Fred sala

Hyperbolic Embedding

→ discrete & structured

(·While much of ML works continuous & unstructur data

→ structural information + $\underbrace{\text{continuous represent}^n}_{\text{suitable for ML}}$

* embedding a space into another

↦ preserve distance & more complex relationships

↦ hyperbolic graph ← can better embed graphs/trees:

angle in hyperbolic world → euclidean space.

Exact recovery problem :- Given only dist info.

↓

recover underlying hyperbolic points

idea: embed root at the origin. & recursively embed the children of each node in the tree by spacing them around a sphere centered at the parent.

→ locally, children of each node should be spread out on the sphere around the paren

[illegible] separated from each othe

→ Globally, subtrees should be separated from each o

hyperbolic space = separation is determined by sphere radius;

<u>quality of embedding</u> :-

local measure → whether neighbors remain closest to each other. (MAP)

global measure → explicit values.

(<u>distortion</u>) $\sum \frac{d(x,y) - d_H(x,y)}{d(x,y)}$

Manifold, M → object in $\mathbb{R}^n$, but only locally.

↳ union of collectⁿ of <u>charts</u>.

bijective mappings b/n open subsets of M & open subsets of $\mathbb{R}^n$

Core Motivation to use hyperbolic space for embedding tree nodes → "mimic discrete trees"

* → Can reproduce an <u>arbitrarily good approxⁿ to tree distance</u>, while still being in continuous space

* distance b/n siblings = 2 → hyperbolic ✓
Euclidean ✗

arbitrarily low distortion, just with 2 dim

* trade-off b/n – embedding fidelity
– properties of the tree (maximal degree & depth –
– embedding dimensions

Vs

Precision.

Numerical issues → for long chains
Hyperbolic ~~good~~ for short / bushy trees
struggles with trees with long paths.

<u>product manifold</u> :-

Training an embedding in the product space is no harder than training an embedding for one space.

(decomposability of spaces → sph., eucl, hyperbolic

☆ Challenge → Generalizing message passing algorithms to non-Euclidean geometry is a challenge.

GNNs not only embed structure, but also preserve semantic information that might come with the graph in the form of node features. Most GNNs models are based on a message passing algorithm, where nodes aggregate information from their neighbors at each layer in the GNN network. Generalizing message passing algorithms to non-Euclidean geometry is a challenge: we do so is by using the tangent space. Recent work leverages gyrovector theory to define useful operations in ML such as addition ⊕ or matrix-vector multiplication ⊗. These operations are applied in the Euclidean tangent space at the origin, and mappings from and to this space are done via the exp and log maps. In HGCN, we use these techniques to perform message passing in hyperbolic space in a two step-process:

* $h_i = W \otimes x_i \oplus b$ <u>feature transform^n</u>

* $\exp_{x_i}\left(\sum_{j \in N(i)} w_{ij} \log_{x_i}(x_j) \right)$ <u>Aggregation.</u>

capture nodes' hierarchies

Linear transf^n → most diff op^n in manifolds

* Hyperbolic Graph ~~CNN~~ (HGCN)

embedding real-world graphs — less distortion ✓

→ feature transformation / aggregation [Not easy].

expressiveness of GCN

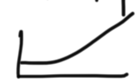
⊕

hyperbolic geometry to learn inductive node represent^n for <u>hierarchial</u> & <u>scale-free</u> graphs

GCN operations → in the hyperboloid model.

Euclidean features → embedding in hyperbolic spaces (with diff trainable curvature at each layer).

# of nodes within some distance of center node → exp



Volume of ball in Euclidean space → polynomial.

"High" distortion tree embedding.

grows exponentially in hyperbolic space

* only accounts for graph structure & don't leverage rich node features

→ learns only shallow embeddings with hyperbolic distance metric

→ doesn't take into a/c features of node
lack scalability & inductive capacity