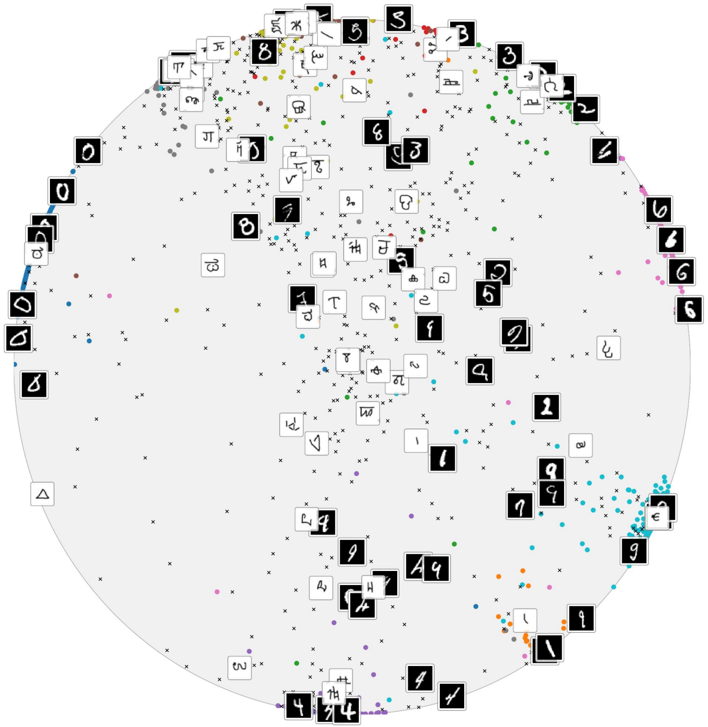


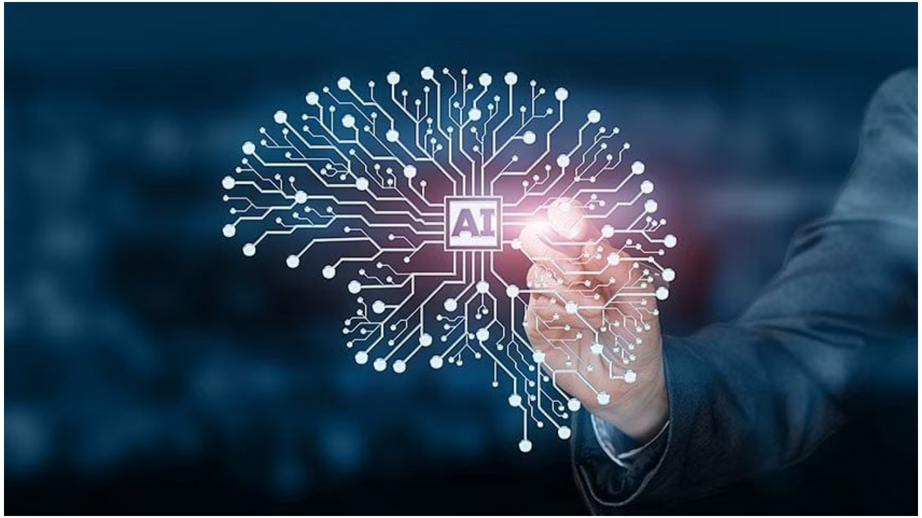
Hyperbolic Image Embeddings

CVPR 2020

이미지처리팀

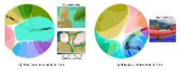
류채은, 강인하, 김병현, 김선옥, 김준철, 김현진, 남종우, 안종식, 이찬혁, 이희재, 조경민, 최승준, 현청천

Table of Contents



01

Introductio



Hyperbolic Image Segmentation

1 code implementation • CVPR 2022

For image segmentation, the current standard is to perform pixel-level optimization and inference in Euclidean output embedding spaces through linear hyperplanes.

Image Segmentation | Semantic Segmentation

HypLiLoc: Towards Effective LiDAR Pose Regression with Hyperbolic Fusion

2 code implementations • CVPR 2023

LiDAR relocalization plays a crucial role in many fields, including robotics, autonomous driving, and computer vision.

Ranked #1 on Visual Localization on Oxford Radar RobotCar (Full-6)

Autonomous Driving | lidar absolute pose regression | +3

Fully Hyperbolic Neural Networks

1 code implementation • ACL 2022

Hyperbolic neural networks have shown great potential for modeling complex data.

OpenMetricLearning

Hyperbolic Vision Transformers: Combining Improvements in Metric Learning

2 code implementations • CVPR 2022

Following this line of work, we propose a new hyperbolic-based model for metric learning.

Ranked #1 on Metric Learning on CUB-200-2011

Metric Learning

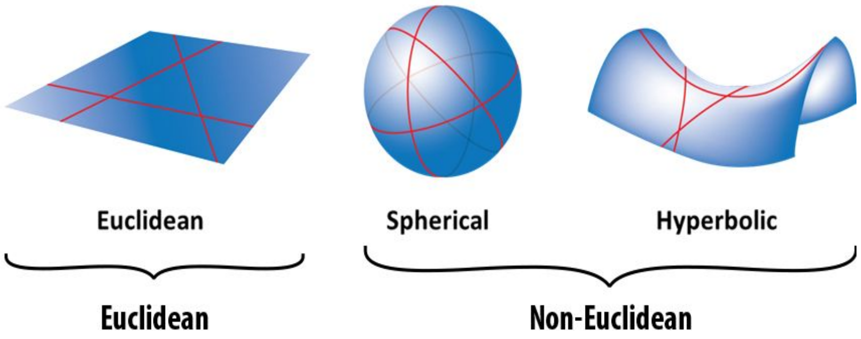
Hyperbolic Visual Embedding Learning for Zero-Shot Recognition

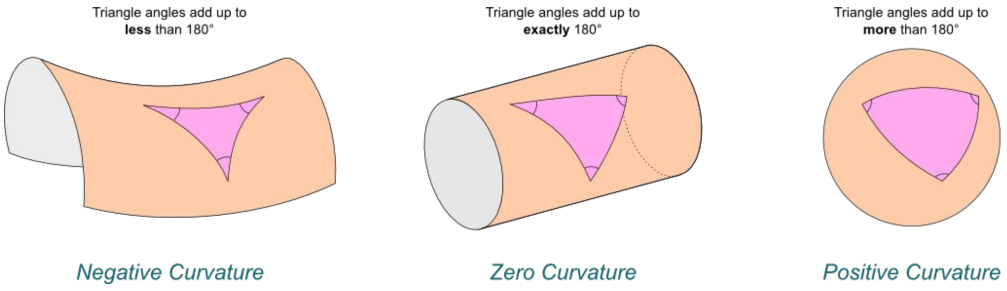
1 code implementation • CVPR 2020

This paper proposes a Hyperbolic Visual Embedding Learning Network for zero-shot recognition.

