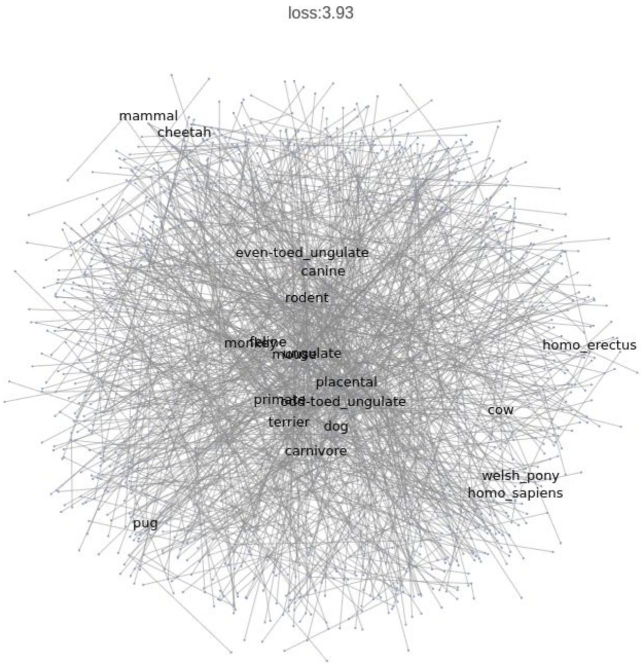
Hyperbolic Machine Learning



Hyperbolic Geometry & Data Science Seminar, 21.01.21, Sebastian Damrich

Overview

Trees in Hyperbolic Space

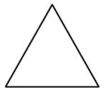
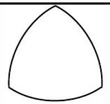
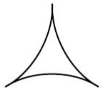
Riemannian Gradient Descent

Shallow hyperbolic ML

Hyperbolic Deep ML

Euclidean, spherical and hyperbolic geometry

TABLE I: Characteristic properties of Euclidean, spherical, and hyperbolic geometries. *Parallel lines* is the number of lines that are parallel to a line and that go through a point not belonging to this line, and $\zeta = \sqrt{|K|}$.

Property	Euclidean	Spherical	Hyperbolic
Curvature K	0	> 0	< 0
Parallel lines	1	0	∞
Triangles are	normal	thick	thin
Shape of triangles			
Sum of $\triangle$ angles	π	$> \pi$	$< \pi$
Circle length	$2\pi r$	$2\pi \sin \zeta r$	$2\pi \sinh \zeta r$
Disk area	$2\pi r^2/2$	$2\pi(1 - \cos \zeta r)$	$2\pi(\cosh \zeta r - 1)$

Hyperbolic Geometry of Complex Networks, Krioukov, …, Boguñá

Equivalent models

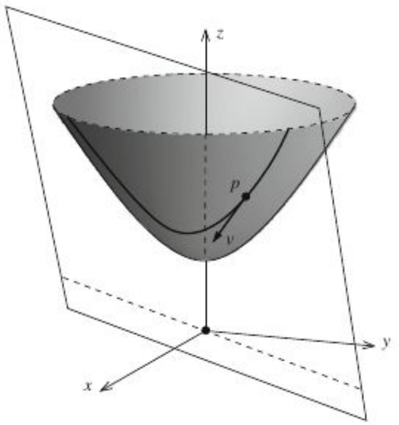


Fig. 5.5: A great hyperbola

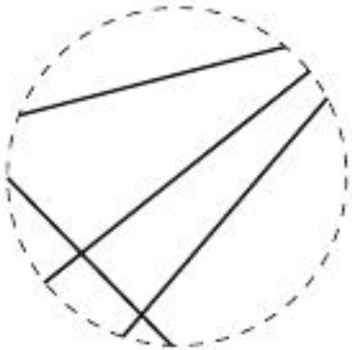


Fig. 5.6: Geodesics of $\mathbb{K}^n(R)$

Fig. 5.7: Geodesics of $\mathbb{B}^n(R)$

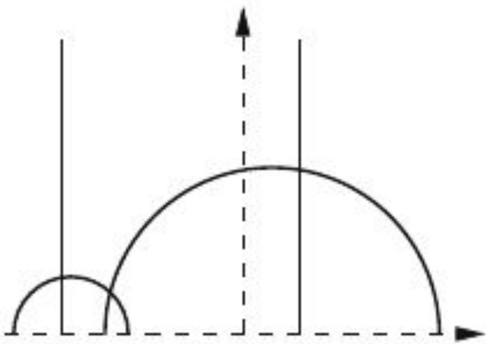


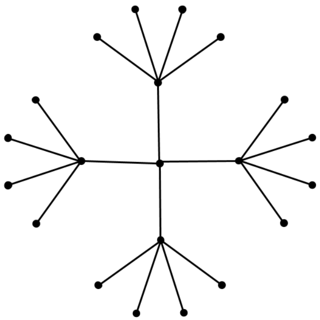
Fig. 5.8: Geodesics of $\mathbb{U}^n(R)$

Introduction to Riemannian Manifolds, Lee, Springer

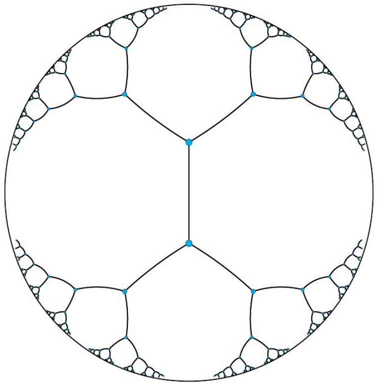
Trees in hyperbolic space

“hyperbolic space is a continuous analogue of trees;
trees are a discretised hyperbolic space”

hyperbolic space has “more space” than Euclidean space



not enough space for tree with constant branching factor in euclidean space
http://bjlkeng.github.io/posts/hyperbolic-geometry-and-poincare-embeddings/



Area grows exponentially and can fit a tree with constant branching factor
Poincaré Embeddings for Learning Hierarchical Representations, Nickel, Kiela Neurips’17

Trees in hyperbolic space

Theorem:

Given $\varepsilon > 0$ and a positively weighted tree $T = (V, E, w)$, then there is some $\eta > 0$ such that V can be embedded into the Poincaré disk such that ηT is the MST of the embedded points and the distortion (product of max distance contraction and elongation) is at most $1 + \varepsilon$.

