Numerically Accurate Hyperbolic Embeddings Using Tiling-Based Models

Tao Yu & Christopher De Sa

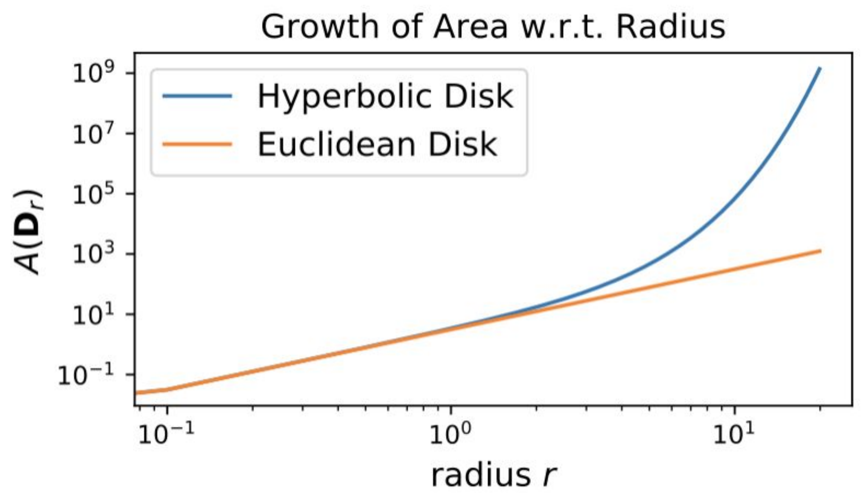
{tyu,cdesa}@cs.cornell.edu

Cornell University Computer Science

Hyperbolic Space

Machine learning has achieved great success by embedding objects into Euclidean space,recent some exciting work [1,2,3,4,5] proposed embeddings in hyperbolic space.

- ➔ Hyperbolic space is a maximally symmetric, simply connected Riemannian manifold with a constant negative sectional curvature.
- ➔ Hyperbolic space contains more space for embeddings: Area (Volume) of a disk (ball) in the space increases exponentially over the radius (polynomially in Euclidean space).



Standard models:

- ❏ Poincare ball model:

$$(\mathcal{B}^n, g_p) : \mathcal{B}^n = \{\boldsymbol{x} \in \mathbb{R}^n : \|\boldsymbol{x}\| < 1\} \ , \ g_p(\boldsymbol{x}) = \left(\frac{2}{1-\|\boldsymbol{x}\|^2}\right)^2 g_e$$

where g_e is the Euclidean metric.

- ❏ Lorentz hyperboloid model:

$$(\mathcal{L}^n, g_l) : \mathcal{L}^n = \{\boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{x}^T g_l \boldsymbol{x} = -1, x_0 > 0\} \ , \ g_l = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

- ❏ Poincare half-space model:

$$(\mathcal{U}^n, g_u) : \mathcal{U}^n = \{\boldsymbol{x} \in \mathbb{R}^n : x_n > 0\} \ , \ g_u(\boldsymbol{x}) = \frac{g_e}{x_n^2}$$

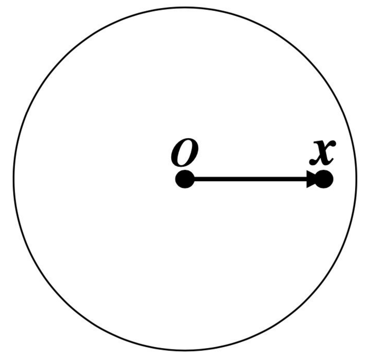
The NaN Problem

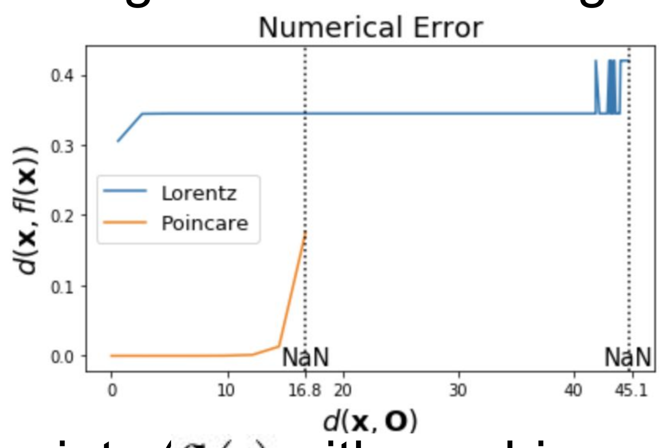
Hyperbolic embeddings are limited by numerical issues when the space is represented by floating-points.

- ❏ Computing the distance produces NaNs as points get far from the origin.

