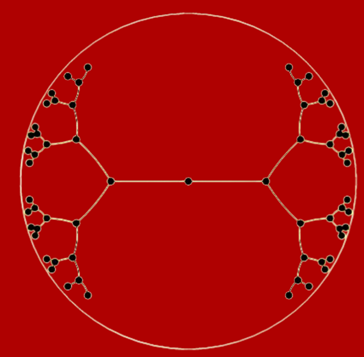


Tradeoffs Between Embeddings in Different Models of the Hyperbolic Space

Eric Lou — erlou@stanford.edu
Shawn Zhang — szhang22@stanford.edu
Ishan Gaur — ishang@stanford.edu

Stanford

INTRODUCTION

Motivation:
Hyperbolic embeddings have achieved recent success in capturing hierarchical information (e.g. WordNet). However, such optimizations that come from hyperbolic space are complex, and there is a lack of solid theoretical framework for understanding the tradeoffs that come from employing different hyperbolic models. We wish to elucidate the tradeoffs between such models, both in theory and in practice.

- This project tests with 4 hyperbolic models: **hyperboloid**, **Poincaré disk**, **half-plane**, and **Beltrami-Klein**.
- We experiment with various graphs consisting of various trees, densities, social networks, and cycles.
- We evaluate performance from the following metrics: **running time, loss, distortion**, and **MAP**.

GRAPHS

We synthesized 8 knowledge graphs with NetworkX by choosing parameters for tree radius, tree height, and/or graph density. These graphs fall into 4 broad categories.

Balanced Trees:

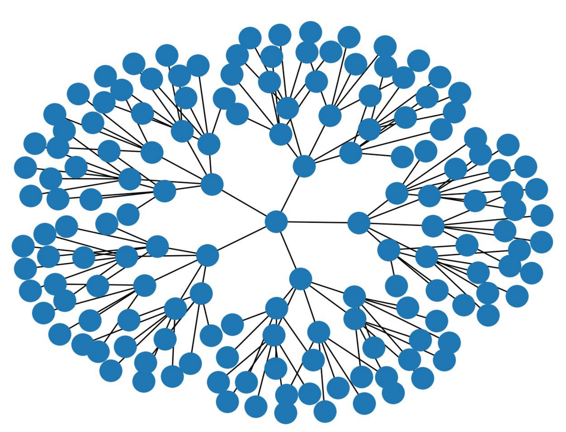


Figure 1a: Tree with radius 3, height 5.

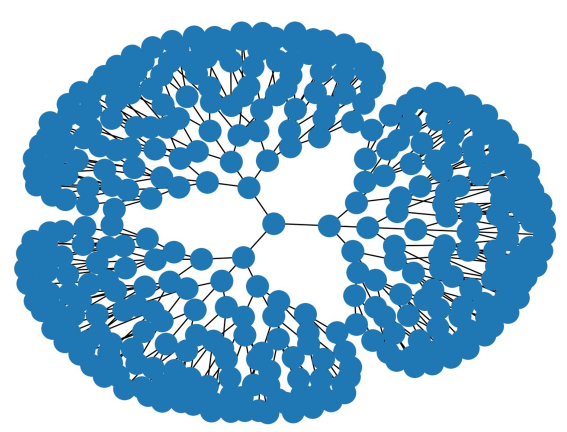


Figure 1b: Tree with radius 5, height 3.