Low Distortion Delaunay Embedding of Trees in Hyperbolic Plane, Sarkar, 2012

Hierarchical data is everywhere

Language: Hypernymys, Entailment of sentences, Translation…

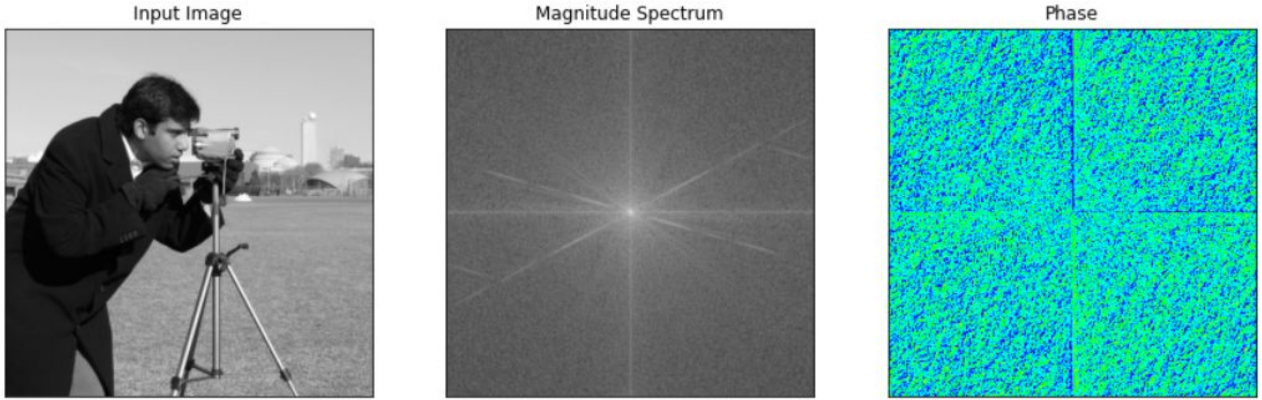
Images: Tracking of dividing cells, different resolutions, crops

Biology: Developmental processes

Many graphs are rather tree-like than flat

Representation learning

Representing data in a way that makes downstream tasks easy, e.g.



Representation learning

Representing data in a way that makes downstream tasks easy, e.g.

- Embedding data points in a metric space
  - Use distance for clustering, retrieving similar data points, …
- More compact than relational information
- Deep Neural Networks
  - sequence of transformed representation

Often representation as point in Euclidean space, but not well suited for hierarchies.

→ Learn representations in hyperbolic space instead

Riemannian Stochastic Gradient Descent

Gradient descent is ubiquitous in Machine Learning

$$\min_{\theta} f(\theta)$$

$$\theta_{t+1} = \theta_t - \eta_t \operatorname{grad} f|_{\theta_t}$$

Riemannian Stochastic Gradient Descent

Gradient descent is ubiquitous in Machine Learning

$$\min_{\theta} f(\theta)$$

$$\theta_{t+1} = \theta_t - \eta_t \operatorname{grad} f|_{\theta_t}$$

$$= \exp_{\theta_t}\left(-\eta_t \cdot \operatorname{grad} f|_{\theta_t}\right)$$

Riemannian Gradient Descent

$$\min_{\theta \in \mathcal{M}} f(\theta)$$

$$\theta_{t+1} = \exp^{\mathcal{M}}_{\theta_t}\left(-\eta_t \cdot \mathrm{grad}^{\mathcal{M}} f|_{\theta_t}\right)$$

$$\mathrm{grad}^{\mathrm{M}} f = g^{ij} \frac{\partial f}{\partial x^i} \frac{\partial}{\partial x^j}$$

Riemannian Stochastic Gradient Descent

Theorem (Bonnabel):

Given a cost function f: M → R on a Riemannian Manifold, (an approximation of) Riemannian Stochastic Gradient Descent converges almost surely to a critical point of f and grad f to 0 under mild conditions.

Stochastic gradient descent on Riemannian manifolds, Bonnabel 2013

Shallow hyperbolic ML

Embed a graph G=(V, E) (representing a hierarchy) into hyperbolic space.

Let v_i is embedded as p_i in hyperbolic space.

Perform Riemannian gradient descent on loss function L(p, E).

Poincaré Embeddings for Learning Hierarchical Representations Nickel, Kiela NeurIPS'17

Data: Undirected version of transitive closure of directed WordNet dataset

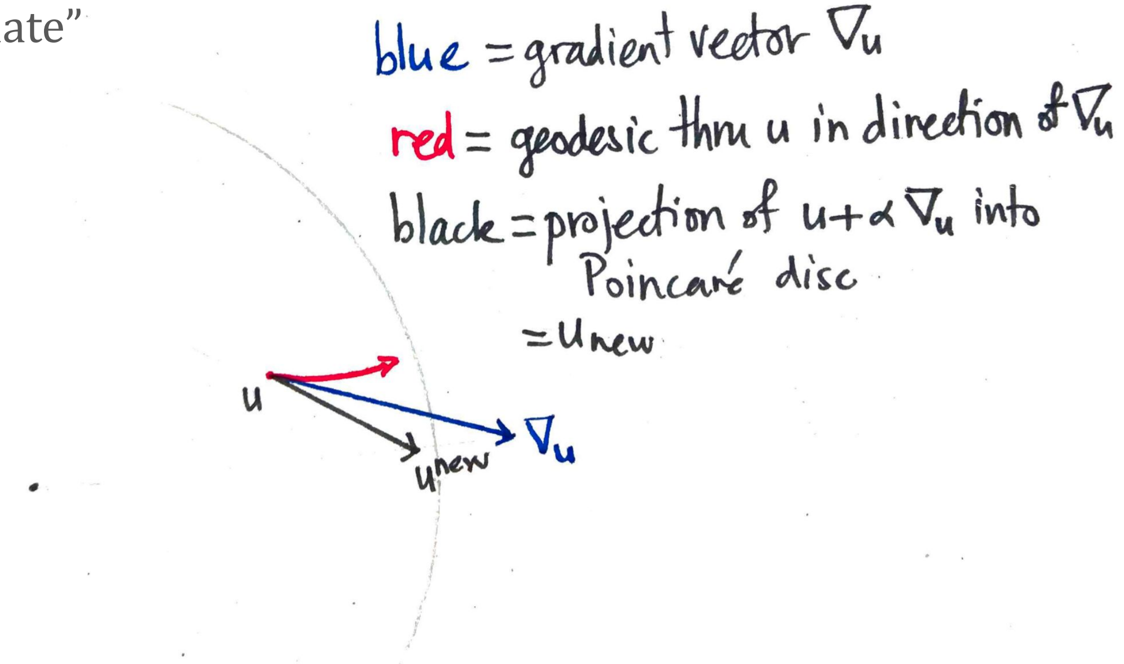
Loss: $$\mathcal{L}(\Theta) = \sum_{(u,v)\in\mathcal{D}} \log \frac{e^{-d(\boldsymbol{u},\boldsymbol{v})}}{\sum_{\boldsymbol{v}'\in\mathcal{N}(u)} e^{-d(\boldsymbol{u},\boldsymbol{v}')}}$$

