2021

인공지능세미나

정보찬

AILab.

2021.05.27.

주식회사 바스젠바이오

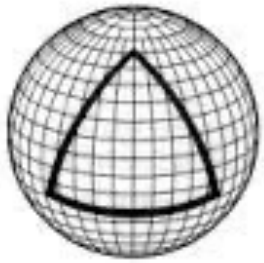
Hyperbolic Image Embeddings

1. Introduction
2. Motivation
3. Methodology
4. Result
5. Discussion

- hyperbolic space란

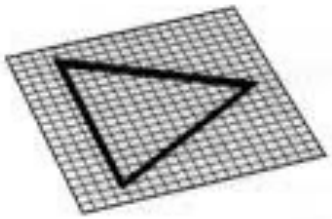
Formally, n-dimensional hyperbolic space denoted as $\mathbb{H}^n$ is defined as the homogeneous, simply connected n-dimensional Riemannian manifold of constant negative sectional curvature.

- 삼각형의 세 내각의 크기의 합이 180도 보다 작은 공간.



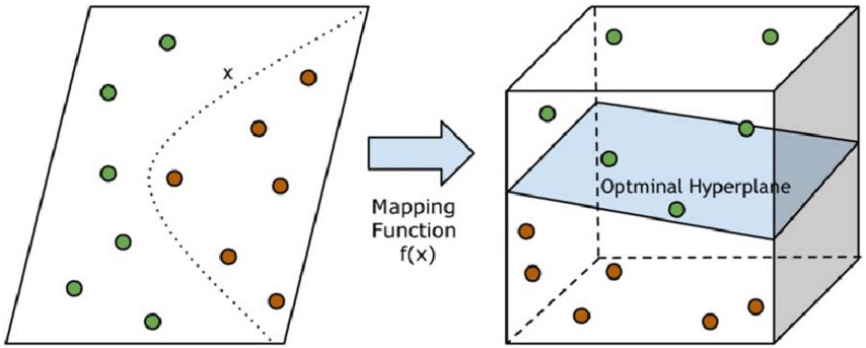
Positive Curvature

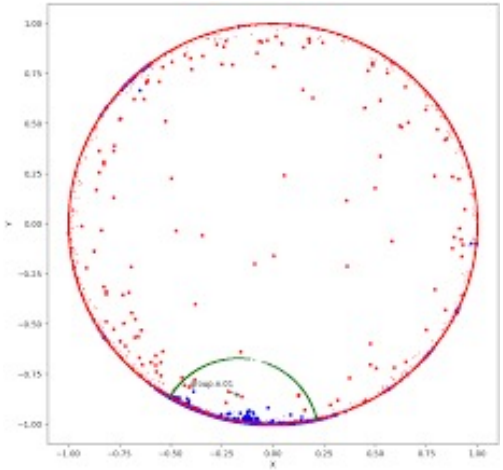
Negative Curvature



Flat Curvature

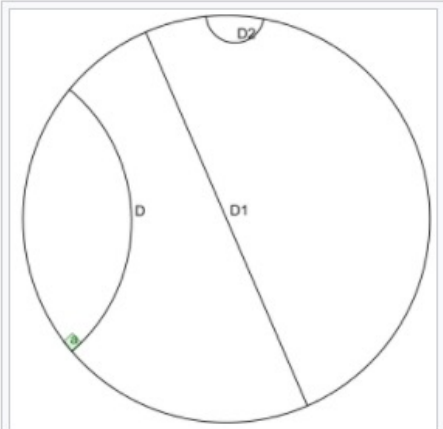
• hyperbolic embedding이란





• data embedding을 할 때 Euclidean space가 아닌 Hyperbolic space로 하는 embedding

- Poincare ball



Poincaré disk with 3 ultraparallel (hyperbolic) straight lines

$$B = \left\{ x \in \mathbb{R}^n : \sum_{i=1}^{n} x_i^2 < 1 \right\}$$

- 단순히 집합으로만 보자면 경계가 없는 disk지만 geodesic과 metric을 Euclidean space와 다르게 줌

- hyperbolic embedding이 필요한 이유

Although embedding methods have proven successful in numerous applications, they suffer from a fundamental limitation: their ability to model complex patterns is inherently bounded by the dimensionality of the embedding space. For instance, Nickel et al. [20] showed that linear embeddings of graphs can require a prohibitively large dimensionality to model certain types of relations. Although non-linear embeddings can mitigate this problem [7], complex graph patterns can still require a computationally infeasible embedding dimensionality. As a consequence, no method yet exists that is

- linear embedding은 복잡한 패턴을 파악하기 위해선 차원이 커져야 하는 한계가 있음.
- 이를 보완하기 위해 hyperbolic embedding이 연구됨.