Density Graphs:

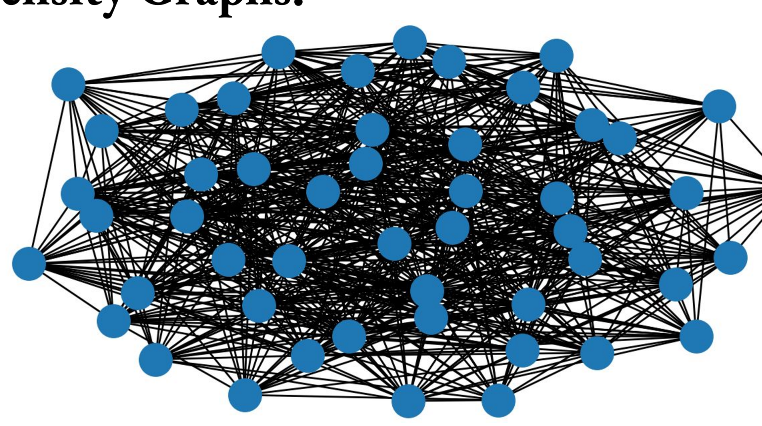


Figure 2: Erdos-Renyl graphs of 400 nodes with densities 0.25, 0.5, 0.75, and 1.

Note: The figure only displays 50 nodes with density 0.5 for clarity.

Social Network Graph:

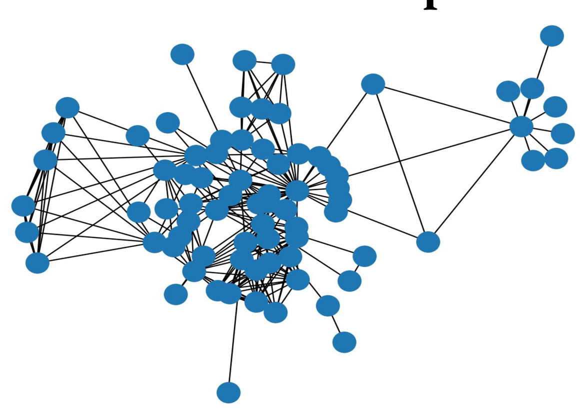


Figure 3: Les Misérables coappearances.

Sierpinski Graph:

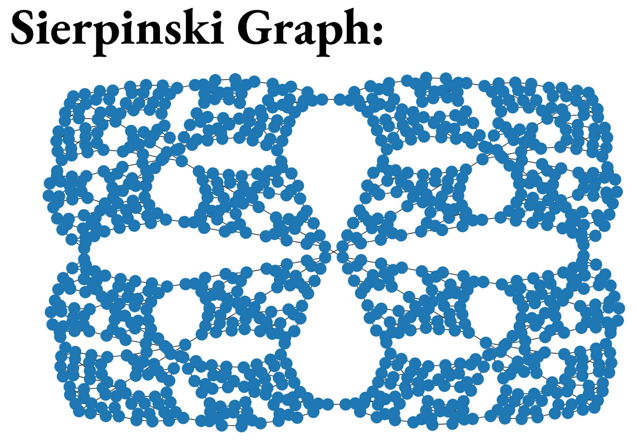


Figure 4: Sierpiński Triangle fractal.

MODELS & METHODS

Models:
where K = Klein, L = Hyperboloid, H = Half-plane, I = Poincaré.

Domains

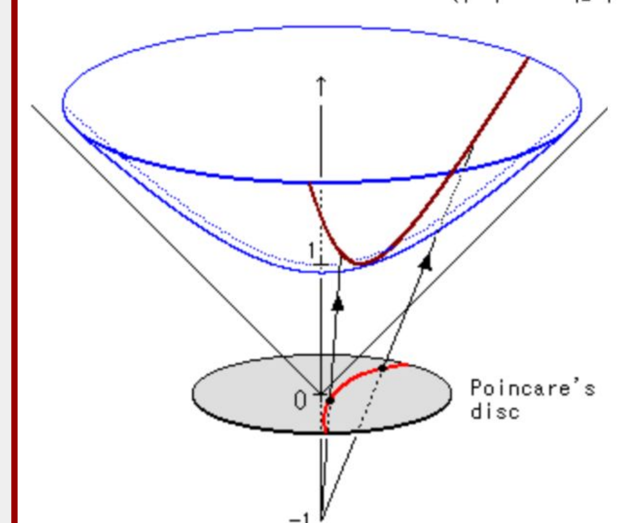
$$K = \{(x_1, \dots, x_n, 1) : x_1^2 + \cdots + x_n^2 < 1\}$$
$$L = \{(x_1, \dots, x_n, x_{n+1}) : x_1^2 + \cdots + x_n^2 - x_{n+1}^2 = -1 \text{ and } x_{n+1} > 0\}$$
$$H = \{(1, x_2, \dots, x_{n+1}) : x_{n+1} > 0\}$$
$$I = \{(x_1, \dots, x_n, 0) : x_1^2 + \cdots + x_n^2 < 1\}$$