No exponential, but approximation $$\boldsymbol{\theta}_{t+1} \leftarrow \mathrm{proj}\left(\boldsymbol{\theta}_t - \eta_t \frac{(1-\|\boldsymbol{\theta}_t\|^2)^2}{4} \nabla_E\right)$$

$$\mathrm{proj}(\boldsymbol{\theta}) = \begin{cases} \boldsymbol{\theta}/\|\boldsymbol{\theta}\| - \varepsilon & \text{if } \|\boldsymbol{\theta}\| \geq 1 \\ \boldsymbol{\theta} & \text{otherwise ,} \end{cases}$$

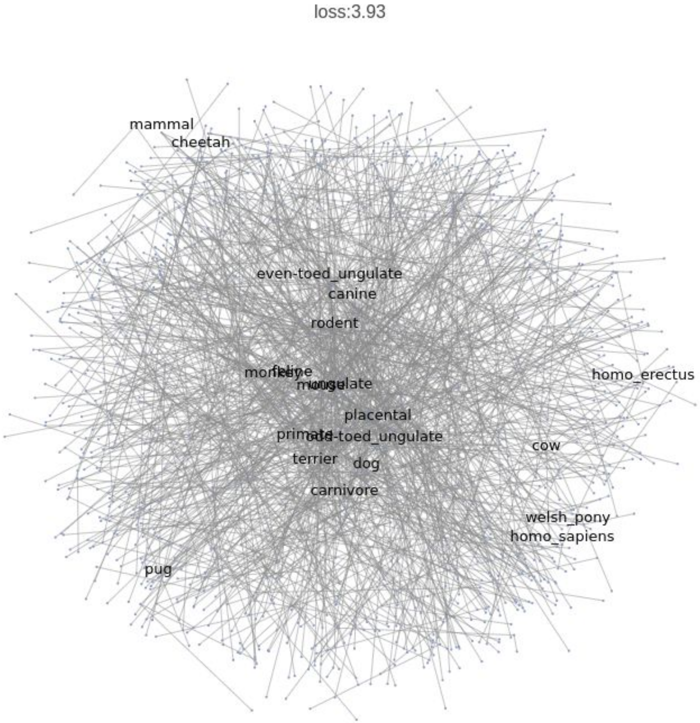
Poincaré Embeddings for Learning Hierarchical Representations Nickel, Kiela NeurIPS'17

"Buzz lightyear update"



http://building-babylon.net/2018/04/10/gradient-optimisation-on-the-poincare-disc/

Poincaré Embeddings for Learning Hierarchical Representations Nickel, Kiela NeurIPS’17



Poincaré Embeddings for Learning Hierarchical Representations Nickel, Kiela NeurIPS'17

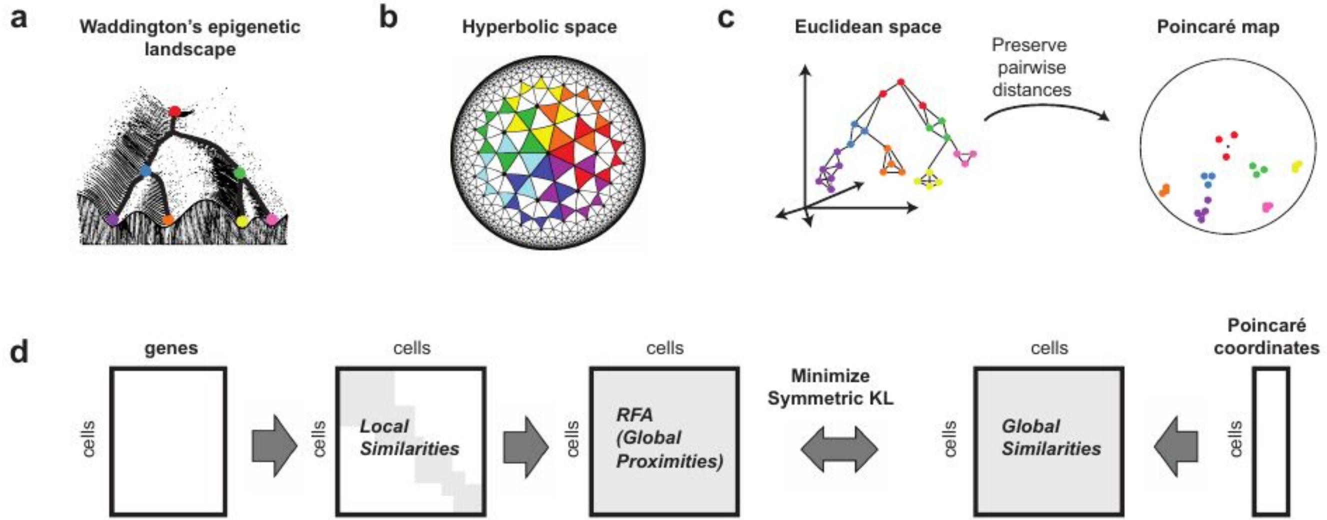
			Dimensionality					
			5	10	20	50	100	200
WordNet Reconstruction	Euclidean	Rank	3542.3	2286.9	1685.9	1281.7	1187.3	1157.3
		MAP	0.024	0.059	0.087	0.140	0.162	0.168
	Poincaré	Rank	4.9	4.02	3.84	3.98	3.9	**3.83**
		MAP	0.823	0.851	0.855	0.86	0.857	**0.87**
WordNet Link Pred.	Euclidean	Rank	3311.1	2199.5	952.3	351.4	190.7	81.5
		MAP	0.024	0.059	0.176	0.286	0.428	0.490
	Poincaré	Rank	5.7	**4.3**	4.9	4.6	4.6	4.6
		MAP	0.825	0.852	0.861	**0.863**	0.856	0.855

Clemens Fruböse improved this with a clever initialisation.

