{K_j\}$ and queries $\{Q_i\}$

$$Z_i = \sum_{j=1} \frac{\exp(Q_i K_j^T / \sqrt{d'})}{\sum_{j=1} \exp(Q_i K_j^T / \sqrt{d'})} V_j$$

How to generalize midpoint operations to hyperbolic space?

Hyperbolic Midpoint Operations

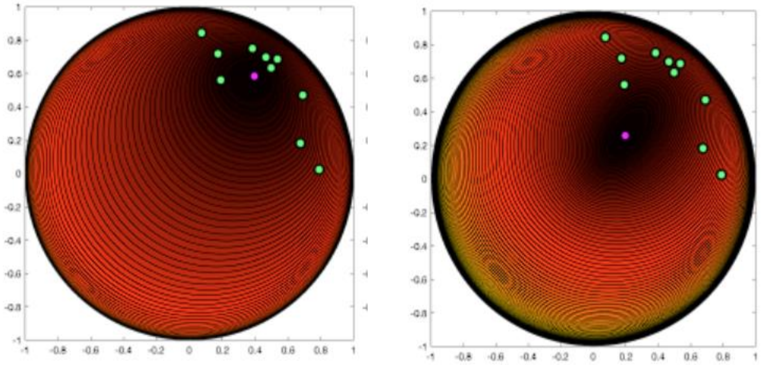
Hyperbolic midpoint has close forms in Lorentz model $LMid_K$, Poincare mode $PMid_K$, and Klein model $KMid_K$ (Einstein Midpoint)

- All of these operations are ***equivalent*** under ***isometric mappings***

Lorentzian Midpoint

$$LMid_K(x_1, \dots, x_N; \{v_i\}) = \frac{\sum_j v_j x_j}{\sqrt{-K}\left\| \, |\sum_j v_j x_j| \, \right\|_{\mathcal{L}}}$$

Plot of Lorentzian Midpoint (purple)



Poincaré Midpoint

$$PMid_K(x_1, \dots, x_N; \{v_i\}) = \frac{1}{2} \otimes_K \frac{\sum_j v_j \lambda_{x_i}^K x_j}{\sum_j |v_j| (\lambda_{x_i}^K - 1)}$$

$$\lambda_x^K = \frac{2}{1 + K||x||^2}$$

Gyrovector space scalar multiplication: implemented through $f^{T,K}$

References and Image Source: Marc Law, Renjie Liao, Jake Snell, and Richard Zemel. 2019. Lorentzian distance learning for hyperbolic representations. In ICML. PMLR, 3672–3681.
Ryohei Shimizu, Yusuke Mukuta, and Tatsuya Harada. 2020. Hyperbolic Neural Networks++. In ICLR

Hyperbolic Self-Attention

Hyperbolic self-attention can be formulated with hyperbolic midpoint operations and similarity score computed using negative hyperbolic distance

Hyperbolic Self-Attention

$$LAtten(Q,K,V) = LMid\left(v_1, \dots, v_N, \{\alpha_{i,j}\}_{j=1}\right)$$

$$PAtten(Q,K,V) = PMid\left(v_1, \dots, v_N, \{\alpha_{i,j}\}_{j=1}\right)$$

Attention Score

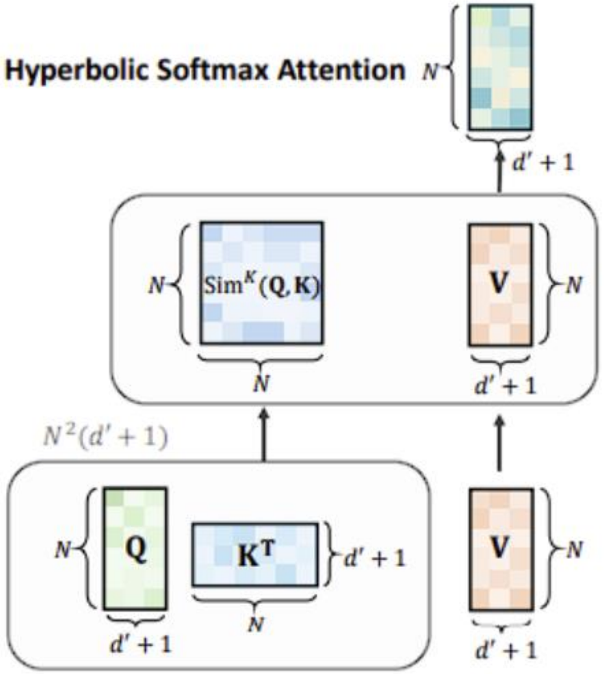
$$\alpha_{i,j} = \frac{\exp(-d_H^2(q_i, v_j))}{\sum_\ell \exp(-d_H^2(q_i, v_\ell))}$$

References: Ryohei Shimizu, Yusuke Mukuta, and Tatsuya Harada. 2020. Hyperbolic Neural Networks++. In ICLR
Weize Chen, Xu Han, Yankai Lin, Hexu Zhao, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2021. Fully Hyperbolic Neural Networks. arXiv:2105.14686 (2021).

Hyperbolic Linear-Attention (1)

Hyperbolic self-attention requires *quadratic time complexity* w.r.t. input tokens:

Many applications such as graph Transformers requires the model to handle *long context*



***Solution:* linear time approximation for attention mechanism**

References: Menglin Yang, Harshit Verma, Delvin Ce Zhang, Jiahong Liu, Irwin King, and Rex Ying. 2024. Hypformer: Exploring efficient transformer fully in hyperbolic space. In KDD. 3770–3781.

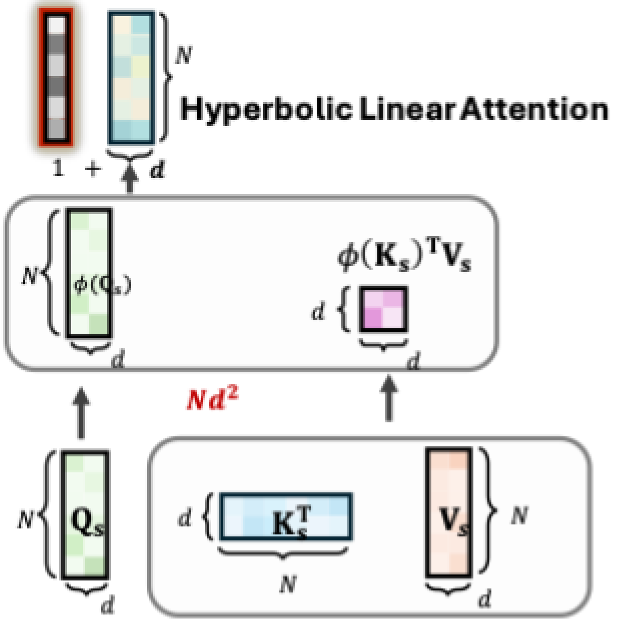
Hyperbolic Linear-Attention (2)

Hyperbolic Linear Attention

$$Q' = \phi(Q_s), K' = \phi(K_s), V' = \phi(V_s)$$

$$LiAttn_{K_1,K_2}(Q,K,V) = \left[\sqrt{||Z||^2 - \frac{1}{K_2}}, Z\right]^T + f^F_{K_1,K_2}(V_s)$$

$$Z = \frac{Q'(K'^T V')}{Q'(K'^T \mathbf{1})}$$



Notations

$$Q' = \phi(Q_s), K' = \phi(K_s), V' = \phi(V_s)$$

$$\phi(x) = \frac{||\tilde{x}||}{||\tilde{x}^p||}\tilde{x}^p$$

$$\tilde{x} = ReLU(x)/t$$

t, p hyperparameters

X_s denotes the space-like dimension

References: Menglin Yang, Harshit Verma, Delvin Ce Zhang, Jiahong Liu, Irwin King, and Rex Ying. 2024. Hypformer: Exploring efficient transformer fully in hyperbolic space. In KDD. 3770–3781

Hyperbolic Normalization Methods

Normalization methods are critical for neural network and foundation models, e.g.

- Layer normalization in Transformers
- Batch normalization in Convolutional Neural Networks

Considerations:

- *Meaningful normalizing operations*
- *Computational efficiency*

Hyperbolic Normalization Methods Cont'd

Consideration 1: Meaningful normalization – similar to the Euclidean case, the goal is to *center the feature vectors across batches/layers* and scale the *keep the variance of their norms within a manageable range*

- Initial work proposed using the ***Fréchet Mean***
- However, this is ***computational expensive***
    - Up to 77% of all compute in the forward pass in hyperbolic CNNs!

Consideration 2: Computational efficiency

References: Max van Spengler, Erwin Berkhout, and Pascal Mettes. 2023. Poincaré ResNet. CVPR (2023)

Hyperbolic Batch Normalization

Method 1: use ***hyperbolic midpoint operations*** instead of Fréchet mean

- Approximately centering the vectors at the origin

Compute mean $\mu = PMid_K(x_1, \ldots, x_N, \{1\})$ (or $\mu = LMid_K(x_1, \ldots, x_N, \{1\})$)

Compute variance $\sigma^2 = \frac{1}{N}\sum_i d_H^2(x_i, \mu)$

Return normalization term $\tilde{x}_i = \exp_\beta^K(\frac{\sqrt{\gamma}}{\sigma} P_{\mu\to\beta}(\log_\mu^K(x_i)))$

Set new mean as learnable β

Optional: re-centering at the origin first: simple geodesics at the origin

$$P_{o\to\beta}(\frac{\sqrt{\gamma}}{\sigma} P_{\mu\to o}(\log_\mu^K(x_i)))$$

Learnable parameters

References: Max van Spengler, Erwin Berkhout, and Pascal Mettes. 2023. Poincaré ResNet. CVPR (2023)
Ahmad Bdeir, Kristian Schwethelm, and Niels Landwehr. 2024. Fully Hyperbolic Convolutional Neural Networks for Computer Vision. In ICLR.

Hyperbolic Layer Normalization

Method 2: use *fully hyperbolic* formulation in *Lorentz space*

- Computationally efficient
- Retain normalizing capabilities

Normalizing the space-like dimension: $y_s = LayerNorm(x_s)$(or $y_s = RSMNorm(x_s)$, etc)

↓

Compute the time-like dimension and return normalized vectors: $\left[\sqrt{||y_s||^2 - \frac{1}{K}}, y_s\right]^T$

→ Normalizing space dimension approximates normalization locally and centers around the origin: $o = \left[\sqrt{-\frac{1}{K}}, 0, \ldots, 0\right]$

References: Menglin Yang, Harshit Verma, Delvin Ce Zhang, Jiahong Liu, Irwin King, and Rex Ying. 2024. Hypformer: Exploring efficient transformer fully in hyperbolic space. In KDD. 3770–3781
Neil He, Rishabh Anand, Hiren Madhu, Ali Maatouk, Smita Krishnaswamy, Leandros Tassiulas, Menglin Yang, and Rex Ying. 2025. HELM: Hyperbolic Large Language Models via Mixture-of-Curvature Experts. arXiv preprint arXiv:2505.24722 (2025).

Hyperbolic Positional Encoding (1)

Positional encodings (PE) enables the model to ***learn ordering information of tokens*** in the input sequence

Learn ***relative positional information***:

- Though hyperbolic addition: $PE_K(x) = x \oplus_L [\varepsilon f^{F,K}(x)]$; ϵ learnable parameters
- Adding positional encoding as bias term in $f^{F,K}$:
  - Assumes PE also follows a linear layer

References: Menglin Yang, Harshit Verma, Delvin Ce Zhang, Jiahong Liu, Irwin King, and Rex Ying. 2024. Hypformer: Exploring efficient transformer fully in hyperbolic space. In KDD. 3770–3781
Neil He, Rishabh Anand, Hiren Madhu, Ali Maatouk, Smita Krishnaswamy, Leandros Tassiulas, Menglin Yang, and Rex Ying. 2025. HELM: Hyperbolic Large Language Models via Mixture-of-Curvature Experts. arXiv preprint arXiv:2505.24722 (2025).
Weize Chen, Xu Han, Yankai Lin, Hexu Zhao, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2021. Fully Hyperbolic Neural Networks. arXiv:2105.14686 (2021).

Hyperbolic Positional Encoding (2)

Pros of relative positional encoding:
- Improves generalizability to different sequence length
- Improves context understanding

Cons of relative positional encoding:
- Introduces additional parameters and computational/memory costs
- Potential overfitting & requires further tuning

References: Menglin Yang, Harshit Verma, Delvin Ce Zhang, Jiahong Liu, Irwin King, and Rex Ying. 2024. Hypformer: Exploring efficient transformer fully in hyperbolic space. In KDD. 3770–3781
Neil He, Rishabh Anand, Hiren Madhu, Ali Maatouk, Smita Krishnaswamy, Leandros Tassiulas, Menglin Yang, and Rex Ying. 2025. HELM: Hyperbolic Large Language Models via Mixture-of-Curvature Experts. arXiv preprint arXiv:2505.24722 (2025).
Weize Chen, Xu Han, Yankai Lin, Hexu Zhao, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2021. Fully Hyperbolic Neural Networks. arXiv:2105.14686 (2021).

Hyperbolic Rotary Positional Encoding (1)

Alternative: Rotary incorporates aspects from both *absolute and relative* encoding method

- Euclidean RoPE: apply ***rotational matrix*** to feature vectors

Apply ***Lorentzian*** :

$$HoPE(z_i) = \left[\sqrt{||R_{i,\Theta}(z_i)_s||^2 - \frac{1}{K}}, R_{i,\Theta}(z_i)_s\right]^T$$

$$\Theta = \{\theta_1, \dots, \theta_{\frac{d}{2}}\}$$

$R_{i,\Theta} \in \mathbb{R}^{d\times d}$ where the *diagonal are 2×2 block matrices* R_{i,θ_j}, which are 2×2 rotation matrices of angle $i\theta_j$

z_i can either be query q_i or key k_i

Neil He, Rishabh Anand, Hiren Madhu, Ali Maatouk, Smita Krishnaswamy, Leandros Tassiulas, Menglin Yang, and Rex Ying. 2025. HELM: Hyperbolic Large Language Models via Mixture-of-Curvature Experts. arXiv preprint arXiv:2505.24722 (2025).

Hyperbolic Rotary Positional Encoding (2)

- ***Long-term decay***: the attention score between a *key-query pair decays* when the *relative position increases*
- ***Robustness***: *robust* attention across *arbitrary relative distances*
- ***Learning Complex Relations***: attention heads with HoPE can learn *diagonal(attends to only itself)* and *off-diagonal(attends to only predecessor)* attention patterns

Neil He, Rishabh Anand, Hiren Madhu, Ali Maatouk, Smita Krishnaswamy, Leandros Tassiulas, Menglin Yang, and Rex Ying. 2025. HELM: Hyperbolic Large Language Models via Mixture-of-Curvature Experts. arXiv preprint arXiv:2505.24722 (2025).

Hyperbolic Concatenation

Hyperbolic concatenation and splitting for ***merging heads in multi-head attention***

- Poincare Concatenation: $Cat_P(x_1, \dots, x_n) = [\exp_o \gamma \beta_1^{-1}(\log_o(x_1))^T, \dots, \exp_o \gamma \beta_n^{-1}(\log_o(x_n))^T]$
  - $\gamma, \beta_i \in B\left(\frac{n}{2}, \frac{1}{2}\right), B\left(\frac{n}{2}, \frac{1}{2}\right)$ beta distribution
- Lorentz Concatenation: $Cat_L(x_1, \dots, x_n) = \left[\sqrt{||y||^2 - \frac{1}{k}}, y\right], y = [(x_1)_s^T, \dots, (x_n)_s^T]$

Other Hyperbolic Neural Operations

- Hyperbolic convolutional layers
- Hyperbolic neighborhood aggregation

References: Ryohei Shimizu, Yusuke Mukuta, and Tatsuya Harada. 2020. Hyperbolic Neural Networks++. In ICLR
Eric Qu and Dongmian Zou. 2022. Lorentzian fully hyperbolic generative adversarial network. arXiv:2201.12825 (2022).