Riemannian Metrics

$$ds_K^2 = \frac{dx_1^2 + \cdots + dx_n^2}{(1 - x_1^2 - \cdots - x_n^2)} + \frac{(x_1\,dx_1 + \cdots + x_n\,dx_n)^2}{(1 - x_1^2 - \cdots - x_n^2)^2}$$
$$ds_L^2 = dx_1^2 + \cdots + dx_n^2 - dx_{n+1}^2$$
$$ds_H^2 = \frac{dx_2^2 + \cdots + dx_{n+1}^2}{x_{n+1}^2} \qquad ds_I^2 = 4\frac{dx_1^2 + \cdots + dx_n^2}{(1 - x_1^2 - \cdots - x_n^2)^2}$$

Distance Functions

$$d_K(p,q) = \frac{1}{2}\log\frac{|aq||pb|}{|ap||qb|} \qquad d_L(p,q) = \operatorname{arcosh}(p_1q_1 - p_2q_2 - \dots - p_nq_n)$$
$$d_H(p,q) = \operatorname{arcosh}\left(1 + \frac{(q_1-p_1)^2 + (q_2-p_2)^2 + \dots + (q_n-p_n)^2}{2p_nq_n}\right)$$
$$d_I(p,q) = \operatorname{arcosh}\left(1 + \frac{2|pq|^2|r|^2}{(|r|^2 - |p|^2)(|r|^2 - |q|^2)}\right)$$



From the above equations, we mapped from Euclidean to hyperbolic space. In implementation, we also mapped hyperbolic models between each other.

Figure 5: Demonstrating projection from hyperboloid to Poincaré.

Evaluation Metrics:

Mean Average Precision (MAP)

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

Distortion

$$D(f) = \frac{1}{\binom{n}{2}}\left(\sum_{u,v \in U: u \neq v}\frac{|d_V(f(u), f(v)) - d_U(u,v)|}{d_U(u,v)}\right)$$

where $f: U \rightarrow V$ with distances d_U, d_V, $a \in V$ with $N_a = \{b_1, b_2, \dots b_{deg(a)}\}$, and $R_{a,bi}$ is the smallest set of nearest points for b_i of $f(a)$.

Loss

$$\mathcal{L}(x) = \sum_{1 \leq i \leq j \leq n}\left|\left(\frac{d_P(x_i, x_j)}{d_G(X_i, X_j)}\right)^2 - 1\right|$$

with graph G and Riemannian manifold P.

RESULTS

Balanced Tree: R=3, H=5						
Model	Dim	Learning Rt	Loss	MAP	Distortion	WC
Hyperboloid	10	5	0.12707	0.8811	0.2925	9.57
Poincare	10	5	0.12707	0.8811	0.2925	9.57
Halfplane	**10**	**5**	**0.109614**	**0.8967**	**0.2754**	**9.3912**
Klein	10	0.1	0.310093	0.7703	0.5353	11.8158
Balanced Tree: R=5, H=3						
Model	Dim	Learning Rt	Loss	MAP	Distortion	WC
Hyperboloid	10	10	0.004236	0.5888	0.0426	1.5709
Poincare	10	10	0.004236	0.5888	0.0426	1.5709
Halfplane	**10**	**5**	**0.004787**	**0.8641**	**0.046**	**1.6317**
Klein	10	0.1	0.143739	0.8915	0.3439	6.371