Poincaré Maps for Analyzing Complex Hierarchies in Single-Cell Data Klimovskaia, ..., Bottou, Nickel, 2020

Application to scRNA data of developing cell population

- Gene expression measurements for cells



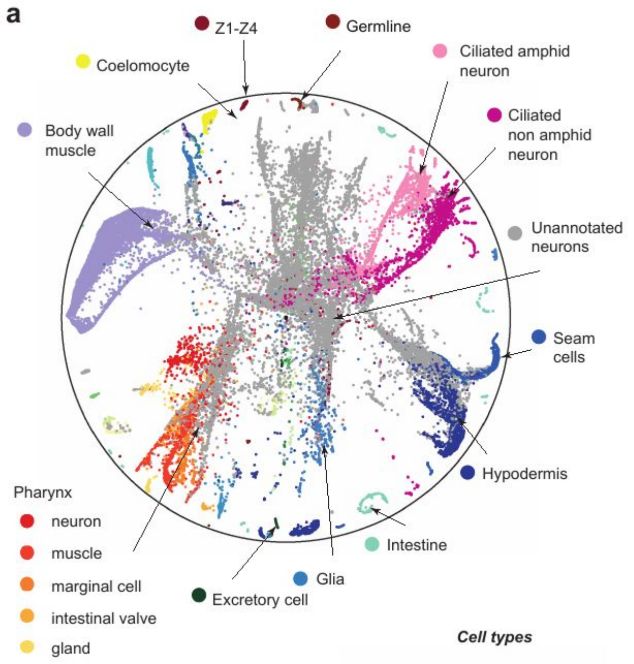
p_{ij}=random forest accessibility index on kernel-transformed connected mutual kNN graph

$$q_{ij} = \frac{\exp(-d_p(\boldsymbol{y}_i, \boldsymbol{y}_j)/\gamma)}{\sum_k \exp(-d_p(\boldsymbol{y}_i, \boldsymbol{y}_k)/\gamma)}$$

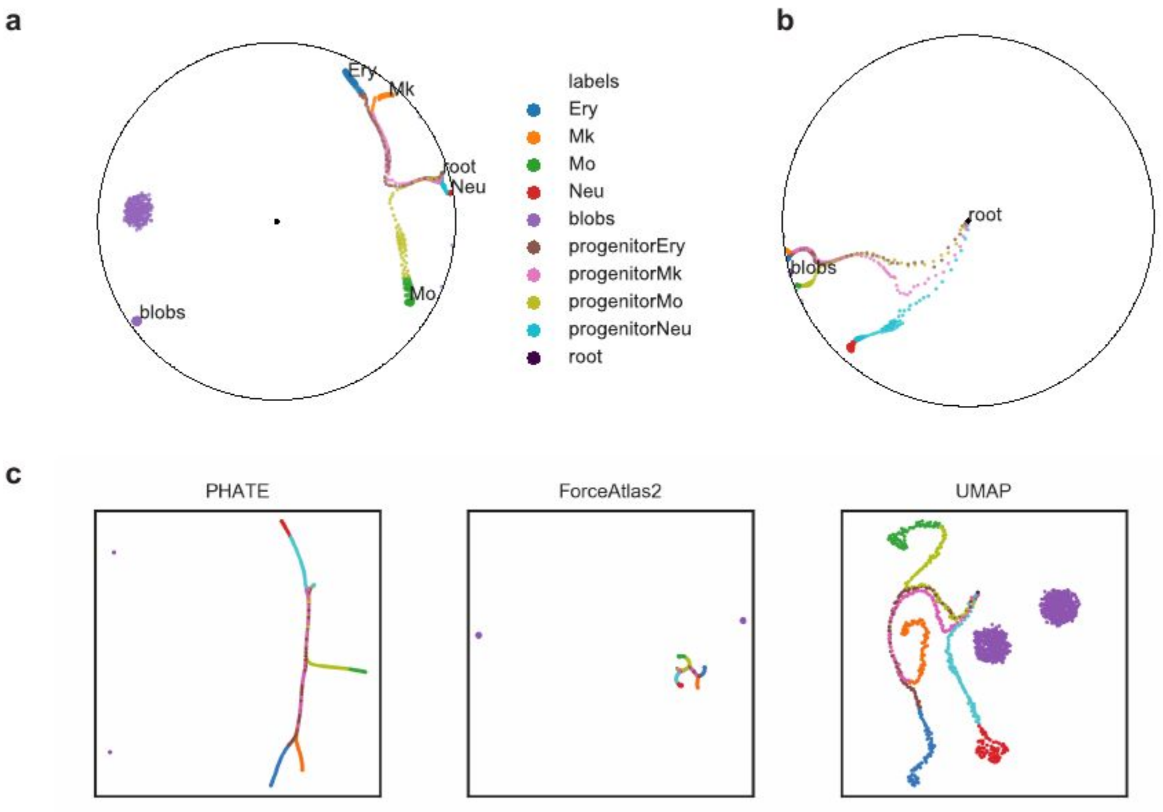
Poincaré Maps for Analyzing Complex Hierarchies in Single-Cell Data Klimovskaia, ..., Bottou, Nickel, 2020

Versatile embedding:

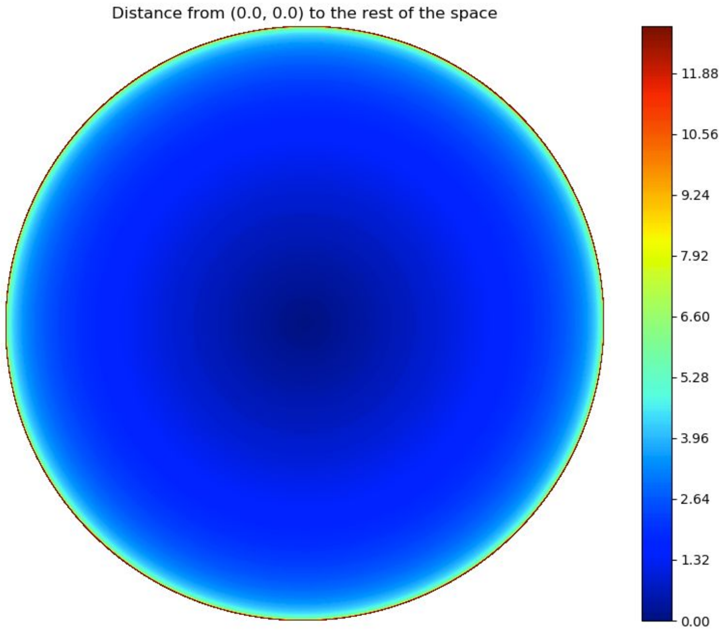
- agglomerative clustering for lineage detection
- pseudotime inference
- visualisation

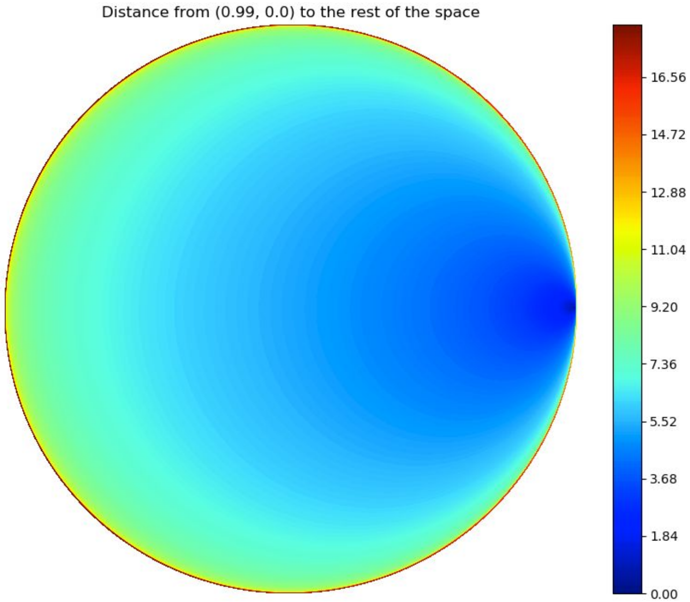


Poincaré Maps for Analyzing Complex Hierarchies in Single-Cell Data Klimovskaia, ..., Bottou, Nickel, 2020



Distance distortion in Poincaré disk





https://github.com/dwright37/lorentz_embeddings_recreation

Gyrovectors

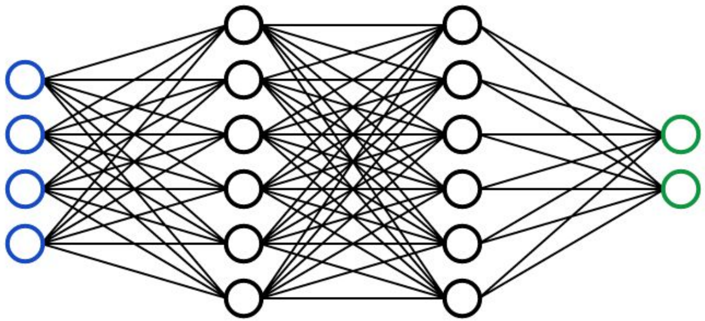
Hyperbolic space is not a vector space.
But carries more complicated structure of “gyrovector space”.

Addition $x \oplus_c y$, scalar multiplication $r \otimes_c x$

Geodesics, translation, exponential, logarithm can be expressed in terms of gyrovector operations.

Parameter c is negative curvature c → 0 give normal real vector space.

Hyperbolic Deep Learning



https://victorzhou.com/series/neural-networks-from-scratch/

Remember: Simple fully connected neural network is sequence of affine maps and non-linear maps:

$$f(x) = \varphi_N(W_N(\dots \varphi_2(W_2(\varphi_1(W_1x+b_1))+b_2)\dots)+b_N)$$

With Euclidean input x, Euclidean parameter matrices W_i and biases b_i, learnt by Gradient descent.

Hyperbolic Neural Network:
Neural network with hidden representation and / or parameters in hyperbolic space.

Hyperbolic Neural Networks Ganea, …, Hofmann, NeurIPS'18

Define

multinomial logistic regression

linear layer

for hyperbolic activations.

Map between Euclidean space input and hyperbolic activations via exponential at origin.

Hyperbolic Neural Networks Ganea, …, Hofmann, NeurIPS'18

Linear layer:

For $f : \mathbb{R}^n \to \mathbb{R}^m$, we define $f^{\otimes_c}(x) := \exp_{\mathbf{0}}^c(f(\log_{\mathbf{0}}^c(x)))$

→ do map in tangent space of the origin, which is isomorphic to R^n.

Bias b in D^n $x \leftarrow x \oplus_c b = \exp_x^c(P_{\mathbf{0} \to x}^c(\log_{\mathbf{0}}^c(b)))$

Non-linearity also in tangent space of origin

→ defines fully connected feed-forward hyperbolic network

→ Euclidean gradient descent wrt Euclidean weight matrix,
Riemannian GD wrt hyperbolic bias parameter

Hyperbolic Neural Networks Ganea, …, Hofmann, NeurIPS'18

Euclidean multinomial logistic regression

Simple classifier method, for each input x, probability distribution over K outputs

$$p(y = k|x) \propto \exp\left(\left(\langle a_k, x\rangle - b_k\right)\right), \quad \text{where } b_k \in \mathbb{R},\ x, a_k \in \mathbb{R}^n$$

i.e. single layer neural network with output dimension K and softmax non-linearity:

$p(Y|\, x) = \text{softmax}((a_1, \dots, a_K)^T x - b)$

but hyperbolic modelling is different than hyperbolic feed-forward layer

Hyperbolic Neural Networks Ganea, ..., Hofmann, NeurIPS'18

Define hyperplane and rewrite linear map as signed distance to hyperplane:

$$H_{a,b} = \{x \in \mathbb{R}^n : \langle a, x \rangle - b = 0\}$$

$$p(y = k|x) \propto \exp(\mathbf{sign}(\langle a_k, x \rangle - b_k)\|a_k\| d(x, H_{a_k, b_k})), \; b_k \in \mathbb{R}, x, a_k \in \mathbb{R}^n.$$

Choose point p_k on hyperplane and rewrite again:

$$p(y = k|x) \propto \exp(\mathbf{sign}(\langle -p_k + x, a_k \rangle)\|a_k\| d(x, \tilde{H}_{a_k, p_k})), \; \text{with } p_k, x, a_k \in \mathbb{R}^n$$

Hyperbolic Neural Networks Ganea, ..., Hofmann, NeurIPS'18

This definition carries over to hyperbolic space

$$\tilde{H}^c_{a,p} := \{x \in \mathbb{D}^n_c : \langle \log^c_p(x), a \rangle_p = 0\} = \exp^c_p(\{a\}^\perp)$$

$$p(y=k|x) \propto \exp(\textbf{sign}(\langle \log^c_p(x), a_k \rangle_p)\sqrt{g^c_{p_k}(a_k, a_k)}\, d_c(x, \tilde{H}^c_{a_k,p_k})), \quad \forall x \in \mathbb{D}^n_c,$$

Input x and parameter p_k in D^n but parameters a_k in $T_pD^n = R^n$ and output in R.

→ Gradients wrt a_k are Euclidean,
Gradients wrt x, p_k are Riemannian.

Hyperbolic Neural Networks Ganea, …, Hofmann, NeurIPS'18

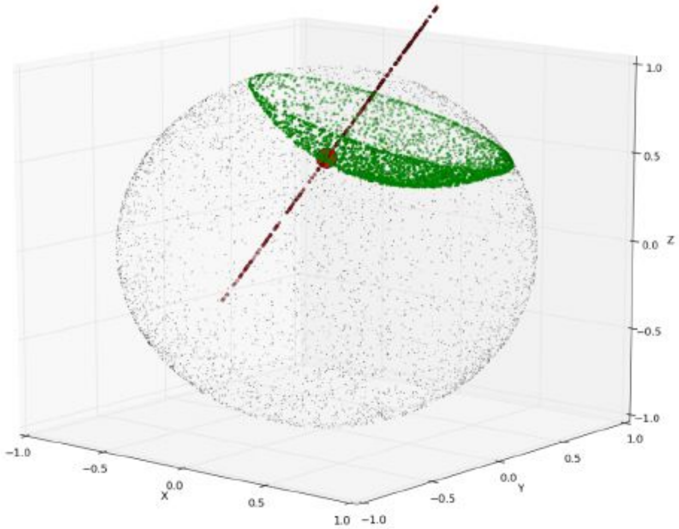


Figure 1: *An example of a hyperbolic hyperplane in* $\mathbb{D}_1^3$ *plotted using sampling. The red point is* p*. The shown normal axis to the hyperplane through* p *is parallel to* a*.*

Hyperbolic Neural Networks++ Shimizu, ..., Harada 2020

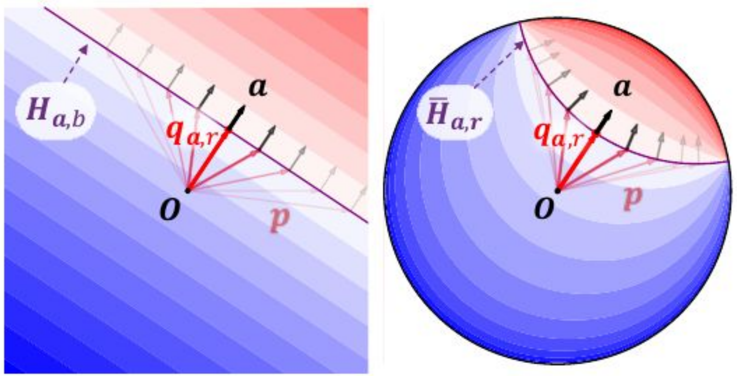
Reformulate several parts of the Hyperbolic Neural Network, in particular unify fully connected layer with multinomial logistic regression

Hyperbolic Neural Networks++ Shimizu, ..., Harada 2020

Old MLR (Ganea et al):

$$p(y=k|x) \propto \exp(\mathbf{sign}(\langle \log_p^c(x), a\rangle_p)\sqrt{g_{p_k}^c(a_k, a_k)}d_c(x, \tilde{H}_{a_k,p_k}^c)), \quad \forall x \in \mathbb{D}_c^n,$$

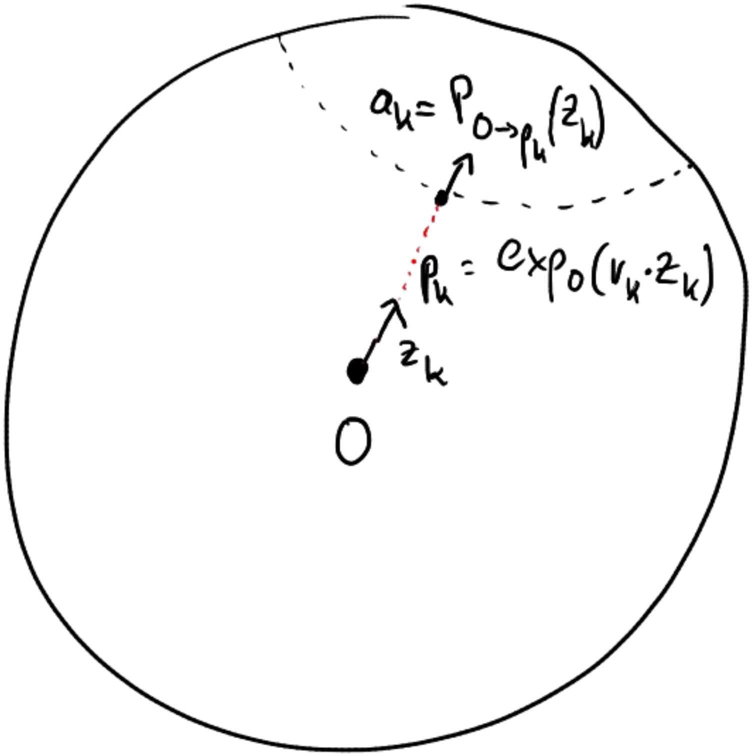
is overparametrized



(a) Reformulation in $\mathbb{R}^n$ (b) Generalization to $\mathbb{B}_c^n$

Figure 2: Whichever pair of $\boldsymbol{a}$ and $\boldsymbol{p}$ is chosen, a determined discriminative hyperplane is the same. Considering one bias point $\boldsymbol{q}_{\boldsymbol{a},r}$ per one discriminative hyperplane solves this over-parameterization.