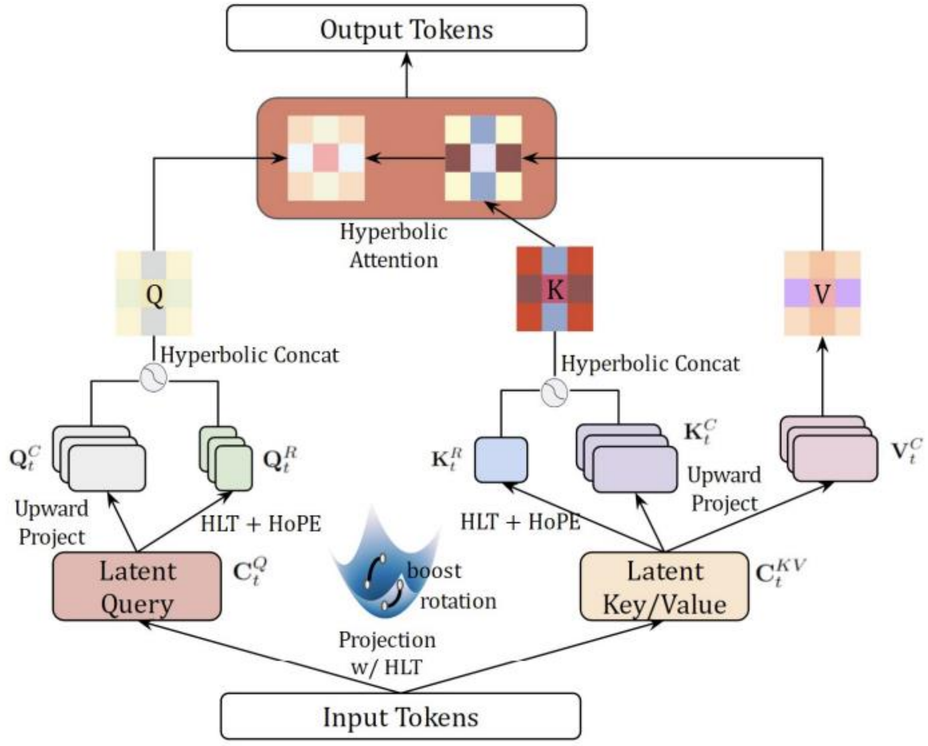
Hyperbolic Latent-Attention

Size of KV-Cache for Hyperbolic MHA per Layer: $O(nn_h)$

- $n =$ number of heads
- $n_h =$ dimension per head

Reduce the KV-Cache: Hyperbolic MLA

1. Project input token x to latent vectors c^Q, c^{KV} of dimensions n_q, n_{kv}
   - $n_q, n_{kv} \ll n$
2. Project latent vectors back to dimension n, obtain $[k_i^C]_{i\leq n}, [v_i^C]_{i\leq n}$ from c^{KV} and $[q_i^C]_{i\leq n}$ from c^Q

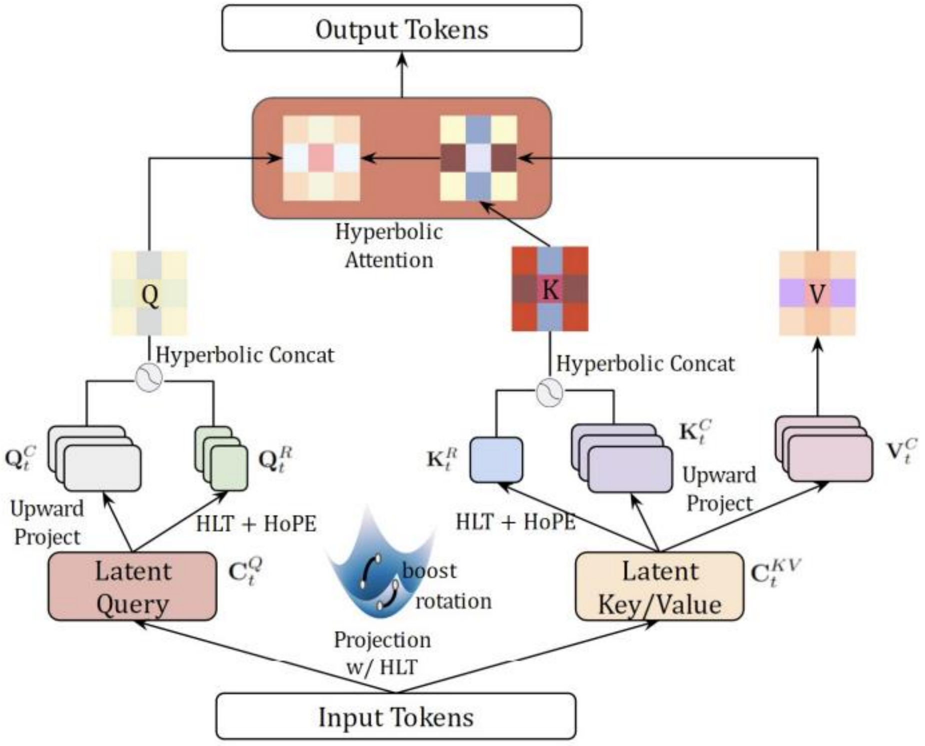


References: Neil He, Rishabh Anand, Hiren Madhu, Ali Maatouk, Smita Krishnaswamy, Leandros Tassiulas, Menglin Yang, and Rex Ying. 2025. HELM: Hyperbolic Large Language Models via Mixture-of-Curvature Experts. arXiv preprint arXiv:2505.24722 (2025)

Hyperbolic Latent-Attention (2)

Reduce the KV-Cache: Hyperbolic MLA

3. Decoupled positional encoding: account to dependency on token index
   - Project latent vectors to rotational queries $[q_i^R]_{i \le n}$ and a shared key k^Q of dimensions nn_r , n_r
   - Perform HoPE on these vectors
4. Concatenate $[q_i^C]_{i \le n}, [q_i^R]_{i \le n}$ and $[k_i^C]_{i \le n}, k^R$ through Lorentzian concatenation
5. Compute hyperbolic attention as usual



We only need to store the latent vectors in the cache

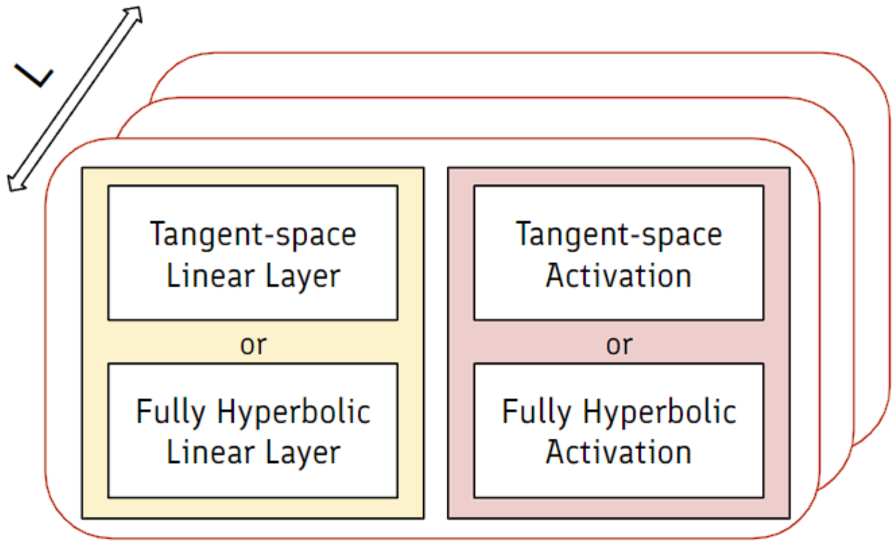
- Complexity $O\left(n(n_q, n_{kv})\right) \ll O(nn_h)$

References: Neil He, Rishabh Anand, Hiren Madhu, Ali Maatouk, Smita Krishnaswamy, Leandros Tassiulas, Menglin Yang, and Rex Ying. 2025. HELM: Hyperbolic Large Language Models via Mixture-of-Curvature Experts. arXiv preprint arXiv:2505.24722 (2025)

Hyperbolic Operations in Practice: HNNs

Hyperbolic MLP

- Hyperbolic Linear layer with hyperbolic activation
- Either tangent-space based methods $f^{T}_{K_1,K_2}$ or fully hyperbolic methods $f^{F}_{K_1,K_2}$



Hyperbolic CNNs and GNNs

Can build hyperbolic **CNNs** and **GNNs** as well!

Hyperbolic CNN

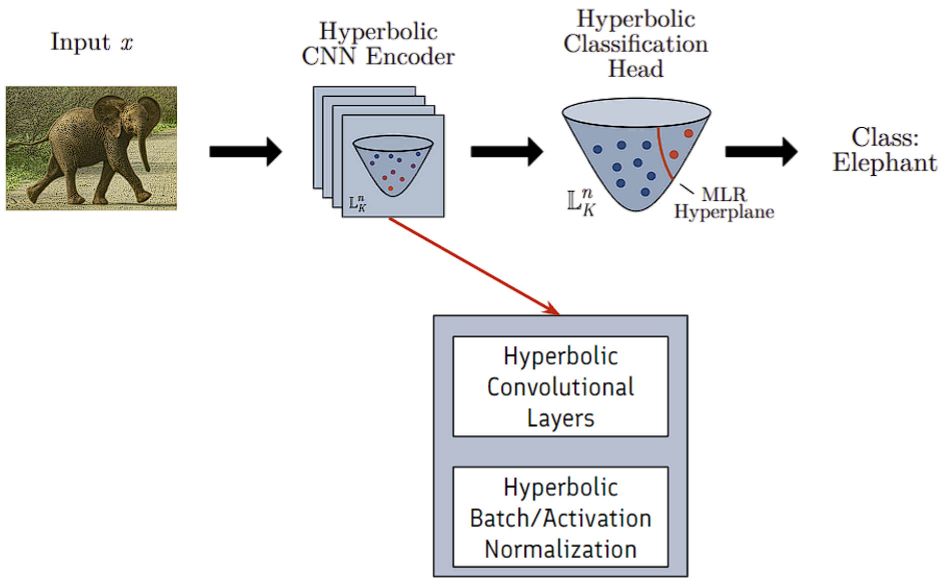
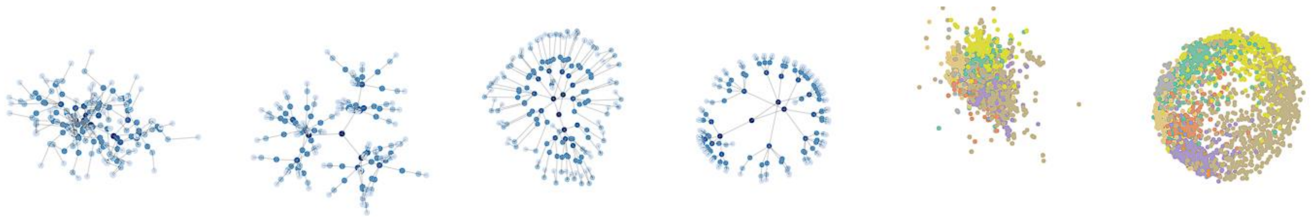


Image Source: Ahmad Bdeir, Kristian Schwethelm, and Niels Landwehr. 2024. Fully Hyperbolic Convolutional Neural Networks for Computer Vision. In ICLR.

Hyperbolic GNN Embeddings



(a) GCN layers. (b) HGCN layers. (c) GCN (left), HGCN (right).

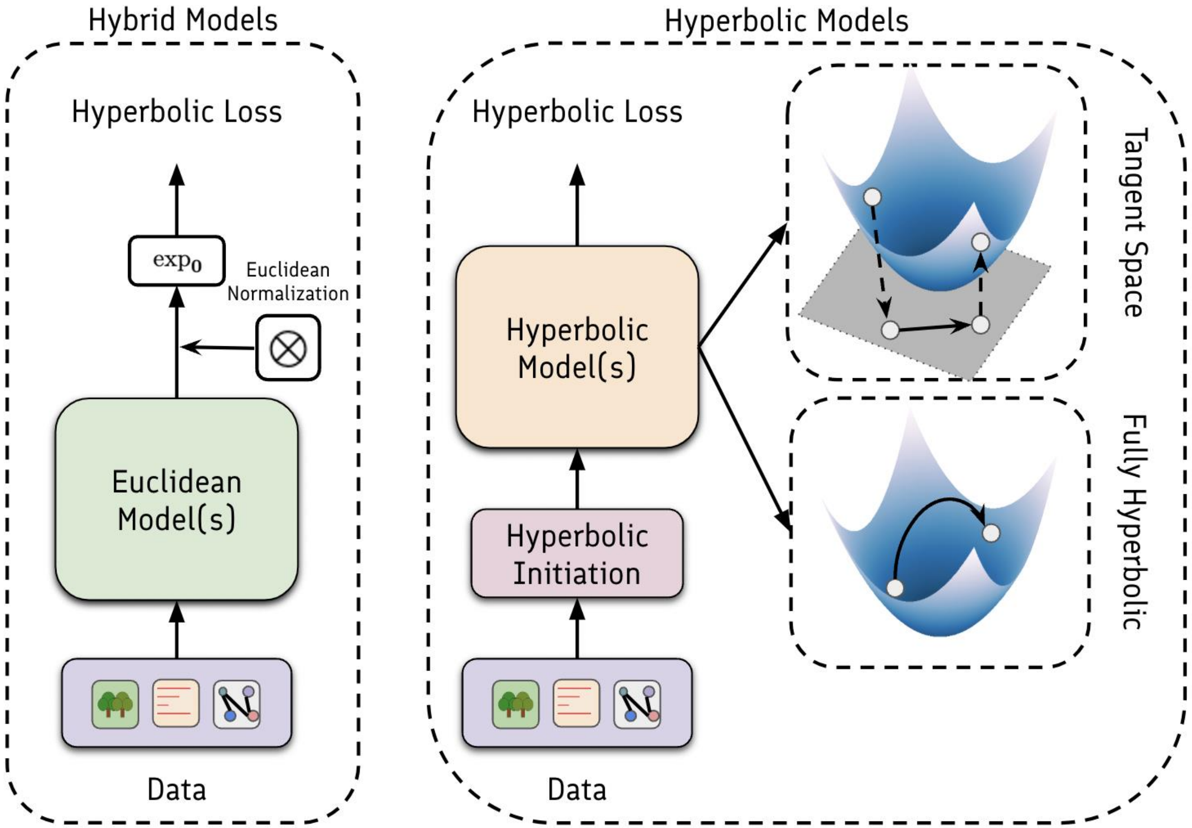
Image Source: Chami, Ines, et al. "Hyperbolic graph convolutional neural networks." Advances in neural information processing systems 32 (2019).

Part 3: Hyperbolic Foundation Models

Hyperbolic Foundation Models: Geometric Modes

Division of Hyperbolic Foundation Models based on their *geometric modes*

- Hybrid Models
- Hyperbolic models



Hybrid Models

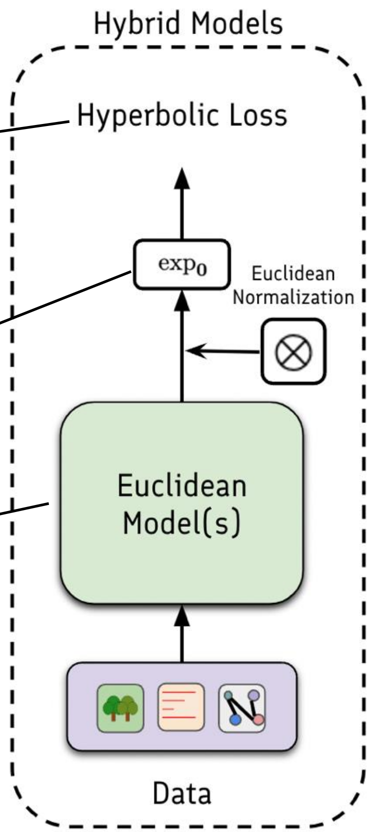
Hybrid consists of two components

- **First component:** Euclidean neural network
- **Second component:** Hyperbolic loss function

3. Compute hyperbolic loss (possibly in combination with Euclidean loss)

2. Lift the Euclidean output to hyperbolic space through a ***projection map***: e.g. $\exp_{\boldsymbol{o}}(x)$

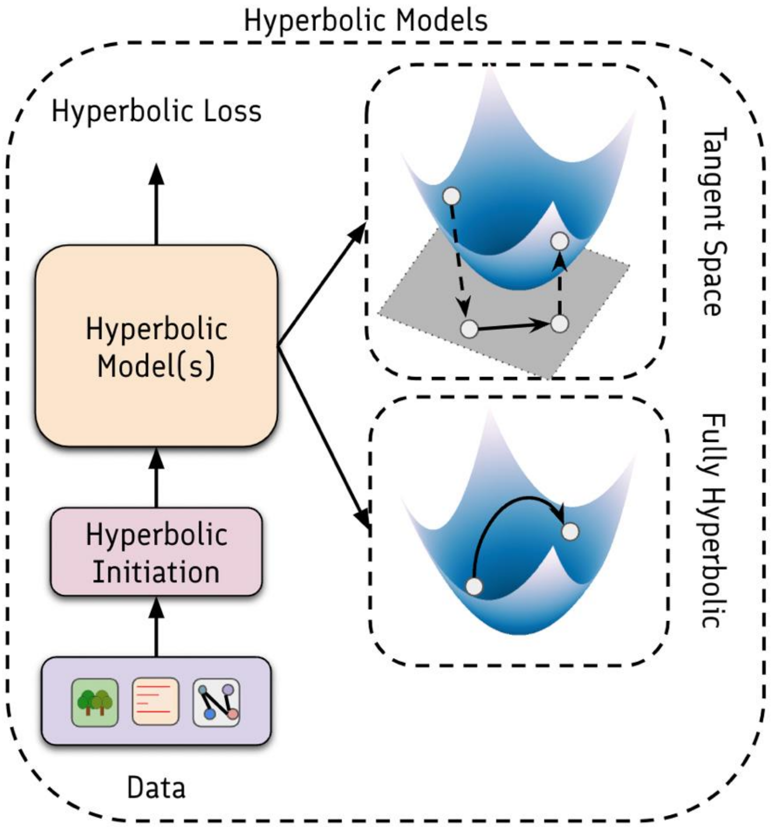
1. Process the data via one or multiple Euclidean mode (s)



Hyperbolic Models (1)

Hyperbolic models
- ***ALL*** components are hyperbolic
  - Hyperbolic neural networks + hyperbolic loss function

Hyperbolic Initiation: often data does not come in the form of hyperbolic vectors, therefore they need to be initialized in hyperbolic space
- If data is already vectorized (Euclidean): lift the data to hyperbolic through ***projection maps***: e.g. $\exp_o(x)$
- If the data is not vectorized:
  - E.g., token indices: map indices to hyperbolic embeddings vectors and optimized with tailored hyperbolic optimizers

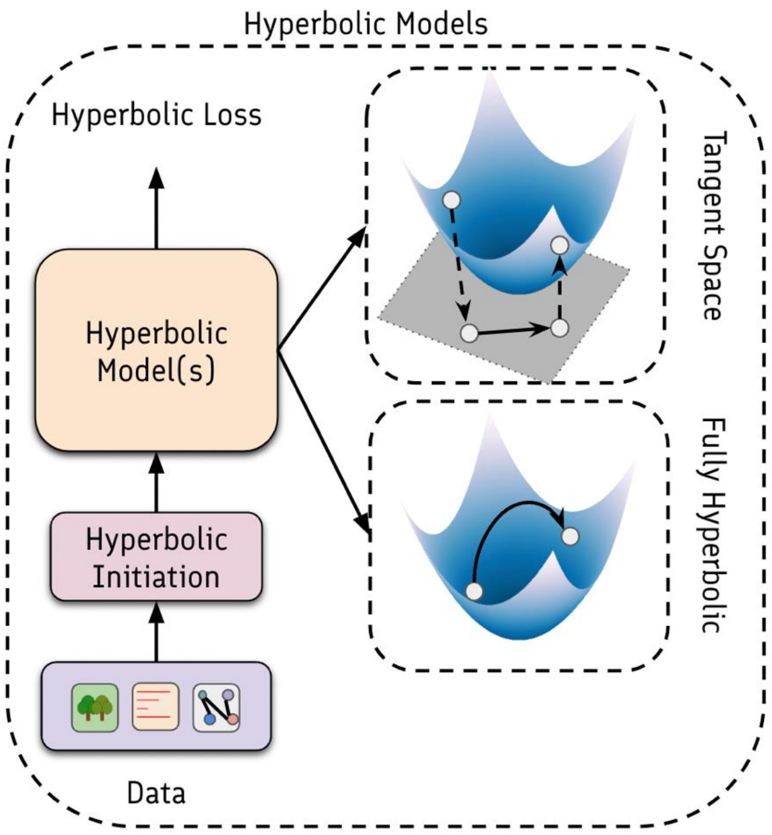


Hyperbolic Models (2)