Möbius addition. For a pair $\mathbf{x}, \mathbf{y} \in \mathbb{D}_c^n$, the Möbius addition is defined as follows:

$$\mathbf{x} \oplus_c \mathbf{y} := \frac{(1 + 2c\langle \mathbf{x}, \mathbf{y} \rangle + c\|\mathbf{y}\|^2)\mathbf{x} + (1 - c\|\mathbf{x}\|^2)\mathbf{y}}{1 + 2c\langle \mathbf{x}, \mathbf{y} \rangle + c^2\|\mathbf{x}\|^2\|\mathbf{y}\|^2}. \quad (6)$$

Exponential and logarithmic maps. To perform operations in the hyperbolic space, one first needs to define a bijective map from $\mathbb{R}^n$ to $\mathbb{D}_c^n$ in order to map Euclidean vectors to the hyperbolic space, and vice versa. The so-called exponential and (inverse to it) logarithmic map serves as such a bijection.

The *exponential* map $\exp_{\mathbf{x}}^c$ is a function from $T_{\mathbf{x}}\mathbb{D}_c^n \cong \mathbb{R}^n$ to $\mathbb{D}_c^n$, which is given by

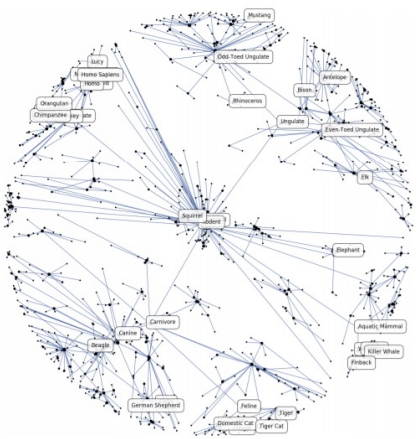
$$\exp_{\mathbf{x}}^c(\mathbf{v}) := \mathbf{x} \oplus_c \left(\tanh\left(\sqrt{c} \frac{\lambda_{\mathbf{x}}^c \|\mathbf{v}\|}{2} \right) \frac{\mathbf{v}}{\sqrt{c}\|\mathbf{v}\|} \right). \quad (8)$$

Numerical stability. While implementing most of the formulas described above is straightforward, we employ some tricks to make the training more stable. In particular, to ensure numerical stability, we perform clipping by norm after applying the exponential map, which constrains the norm not to exceed $\frac{1}{\sqrt{c}}(1 - 10^{-3})$.

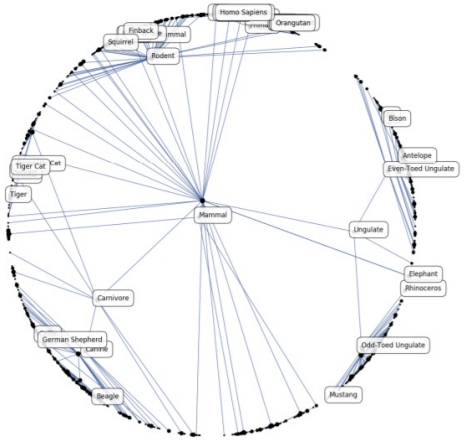
3 Methodology

- main hypothesis

In this subsection, we validate our hypothesis, which claims that if one trains a hyperbolic classifier, then the distance of the Poincaré ball embedding of an image to the origin can serve as a good measure of confidence of a model. We start by training a simple hyperbolic convolutional neural network on the MNIST dataset (we hypothesized that such a simple dataset contains a very basic hierarchy, roughly corresponding to visual ambiguity of images, as demonstrated by a trained network on Figure 1). The output of the last hidden layer was mapped to the Poincaré ball using the exponential map (8) and was followed by the hyperbolic multi-linear regression (MLR) layer [11].



(a) Intermediate embedding after 20 epochs



(b) Embedding after convergence

- Poincare ball 가운데 있을수록 애매한 데이터이고, 경계 쪽에 있을수록 확실한 데이터이다.