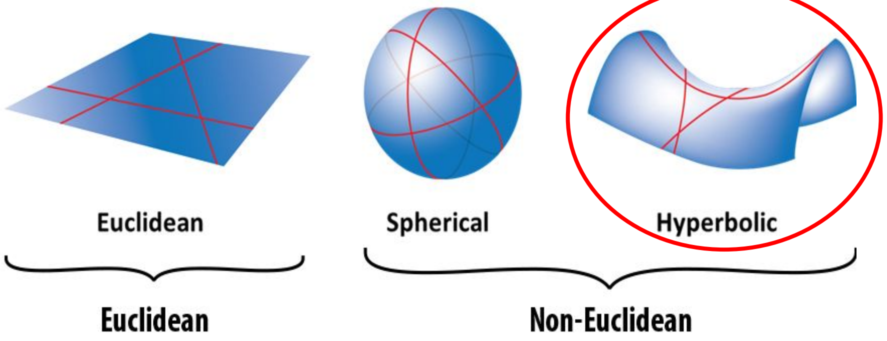
Zero-Shot Learning

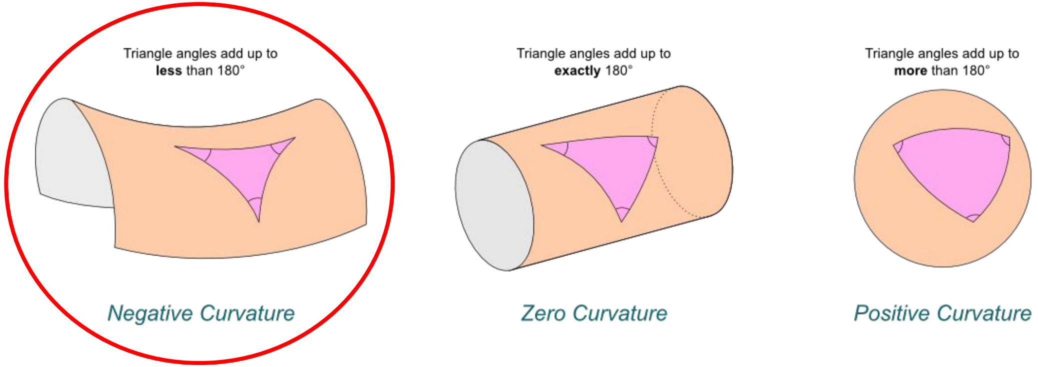
Introduction



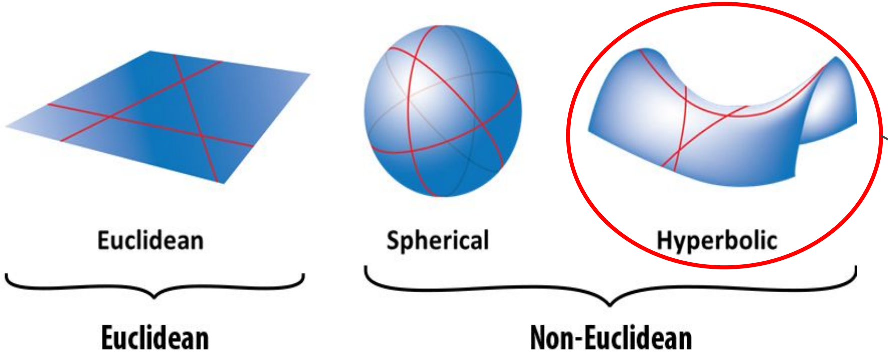


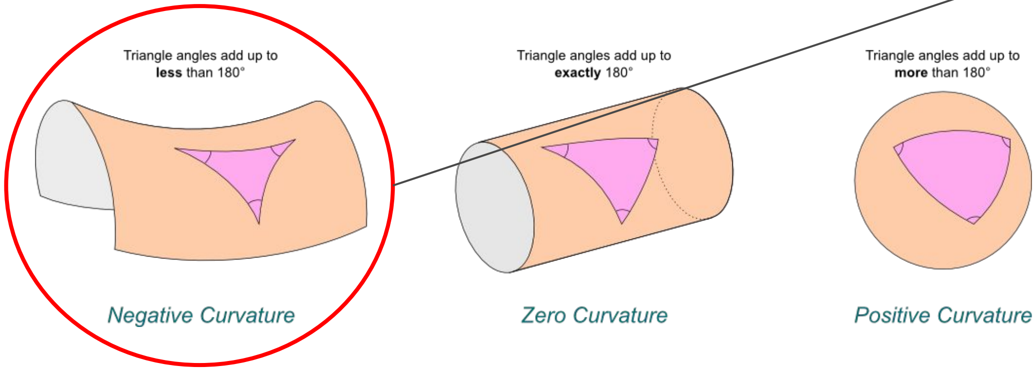
Introduction

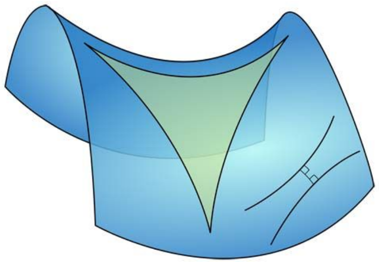




Introduction





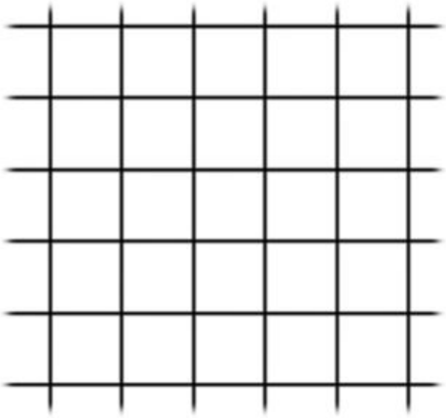


Hyperbolic spaces with negative curvature might often be more appropriate for learning embedding of images

Prerequisites–Comparing Geometries

- **All geometries shown are 2D!**
- **If we cover geometries with squares:**

Euclidean Geometry



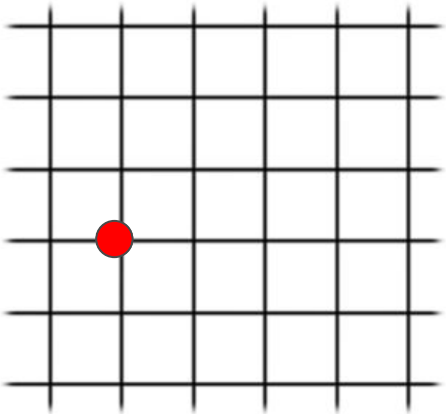
Hyperbolic Geometry

Ref: https://www.youtube.com/watch?v=zQo_S3yNa2w

Prerequisites–Comparing Geometries

- All geometries shown are 2D!
- If we cover geometries with squares:

Euclidean Geometry



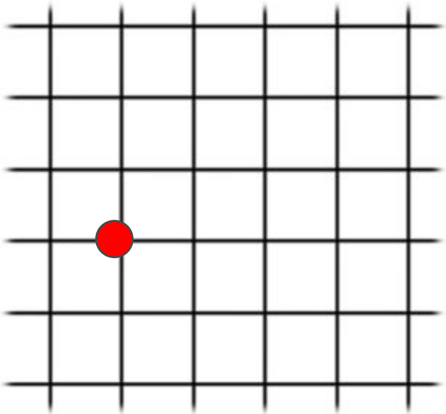
Hyperbolic Geometry

Ref: https://www.youtube.com/watch?v=zQo_S3yNa2w

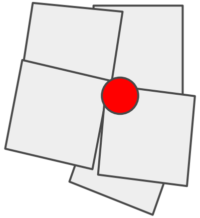
Prerequisites–Comparing Geometries

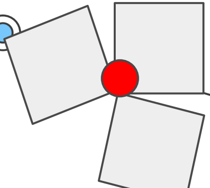
- **All geometries shown are 2D!**
- **If we cover geometries with squares:**

Euclidean Geometry



Hyperbolic Geometry



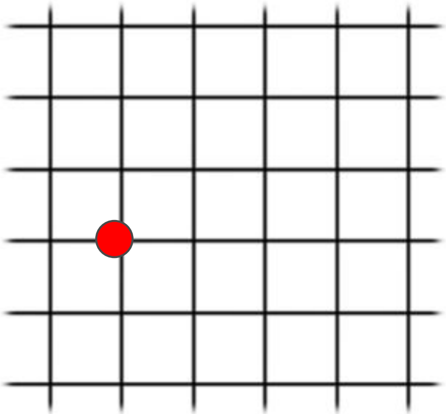


Ref: https://www.youtube.com/watch?v=zQo_S3yNa2w

Prerequisites–Comparing Geometries

- All geometries shown are 2D!
- If we cover geometries with squares:

Euclidean Geometry



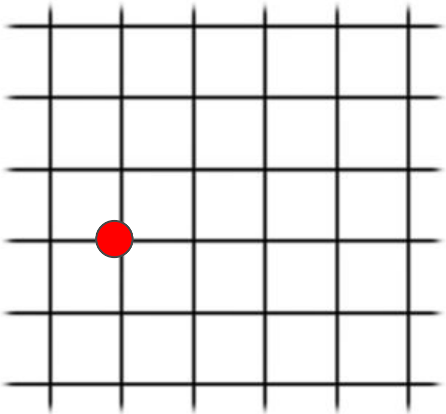
Hyperbolic Geometry

Ref: https://www.youtube.com/watch?v=zQo_S3yNa2w

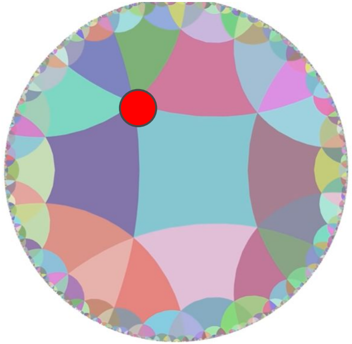
Prerequisites-Comparing Geometries

- All geometries shown are 2D!
- If we cover geometries with squares:

Euclidean Geometry



Hyperbolic Geometry



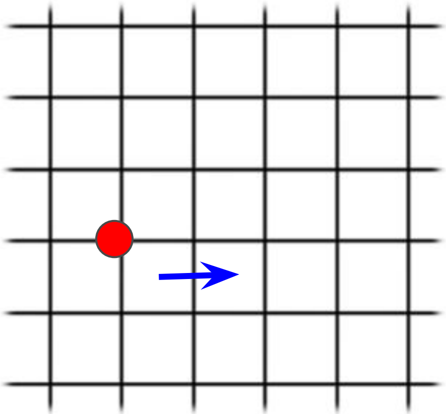
Area of Hyperbolic plane grows rapidly than Euclidean plane!