Hyperbolic Model(s): the initialized hyperbolic vectors are then process by one or multiple hyperbolic model(s)

- Two additional geometric modes:
  - Tangent space models: models that relies on tangent-space-based methods for its operations
  - Fully hyperbolic models: models that uses only fully hyperbolic methods for its operations

Hyperbolic Loss: the output of the hyperbolic models are then used to compute hyperbolic losses



Hyperbolic Foundation Model Overview

Overview of hyperbolic foundation models organized by model architecture + modality

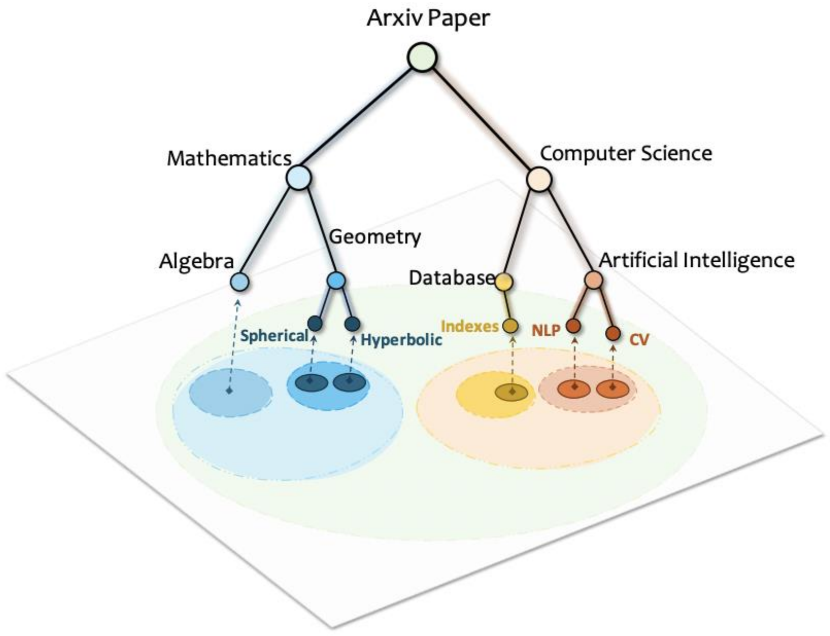
	Architecture	Method	Modality	Geometric Mode	Manifold
Transformers and Language Models	Transformer, Recursive Transformer	HAN [44]	Text, Graph	Hybrid	$\mathbb{K}$
	Transformer	HNN++ [101]	Text	Tangent Space	$\mathbb{P}$
	Transformer	FNN [18]	Text	Fully Hyperbolic	$\mathbb{L}$
	Transformer	H-BERT [17]	Text	Fully Hyperbolic	$\mathbb{L}$
	Transformer, Graph Transformer	Hypformer [115]	Text, Graph, Image	Fully Hyperbolic	$\mathbb{L}$
	Fine-Tuning	HypLoRA [113]	Text	Hybrid	$\mathbb{L}$
	LLM	HELM [47]	Text	Fully Hyperbolic	$\mathbb{L}$
Vision Foundation Models	Vision Transformer	Hyp-ViT [34]	Image	Hybrid	$\mathbb{L}, \mathbb{P}$
	Vision Transformer	HVT [35]	Image	Tangent Space	$\mathbb{P}$
	Vision Transformer	LViT [49]	Image	Fully Hyperbolic	$\mathbb{L}$
	MoCo	HCL [41]	Image	Hybrid	$\mathbb{P}$
	SimCLR/RoCL	RHCL [120]	Image	Hybrid	$\mathbb{P}$
Vision Language Foundation Models	CLIP	MERU [29]	Text, Image	Hybrid	$\mathbb{L}$
	BLIP	H-BLIP-2 [79]	Text, Image	Hybrid	$\mathbb{P}$
	CLIP	HyCoCLIP [88]	Text, Image	Hybrid	$\mathbb{L}$
	CLIP	L-CLIP [49]	Text, Image	Fully Hyperbolic	$\mathbb{L}$

K: Klein Model
P: Poincare Ball Model
L: Lorentz Hyperboloid

Language Transformer: Further Motivation

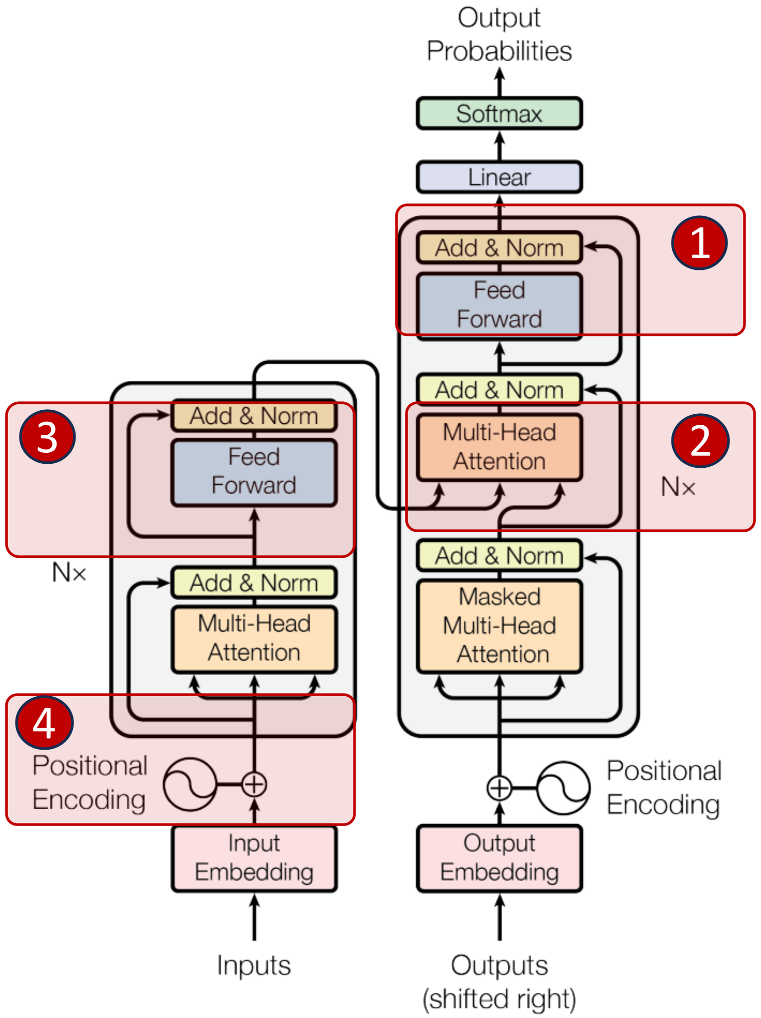
We saw earlier that on the level of *token distribution*, there is *inherent hierarchy* in texts

- This is also the case when it comes to texts on the level of *concepts*
- This naturally hyperbolic embeddings!



References: Menglin Yang, Harshit Verma, Delvin Ce Zhang, Jiahong Liu, Irwin King, and Rex Ying. 2024. Hypformer: Exploring efficient transformer fully in hyperbolic space. In KDD. 3770–3781

Designing Hyperbolic Transformers



Core modules in Transformer

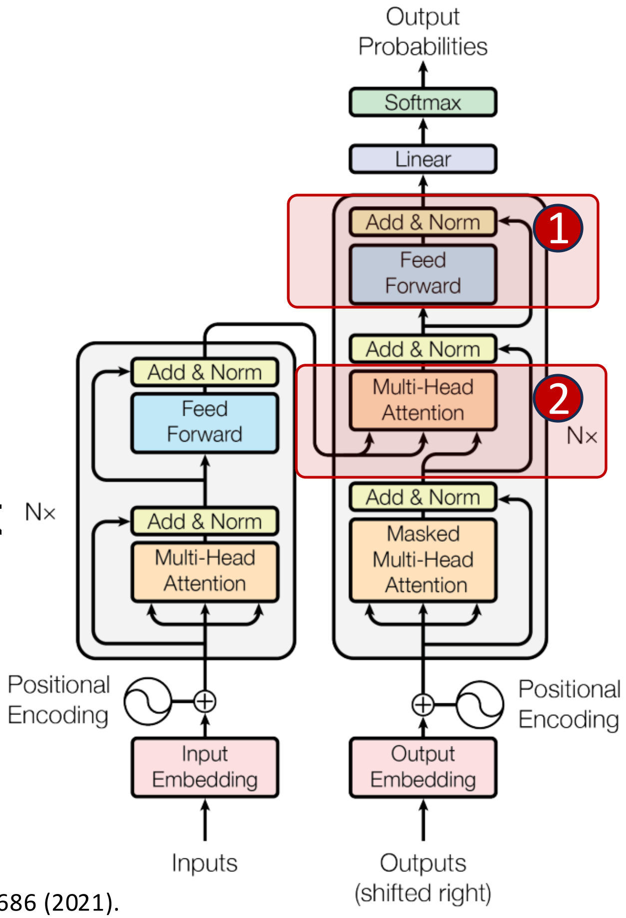
1. FeedForward Layer
2. Multi-Head Attention
3. Addition and LayerNorm
4. Positional Encoding

Language Transformer Example: FNN (1)

The first fully hyperbolic Transformer: Fully Hyperbolic Neural Networks Chen et al. (FNN)

Core modules in Transformer

1. FeedForward Layer
   - Uses fully hyperbolic linear layers: $f^{F,K}(x)$
2. Multi-Head Attention
   - Uses *Lorentzian centroid* based method for hyperbolic mult
     - $LAtten(Q, K, V)$
   - Uses *Lorentzian concatenation* to combine the heads



References: Weize Chen, Xu Han, Yankai Lin, Hexu Zhao, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2021. Fully Hyperbolic Neural Networks. arXiv:2105.14686 (2021).

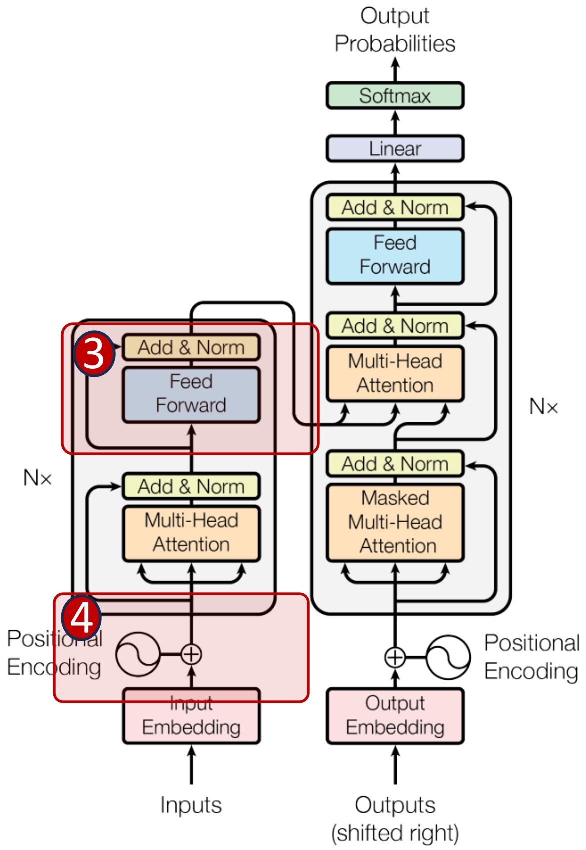
Language Transformer Example: FNN (2)

Core modules in Transformer ***that are missing/limited in FNN***

3. Addition and LayerNorm

4. Positional Encoding

- FNN lacked separate modules for these – they are *built in* into the feedforward layers
- Normalization is performed within $f^{F,K}(x)$
- Residual connection and positional encoding are added as *bias terms* in $f^{F,K}(x)$
  - Assumes they are followed/preceded by linear layers!



References: Weize Chen, Xu Han, Yankai Lin, Hexu Zhao, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2021. Fully Hyperbolic Neural Networks. arXiv:2105.14686 (2021).

Language Transformer Example: FNN (3)

Experimental Snapshot of FNN in machine translation (English <--> German):

- Compared with
  - Euclidean Transformer
  - Hyperbolic Transformers

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**

Euclidean: Transformer
Tangent Space Based: HyperNN++
Hybrid: HAtt

Model	WMT'14
$\text{Transformer}_{base}$ (Vaswani et al., 2017)	27.3
Transformer_{big} (Vaswani et al., 2017)	28.4
HAtt_{base} (Gulcehre et al., 2018)	27.5
HyboNet_{base}	**28.2**

Euclidean: Transformer
Hybrid: HAtt

Outperforms Euclidean & Hyperbolic Transformers (of other geometric modes) across *all dimensions*

Efficient (Graph) Transformer: Hypformer (1)

Missing modules and limitations of FNN

- Lack of layer normalization, residual connections, and positional encoding
- For processing large graphs: inefficient, quadratic time attention mechanism

Solution by Hypformer:

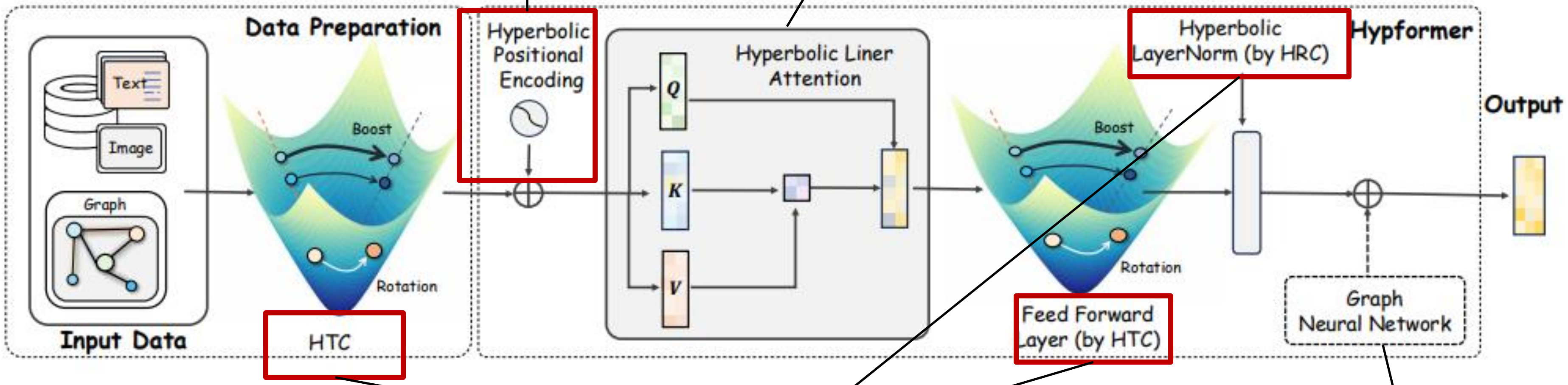
- Implements *layer normalization* through *fully hyperbolic operations*: $f^F_{K_1,K_2}$
- Implements *residual connections* similarly special case of LResNet: $x \oplus_L y$
- Implements *positional encoding* by adding learned relative encodings: $PE_K(x) = x \oplus_L \epsilon f^F_{K_1,K_2}(x)$
- Uses *hyperbolic linear attention* for efficient processing of long sequences: $LiAtten_L$

References: Menglin Yang, Harshit Verma, Delvin Ce Zhang, Jiahong Liu, Irwin King, and Rex Ying. 2024. Hypformer: Exploring efficient transformer fully in hyperbolic space. In KDD. 3770–3781

Efficient (Graph) Transformer: Hypformer (2)

Learned relative encoding: PE_K

$LiAtten_L$



Fully hyperbolic operations: $f^F_{K_1,K_2}$

Optional hyperbolic GNN for processing graphs

References: Menglin Yang, Harshit Verma, Delvin Ce Zhang, Jiahong Liu, Irwin King, and Rex Ying. 2024. Hypformer: Exploring efficient transformer fully in hyperbolic space. In KDD. 3770–3781

Experiment Snapshot: Scalability Evaluation of Hypformer (1)

Method	ogbn-proteins	Amazon2m	ogbn-arxiv	Papers100M
#Nodes	132, 534	2, 449, 029	169, 343	111, 059, 956
#Edges	39, 561, 252	61, 859, 140	1, 166, 243	1, 615, 685, 872
MLP	72.0 ± 0.5	63.5 ± 0.1	55.5 ± 0.2	47.2 ± 0.3
GCN [33]	72.5 ± 0.4	83.9 ± 0.1	71.7 ± 0.3	OOM
SGC [70]	70.3 ± 0.2	81.2 ± 0.1	67.8 ± 0.3	63.3 ± 0.2
GCN-NSampler	73.5 ± 1.3	83.8 ± 0.4	68.5 ± 0.2	62.0 ± 0.3
GAT-NSampler	74.6 ± 1.2	85.2 ± 0.3	67.6 ± 0.2	63.5 ± 0.4
SIGN [21]	71.2 ± 0.5	81.0 ± 0.3	70.3 ± 0.3	65.1 ± 0.1
GraphFormer [83]	OOM	OOM	OOM	OOM
GraphTrans [73]	OOM	OOM	OOM	OOM
GraphGPS [54]	OOM	OOM	OOM	OOM
HAN [25]	OOM	OOM	OOM	OOM
HNN++ [60]	OOM	OOM	OOM	OOM
F-HNN [9]	OOM	OOM	OOM	OOM
NodeFormer [71]	77.5 ± 1.2	87.9 ± 0.2	59.9 ± 0.4	OOT
SGFormer [72]	79.5 ± 0.3	89.1 ± 0.1	72.4 ± 0.3	65.8 ± 0.5
Hypformer	**80.4 ± 0.5**	**89.4 ± 0.3**	**73.2 ± 0.2**	**66.1 ± 0.4**

