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: \( \mathbb{B}^n_r \), \( \mathbb{B}^n = \{ x \in \mathbb{R}^n : \| x \| < 1 \} \), \( g_B(x) = \left( \frac{2}{1 - \| x \|^2} \right) x \), where \( g_B \) is the Euclidean metric.
- Lorentz hyperboloid model: \( (C^n, g_C) = \{ x \in \mathbb{R}^n : x^t g_C x = -1, x_n > 0 \} \), \( g_C = \frac{1}{x_n} \delta_{II} + \frac{x_n}{x_n^2} g_{C,0} \).
- Poincare half-space model: \( \mathbb{H}^n \), \( \mathbb{H}^n = \{ x \in \mathbb{R}^n : x_n > 0 \} \), \( g_H(x) = \frac{x^2}{x_n^2} \).

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.
- The NaN Problem happens in standard models when the embeddings are far from the origin and affects the embedding performances.

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 \( \mathcal{X} \) in the plane with a tuple: 
  1. Integer Coordinates \((i, j)\) of the square where \( \mathcal{X} \) is in;  
  2. Offsets of \( \mathcal{X} \) within the square as floating-points. 

- Do the same thing in the hyperbolic space: construct a tiling and represent \( \mathcal{X} \) with a tuple: 
  1. The tile where \( \mathcal{X} \) is in;  
  2. Offsets of \( \mathcal{X} \) 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: \( g \in \mathbb{R}^{2x2} \), \( g' g g^{-1} = g \).
- Construct a subgroup \( G \) of the set of isometries: \( G = \{ g | g = L^{-1} Z L, Z \in \mathbb{Z}^{2x2} \} \), where \( Z \) is in a group generated by \( g_2 = \begin{pmatrix} 2 & 1 \\ 1 & 0 \end{pmatrix}, g_0 = \begin{pmatrix} 2 & -1 \\ 1 & 0 \end{pmatrix} \) and \( L = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \).
- Represent \( \mathcal{X} \) with a tuple \((Z, x^t)\) (L-tiling model): 
  - Exact integer matrix \( Z \)  
  - \( x^t \in F: x^t g \mathcal{X} = -1 \). Where \( g \mathcal{X} \) is in floating-points and \( F \) is unbounded.
- 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: 
  - \((f, p) = (p(g + k), (j,k) \in Z \times \{ \pm 1 \})\) 
  - Guarantees: Numerical error to represent \( \mathcal{X} \) and the relative numerical error to compute distance and its gradient is \( O(\epsilon_{tile}) \).

Applications: 

- Compression: Represent hyperbolic embeddings in tiling-based models with way fewer bits than standard models using BigFloat on the WordNet dataset.
- 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 occurs happens in standard models when the embeddings are far from the origin and affects the embedding performances.

References: 