Ref: https://www.youtube.com/watch?v=zQo_S3yNa2w

Prerequisites-Comparing Geometries

- **All geometries shown are 2D!**
- **If we cover geometries with squares:**

Euclidean Geometry



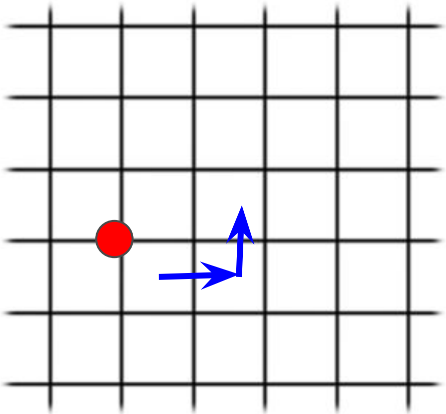
Hyperbolic Geometry

Ref: https://www.youtube.com/watch?v=zQo_S3yNa2w

Prerequisites–Comparing Geometries

- All geometries shown are 2D!
- If we cover geometries with squares:

Euclidean Geometry



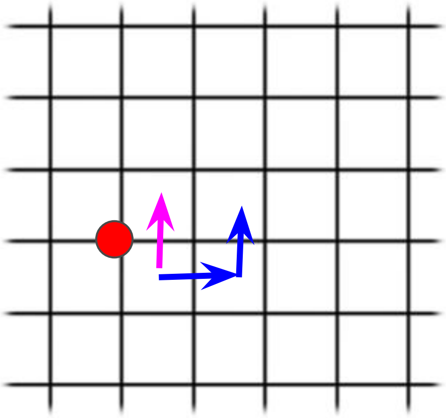
Hyperbolic Geometry

Ref: https://www.youtube.com/watch?v=zQo_S3yNa2w

Prerequisites–Comparing Geometries

- All geometries shown are 2D!
- If we cover geometries with squares:

Euclidean Geometry



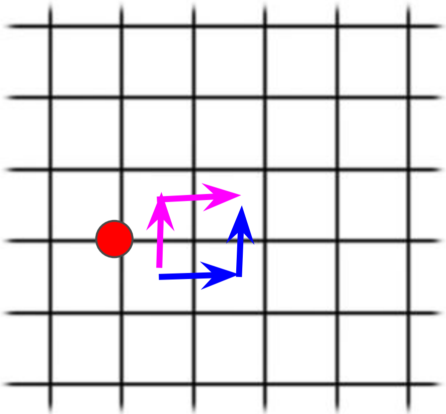
Hyperbolic Geometry

Ref: https://www.youtube.com/watch?v=zQo_S3yNa2w

Prerequisites–Comparing Geometries

- All geometries shown are 2D!
- If we cover geometries with squares:

Euclidean Geometry



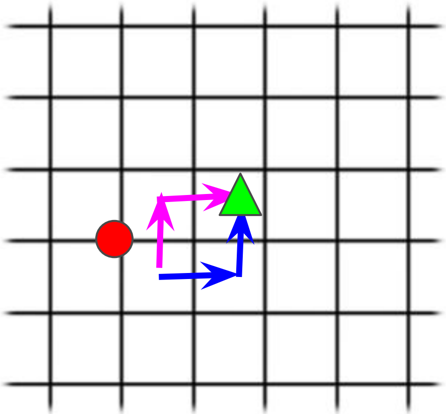
Hyperbolic Geometry

Ref: https://www.youtube.com/watch?v=zQo_S3yNa2w

Prerequisites–Comparing Geometries

- **All geometries shown are 2D!**
- **If we cover geometries with squares:**

Euclidean Geometry



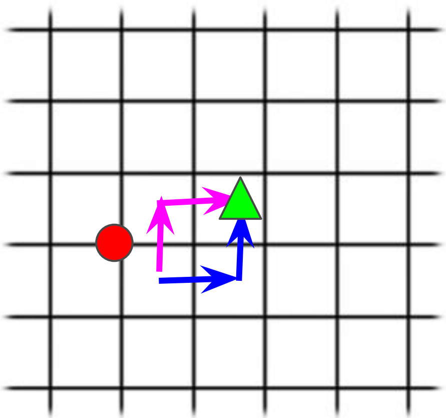
Hyperbolic Geometry

Ref: https://www.youtube.com/watch?v=zQo_S3yNa2w

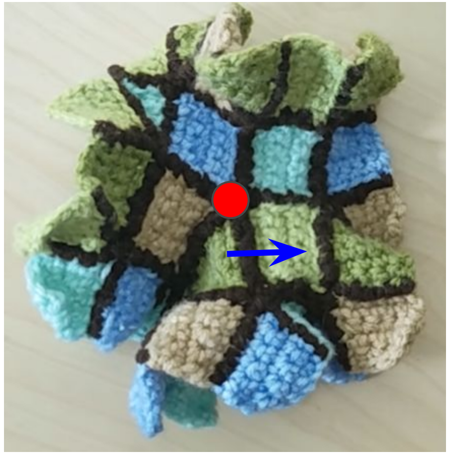
Prerequisites–Comparing Geometries

- All geometries shown are 2D!
- If we cover geometries with squares:

Euclidean Geometry



Hyperbolic Geometry

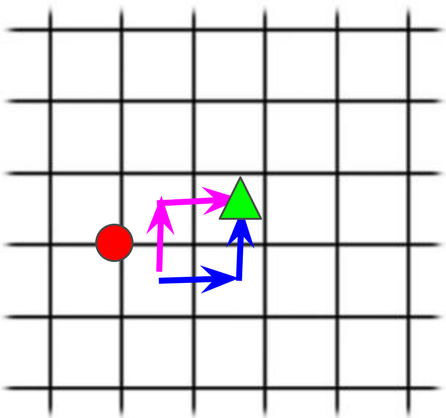


Ref: https://www.youtube.com/watch?v=zQo_S3yNa2w

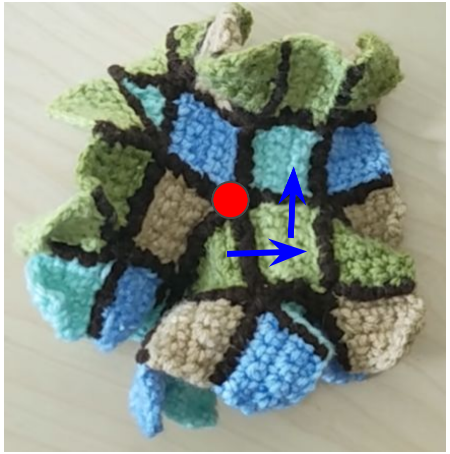
Prerequisites-Comparing Geometries

- All geometries shown are 2D!
- If we cover geometries with squares:

Euclidean Geometry



Hyperbolic Geometry

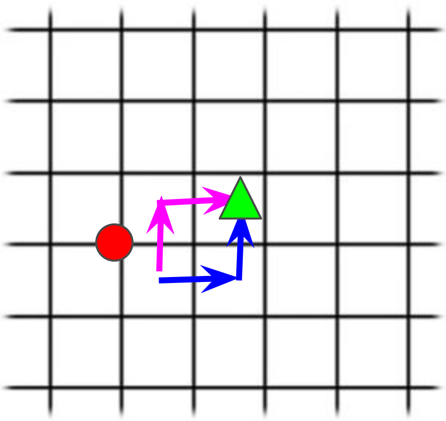


Ref: https://www.youtube.com/watch?v=zQo_S3yNa2w

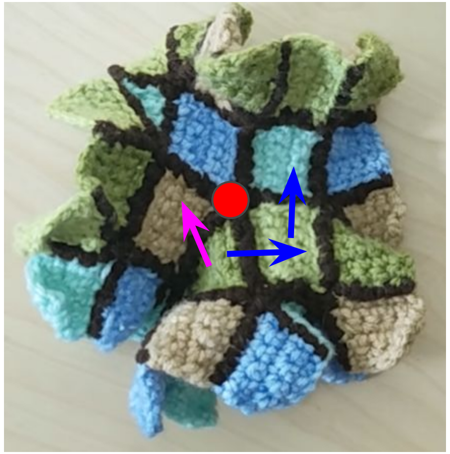
Prerequisites–Comparing Geometries

- All geometries shown are 2D!
- If we cover geometries with squares:

Euclidean Geometry



Hyperbolic Geometry

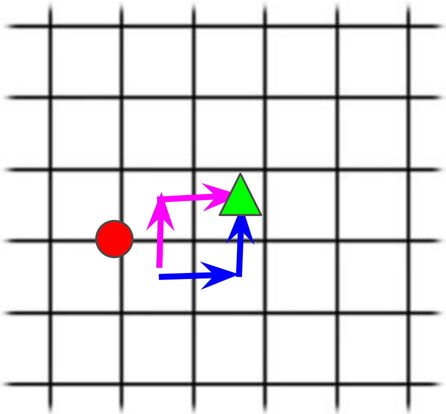


Ref: https://www.youtube.com/watch?v=zQo_S3yNa2w

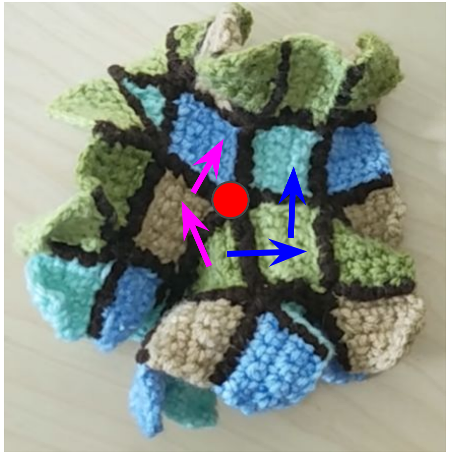
Prerequisites–Comparing Geometries

- All geometries shown are 2D!
- If we cover geometries with squares:

Euclidean Geometry



Hyperbolic Geometry

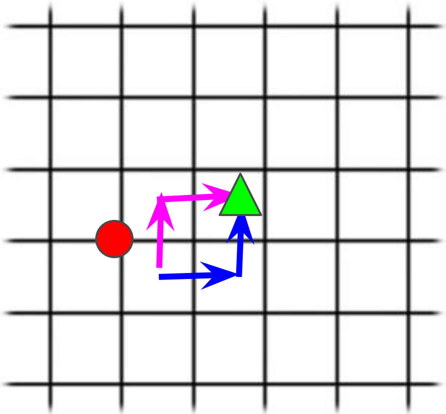


Ref: https://www.youtube.com/watch?v=zQo_S3yNa2w

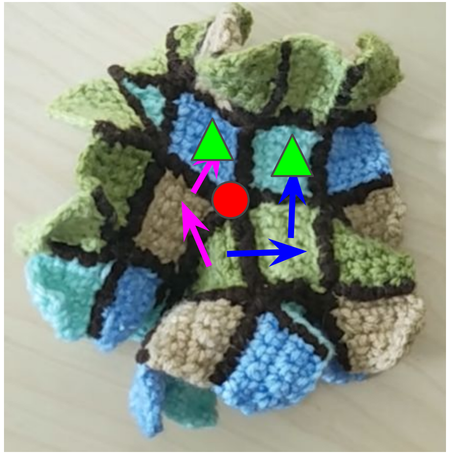
Prerequisites–Comparing Geometries

- All geometries shown are 2D!
- If we cover geometries with squares:

Euclidean Geometry



Hyperbolic Geometry



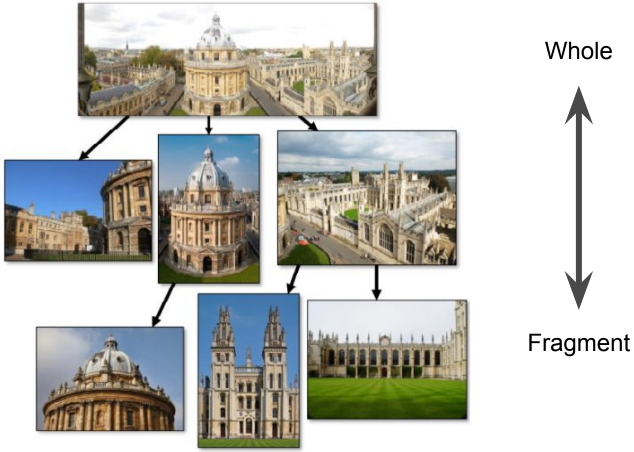
Basic euclidean operations not defined in hyperbolic geometry!

Ref: https://www.youtube.com/watch?v=zQo_S3yNa2w

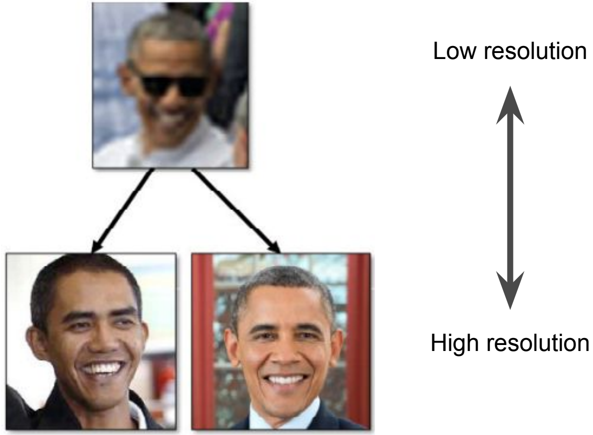
Introduction

How can images have hierarchies?

Example 1) Image Retrieval



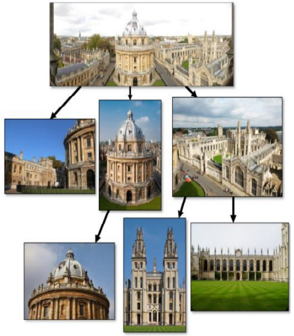
Example 2) Recognition tasks



Introduction

Why is hyperbolic space useful for embedding hierarchy?

Example 1) Image Retrieval



Whole

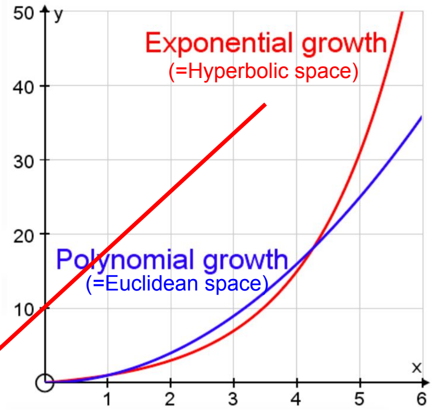
Fragment

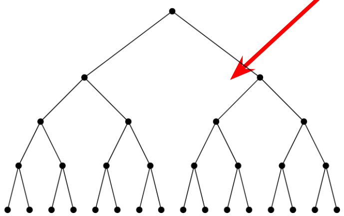
Example 2) Recognition tasks

Low resolution

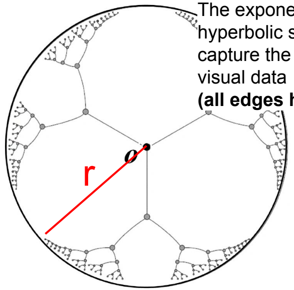
High resolution

Volume Expansion



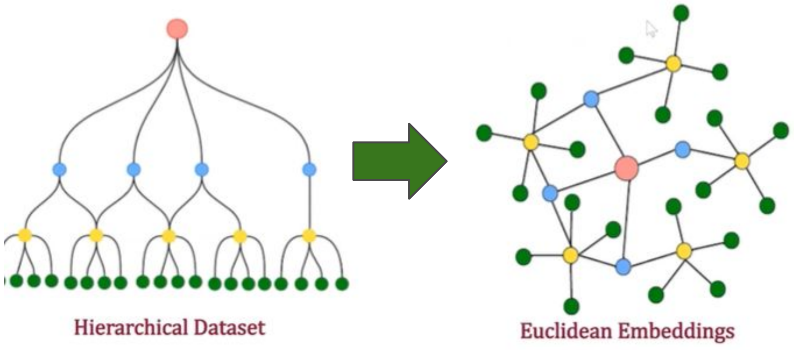


Given h depth, number of total Nodes in binary tree
= 2^h+1
= the number of tree nodes **grows exponentially** with tree depth

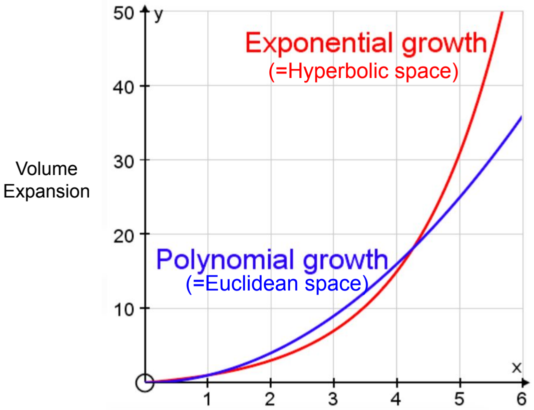


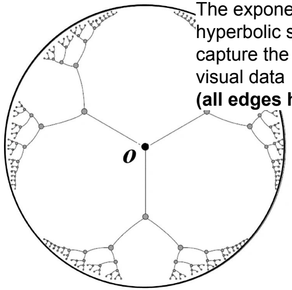
The exponentially expanding hyperbolic space will be able to capture the underlying hierarchy of visual data
(all edges have equal length)