A simple task::

1. Start from the origin
2. Move in a direction for a distance



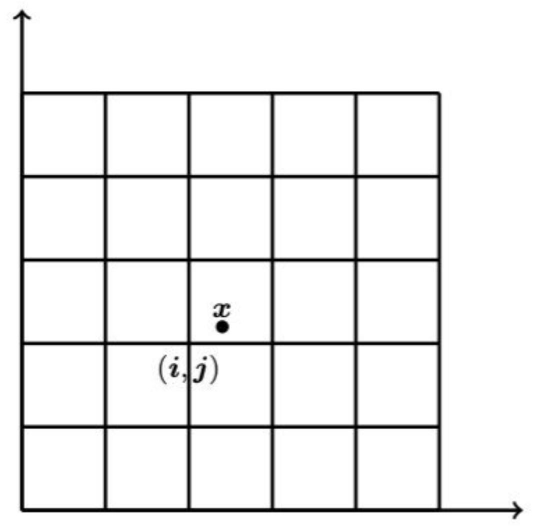


Proved: If the space is represented with floating-points ($\mathrm{fl}(\cdot)$ with machine epsilon ϵ_m) in standard models, the worst case representation error is $d(\boldsymbol{x}, \mathrm{fl}(\boldsymbol{x})) = \Omega(\epsilon_m \exp(d(\boldsymbol{x}, \boldsymbol{O})))$ the worst case relative numerical error to compute the distance $d(\boldsymbol{x}, \boldsymbol{y})$ and its gradient is $\Omega(\epsilon_m \exp(d(\boldsymbol{x}, \boldsymbol{O}) + d(\boldsymbol{y}, \boldsymbol{O})))$

An Everywhere-Accurate Solution?

- ❏ A potential solution: BigFloats, floating-points with a large quantity of bits. However:
  - ➢ The numerical issues still happen for points sufficiently far away from the origin.
  - ➢ No amount of bits are sufficient to accurately represent points everywhere in hyperbolic space [3].
- ❏ A solution in the Euclidean plane with constant error: using the integer-lattice square tiling, represent a point $\boldsymbol{x}$ in the plane with a tuple
  1. Integer Coordinates $(\boldsymbol{i}, \boldsymbol{j})$ of the square where $\boldsymbol{x}$ is in;
  2. Offsets of $\boldsymbol{x}$ within the square as floating-points.

  Proved: Numerical error to represent $\boldsymbol{x}$ and the relative numerical error to compute distance and its gradient is $O(\epsilon_m)$.



- ❏ Do the same thing in the hyperbolic space: construct a tiling and represent $\boldsymbol{x}$ with a tuple:
  1. the tile where $\boldsymbol{x}$ is in;
  2. Offsets of $\boldsymbol{x}$ within that tile as floating-points.

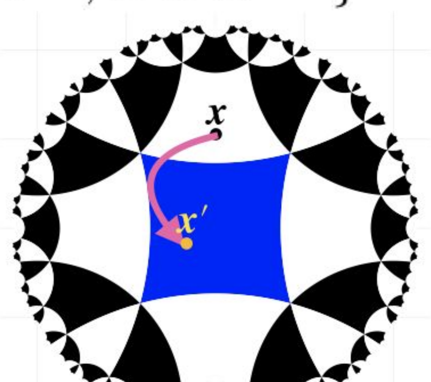
Tiling ⟷ Isometries

How to identify a tile in the tiling of the hyperbolic space? (← Isometries)

← Each tile can be mapped onto the central tile F with a unique isometry g^{-1}

- ❏ Isometries of the 2-dimensional Lorentz model: $g \in \mathbb{R}^{3\times 3}$ *s.t.* $g^T g_l g = g_l$.
  - ❏ Construct a subgroup G of the set of isometries: $G = \{g \mid g = LZL^{-1}, Z \in \mathbb{Z}^{3\times 3}\}$ where Z is in a group generated by

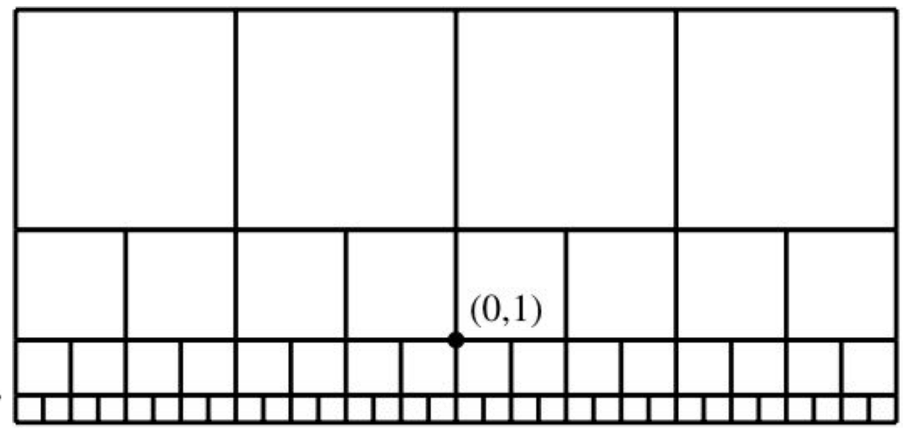
$$g_a = \begin{bmatrix} 2 & 1 & 0 \\ 0 & 0 & -1 \\ 3 & 2 & 0 \end{bmatrix}, \ g_b = \begin{bmatrix} 2 & -1 & 0 \\ 0 & 0 & -1 \\ -3 & 2 & 0 \end{bmatrix}, \ \textit{and } L = \begin{bmatrix} \sqrt{3} & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \Longleftrightarrow$$