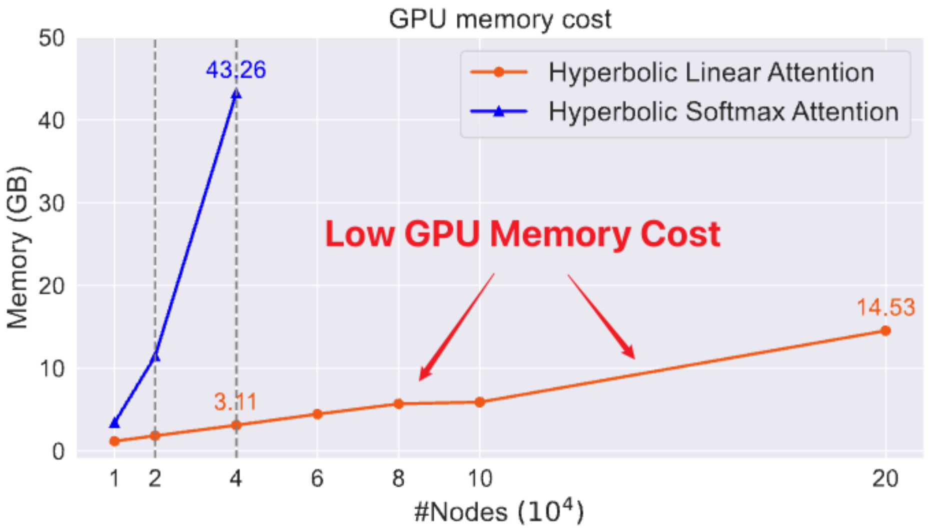
GraphFormer Model

Hyperbolic (Graph)Transformer (*failed!!*)

Successfully working on billion-level graph data and process 10K~200K input tokens

References: Menglin Yang, Harshit Verma, Delvin Ce Zhang, Jiahong Liu, Irwin King, and Rex Ying. 2024. Hypformer: Exploring efficient transformer fully in hyperbolic space. In KDD. 3770–3781

Experiment Snapshot: Scalability Evaluation of Hypformer (2)



More efficiency and save half of running time

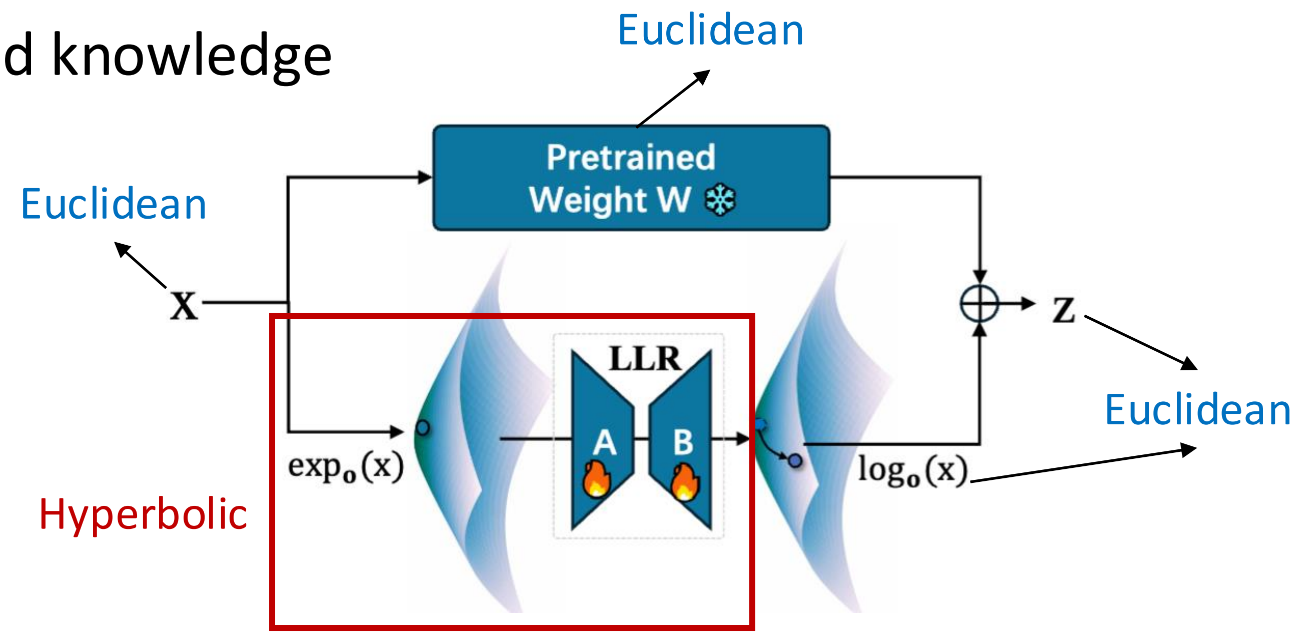
Method	ogbn-proteins Train	ogbn-proteins Test	Amazon2M Train	Amazon2M Test	ogbn-arxiv Train	ogbn-arxiv Test
Hypformer (Softmax)	11.9	-	37.38	-	7.8	-
Hypformer (Linear)	5.3	2.4	16.32	2.5	3	2.5

References: Menglin Yang, Harshit Verma, Delvin Ce Zhang, Jiahong Liu, Irwin King, and Rex Ying. 2024. Hypformer: Exploring efficient transformer fully in hyperbolic space. In KDD. 3770–3781

LLM Integration: Hyperbolic Fine-Tuning (HypLoRA) (1)

Building on existing Euclidean LLMs: a *hybrid model*

- Maintains flexibility while producing hyperbolic representations
- Leverages pre-trained knowledge



LLR(BA, X) is based on fully hyperbolic operation $f^F_{K_1,K_2}$

References: Menglin Yang, Aosong Feng, Bo Xiong, Jihong Liu, Irwin King, and Rex Ying. 2024. Hyperbolic Fine-tuning for Large Language Models. ICML LLM Cognition Workshop (2024).

LLM Integration: Hyperbolic Fine-Tuning (HypLoRA) (2)

Review of Euclidean LoRA:

$$\boldsymbol{z} = \boldsymbol{W}\boldsymbol{x} + BA\boldsymbol{x}, B \in \mathbb{R}^{d\times r}, A \in \mathbb{R}^{r\times k}$$

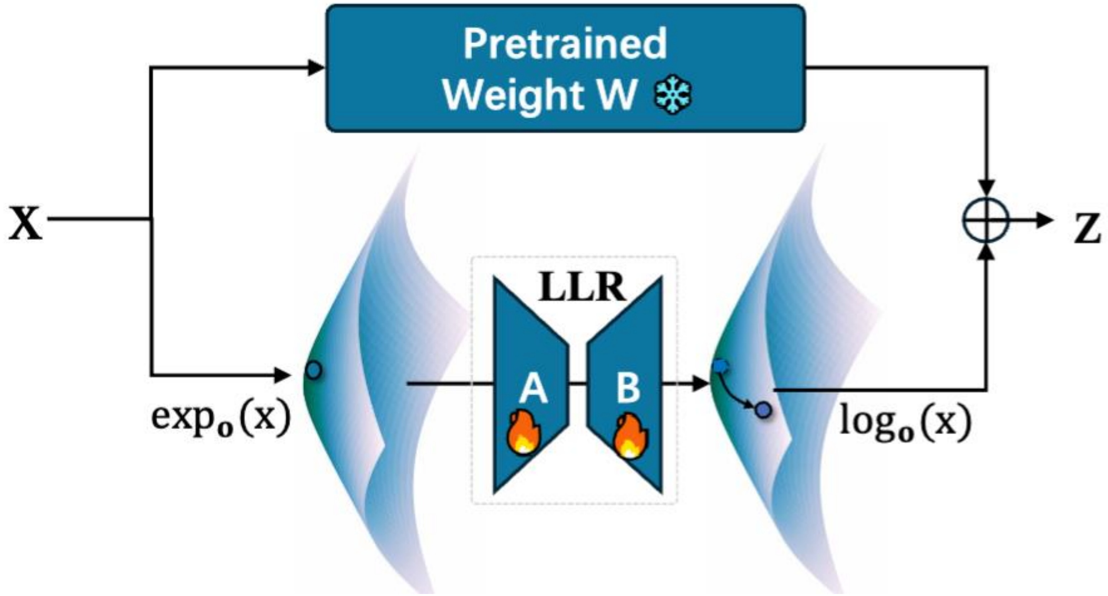
HypLoRA:

Transformation on $\boldsymbol{x}^H$ $\quad \boldsymbol{x}^H$

$$\boldsymbol{z}^E = \boldsymbol{W}\boldsymbol{x}^E + \log_{\boldsymbol{o}}^K\left(\boldsymbol{LLR}(BA, \exp_{\boldsymbol{o}}^K(\boldsymbol{x}^E))\right);$$

$$\boldsymbol{LLR}(BA, \boldsymbol{x}^H) = \left(\sqrt{||B\boldsymbol{y}^H||^2 + \frac{1}{K}}, B\boldsymbol{y}^H\right);$$

$$\boldsymbol{y}^H = \left(\sqrt{||A\boldsymbol{x}^H||^2 + \frac{1}{K}}, A\boldsymbol{x}^H\right)$$



References: Menglin Yang, Aosong Feng, Bo Xiong, Jihong Liu, Irwin King, and Rex Ying. 2024. Hyperbolic Fine-tuning for Large Language Models. ICML LLM Cognition Workshop (2024).

Experiment Snapshot: Mathematical Reasoning

Dataset	Domain	# Train	# Test	Answer
MAWPS	Math	-	239	Number
GSM8K	Math	8.8K	1,319	Number
AQuA	Math	100K	254	Option
SVAMP	Math	-	1,000	Number

MAWPS: Paul had 95 pens and 153 books. After selling some books and pens in a garage sale he had 13 books and 23 pens left. How many books did he sell in the garage sale?

GSM8K: James decides to run 3 sprints 3 times a week. He runs 60 meters each sprint. How many total meters does he run a week?

AQuA: Find out which of the following values is the multiple of X, if it is divisible by 9 and 12?
"options": ["A)36", "B)12", "C)3", "D)9", "E)6"]

References: Menglin Yang, Aosong Feng, Bo Xiong, Jihong Liu, Irwin King, and Rex Ying. 2024. Hyperbolic Fine-tuning for Large Language Models. ICML LLM Cognition Workshop (2024).

Experiment Snapshot: Mathematical Reasoning

Model	PEFT Method	MAWPS(8.5%)	SVAMP(35.6%)	GSM8K(46.9%)	AQuA(9.0%)	M.AVG
GPT-3.5	None	**87.4**	**69.9**	**56.4**	**38.9**	**62.3**
LLaMA-7B	None	51.7	32.4	15.7	16.9	24.8
	Prefix*	63.4	38.1	24.4	14.2	31.7
	Series*	77.7	52.3	33.3	15.0	42.2
	Parallel*	82.4	49.6	35.3	18.1	42.8
	LoRA*	79.0	52.1	37.5	18.9	44.6
	LoRA†	81.9	48.2	38.3	18.5	43.7
	DoRA	80.0	48.8	39.0	16.4	43.9
	HypLoRA (Ours)	79.0	49.1	39.1	20.5 +11%	44.4
LLaMA-13B	None	65.5	37.5	32.4	15.0	35.5
	Prefix*	66.8	41.4	31.1	15.7	36.4
	Series*	78.6	50.8	44.0	22.0	47.4
	Parallel*	81.1	55.7	43.3	20.5	48.9
	LoRA*	83.6	54.6	47.5	18.5	50.5
	LoRA†	83.5	54.7	48.5	18.5	51.0
	DoRA	83.0	54.6	OOT	18.9	NA
	HypLoRA (Ours)	83.2	54.8	49.0	21.5 +16%	51.5
Gemma-7B	None	76.5	60.4	38.4	25.2	48.3
	LoRA	91.6	76.2	66.3	28.9	68.6
	DoRA	91.7	75.9	65.4	27.7	68.0
	HypLoRA (Ours)	91.5	78.7	69.5	32.7 +13%	71.3
LLaMA3-8B	None	79.8	50.0	54.7	21.0	52.1
	LoRA	**92.7**	78.9	70.8	30.4	71.9
	DoRA	92.4	79.3	71.3	33.1	72.5
	HypLoRA (Ours)	91.6	**80.5**	**74.0**	**34.2** +13.4%	**74.2**

Improvements over LoRA

HypLoRA performs better on harder questions.

HypLoRA introduce higher-order interaction and hierarchies-related terms compared with LoRA.

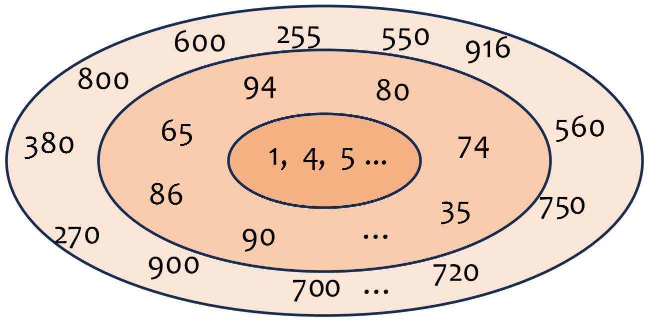
The update of query Q is related to high-order Information and token's norm

$$\Delta Q^{\mathrm{Hyp}} \approx (BA)\mathbf{x} + \frac{\|\mathbf{x}\|^2}{6R^2}(BA)\mathbf{x}.$$

$$\Delta Q^{\mathrm{LoRA}} = (BA)\mathbf{x}.$$

References: Menglin Yang, Aosong Feng, Bo Xiong, Jihong Liu, Irwin King, and Rex Ying. 2024. Hyperbolic Fine-tuning for Large Language Models. ICML LLM Cognition Workshop (2024).

Case Study



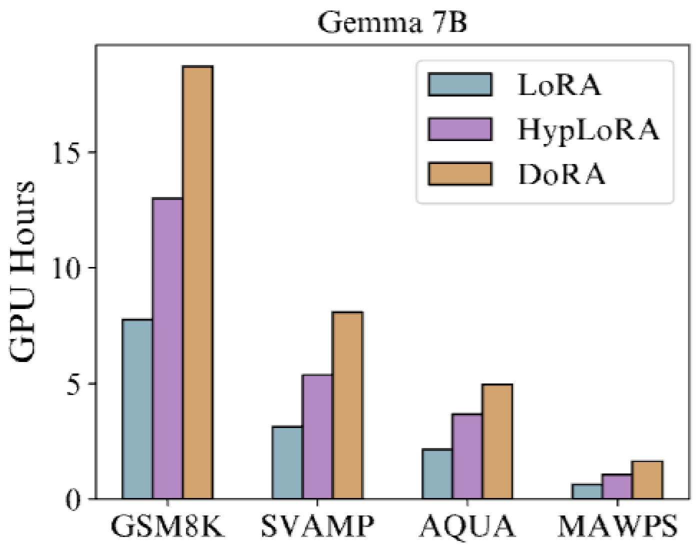
Numbers (token) are arranaged based on their norms in LLaMA 3

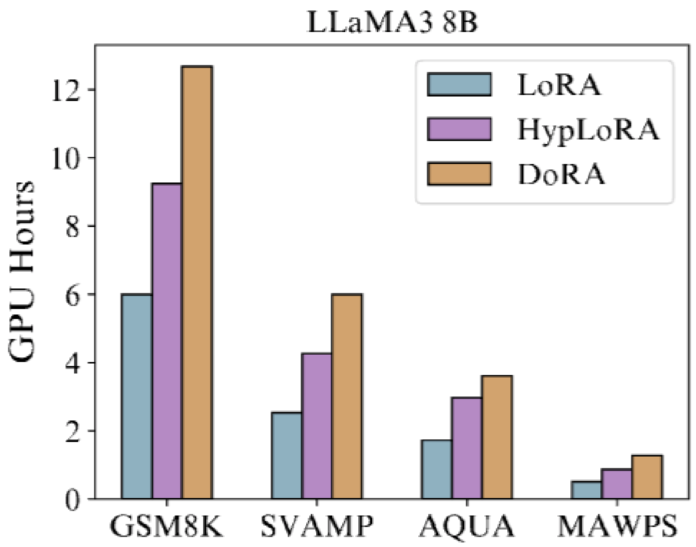
Question	To have the car for the weekend, Wilson's report card needs to show that he received 80 or higher in his math class. His math grade is calculated by the average of 5 test scores. On those tests, he received: 65, 94, 81, 86, and 74. What is his current math grade? **Answer:** 80.0
LoRA	To find Wilson's current math grade, we need to take the average of his 5 test scores. First, we add up all 5 test scores: 65 + 94 + 81 + 86 + 74 = 380. Then, we divide the sum by the number of tests: 380 ÷ 5 = 76. Therefore, Wilson's current math grade is 76. **Answer: 76**.
HypLoRA	To find Wilson's current math grade, we need to take the average of his 5 test scores. To do this, we add up the scores and divide by 5: 65 + 94 + 81 + 86 + 74 = 400. Then, we divide by 5: 400 ÷ 5 = 80. Therefore, Wilson's current math grade is 80. **Answer: 80**.

HypLoRA provides better understanding of number's hierarchies (especially for these leaf tokens) for prediction and accurate computation

References: Menglin Yang, Aosong Feng, Bo Xiong, Jihong Liu, Irwin King, and Rex Ying. 2024. Hyperbolic Fine-tuning for Large Language Models. ICML LLM Cognition Workshop (2024).

Efficiency





Although the proposed method increases the computational burden compared to the original LoRA, it remains significantly more efficient than DoRA, one of the state-of-the-art adapters.