- main hypothesis

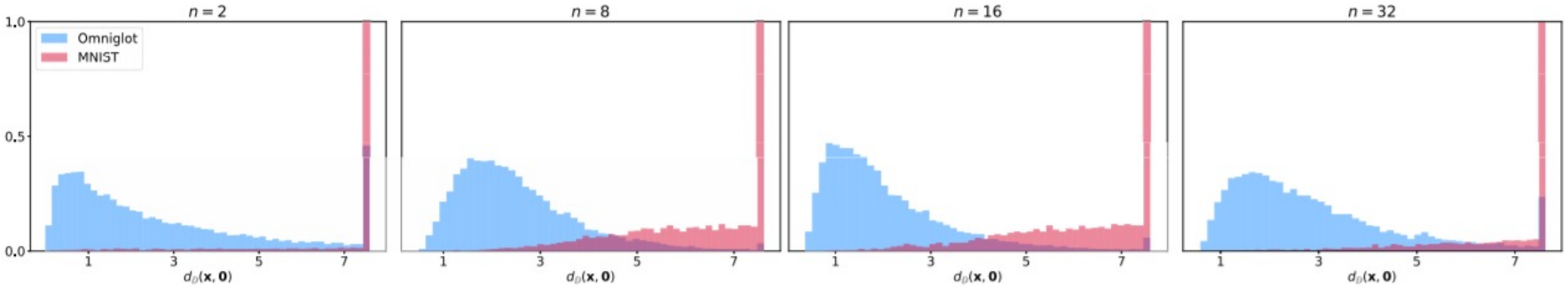


Figure 5: Distributions of the hyperbolic distance to the origin of the MNIST (red) and Omniglot (blue) datasets embedded into the Poincaré ball; parameter n denotes embedding dimension of the model trained for MNIST classification. Most Omniglot instances can be easily identified as out-of-domain based on their distance to the origin.

4 result

- MiniImagenet – few shot learning

Baselines	Embedding Net	1-Shot 5-Way	5-Shot 5-Way
MatchingNet [53]	4 Conv	43.56 ± 0.84%	55.31 ± 0.73%
MAML [9]	4 Conv	48.70 ± 1.84%	63.11 ± 0.92%
RelationNet [48]	4 Conv	50.44 ± 0.82%	65.32 ± 0.70%
REPTILE [28]	4 Conv	49.97 ± 0.32%	65.99 ± 0.58%
ProtoNet [43]	4 Conv	49.42 ± 0.78%	68.20 ± 0.66%
Baseline* [4]	4 Conv	41.08 ± 0.70%	54.50 ± 0.66%
Spot&learn [6]	4 Conv	51.03 ± 0.78%	67.96 ± 0.71%
DN4 [23]	4 Conv	51.24 ± 0.74%	71.02 ± 0.64%
Hyperbolic ProtoNet	4 Conv	**54.43 ± 0.20%**	**72.67 ± 0.15%**
SNAIL [27]	ResNet12	55.71 ± 0.99%	68.88 ± 0.92%
ProtoNet+ [43]	ResNet12	56.50 ± 0.40%	74.2 ± 0.20%
CAML [16]	ResNet12	59.23 ± 0.99%	72.35 ± 0.71%
TPN [25]	ResNet12	59.46%	75.65%
MTL [47]	ResNet12	61.20 ± 1.8%	75.50 ± 0.8%
DN4 [23]	ResNet12	54.37 ± 0.36%	74.44 ± 0.29%
TADAM [32]	ResNet12	58.50%	76.70%
Qiao-WRN [34]	Wide-ResNet28	59.60 ± 0.41%	73.74 ± 0.19%
LEO [38]	Wide-ResNet28	**61.76 ± 0.08%**	**77.59 ± 0.12%**
Dis. k-shot [2]	ResNet34	56.30 ± 0.40%	73.90 ± 0.30%
Self-Jig(SVM) [5]	ResNet50	58.80 ± 1.36%	76.71 ± 0.72%
Hyperbolic ProtoNet	ResNet18	59.47 ± 0.20%	76.84 ± 0.14%

- Caltech-UCSD Birds – few shot learning

Baselines	Embedding Net	1-Shot 5-Way	5-Shot 5-Way
MatchingNet [53]	4 Conv	61.16 ± 0.89	72.86 ± 0.70
MAML [9]	4 Conv	55.92 ± 0.95%	72.09 ± 0.76%
ProtoNet [43]	4 Conv	51.31 ± 0.91%	70.77 ± 0.69%
MACO [15]	4 Conv	60.76%	74.96%
RelationNet [48]	4 Conv	62.45 ± 0.98%	76.11 ± 0.69%
Baseline++ [4]	4 Conv	60.53 ± 0.83%	79.34 ± 0.61%
DN4-DA [23]	4 Conv	53.15 ± 0.84%	81.90 ± 0.60%
Hyperbolic ProtoNet	4 Conv	**64.02 ± 0.24%**	**82.53 ± 0.14%**

• hyperbolic geometry에 대한 더 나은 이해가 필요하다.

Future work may include several potential modifications of the approach. We have observed that the benefit of hyperbolic embeddings may be substantially bigger in some tasks and datasets than in others. A better understanding of when and why the use of hyperbolic geometry is warranted is therefore needed. Finally, we note that while all hyperbolic geometry models are equivalent in the continuous setting, fixed-precision arithmetic used in real computers breaks this equivalence. In practice, we observed that care should be taken about numeric precision effects. Using other models of hyperbolic geometry may result in a more favourable floating point performance.