Figure 6: Performance on Balanced Trees

Random Graph: 400 Nodes, P=0.25						
Model	Dim	Learning Rt	Loss	MAP	Distortion	WC
Hyperboloid	**5**	**5**	**0.003007**	**0.2579**	**0.0259**	**2.2029**
Poincare	**5**	**5**	**0.003007**	**0.2579**	**0.0259**	**2.2029**
Halfplane	10	5	995.505658	0.2632	31.5376	1.2922
Klein	10	0.1	0.000832	0.2475	0.0184	1.4653
Random Graph: 400 Nodes, P=0.50						
Model	Dim	Learning Rt	Loss	MAP	Distortion	WC
Hyperboloid	**10**	**5**	**0**	**0.5001**	**0.0001**	**1.0076**
Poincare	5	5	0.001961	0.5049	0.0163	2.0725
Halfplane	10	5	1003.58405	0.5076	31.6572	1.1727
Klein	10	0.1	0.000472	0.494	0.0095	1.4156
Random Graph: 400 Nodes, P=0.75						
Model	Dim	Learning Rt	Loss	MAP	Distortion	WC
Hyperboloid	**10**	**5**	**0**	**0.7526**	**0**	**1**
Poincare	**10**	**5**	**0**	**0.7526**	**0**	**1**
Halfplane	10	5	1006.02325	0.7546	31.7221	1.1718
Klein	10	0.1	0.000239	0.7491	0.0051	1.4001
Random Graph: 400 Nodes, P=1.00						
Model	Dim	Learning Rt	Loss	MAP	Distortion	WC
Hyperboloid	10	5	0.000341	1	0.0028	1.397
Poincare	10	5	0.000341	1	0.0028	1.397
Halfplane	**10**	**0.1**	**0.000276**	**1**	**0.0013**	**1.4486**
Klein	**10**	**0.1**	**0.000276**	**1**	**0.0013**	**1.4486**

Figure 7: Performance on Erdos-Renyl Random Graphs

Sierpinski: K=4, H=5						
Model	Dim	Learning Rt	Loss	MAP	Distortion	WC
Hyperboloid	10	5	0.308715	0.748	0.4733	38.4042
Poincare	10	5	0.308715	0.748	0.4733	38.4042
Halfplane	**10**	**5**	**0.306268**	**0.764**	**0.4684**	**38.453**
Klein	10	0.1	0.583595	0.1828	0.6895	53.2123
Social Graph: Les Miserables						
Model	Dim	Learning Rt	Loss	MAP	Distortion	WC
Hyperboloid	5	5	0.013329	0.8202	0.0869	2.1958
Poincare	5	5	0.013329	0.8202	0.0869	2.1958
Halfplane	**10**	**5**	**0.009316**	**0.9176**	**0.0688**	**2.214**
Klein	10	0.1	0.028229	0.9274	0.1339	2.8731