Hyperbolic Neural Networks++ Shimizu, ..., Harada 2020



Hyperbolic Neural Networks++ Shimizu, …, Harada 2020

New MLR:

Geodesic from origin to hyperplane is orthogonal.
→ use direction of this geodesic and scalar to define the point
→ parallel transport direction from origin to tangent space of point on hyperplane

Instead of p_k in D^n and a_k in $T_{pk}D^n = R^n$, only z_k in $T_0D^n = R^n$ and r_k in R

Hyperbolic Neural Networks++ Shimizu, ..., Harada 2020

Fully connected layer in Euclidean space is stack of translated scalar products

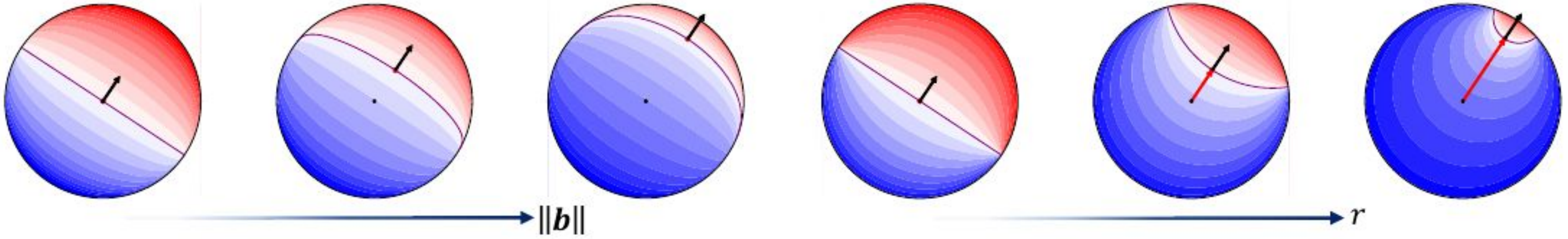
$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}$$
$$y_k = \langle \boldsymbol{a}_k, \boldsymbol{x} \rangle - b_k$$

In MLR, we would do $v_k(\boldsymbol{x}) = \mathrm{sign}(\langle \boldsymbol{a}_k, \ominus_c \boldsymbol{q}_{\boldsymbol{a}_k, r_k} \oplus_c \boldsymbol{x} \rangle)\, d_c\left(\boldsymbol{x}, \bar{H}^c_{\boldsymbol{a}_k, r_k}\right) \|\boldsymbol{a}_k\|^c_{\boldsymbol{q}_{\boldsymbol{a}_k, r_k}}$

But in FC layer Ganea et al do $\boldsymbol{y} = \exp^c_{\mathbf{0}}(\boldsymbol{A} \log^c_{\mathbf{0}}(\boldsymbol{x})) \oplus_c \boldsymbol{b}$

→ use hyperplane method everywhere to get scores and map back to hyperbolic space by using them as distance to axis orthogonal hyperplanes at the origin.

Hyperbolic Neural Networks++ Shimizu, ..., Harada 2020



(a) $\exp_{\mathbf{0}}^c\left(\boldsymbol{A}\,\log_{\mathbf{0}}^c(\boldsymbol{x})\right) \oplus_c \boldsymbol{b}$ (b) $\mathcal{F}^c(\boldsymbol{x}; \boldsymbol{Z}, \boldsymbol{r})$ (ours)

Figure 3: Comparison of FC layers in input spaces $\mathbb{B}_c^n$. The values at a certain dimension of output spaces are illustrated as contour plots. Black arrows depict the orientation parameters, and they are fixed for the comparison. Their orthogonal curves show discriminative hyperplanes where the values are zeros. As a bias parameter $\boldsymbol{b}$ or r_k changes, the outline of the contour landscape in **(a)** remains unchanged, whereas in **(b)** the focused regions are dynamically squeezed according to the geodesics.

Application of Hyperbolic Neural Networks

Chami, …, Leskovec NeurIPS'19

HNN + Graph Convolution Network + Attention + trainable curvature

link prediction and node classification in various transductive / inductive graph settings

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

Table 1: ROC AUC for Link Prediction (LP) and F1 score for Node Classification (NC) tasks. For inductive datasets, we only evaluate inductive methods since shallow methods cannot generalize to unseen nodes/graphs. We report graph hyperbolicity values δ (lower is more hyperbolic).

Application of Hyperbolic Neural Networks

Chami, ..., Leskovec NeurIPS'19

1. **Citation networks**. CORA [36] and PUBMED [27] are standard benchmarks describing citation networks where nodes represent scientific papers, edges are citations between them, and node labels are academic (sub)areas. CORA contains 2,708 machine learning papers divided into 7 classes while PUBMED has 19,717 publications in the area of medicine grouped in 3 classes.
2. **Disease propagation tree**. We simulate the SIR disease spreading model [2], where the label of a node is whether the node was infected or not. Based on the model, we build tree networks, where node features indicate the susceptibility to the disease. We build transductive and inductive variants of this dataset, namely DISEASE and DISEASE-M (which contains multiple tree components).
3. **Protein-protein interactions (PPI) networks**. PPI is a dataset of human PPI networks [37]. Each human tissue has a PPI network, and the dataset is a union of PPI networks for human tissues. Each protein has a label indicating the stem cell growth rate after 19 days [40], which we use for the node classification task. The 16-dimensional feature for each node represents the RNA expression levels of the corresponding proteins, and we perform log transform on the features.
4. **Flight networks**. AIRPORT is a transductive dataset where nodes represent airports and edges represent the airline routes as from OpenFlights.org. Compared to previous compilations [49], our dataset has larger size (2,236 nodes). We also augment the graph with geographic information (longitude, latitude and altitude), and GDP of the country where the airport belongs to. We use the population of the country where the airport belongs to as the label for node classification.