  - ❏ Represent $\boldsymbol{x}$ with a tuple $(Z, \boldsymbol{x}')$ (L-tiling model):
    - ❏ Exact integer matrix Z
    - ❏ $\boldsymbol{x}' \in F : \boldsymbol{x}'^T g_l \boldsymbol{x}' = -1$, where $\boldsymbol{x}'$ is in floating-points and F is bounded.
- ❏ Higher dimensions:

  Problem: Deriving a tiling induced by a subgroup is impossible in higher dimensions!
  - ❏ Construct an ensemble of isometries in the Poincare half-space model: ⟷

$$\{f \mid f(p) = 2^j(p+k), (j,k) \in \mathbb{Z} \times (\mathbb{Z}^{n-1} \times \{0\})\}.$$

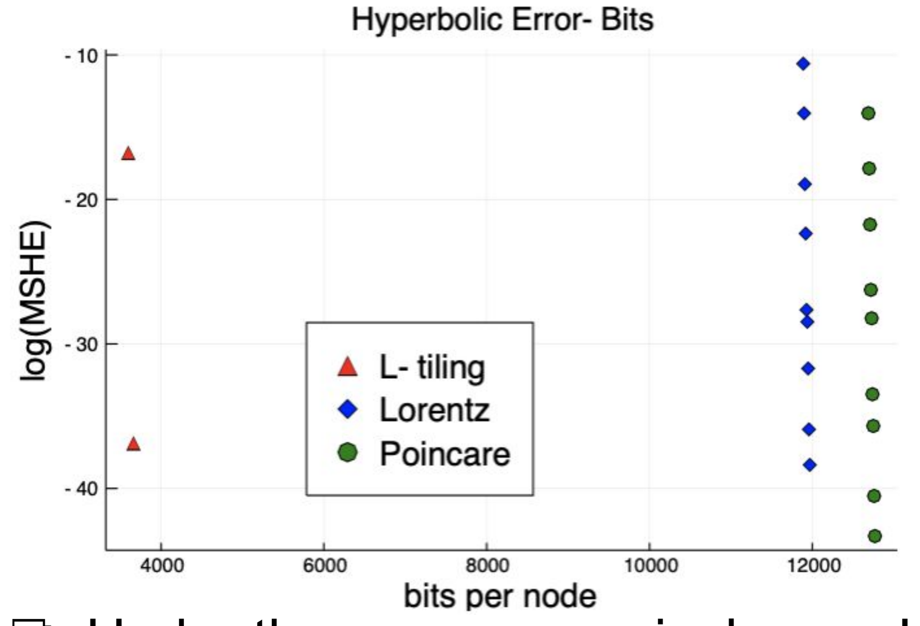


Guarantees: Numerical error to represent $\boldsymbol{x}$ and the relative numerical error to compute distance and its gradient is $O(\epsilon_m)$.

Applications:

- ❏ Compression:

  Represent hyperbolic embeddings in tiling-based models with way fewer bits than standard models using BigFloat on the WordNet dataset.



Models	size (MB)	bzip (MB)
Poincaré	372	119
Poincaré	287	81
Lorentz	396	171
L-Tiling	37.35	7.13

  - ❏ Under the same numerical error, L-tiling model uses 2/3 less bits to store per node compared to that of Lorentz and Poincare models using BigFloat.
  - ❏ L-tiling model can accurately represent an embedding to 2% (7.13 MB) of its original size (372 MB), while at least 81 MB is required for any accurate baseline model using BigFloat.
- ❏ Learning:

  Compute efficiently using integers in tiling-based models and learn high-precision embeddings without using BigFloats.
  - ❏ On the largest WordNet-Nouns dataset, tiling-based model outperforms previous standard floating-points implementations.
  - ❏ Numerical issue happens in standard models when the embeddings are far from the origin and affects the embedding performances.

Dimension	Models	MAP	MR
2	Poincaré	0.124±0.001	68.75±0.26
	Lorentz	0.382±0.004	17.80±0.55
	Tiling	**0.413**±0.007	**15.26**±0.57
5	Poincaré	0.848±0.001	4.16±0.04
	Lorentz	0.865±0.005	**3.70**±0.12
	Tiling	**0.869**±0.001	**3.70**±0.06
10	Poincaré	0.876±0.001	3.47±0.02
	Lorentz	0.865±0.004	3.36±0.04
	Tiling	**0.888**±0.004	**3.22**±0.02

References:

[1] M. Nickel et al. Poincaré embeddings for learning hierarchical representations. NeurIPS 2017.
[2] M. Nickel et al. Learning Continuous Hierarchies in the Lorentz Model of Hyperbolic Geometry. ICML 2018.
[3] C. De Sa et al. representation tradeoffs for hyperbolic embeddings, ICML 2018.
[4] B. Chamberlain et al. Neural embeddings of graphs in hyperbolic space. KDD workshop, 2017.
[5] A. Gu et al. Learning mixed-curvature representations in product spaces. ICLR 2019.