Introduction



Points get clustered

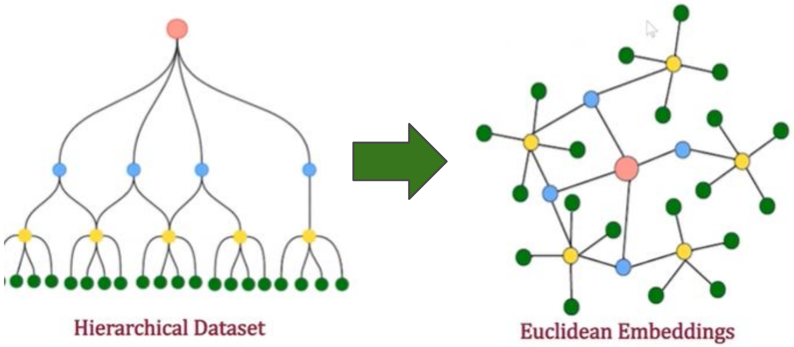




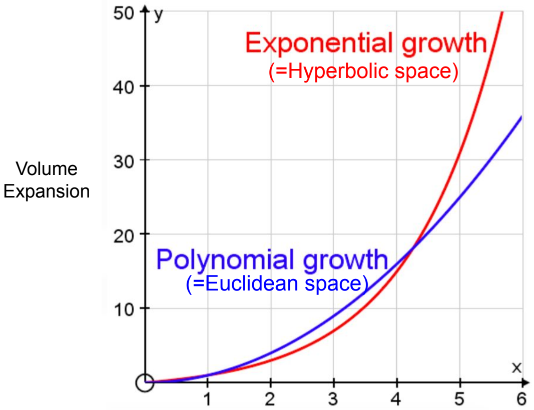
The exponentially expanding hyperbolic space will be able to capture the underlying hierarchy of visual data
(all edges have equal length)

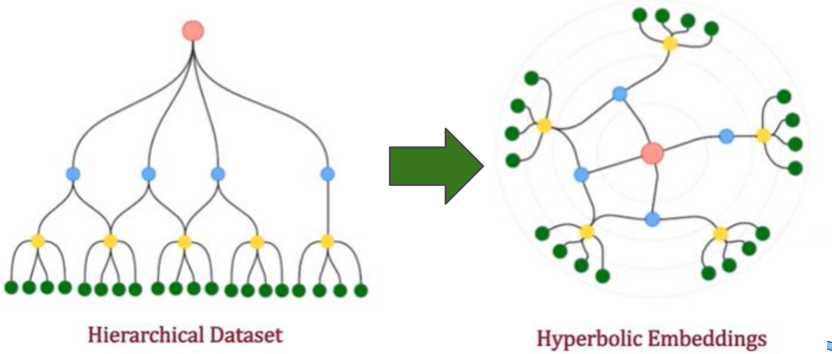
Ref: https://nurendra.me/hyperbolic-networks-tutorial/

Introduction

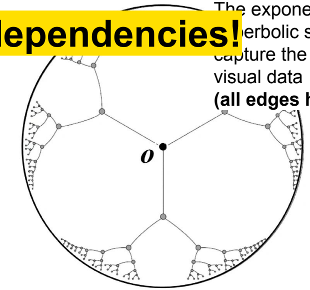


Points get clustered





Better capture hierarchical dependencies!

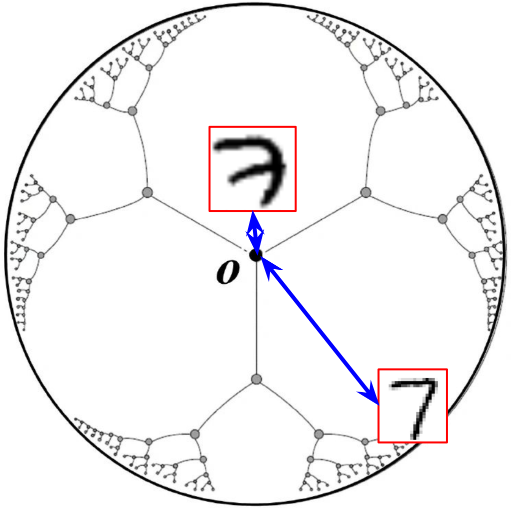


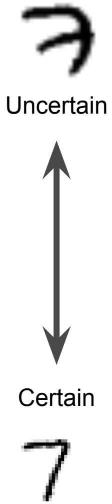
The exponentially expanding hyperbolic space will be able to capture the underlying hierarchy of visual data
(all edges have equal length)

Ref: https://nurendra.me/hyperbolic-networks-tutorial/

Introduction

Hierarchy in this paper





Network: “I have 0.1 confident score that this is 7”

$\propto$ **P(y=c|x)**

Network: “I have 0.9 confident score that this is 7”

02

Related Work

Related Works

Natural Language Preprocessing

Ref: Hyperbolic Neural Networks (Advances in Neural Information Processing Systems,2018)

sentence entailment classification task

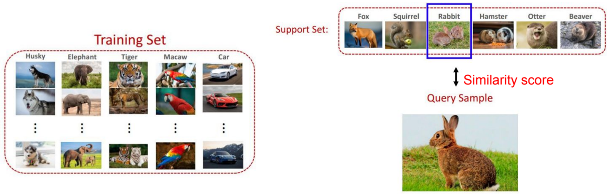
WordNet subtree	Model	D = 2	D = 3	D = 5	D = 10
animal.n.01 3218 / 798	Hyp	**47.43 ± 1.07**	**91.92 ± 0.61**	**98.07 ± 0.55**	**99.26 ± 0.59**
	Eucl	41.69 ± 0.19	68.43 ± 3.90	95.59 ± 1.18	**99.36 ± 0.18**
	$\log_0$	38.89 ± 0.01	62.57 ± 0.61	89.21 ± 1.34	98.27 ± 0.70
group.n.01 6649 / 1727	Hyp	**81.72 ± 0.17**	**89.87 ± 2.73**	**87.89 ± 0.80**	**91.91 ± 3.07**
	Eucl	61.13 ± 0.42	63.56 ± 1.22	67.82 ± 0.81	**91.38 ± 1.19**
	$\log_0$	60.75 ± 0.24	61.98 ± 0.57	67.92 ± 0.74	**91.41 ± 0.18**
worker.n.01 861 / 254	Hyp	**12.68 ± 0.82**	**24.09 ± 1.49**	**55.46 ± 5.49**	**66.83 ± 11.38**
	Eucl	10.86 ± 0.01	22.39 ± 0.04	35.23 ± 3.16	47.29 ± 3.93
	$\log_0$	9.04 ± 0.06	22.57 ± 0.20	26.47 ± 0.78	36.66 ± 2.74
mammal.n.01 953 / 228	Hyp	**32.01 ± 17.14**	**87.54 ± 4.55**	**88.73 ± 3.22**	**91.37 ± 6.09**
	Eucl	**15.58 ± 0.04**	44.68 ± 1.87	59.35 ± 1.31	77.76 ± 5.08
	$\log_0$	13.10 ± 0.13	44.89 ± 1.18	52.51 ± 0.85	56.11 ± 2.21

Table 2: Test F1 classification scores (%) for four different subtrees of WordNet noun tree. 95% confidence intervals for 3 different runs are shown for each method and each dimension. "Hyp" denotes our hyperbolic MLR, "Eucl" denotes directly applying Euclidean MLR to hyperbolic embeddings in their Euclidean parametrization, and $\log_0$ denotes applying Euclidean MLR in the tangent space at $\mathbf{0}$, after projecting all hyperbolic embeddings there with $\log_0$.

WordNet Hierarchy

Our work is inspired by the recent body of works that demonstrate the advantage of learning hyperbolic embeddings for language entities such as taxonomy entries, common words, phrases, and for other NLP tasks, such as neural machine translation

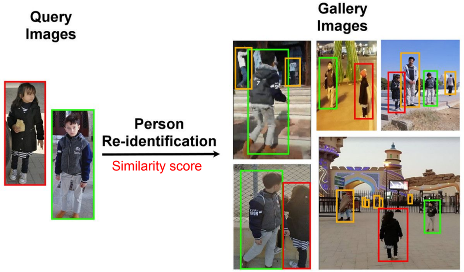
Few-shot Learning



The overall ability of the model to generalize to unseen data during training

While the previous methods employ either Euclidean or spherical geometries, there was no extension to hyperbolic spaces.