Application of Hyperbolic Neural Networks

Chami, …, Leskovec NeurIPS’19

Name	Nodes	Edges	Classes	Node features
CORA	2708	5429	7	1433
PUBMED	19717	88651	3	500
HUMAN PPI	17598	5429	4	17
AIRPORT	3188	18631	4	4
DISEASE	1044	1043	2	1000
DISEASE-M	43193	43102	2	1000

Table 3: Benchmarks’ statistics

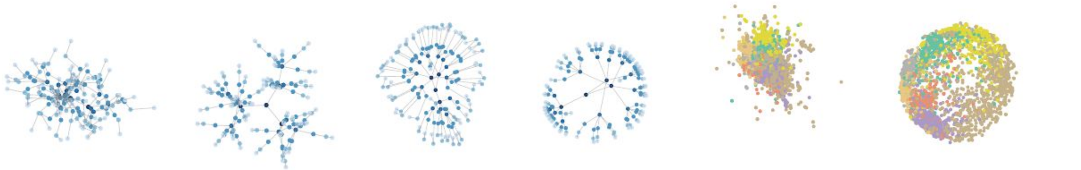
Citation network features: Word frequencies

Airport

Application of Hyperbolic Neural Networks

Chami, ..., Leskovec NeurIPS'19

HNN + Graph Convolution Network (Chami, ..., Leskovec NeurIPS'19)



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

Figure 3: Visualization of embeddings for LP on DISEASE and NC on CORA (visualization on the Poincaré disk for HGCN). (a) GCN embeddings in first and last layers for DISEASE LP hardly capture hierarchy (depth indicated by color). (b) In contrast, HGCN preserves node hierarchies. (c) On CORA NC, HGCN leads to better class separation (indicated by different colors).

Not enough time …

More hyperbolic neural networks

Hyperbolic Graph Neural Networks; Liu, Nickel, Kiela NeurIPS'19

Hyperbolic Graph Convolutional Neural Networks; Chami, …, Leskovec NeurIPS'19

Hyperbolic Attention Networks; Gulcehre, …, Pascanu, de Freitas ICLR'19

Hyperbolic Autoencoder

Mixed-Curvature Variational Autoencoders; Skopek, Ganea, Bécigneul ICLR'20

Adversarial Autoencoders with Constant-Curvature Latent Manifolds; Grattarola, Livi, Alippi '20

A Wrapped Normal Distribution on Hyperbolic Space for Gradient-Based Learning; Nagano, ... , Koyama ICML'19

Continuous Hierarchical Representations with Poincaré Variational Auto-Encoders; Mathieu, …, Teh NeurIPS'19

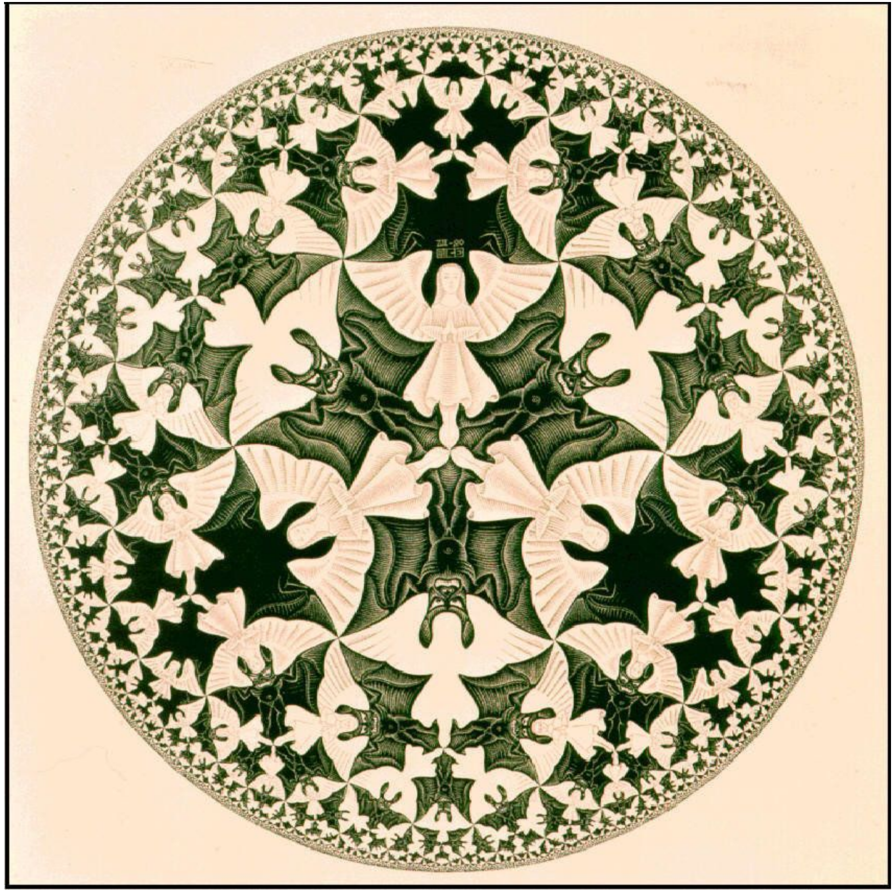
Learning Hierarchies

Hyperbolic Entailment Cones for Learning Hierarchical Embeddings; Ganea, …, Hofmann ICML'18

Hyperbolic Disk Embeddings for Directed Acyclic Graphs; Suzuki, …, Onoda ICML'18

Hierarchical Image Classification Using Entailment Cone Embeddings; Dhall, …, Krause CVPR'20

Learning Continuous Hierarchies in the Lorentz Model of Hyperbolic Geometry; Nickel, Kiela ICML'18



https://www.researchgate.net/figure/The-work-Circle-Limit-IV-Heaven-and-Hell-by-MC-Escher-dated-1960-exemplifies-the_fig1_332186402