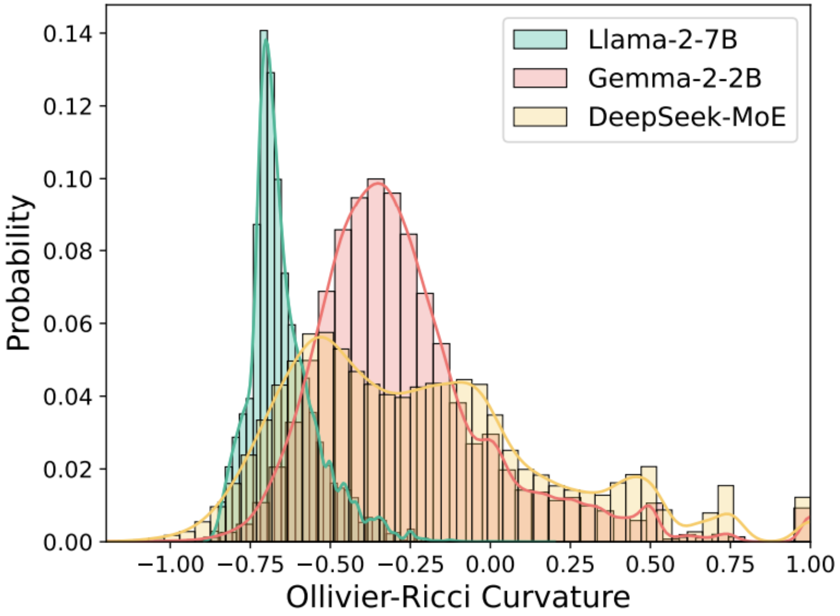
References: Menglin Yang, Aosong Feng, Bo Xiong, Jihong Liu, Irwin King, and Rex Ying. 2024. Hyperbolic Fine-tuning for Large Language Models. ICML LLM Cognition Workshop (2024).

Hyperbolic MoE & LLM: HELM (1)

Mixture of Curvature Experts (MiCE)
- Intuition: not all tokens exhibit the exact same geometric property
- It is advantageous to embed each token in a geometric space that is the most suitable for that specific token

Observation: there is a wide variety of Ollivier-Ricci values for the tokens in LLMs

Mixture of Experts (MoE) provides a *natural framework*



References: Neil He, Rishabh Anand, Hiren Madhu, Ali Maatouk, Smita Krishnaswamy, Leandros Tassiulas, Menglin Yang, and Rex Ying. 2025. HELM: Hyperbolic Large Language Models via Mixture-of-Curvature Experts. arXiv preprint arXiv:2505.24722 (2025).

Hyperbolic MoE & LLM: HELM (2)

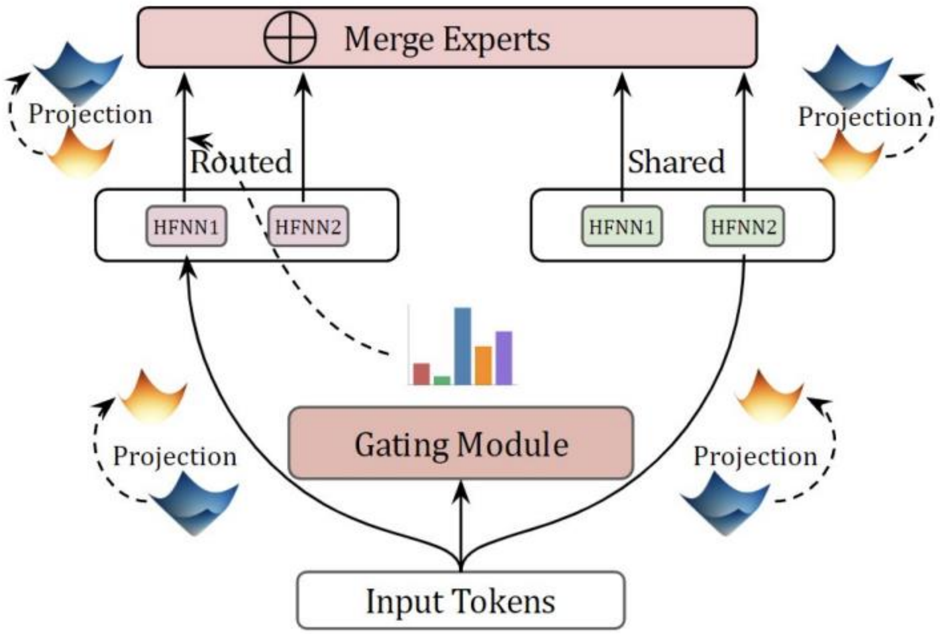
Employ (K_R) routed experts R_i and (K_S) shared experts S_i

Selecting routed experts:

The routing score is $g_{t,i} = \frac{g'_{t,i}}{\Sigma_j g'_{t,j}}$ where

$g'_{t,i} = s_{t,i}$ if $s_{t,i} \in \text{topk}(\{s_{t,j}\}, K_R)$ and 0 otherwise,
where $s_{t,i} = (x_t)_s^\top y_s$

$(x_t)_s$ = space dimension of t-th token
y_s = space dimension of centroid weighting vector

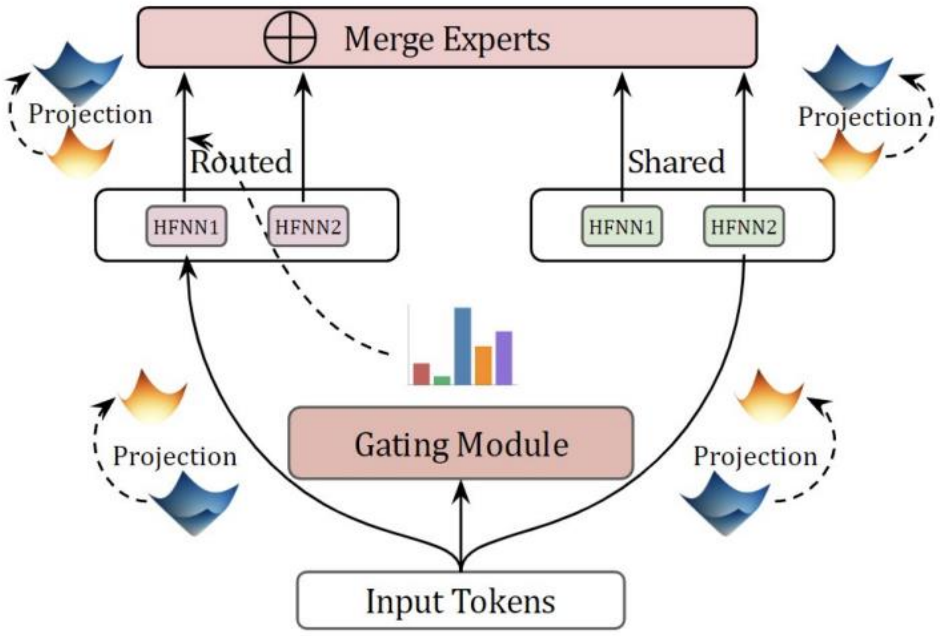


References: Neil He, Rishabh Anand, Hiren Madhu, Ali Maatouk, Smita Krishnaswamy, Leandros Tassiulas, Menglin Yang, and Rex Ying. 2025. HELM: Hyperbolic Large Language Models via Mixture-of-Curvature Experts. arXiv preprint arXiv:2505.24722 (2025).

Hyperbolic MoE & LLM: HELM (3)

Expert Processing

- The overall model's curvature is K
- The routed experts' curvatures are $K_{R,i}$
- The shared experts' curvatures are $K_{S,i}$



Aligning the Manifolds Through Projections

$$z_{t,i} = \sqrt{\frac{K_{R,i}}{K}} R_i \left(\sqrt{\frac{K}{K_{R,i}}} x_t \right)$$

$$y_{t,i} = \sqrt{\frac{K_{S,i}}{K}} S_i \left(\sqrt{\frac{K}{K_{S,i}}} x_t \right)$$

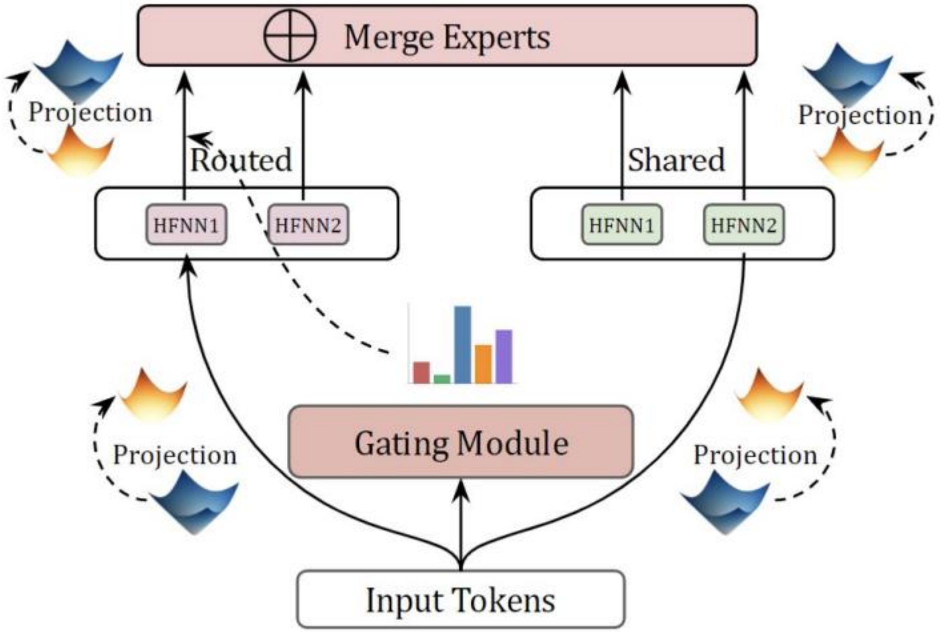
References: Neil He, Rishabh Anand, Hiren Madhu, Ali Maatouk, Smita Krishnaswamy, Leandros Tassiulas, Menglin Yang, and Rex Ying. 2025. HELM: Hyperbolic Large Language Models via Mixture-of-Curvature Experts. arXiv preprint arXiv:2505.24722 (2025).

Hyperbolic MoE & LLM: HELM (4)

Aggregating Final Output

$$x_t \oplus_L Mid_L(y_{t,1}, \ldots, y_{t,K_S}, z_{t,1}, \ldots z_{t,K_R}; \{1, \ldots, 1, g_{t,1}, \ldots, g_{t,K_R}\})$$

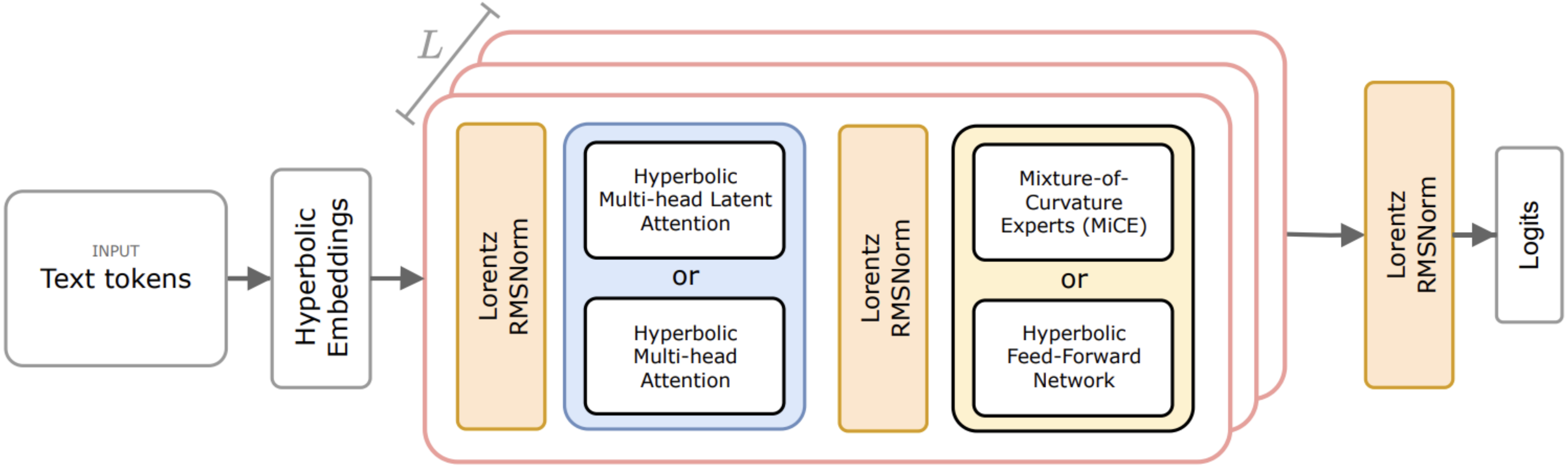
MiCE enables better representation of finer-grained geometric structures



References: Neil He, Rishabh Anand, Hiren Madhu, Ali Maatouk, Smita Krishnaswamy, Leandros Tassiulas, Menglin Yang, and Rex Ying. 2025. HELM: Hyperbolic Large Language Models via Mixture-of-Curvature Experts. arXiv preprint arXiv:2505.24722 (2025).

Hyperbolic MoE & LLM: HELM (4)

Hyperbolic LLM Architecture



References: Neil He, Rishabh Anand, Hiren Madhu, Ali Maatouk, Smita Krishnaswamy, Leandros Tassiulas, Menglin Yang, and Rex Ying. 2025. HELM: Hyperbolic Large Language Models via Mixture-of-Curvature Experts. arXiv preprint arXiv:2505.24722 (2025).

Hyperbolic MoE & LLM: HELM (5)

Hyperbolic LLM results v.s. Euclidean Baselines trained on the 5B tokens

Model	# Params	CommonsenseQA 0-Shot	HellaSwag 0-Shot	OpenbookQA 0-Shot	MMLU 5-Shot	ARC-Challenging 5-Shot	Avg -
LLaMA	115M	**21.1**	25.3	25.3	23.8	21.0	23.3
HELM-D	115M	20.1	25.9	27.0	25.8	21.2	24.0
DeepSeekV3	120M	19.2	25.2	23.4	24.2	21.8	22.8
HELM-MiCE	120M	19.3	26.0	27.4	24.7	23.5	24.2
DeepSeekV3	1B	19.5	26.2	27.4	23.6	22.7	23.9
HELM-MiCE	1B	19.8	**26.5**	**28.4**	**25.9**	**23.7**	**24.9**

Hyperbolic LLM outperforms Euclidean baselines consistently

Case Study: better semantic hierarchy representation

General works (e.g. how, if) lie closer to the origin than specific words (graph, connecting, edges)

HELM-MiCE		DeepseekV3	
Words	**Norm Range**	**Words**	**Norm Range**
A, How, does, if, there, have, is, any, with, of	36.031–36.396	is, a, connecting, graph, there, edges, complete, have, of	33.668–33.768
discrete, vertices, edges, connecting, pair, graph, complete, many, 10	36.506–36.717	discrete, 10, how, if, pair, does, with, A, vertices, any	33.772–33.908

General words (e.g. how, if) and specific words (connecting, edges) are mixed together

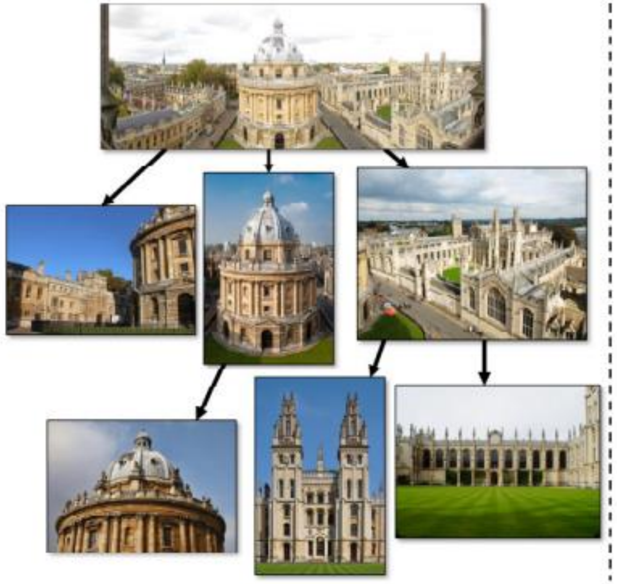
References: Neil He, Rishabh Anand, Hiren Madhu, Ali Maatouk, Smita Krishnaswamy, Leandros Tassiulas, Menglin Yang, and Rex Ying. 2025. HELM: Hyperbolic Large Language Models via Mixture-of-Curvature Experts. arXiv preprint arXiv:2505.24722 (2025).

Hyperbolic Vision Foundation Models: Hyp-ViT(1)

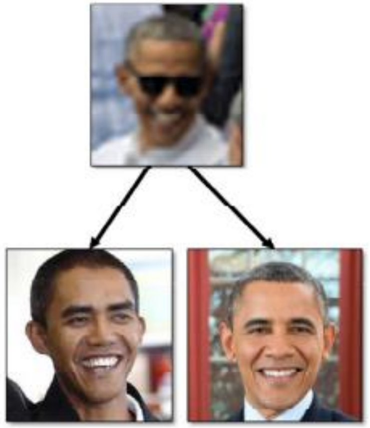
Hierarchical structures are prevalent in vision data as well!