Person re-identification



To match pedestrian images captured by possibly non-overlapping surveillance cameras

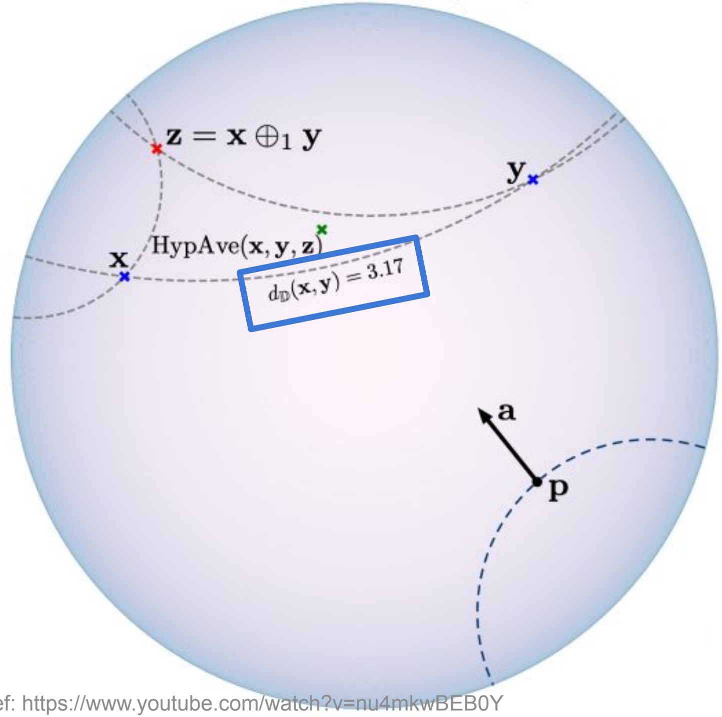
Papers adopt the pairwise models that accept pairs of images and output their similarity scores. The resulting similarity scores are used to classify the input pairs as being matching or non-matching.

03

Methodology

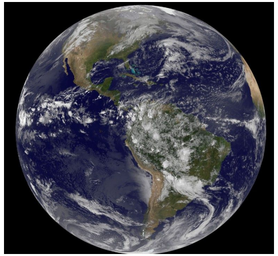
Hyperbolic Spaces & Hyperbolicity Estimation

$$\mathbb{D}^2 = \{\mathbf{x} \in \mathbb{R}^2 : \|\mathbf{x}\| < 1\}$$



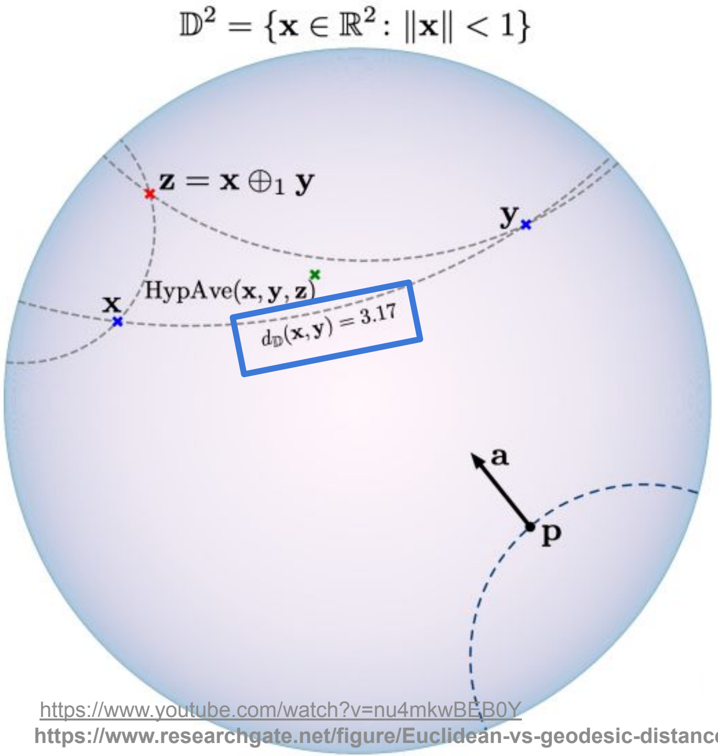
Geodesic distance

$$d_{\mathbb{D}}(\mathbf{x}, \mathbf{y}) = \operatorname{arccosh}\left(1 + 2\frac{\|\mathbf{x} - \mathbf{y}\|^2}{(1 - \|\mathbf{x}\|^2)(1 - \|\mathbf{y}\|^2)}\right)$$



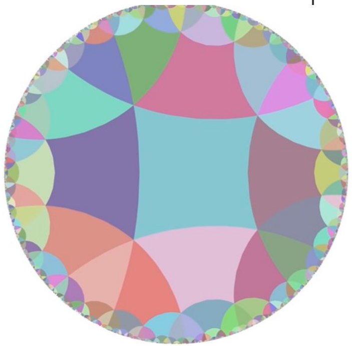
Ref: https://www.youtube.com/watch?v=nu4mkwBEB0Y

Hyperbolic Spaces & Hyperbolicity Estimation



Geodesic distance

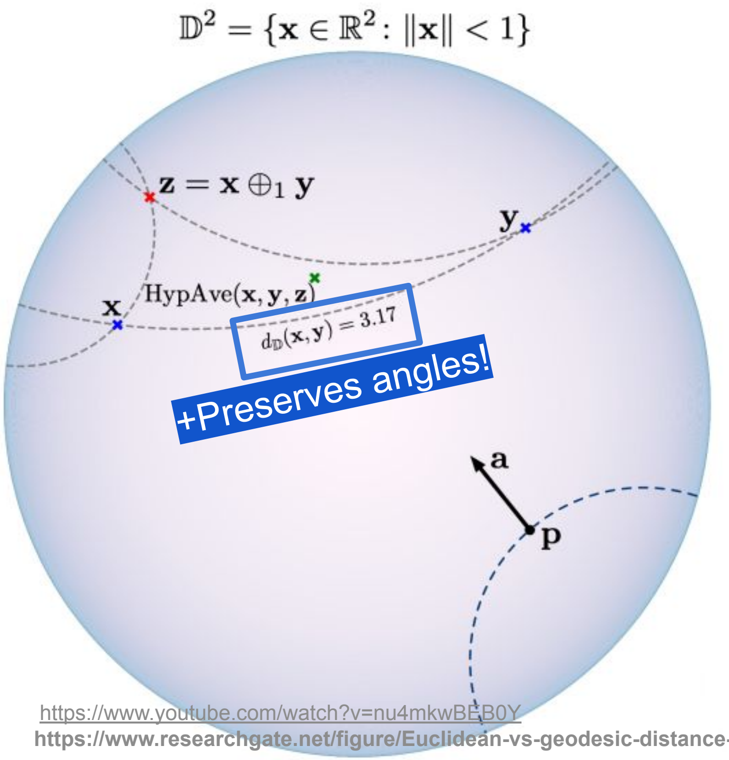
$$d_{\mathbb{D}}(\mathbf{x}, \mathbf{y}) = \operatorname{arccosh}\left(1 + 2\frac{\|\mathbf{x}-\mathbf{y}\|^2}{(1-\|\mathbf{x}\|^2)(1-\|\mathbf{y}\|^2)}\right)$$



Ref:

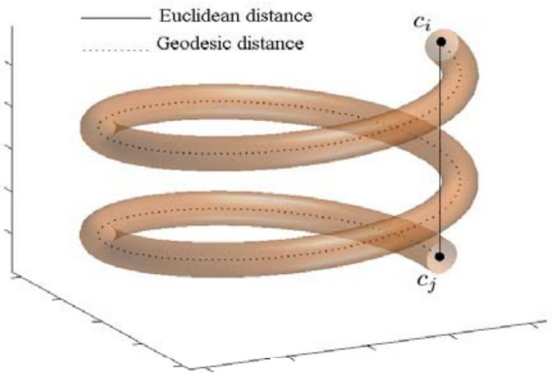
- https://www.youtube.com/watch?v=nu4mkwBEB0Y
- **https://www.researchgate.net/figure/Euclidean-vs-geodesic-distance-on-a-nonlinear-manifold_fig3_2894469**