Figure 8: Performance on Fractal and Social Graph

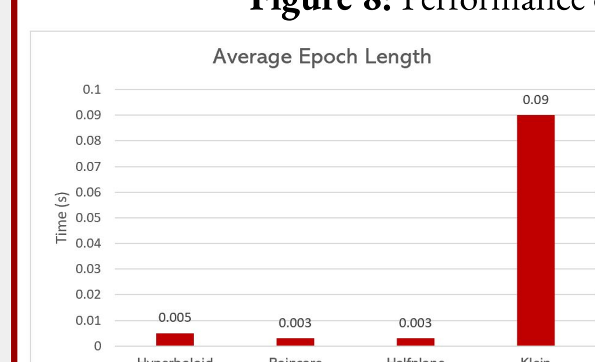


Figure 8: Average Epoch Length

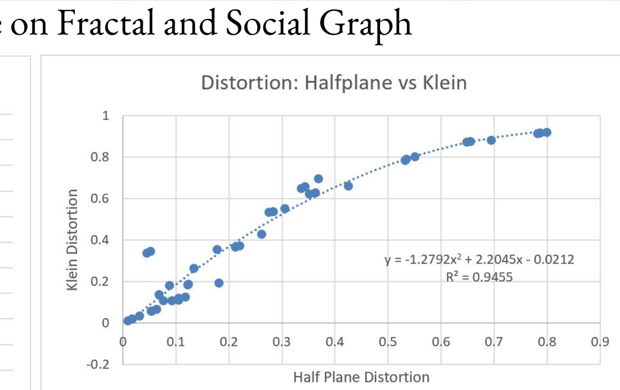


Figure 9: Half-plane vs. Klein Distortion

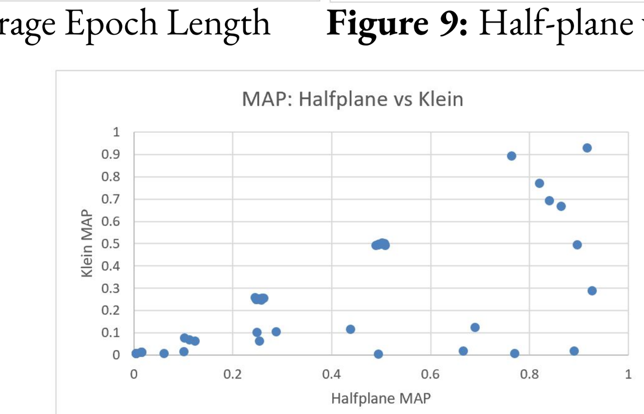


Figure 10: Half-plane vs. Klein MAP

DISCUSSION

Embeddings for the balanced trees were great as expected
- Social graphs, and last two random graphs also performed well
- Great MAP for social graphs means good neighbors preservation
- Big factor is number of nodes--natural facet of certain graphs

Halfplane and Klein's performances were correlated across runs
- MAP results imply neighbors are preserved better in Halfplane
- Seems like MAP of Halfplane is an upperbound for Klein
- Indicates further hyperparameters tuning might help

Discrepancy in MAP and distortion of Halfplane for the Erdos-Renyl graphs
- Might be because uniform probability for each edge pushes MAP to that probability
- The first n randomly selected nodes are in the neighbors with probability p.
- Low distortion for the first three graphs might be overfitting, which would have to tested with different inference tasks

Higher dimensions tended to be better
- Embedding has more space, but higher learning rates fail faster
- Gradient descent becomes unstable because the Euclidian norm of the update stays the same mod dimensionality, but the compression of the space at the edges is increasingly larger so errors from the approximations due to retraction are also larger

Klein takes much longer
- Expect the gradient update to be O(n^2) instead of O(n), where n is the dimension, since more complex metric forces matrix multiplication for the exponential map
- Data shows this: runtimes are off by a factor of 10=100/10

FUTURE WORK

- More fine-grained hyperparameter search to start some predictive effort for learning rate determination
- Analyze computational faults further to attribute more of them to theoretical features of the models
- Test higher dimensional models with much larger datasets to tease out more accurate capabilities for each one
- Compare with Euclidean and Spherical embeddings

REFERENCES

In collaboration with Frederic Sala and Adva Wolf.

- Cannon, J. W., Floyd, W. J., Kenyon, R., & Parry, W. R. (1997). *Hyperbolic Geometry* (Vol. 31). MSRI Publications.
- Ganea, O. E., Bécigneul, G., & Hofmann, T. (2018). Hyperbolic entailment cones for learning hierarchical embeddings. arXiv preprint arXiv:1804.01882.
- Sala, F., de Sa, C., Gu, A., & Re, C. (2018). Representation Tradeoffs for Hyperbolic Embeddings. *International Conference on Machine Learning*.
- Wilson, B., & Leimeister, M. (2018). Gradient descent in hyperbolic space. arXiv preprint arXiv:1805.08207.