- Scale-free distribution in quantized vision foundation models that we showed earlier
- Structural hierarchies in the photo itself and/or recognition

Whole-part hierarchy



Ambiguity hierarchy



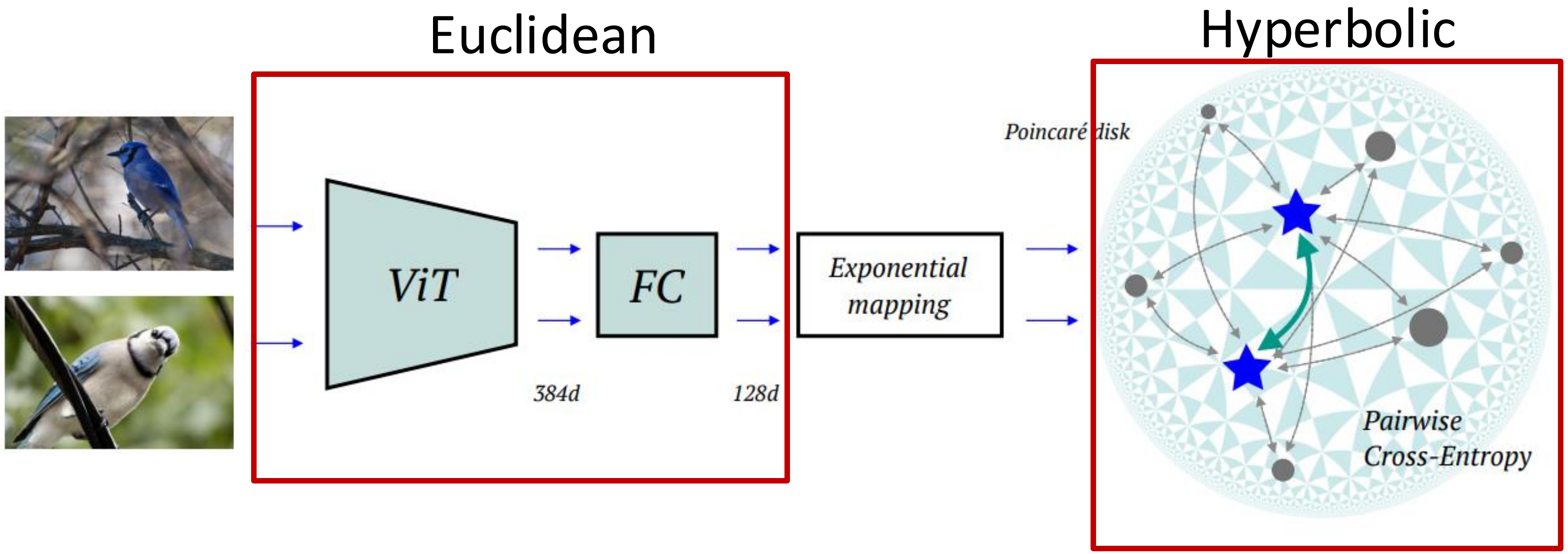
Hyperbolicity in ViTs

	CUB-200	Cars-196	SOP	In-Shop
ViT-S	0.280	0.339	0.271	0.313
DeiT-S	0.294	0.343	0.270	0.323
DINO	0.315	0.327	0.301	0.318

References: Aleksandr Ermolov, Leyla Mirvakhabova, Valentin Khrulkov, Nicu Sebe, and Ivan Oseledets. 2022. Hyperbolic vision transformers: Combining improvements in metric learning. In CVPR. 7409–7419.
Valentin Khrulkov, Leyla Mirvakhabova, Evgeniya Ustinova, Ivan Oseledets, and Victor Lempitsky. Hyperbolic image embeddings. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 6418–6428, 2020

Hyperbolic Vision Foundation Models: Hyp-ViT(2)

Hyp-ViT: hybrid model that adapts existing Euclidean vision Transformers to hyperbolic space by incorporating a *hyperbolic cross-entropy loss*



References: Aleksandr Ermolov, Leyla Mirvakhabova, Valentin Khrulkov, Nicu Sebe, and Ivan Oseledets. 2022. Hyperbolic vision transformers: Combining improvements in metric learning. In CVPR. 7409–7419.

Hyperbolic Vision Foundation Models: Hyp-ViT(3)

Euclidean Entropy Loss

- Cosine similarity (spherical) based

$$L_{CE}^{E}(z_i, z_j) = -\log\left(\frac{\exp\left(-\frac{d_{cos}(z_i, z_j)}{\tau}\right)}{\Sigma_{k=1}^{B}\exp\left(-\frac{d_{cos}(z_i, z_j)}{\tau}\right)}\right);$$

$$d_{cos}(z_i, z_j) = ||\frac{z_i}{||z_i||^2} - \frac{z_j}{||z_j||^2}||^2$$

Hyperbolic Entropy Loss

- Hyperbolic distance based

$$L_{CE}^{E}(z_i, z_j) = -\log\left(\frac{\exp\left(-\frac{d_H(z_i, z_j)}{\tau}\right)}{\Sigma_{k=1}^{B}\exp\left(-\frac{d_H(z_i, z_j)}{\tau}\right)}\right);$$

$$d_H(z_i, z_j) = Poincare\ Distance$$

References: Aleksandr Ermolov, Leyla Mirvakhabova, Valentin Khrulkov, Nicu Sebe, and Ivan Oseledets. 2022. Hyperbolic vision transformers: Combining improvements in metric learning. In CVPR. 7409–7419.

Hyperbolic Vision Foundation Models: Hyp-ViT(4)

Recall@K results

Method	Dim	CUB-200-2011 (K) 1	2	4	8	Cars-196 (K) 1	2	4	8	SOP (K) 1	10	100	1000	In-Shop (K) 1	10	20	30
A-BIER [36]	512	57.5	68.7	78.3	86.2	82.0	89.0	93.2	96.1	74.2	86.9	94.0	97.8	83.1	95.1	96.9	97.5
ABE [24]	512	60.6	71.5	79.8	87.4	85.2	90.5	94.0	96.1	76.3	88.4	94.8	98.2	87.3	96.7	97.9	98.2
SM [49]	512	56.0	68.3	78.2	86.3	83.4	89.9	93.9	96.5	75.3	87.5	93.7	97.4	90.7	97.8	98.5	98.8
XBM [59]	512	65.8	75.9	84.0	89.9	82.0	88.7	93.1	96.1	79.5	90.8	96.1	98.7	89.9	97.6	98.4	98.6
HTL [13]	512	57.1	68.8	78.7	86.5	81.4	88.0	92.7	95.7	74.8	88.3	94.8	98.4	80.9	94.3	95.8	97.2
MS [58]	512	65.7	77.0	86.3	91.2	84.1	90.4	94.0	96.5	78.2	90.5	96.0	98.7	89.7	97.9	98.5	98.8
SoftTriple [37]	512	65.4	76.4	84.5	90.4	84.5	90.7	94.5	96.9	78.6	86.6	91.8	95.4	-	-	-	-
HORDE [20]	512	66.8	77.4	85.1	91.0	86.2	91.9	95.1	97.2	80.1	91.3	96.2	98.7	90.4	97.8	98.4	98.7
Proxy-Anchor [23]	512	68.4	79.2	86.8	91.6	86.1	91.7	95.0	97.3	79.1	90.8	96.2	98.7	91.5	98.1	98.8	**99.1**
NSoftmax [64]	512	61.3	73.9	83.5	90.0	84.2	90.4	94.4	96.9	78.2	90.6	96.2	-	86.6	97.5	98.4	98.8
ProxyNCA++ [52]	512	69.0	79.8	87.3	92.7	86.5	92.5	95.7	97.7	80.7	92.0	96.7	98.9	90.4	98.1	98.8	99.0
IRT_R [8]	384	76.6	85.0	91.1	94.3	-	-	-	-	84.2	93.7	97.3	99.1	91.9	98.1	98.7	98.9
ResNet-50 [18] †	2048	41.2	53.8	66.3	77.5	41.4	53.6	66.1	76.6	50.6	66.7	80.7	93.0	25.8	49.1	56.4	60.5
DeiT-S [53] †	384	70.6	81.3	88.7	93.5	52.8	65.1	76.2	85.3	58.3	73.9	85.9	95.4	37.9	64.7	72.1	75.9
DINO [3] †	384	70.8	81.1	88.8	93.5	42.9	53.9	64.2	74.4	63.4	78.1	88.3	96.0	46.1	71.1	77.5	81.1
ViT-S [48] † §	384	83.1	90.4	94.4	96.5	47.8	60.2	72.2	82.6	62.1	77.7	89.0	96.8	43.2	70.2	76.7	80.5
Sph-DeiT	384	76.2	84.5	90.2	94.3	81.7	88.6	93.4	96.2	82.5	92.9	97.2	99.1	89.6	97.2	98.0	98.4
Sph-DINO	384	78.7	86.7	91.4	94.9	86.6	91.8	95.2	97.4	82.2	92.1	96.8	98.9	90.1	97.1	98.0	98.4
Sph-ViT §	384	85.1	90.7	94.3	96.4	81.7	89.0	93.0	95.8	82.1	92.5	97.1	99.1	90.4	97.4	98.2	98.6
Hyp-DeiT	384	77.8	86.6	91.9	95.1	86.4	92.2	95.5	97.5	83.3	93.5	97.4	99.1	90.5	97.8	98.5	98.9
Hyp-DINO	384	80.9	87.6	92.4	95.6	**89.2**	**94.1**	**96.7**	**98.1**	85.1	94.4	97.8	99.3	92.4	**98.4**	**98.9**	**99.1**
Hyp-ViT §	384	**85.6**	**91.4**	**94.8**	**96.7**	86.5	92.1	95.3	97.3	**85.9**	**94.9**	**98.1**	**99.5**	**92.5**	98.3	98.8	**99.1**

Euclidean ViTs

Euclidean (spherical) Cross-Entropy

Hyperbolic Cross-Entropy

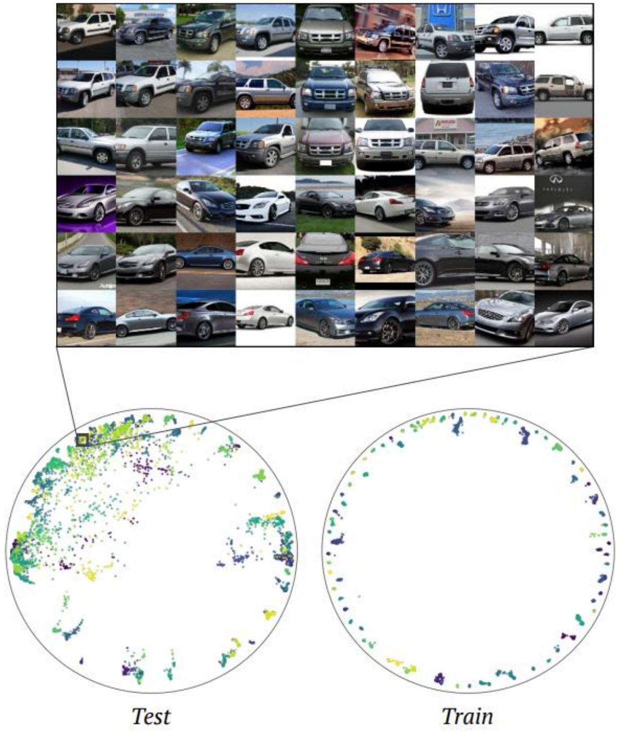
ViTs with hyperbolic cross-entropy loss achieve *better performance* with *fewer* dimensions

References: Aleksandr Ermolov, Leyla Mirvakhabova, Valentin Khrulkov, Nicu Sebe, and Ivan Oseledets. 2022. Hyperbolic vision transformers: Combining improvements in metric learning. In CVPR. 7409–7419.

Hyperbolic Vision Foundation Models: Hyp-ViT(5)

Visualization of embeddings of Hyp-DINO on the Poincare Disk

- Images of different classes are clustered towards the boundary, show that the classes are *well separated*



References: Aleksandr Ermolov, Leyla Mirvakhabova, Valentin Khrulkov, Nicu Sebe, and Ivan Oseledets. 2022. Hyperbolic vision transformers: Combining improvements in metric learning. In CVPR. 7409–7419.

Hyperbolic Language Vision Foundation Models: MERU (1)

Contrastive Language-Image Pre-Training (CLIP) models are foundation models that can process *both language and image data*

- Combines a text encoder (e.g. language Transformer) with an image encoder (e.g. vision Transformer)

The ***natural hierarchies*** in texts and images motivates adapting CLIP models to hyperbolic space

Relies on ***contrastive loss***

$$L_{const}(x_j, y_j) = -\frac{1}{2}\log\frac{e^{-\frac{||x_j - y_j||^2}{\tau}}}{\Sigma_{i\neq j}^{B} e^{-\frac{||x_j - y_i||^2}{\tau}}} - \frac{1}{2}\log\frac{e^{-\frac{||x_j - y_j||^2}{\tau}}}{\Sigma_{i\neq j}^{B} e^{-\frac{||x_i - y_j||^2}{\tau}}}$$

where x_j, y_j are text and image embeddings that form a positive pair

References: Karan Desai, Maximilian Nickel, Tanmay Rajpurohit, Justin Johnson, and Shanmukha Ramakrishna Vedantam. 2023. Hyperbolic image-text representations. In ICML. PMLR, 7694–7731.

Hyperbolic Language Vision Foundation Models: MERU (2)

Adapting contrastive loss to hyperbolic space

- Instead of cosine similarity, use *negative manifold distance*

$$L_{const}(x_j, y_j) = -\frac{1}{2}\log\frac{e^{-\frac{d_H(x_j,y_j)}{\tau}}}{\Sigma_{i\neq j}^{B} e^{-\frac{d_H(x_j,y_i)}{\tau}}} - \frac{1}{2}\log\frac{e^{-\frac{d_H(x_j,y_j)}{\tau}}}{\Sigma_{i\neq j}^{B} e^{-\frac{d_H(x_i,y_j)}{\tau}}}$$

where x_j, y_j are text and image embeddings that form a positive pair

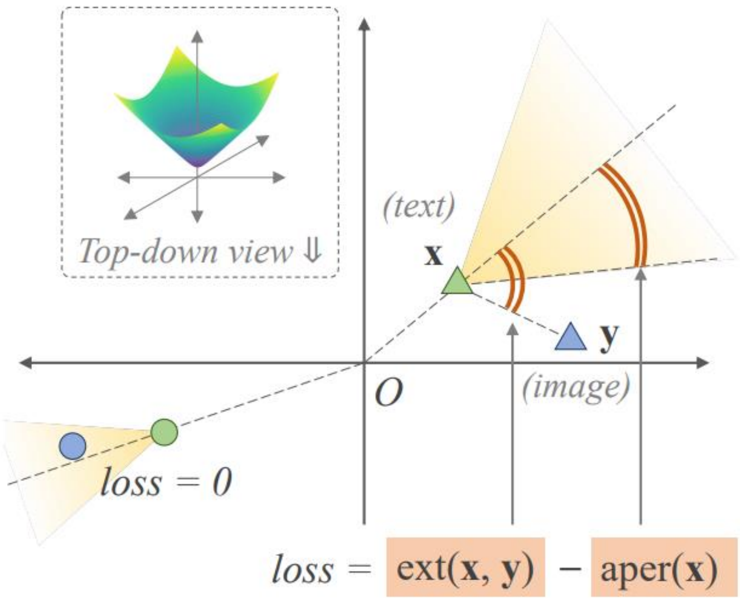
References: Karan Desai, Maximilian Nickel, Tanmay Rajpurohit, Justin Johnson, and Shanmukha Ramakrishna Vedantam. 2023. Hyperbolic image-text representations. In ICML. PMLR, 7694–7731.

Hyperbolic Language Vision Foundation Models: MERU (3)

Hyperbolic Entailment Cone: shining a light cone through a point, where the region is defined by where the light rays hit

- Given a point x, the entailment cone is defined by the ***aperture***: the angle at which the boundary makes with x:

$$aper(x) = \sin^{-1}\left(\frac{2\gamma}{\sqrt{-\frac{1}{K}||x_s||}}\right)$$



References: Karan Desai, Maximilian Nickel, Tanmay Rajpurohit, Justin Johnson, and Shanmukha Ramakrishna Vedantam. 2023. Hyperbolic image-text representations. In ICML. PMLR, 7694–7731.

Hyperbolic Language Vision Foundation Models: MERU (4)

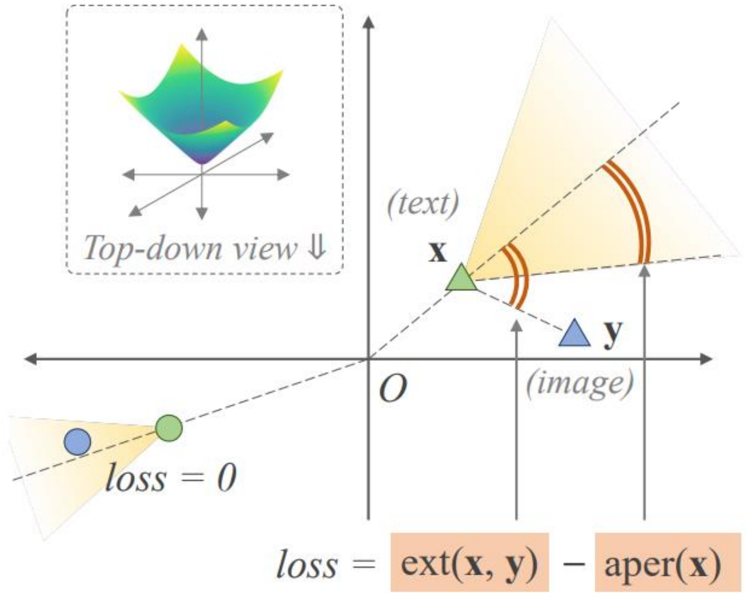
The ***hyperbolic entailment loss*** is defined by deviation from the entailment cone

- Positive pairs should be within the cone
- Negative pairs should be outside of the cone

The deviation is measure by the ***exterior angle***:

$$ext(x,y) = \cos^{-1}\left(\frac{y_t - \frac{x_t}{K}\langle x,y\rangle_L}{||x_s||\sqrt{\left(\frac{-1}{K}\langle x,y\rangle\right)^2 - 1}}\right).$$

Final loss: $L_{entail}(x,y) = ext(x,y) - aper(x)$

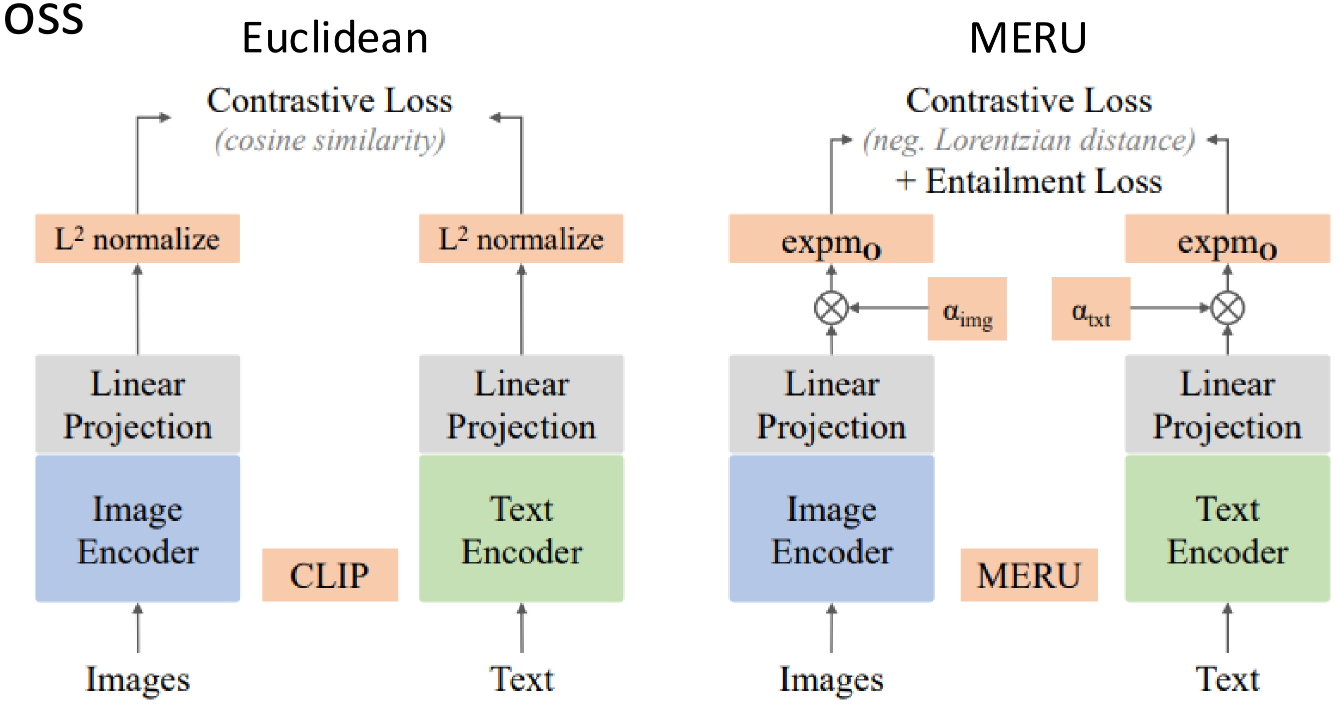


References: Karan Desai, Maximilian Nickel, Tanmay Rajpurohit, Justin Johnson, and Shanmukha Ramakrishna Vedantam. 2023. Hyperbolic image-text representations. In ICML. PMLR, 7694–7731.