Hyperbolic Spaces & Hyperbolicity Estimation



Geodesic distance

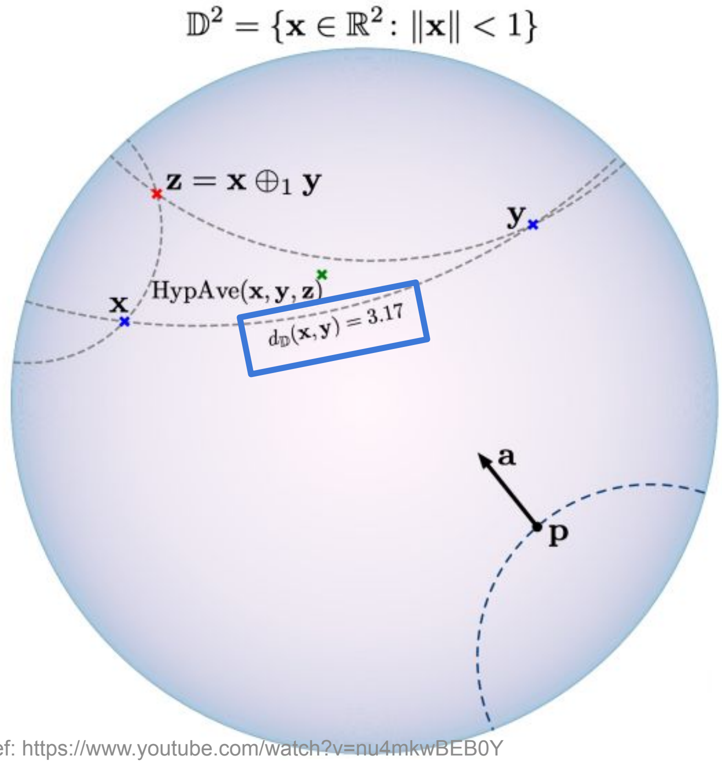
$$d_{\mathbb{D}}(\mathbf{x}, \mathbf{y}) = \operatorname{arccosh}\left(1 + 2\frac{\|\mathbf{x} - \mathbf{y}\|^2}{(1 - \|\mathbf{x}\|^2)(1 - \|\mathbf{y}\|^2)}\right)$$



Ref:

- https://www.youtube.com/watch?v=nu4mkwBEB0Y
- https://www.researchgate.net/figure/Euclidean-vs-geodesic-distance-on-a-nonlinear-manifold_fig3_2894469

Hyperbolic Spaces & Hyperbolicity Estimation

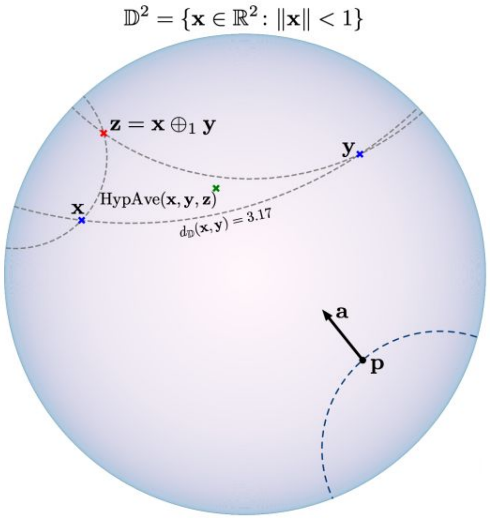


Geodesic distance

$$d_{\mathbb{D}}(\mathbf{x}, \mathbf{y}) = \operatorname{arccosh}\left(1 + 2\frac{\|\mathbf{x} - \mathbf{y}\|^2}{(1 - \|\mathbf{x}\|^2)(1 - \|\mathbf{y}\|^2)}\right)$$

Ref: https://www.youtube.com/watch?v=nu4mkwBEB0Y

Hyperbolic Spaces & Hyperbolicity Estimation



- Most layers use Euclidean operators, such as standard generalized convolutions, while **only the final layers operate within the hyperbolic geometry framework**

- More ambiguous objects closer to the origin while moving more specific objects towards the boundary. **The distance to the origin in our models, therefore, provides a natural estimate of uncertainty**

Ref: https://www.youtube.com/watch?v=nu4mkwBEB0Y

δ-Hyperbolicity

Ref: Hyperbolic Deep Neural Networks: A Survey (IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. 44, NO. 12, DECEMBER 2022)

Gromov δ-Hyperbolicity

: a property that measures how "*curved*" or "*bendy*" a space is

A metric space (X,d) is δ hyperbolic if there exists a δ>0 such that four points *a,b,c,v* in X:

$$\boxed{\langle a,b\rangle_v} \geq \min\{\langle a,c\rangle_v, \langle b,c\rangle_v\} - \delta$$

Minimal value given the constraints!

$$\boxed{\langle a,b\rangle_v = \frac{1}{2}(d(a,v) + d(b,v) - d(a,b))}$$

d: distance function

$$\delta_{rel}(X) = \frac{2\delta(X)}{\text{diam}(X)}$$

δ-Hyperbolicity

Ref: Hyperbolic Deep Neural Networks: A Survey (IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. 44, NO. 12, DECEMBER 2022)

Gromov δ-Hyperbolicity

: a property that measures how "*curved*" or "*bendy*" a space is

$$\boxed{\delta_{rel}(X)} = \frac{2\delta(X)}{\boxed{\text{diam}(X)}}$$

$\in [0, 1]$

the longest distance between any two points within that set

scale-invariant metric

= specifies how close is a dataset X to a hyperbolic space

Strong hyperbolicity

Non hyperbolicity

δ-Hyperbolicity

Ref: Hyperbolic Deep Neural Networks: A Survey (IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. 44, NO. 12, DECEMBER 2022)

Gromov δ-Hyperbolicity

: a property that measures how "*curved*" or "*bendy*" a space is

$$\boxed{\delta_{rel}(X)} = \frac{2\delta(X)}{\boxed{\mathrm{diam}(X)}}$$

$\in [0, 1]$

scale-invariant metric

the longest distance between any two points within that set

= specifies how close is a dataset X to a hyperbolic space

Strong hyperbolicity

Non hyperbolicity

Encoder	Dataset			
	CIFAR10	CIFAR100	CUB	*Mini*ImageNet
Inception v3 [49]	0.25	0.23	0.23	0.21
ResNet34 [14]	0.26	0.25	0.25	0.21
VGG19 [42]	0.23	0.22	0.23	0.17

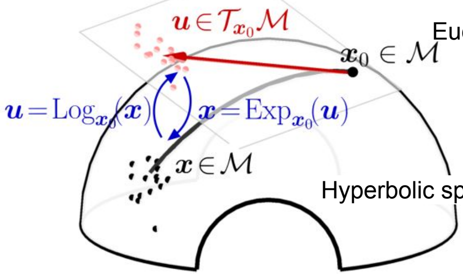
The degree of hyperbolicity in image datasets are quite high

Hyperbolic Operations

We need to define well-known operations in new space!
-> Utilizing the formalism of **Möbius gyrovector spaces to generalize many standard operations to hyperbolic spaces**

- Distance $d_c(\mathbf{x},\mathbf{y}) := \frac{2}{\sqrt{c}} \operatorname{arctanh}(\sqrt{c}\|-\mathbf{x} \oplus_c \mathbf{y}\|).$ $\mathbf{x} \oplus_c \mathbf{y} := \frac{(1+2c\langle \mathbf{x},\mathbf{y}\rangle + c\|\mathbf{y}\|^2)\mathbf{x} + (1-c\|\mathbf{x}\|^2)\mathbf{y}}{1+2c\langle \mathbf{x},\mathbf{y}\rangle + c^2\|\mathbf{x}\|^2\|\mathbf{y}\|^2}$
- Exponential and logarithmic maps

Euclidean vectors



Tangent space – manifold mappin

Euclidean vectors → Hyperbolic space

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

Euclidean vectors ← Hyperbolic space

$$\log_{\mathbf{x}}^c(\mathbf{y}) := \frac{2}{\sqrt{c}\lambda_{\mathbf{x}}^c} \operatorname{arctanh}(\sqrt{c}\|-\mathbf{x} \oplus_c \mathbf{y}\|) \frac{-\mathbf{x} \oplus_c \mathbf{y}}{\|-\mathbf{x} \oplus_c \mathbf{y}\|}$$

Hyperbolic averaging

$$\mathrm{HypAve}(\mathbf{x}_1, \ldots, \mathbf{x}_N) = \sum_{i=1}^{N} \gamma_i \mathbf{x}_i / \sum_{i=1}^{N} \gamma_i$$