Hyperbolic Language Vision Foundation Models: MERU (4)

Overall architecture of MERU

- Process the image and text data with Euclidean image and text encoders
- Normalize the Euclidean outputs for stable norm
- Lift to hyperbolic space and compute loss



References: Karan Desai, Maximilian Nickel, Tanmay Rajpurohit, Justin Johnson, and Shanmukha Ramakrishna Vedantam. 2023. Hyperbolic image-text representations. In ICML. PMLR, 7694–7731.

Hyperbolic Language Vision Foundation Models: MERU (5)

Performance evaluation of MERU

- Image-text retrieval on the COCO dataset

		Embedding width				
		512	256	128	96	64
COCO *text→image*	CLIP	31.7	31.8	31.4	29.6	25.7
	MERU	**32.6**	**32.7**	**32.7**	**31.0**	**26.5**
COCO *image→text*	CLIP	40.6	41.0	40.4	37.9	33.3
	MERU	**41.9**	**42.5**	**42.6**	**40.5**	**34.2**
ImageNet	CLIP	38.4	38.3	37.9	35.2	30.2
	MERU	**38.8**	**38.8**	**38.8**	**37.3**	**32.3**

MERU consistently outperforms the Euclidean CLIP model!

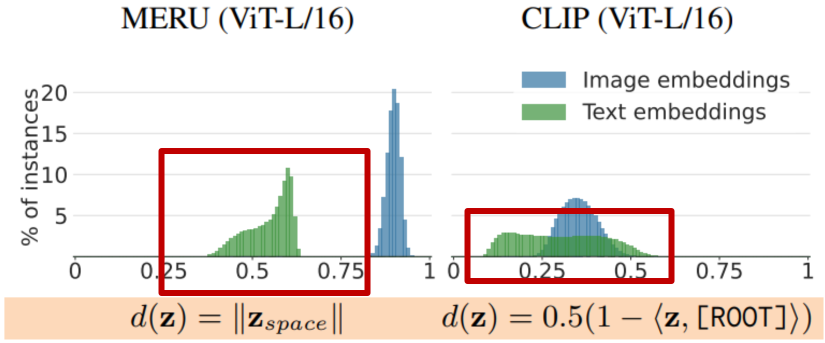
References: Karan Desai, Maximilian Nickel, Tanmay Rajpurohit, Justin Johnson, and Shanmukha Ramakrishna Vedantam. 2023. Hyperbolic image-text representations. In ICML. PMLR, 7694–7731.

Hyperbolic Language Vision Foundation Models: MERU (6)

Embedding distribution of MERU

- Constructing a visual semantic tree

- In Lorentz Space, it is the origin
- In Euclidean space, it is not well defined
  - Use the centroid of all embeddings



MERU better reflects the natural structure – it embeds texts (higher on the visual semantic hierarchy) *closer* to the root than it embeds images!

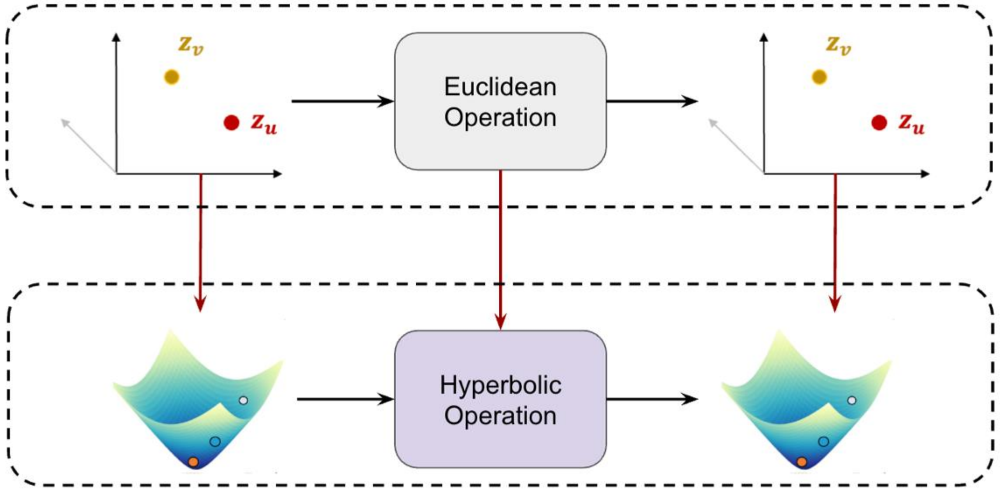
References: Karan Desai, Maximilian Nickel, Tanmay Rajpurohit, Justin Johnson, and Shanmukha Ramakrishna Vedantam. 2023. Hyperbolic image-text representations. In ICML. PMLR, 7694–7731.

Towards Non-Euclidean Foundation Models

"Hyperbolic-fy Operations/Modules in foundation models", e.g.,

- Residual Connection -> LResNet
- Attention Mechanism -> Hyperbolic Attention
- Linear Layer -> $f^{F,K}, f^{T,K}$
- Activation -> Pseudo Lorentz Rotation, tangent-space operations
- LoRA -> HypLoRA

Euclidean Foundation Model



Hyperbolic Foundation Model

But what else?

Goal: Encode geometric structure into the model that the model ***cannot*** *do a good job learning otherwise*

Challenges

- Building hyperbolic foundation models ***would not be simple***
  - Require developing methods with abundance of knowledge in differential geometry
  - Special geometric functions and difficulty in implementing even basic operations, e.g. addition
  - Scattered prior research and incompatibilities
- **Issues with Existing Tools**
  - Limited Modules
  - Inflexibility and Unintuitive-Usage
    - Require extensive geometry knowledge
  - Limited Model Support: difficult to build advanced foundation models
  - Limited to one formulation of hyperbolic space (Poincare or Lorentz)

Hyperbolic Foundation Model Library: HyperCore

- **Flexible** to Create various SoTA models
  - Spotlight Examples: LViT, L-CLIP, Hyperbolic GraphRAG
- **Comprehensive** Modules and Model Support
- **Intuitive** Foundation Model Support
  - Focus on making it easier to build foundation model pipelines
- User **Accessibility**
  - Use the library without being an expert in hyperbolic geometry

Framework	MLPs	GNNs	CNNs	Transformers	ViTs	Fine Tuning	CLIP	Graph RAG	$\mathbb{L}^{n,K}$	$\mathbb{P}^{n,K}$
HypLL [55]	✓	✗	✓	✗	✗	✗	✗	✗	✗	✓
Hyperlib [1]	✓	✓	✗	✗	✗	✗	✗	✗	✓	✓
HyperCore	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

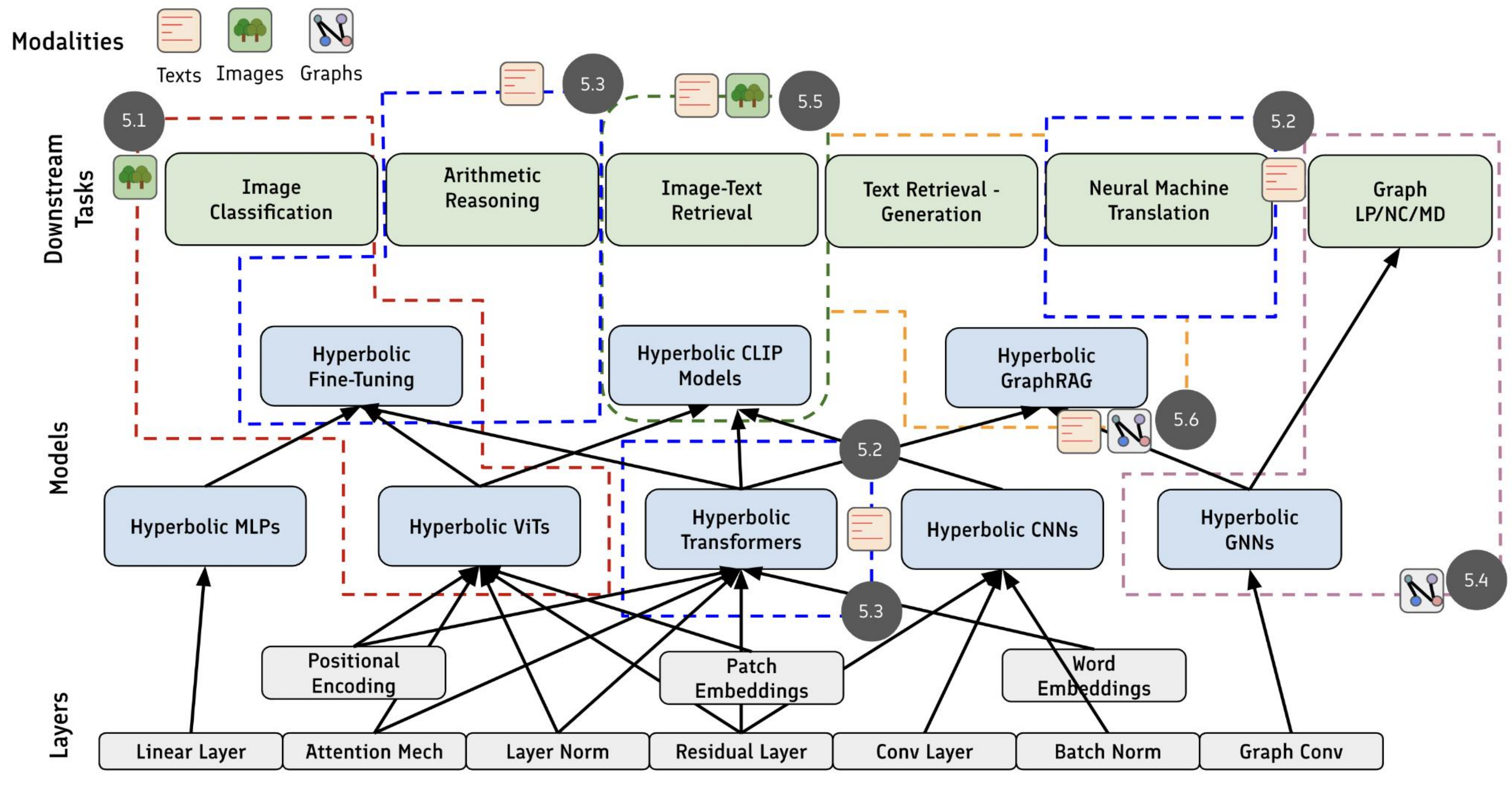
References: Neil He, Menglin Yang, and Rex Ying. 2025. HyperCore: The Core Framework for Building Hyperbolic Foundation Models with Comprehensive Modules.TheWebConf NEGEL Workshop (2025)

Library Overview

- **Modules**
  - Neural network layers (e.g. linear, convolutional, MLR)
  - Transformer layers (e.g. softmax self-attention, linear attention, latent attention, positional encoding, word embedding, patch embedding)
  - Graph related (e.g. graph convolutional layers and neighborhood aggregation)
  - Fine-tuning
  - Essential modules (e.g. layer normalization, residual connection, pooling layers)
- **Optimizers**
  - Support for different training schemes on Euclidean v.s. manifold parameters
- **Manifold**
  - Basic manifold operations and additional operations (e.g. concatenation and splitting vectors, hyperbolic entailment cones)

References: Neil He, Menglin Yang, and Rex Ying. 2025. HyperCore: The Core Framework for Building Hyperbolic Foundation Models with Comprehensive Modules. TheWebConf NEGEL Workshop (2025)

Snapshot of Library Taxonomy



References: Neil He, Menglin Yang, and Rex Ying. 2025. HyperCore: The Core Framework for Building Hyperbolic Foundation Models with Comprehensive Modules. TheWebConf NEGEL Workshop (2025)

Example: Transformer Block

Euclidean Transformer Block

```
import torch
from torch import nn
from collections import OrderedDict

class TransformerBlock(nn.Module):
    def __init__(self, d_model: int, n_head: int):
        super().__init__()

        self.attn = nn.MultiheadAttention(d_model, n_head,
    batch_first=True)
        self.ln_1 = nn.LayerNorm(d_model)
        self.mlp = nn.Sequential(
            OrderedDict(
                [
                    ("c_fc", nn.Linear(d_model, d_model * 4)),
                    ("gelu", nn.GELU()),
                    ("c_proj", nn.Linear(d_model * 4, d_model)),
                ]
            )
        )
        self.ln_2 = nn.LayerNorm(d_model)

    def forward(self, x: torch.Tensor, attn_mask: torch.Tensor |
    None = None):
        lx = self.ln_1(x)
        ax = self.attn(lx, lx, lx, need_weights=False, attn_mask=
    attn_mask)[0]
        x = x + ax
        x = x + self.mlp(self.ln_2(x))
        return x
```

Lorentz Transformer Block w/ HyperCore

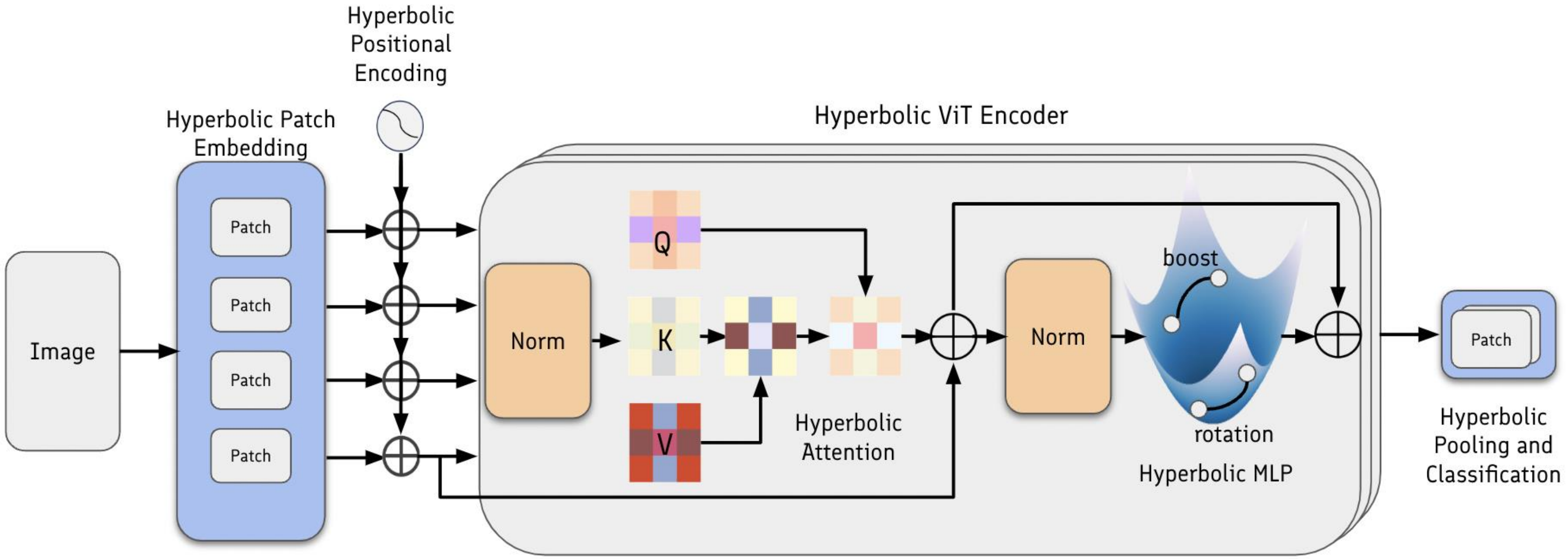
```
import torch
import torch.nn as nn
import hypercore.nn as hnn
from collections import OrderedDict

class LTransformerBlock(nn.Module):
    def __init__(self, manifold, d_model: int, n_head: int):
        super().__init__()
        dim_per_head = d_model // n_head
        self.manifold = manifold
        self.attn = hnn.LorentzMultiheadAttention(manifold,
    dim_per_head, dim_per_head, n_head, attention_type='full',
    trans_heads_concat=True)
        self.ln_1 = hnn.LorentzLayerNorm(manifold, d_model -1)
        self.mlp = nn.Sequential(
            OrderedDict(
                [
                    ("c_fc", hnn.LorentzLinear(manifold, d_model,
    d_model*4-1)),
                    ("gelu", hnn.LorentzActivation(manifold,
    activation=nn.GELU())),
                    ("c_proj", hnn.LorentzLinear(manifold, d_model
    *4, d_model-1)),
                ]
            )
        )
        self.ln_2 = hnn.LorentzLayerNorm(manifold, d_model-1)
        self.res1 = hnn.LResNet(manifold, use_scale=True)
        self.res2 = hnn.LResNet(manifold, use_scale=True)

    def forward(self, x, attn_mask=None):
        lx = self.ln_1(x)
        ax = self.attn(lx, lx, output_attentions=False, mask=
    attn_mask)
        x = self.res1(x, ax)
        x = self.res2(x, self.mlp(self.ln_2(x)))
        return x
```

New Hyperbolic Foundation Models w/ HyperCore: LViT

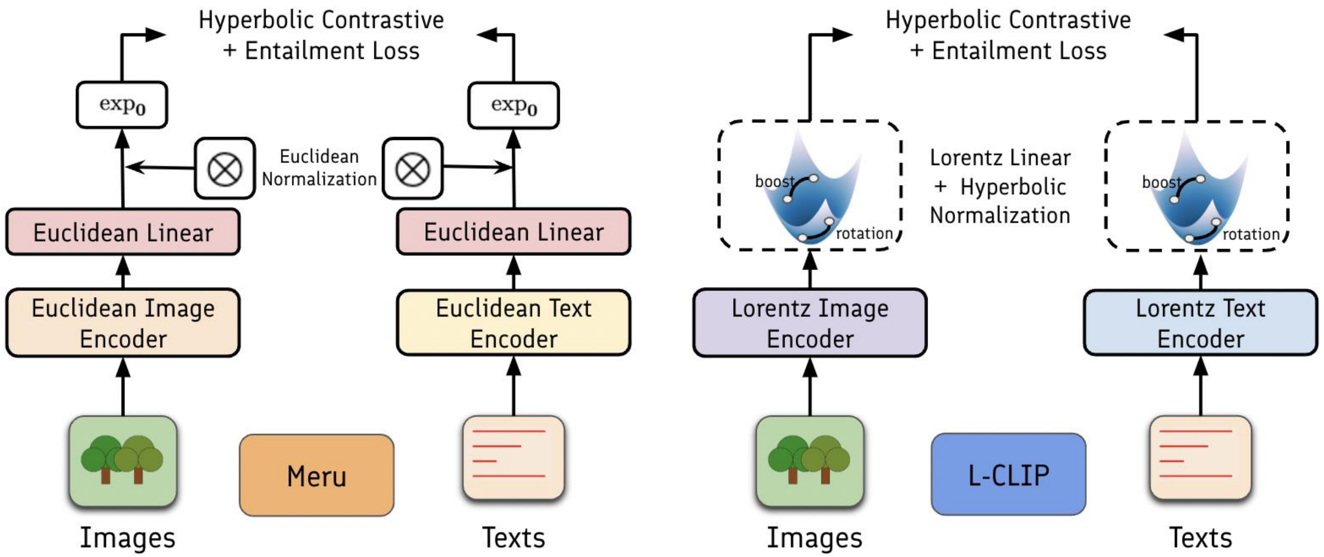
- First fully hyperbolic vision transformer with a fine-tuning pipeline, built with HyperCore



References: Neil He, Menglin Yang, and Rex Ying. 2025. HyperCore: The Core Framework for Building Hyperbolic Foundation Models with Comprehensive Modules. TheWebConf NEGEL Workshop (2025)

New Hyperbolic Foundation Models w/ HyperCore: L-CLIP

- First fully hyperbolic multi-modal CLIP model
  - Compared to MERU, which is a hybrid model



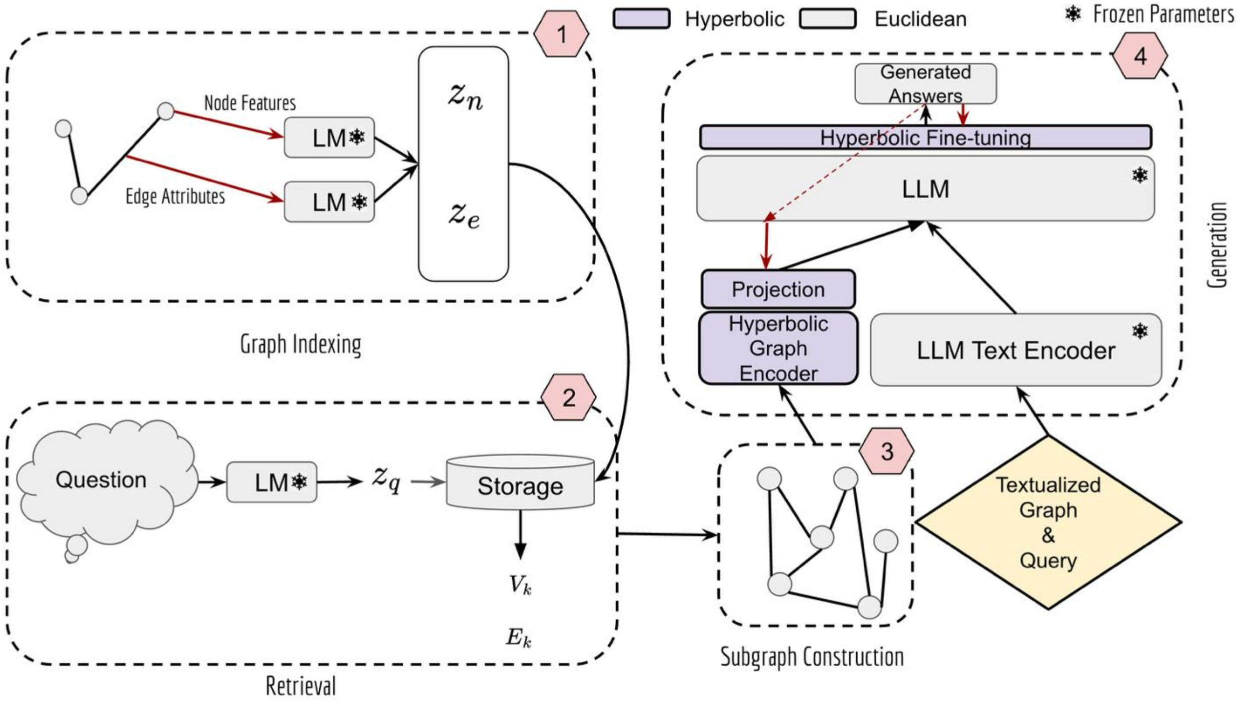
References: Neil He, Menglin Yang, and Rex Ying. 2025. HyperCore: The Core Framework for Building Hyperbolic Foundation Models with Comprehensive Modules. TheWebConf NEGEL Workshop (2025)

New Hyperbolic Foundation Models w/ HyperCore: HypGraphRAG

First Hyperbolic GraphRAG model:

- Uses a hyperbolic graph encoder
- Uses hyperbolic fine-tuning

Better represent the knowledge graph structure



References: Neil He, Menglin Yang, and Rex Ying. 2025. HyperCore: The Core Framework for Building Hyperbolic Foundation Models with Comprehensive Modules. TheWebConf NEGEL Workshop (2025)

Testing New Hyperbolic Models – LViT

- Image Classification with LViT
  - Fine-tuning with HypLoRA on smaller datasets
- Datasets
  - ImageNet-1K: 1.2M images of 1,000 classes
  - CIFAR10 and CIFAR100: 60K images of 10 (100) classes
  - TinyImageNet: 100K images of 200 classes

Every hyperbolic model here is implemented with HyperCore

Dataset **Hyperbolicity**	CIFAR-10 $\delta = 0.26$	CIFAR-100 $\delta = 0.23$	TINY-IMAGENET $\delta = 0.20$	IMAGENET -	
HCNN [54]	95.02 ± 0.19	77.31 ± 0.21	65.01 ± 0.29	-	Hyperbolic ResNets
Poincaré ResNet [6]	94.71 ± 0.13	76.91 ± 0.34	63.11 ± 0.59	-	Hyperbolic ResNets
ViT [21] (Euclidean ViT)	98.13	87.13	-	77.91	
HVT [24] (Tangent Space ViT)	61.44	42.77	40.12	78.2	
LViT (built by us)	85.02	69.11	53.01	79.4	
LViT (fine-tuned w/ HypLoRA)	**98.18**	**87.36**	**74.11**	**79.4**	

References: Neil He, Menglin Yang, and Rex Ying. 2025. HyperCore: The Core Framework for Building Hyperbolic Foundation Models with Comprehensive Modules. TheWebConf NEGEL Workshop (2025)

Testing New Hyperbolic Models – L-CLIP & Hyperbolic GraphRAG

- Image-Text Retrieval on COCO benchmark with L-CLIP
  - Image encoder: LViT; Text encoder: hyperbolic Transformer
- HypGraphRAG: Question-answering tasks in a graph QA dataset (WebQSP)
  - Skip-connected hyperbolic GNN; LLaMA3.1-8B fine-tuned with HypLoRA

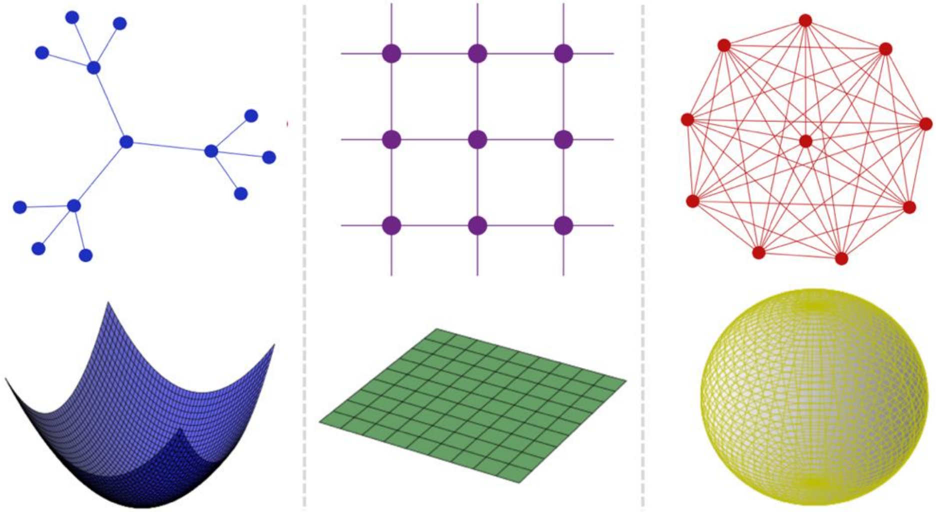
Experimental Goal: To demonstrate what's possible

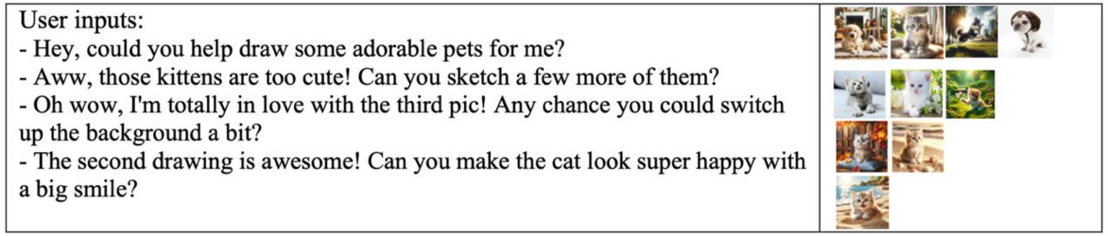
Model	L-CLIP		HypGraphRAG
Dataset	COCO		WebQSP
Task	Image-Text Retrieval		Question-answering
Metric	Recall@5	Recall@10	Hi@1
Restults	28.0	38.1	73.89 ± 1.09

References: Neil He, Menglin Yang, and Rex Ying. 2025. HyperCore: The Core Framework for Building Hyperbolic Foundation Models with Comprehensive Modules. TheWebConf NEGEL Workshop (2025)

Future works

Ultimate goal: Combine non-Euclidean foundation model with large model for Geometric-aware AI





Examples of generating images from corse-grained to fine-grained, aligning human cognition process

From hyperbolic space to adaptive curvature space

From language model to multimodal models

Non-Euclidean Foundation Model

Multimodal LM

Geometric AI

Future works

Training Future Hyperbolic Foundation Models

Fully Hyperbolic Pre-trained Models:

- The majority of current works only consider *Euclidean pre-trained models* as backbones while pre-trained hyperbolic models (e.g. HELM) does not compare in size
- This does not *fully leverage the representation power* of hyperbolic space
- Pre-training hyperbolic models at the scale of Euclidean foundation models could lead to more general hyperbolic representations for downstream tasks

Parameter-efficient Foundation Models:

- Hyperbolic foundation models present the exciting potential for more favorable scaling by compressing geometric information, whereas Euclidean foundation models' performance experience exponentially diminishing returns w.r.t parameter count

Efficient and Intuitive Model Training:

- While libraries such as HyperCore exists, there is a lack of libraries comparable to Euclidean counterparts.
- For instance, it is common for prior works to utilize separate optimizers for Euclidean and hyperbolic parameters, which is not supported by current foundation models libraries such as DeepSpeed.

Future works

Designing Future Hyperbolic Foundation Models

Hyperbolic Retrieval Augmented Generation:

- Hyperbolic retrieval modules, which leverage the hierarchical and scale-free properties of hyperbolic space, could provide a more effective mechanism for document retrieval in knowledge intensive tasks due to the natural hierarchical structure in the external knowledge base
- Hyperbolic nearest neighbor search, ranking mechanisms, and generative architectures could lead to more structured, accurate, and computationally efficient retrieval-augmented generation systems

Hyperbolic Generative Models

- Hyperbolic generative models would be able to better model hierarchical distributions, e.g. series action states

Geometric Insights for Method Design:

- Geometric insights could enhance our understanding and potentially lead to more effective and efficient methods
- Example:
    - Fully hyperbolic operations still have ambiguous geometric meaning for operations other than linear operations and HoPE
    - Designing fully hyperbolic operations for Poincare Ball model
    - Hyperbolic diffusion models lack theoretical guarantees due to the manifold’s uncompactness

Resources

Papers

1. Neil He, Menglin Yang, and Rex Ying. 2025. Lorentzian Residual Neural Networks. In KDD.
2. Menglin Yang, Harshit Verma, Delvin Ce Zhang, Jiahong Liu, Irwin King, and Rex Ying. 2024. Hypformer: Exploring efficient transformer fully in hyperbolic space. In KDD. 3770–3781.
3. Ngoc Bui, Menglin Yang, Runjin Chen, Leonardo Neves, Mingxuan Ju, Rex Ying, Neil Shah, Tong Zhao. Learning Along the Arrow of Time: Hyperbolic Geometry for Backward-Compatible Representation Learning. ICML 2025
4. Menglin Yang, Aosong Feng, Bo Xiong, Jihong Liu, Irwin King, and Rex Ying. 2024. Hyperbolic Fine-tuning for Large Language Models. ICML LLM Cognition Workshop (2024).
5. Qiyao Ma, Menglin Yang, Mingxuan Ju, Tong Zhao, Neil Shah, Rex Ying . 2025. Breaking Information Cocoons: A Hyperbolic Graph-LLM Framework for Exploration and Exploitation in Recommender Systems. In preprint.
6. Weize Chen, Xu Han, Yankai Lin, Hexu Zhao, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2022. Fully Hyperbolic Neural Networks. In ACL
7. Ines Chami, Zhitao Ying, Christopher Ré, and Jure Leskovec. 2019. Hyperbolic graph convolutional neural networks. In NeurIPS. 4868–4879.
8. Neil He, Rishabh Anand, Hiren Madhu, Ali Maatouk, Smita Krishnaswamy, Leandros Tassiulas, Menglin Yang, and Rex Ying. 2025. HELM: Hyperbolic Large Language Models via Mixture-of-Curvature Experts. In Preprint.
9. Neil He, Jiahong Liu, Buze Zhang, Ngoc Bui, Ali Maatouk, Menglin Yang, Irwin King, Melanie Weber, and Rex Ying. 2025. Position: Beyond Euclidean– Foundation Models Should Embrace Non-Euclidean Geometries. In Preprint.
10. Marc Law, Renjie Liao, Jake Snell, and Richard Zemel. 2019. Lorentzian distance learning for hyperbolic representations. In ICML. PMLR, 3672–3681.
11. Ryohei Shimizu, Yusuke Mukuta, and Tatsuya Harada. 2020. Hyperbolic Neural Networks++. In ICLR
12. Max van Spengler, Erwin Berkhout, and Pascal Mettes. 2023. Poincaré ResNet. CVPR (2023)
13. Ahmad Bdeir, Kristian Schwethelm, and Niels Landwehr. 2024. Fully Hyperbolic Convolutional Neural Networks for Computer Vision. In ICLR.

Resources

Papers

14. Eric Qu and Dongmian Zou. 2022. Lorentzian fully hyperbolic generative adversarial network. arXiv:2201.12825 (2022).
15. Aleksandr Ermolov, Leyla Mirvakhabova, Valentin Khrulkov, Nicu Sebe, and Ivan Oseledets. 2022. Hyperbolic vision transformers: Combining improvements in metric learning. In CVPR. 7409–7419.
16. Valentin Khrulkov, Leyla Mirvakhabova, Evgeniya Ustinova, Ivan Oseledets, and Victor Lempitsky. 2020. Hyperbolic image embeddings. In IEEE/CVF CVPR. 6418–6428.
17. Karan Desai, Maximilian Nickel, Tanmay Rajpurohit, Justin Johnson, and Shanmukha Ramakrishna Vedantam. 2023. Hyperbolic image-text representations. In ICML. PMLR, 7694–7731.

Tools:

1. Neil He, Menglin Yang, and Rex Ying. 2025. HyperCore: The Core Framework for Building Hyperbolic Foundation Models with Comprehensive Modules. In Preprint.

Thank You