- ✓ Addition (by extension subtraction) $x \oplus_c a = \frac{(1 + 2ca \cdot x + c \parallel x \parallel^2)a + (1 - c \parallel a \parallel^2)x}{1 + 2ca \cdot x + c^2 \parallel a \parallel^2 \parallel x \parallel^2}$
- ✓ Scalar Multiplication $r \otimes_c x = exp_0^c(r log_0^c(x))$
- ✓ Matrix Multiplication $[M_1 \quad M_2]\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = M_1 \otimes_c x_1 \oplus_c M_2 \otimes_c x_2$
- ✓ Distance Metric $d_{\mathbb{H}}(x,y) = \frac{2}{\sqrt{c}} \tanh^{-1}(\sqrt{c} \parallel -x \oplus_c y \parallel)$
- ✓ Applying Function $f^{\otimes_c}(x) = exp_0^c(f(Log_0^c(x)))$
- ✓ SoftMax - MLR $P(y = k|x) \propto exp\left(\frac{\lambda_{p_k}^c \|a_k\|}{\sqrt{c}} \sinh^{-1}\left(\frac{2\sqrt{c}\langle -p_k \oplus_c x, a_k \rangle}{(1 - c\|-p_k \oplus_c x\|^2)\|a_k\|}\right)\right)$

Ref:
- https://nurendra.me/hyperbolic-networks-tutorial/
- https://arxiv.org/pdf/1805.09112.pdf

04

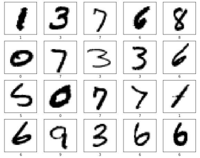
Experiments

Experimental Setup

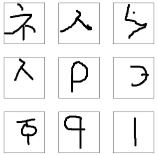
Hypothesis

: the distance to the center in Poincaré ball indicates a model's uncertainty

1. Train a classifier in hyperbolic space on the *MNIST* dataset



2. Evaluate it on the *Omniglot* dataset



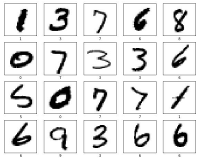
3. Investigate and compare the obtained distributions of <u>distances to the origin of hyperbolic embeddings</u> of the MNIST and Omniglot test sets

+) Few-shot Learning
+) Person re-identification

Distance to the origin as the measure of uncertainty

Validate Hypothesis

(=the distance to the center in Poincaré ball indicates a model's uncertainty)



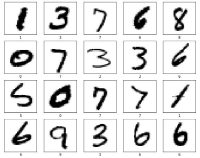
the distance to the center in Poincaré ball will correspond to visual ambiguity of images

```
class Net(nn.Module):
    def __init__(self, args):
        super(Net, self).__init__()
        self.conv1 = nn.Conv2d(1, 20, 5, 1)
        self.conv2 = nn.Conv2d(20, 50, 5, 1)
        self.fc1 = nn.Linear(4 * 4 * 50, 500)
        self.fc2 = nn.Linear(500, args.dim)
        self.tp = hypnn.ToPoincare(
            c=args.c, train_x=args.train_x, train_c=args.train_c, ball_dim=args.dim
        )
        self.mlr = hypnn.HyperbolicMLR(ball_dim=args.dim, n_classes=10, c=args.c)

    def forward(self, x):
        x = F.relu(self.conv1(x))
        x = F.max_pool2d(x, 2, 2)
        x = F.relu(self.conv2(x))
        x = F.max_pool2d(x, 2, 2)
        x = x.view(-1, 4 * 4 * 50)
        x = F.relu(self.fc1(x))
        x = self.fc2(x)
        x = self.tp(x)
        return F.log_softmax(self.mlr(x, c=self.tp.c), dim=-1)
```

The output of the last hidden layer was mapped to the Poincaré ball using the exponential map

Distance to the origin as the measure of uncertainty

Validate Hypothesis
(=the distance to the center in Poincaré ball indicates a model's uncertainty)



the distance to the center in Poincaré ball will correspond to visual ambiguity of images

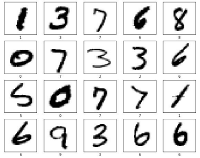
```
class Net(nn.Module):
    def __init__(self, args):
        super(Net, self).__init__()
        self.conv1 = nn.Conv2d(1, 20, 5, 1)
        self.conv2 = nn.Conv2d(20, 50, 5, 1)
        self.fc1 = nn.Linear(4 * 4 * 50, 500)
        self.fc2 = nn.Linear(500, args.dim)
        self.tp = hypnn.ToPoincare(
            c=args.c, train_x=args.train_x, train_c=args.train_c, ball_dim=args.dim
        )
        self.mlr = hypnn.HyperbolicMLR(ball_dim=args.dim, n_classes=10, c=args.c)

    def forward(self, x):
        x = F.relu(self.conv1(x))
        x = F.max_pool2d(x, 2, 2)
        x = F.relu(self.conv2(x))
        x = F.max_pool2d(x, 2, 2)
        x = x.view(-1, 4 * 4 * 50)
        x = F.relu(self.fc1(x))
        x = self.fc2(x)
        x = self.tp(x)
        return F.log_softmax(self.mlr(x, c=self.tp.c), dim=-1)
```

followed by the hyperbolic multi-linear regression (MLR) layer

Distance to the origin as the measure of uncertainty

Validate Hypothesis

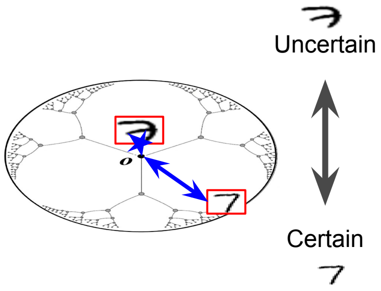
(=the distance to the center in Poincaré ball indicates a model's uncertainty)



the distance to the center in Poincaré ball will **correspond to visual ambiguity of images**

```
class Net(nn.Module):
    def __init__(self, args):
        super(Net, self).__init__()
        self.conv1 = nn.Conv2d(1, 20, 5, 1)
        self.conv2 = nn.Conv2d(20, 50, 5, 1)
        self.fc1 = nn.Linear(4 * 4 * 50, 500)
        self.fc2 = nn.Linear(500, args.dim)
        self.tp = hypnn.ToPoincare(
            c=args.c, train_x=args.train_x, train_c=args.train_c, ball_dim=args.dim
        )
        self.mlr = hypnn.HyperbolicMLR(ball_dim=args.dim, n_classes=10, c=args.c)

    def forward(self, x):
        x = F.relu(self.conv1(x))
        x = F.max_pool2d(x, 2, 2)
        x = F.relu(self.conv2(x))
        x = F.max_pool2d(x, 2, 2)
        x = x.view(-1, 4 * 4 * 50)
        x = F.relu(self.fc1(x))
        x = self.fc2(x)
        x = self.tp(x)
        return F.log_softmax(self.mlr(x, c=self.tp.c), dim=-1)
```

Train the model to ~ 99% test accuracy



Network: "I have 0.1 confident score that this is 7"

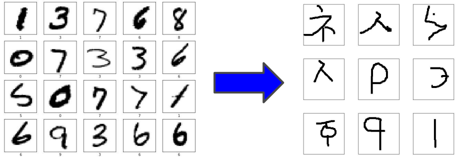
$\propto$ **P(y=c|x)**

Network: "I have 0.9 confident score that this is 7"

Distance to the origin as the measure of uncertainty

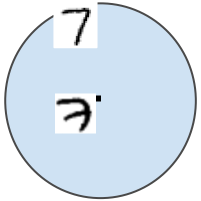
Validate Hypothesis

(=the distance to the center in Poincaré ball indicates a model’s uncertainty)



Train

Evaluate



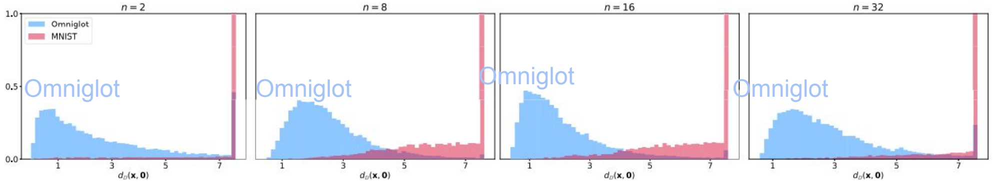


Figure 5: Distributions of the hyperbolic distance to the origin of the MNIST (red) and Omniglot (blue) datasets embedded into the Poincaré ball; parameter n denotes embedding dimension of the model trained for MNIST classification. Most Omniglot instances can be easily identified as out-of-domain based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

Distance to the origin as the measure of uncertainty

Validate Hypothesis

(=the distance to the center in Poincaré ball indicates a model's uncertainty)

observe that more "unclear" images are located near the center, while the images that are easy to classify are located closer to the boundary

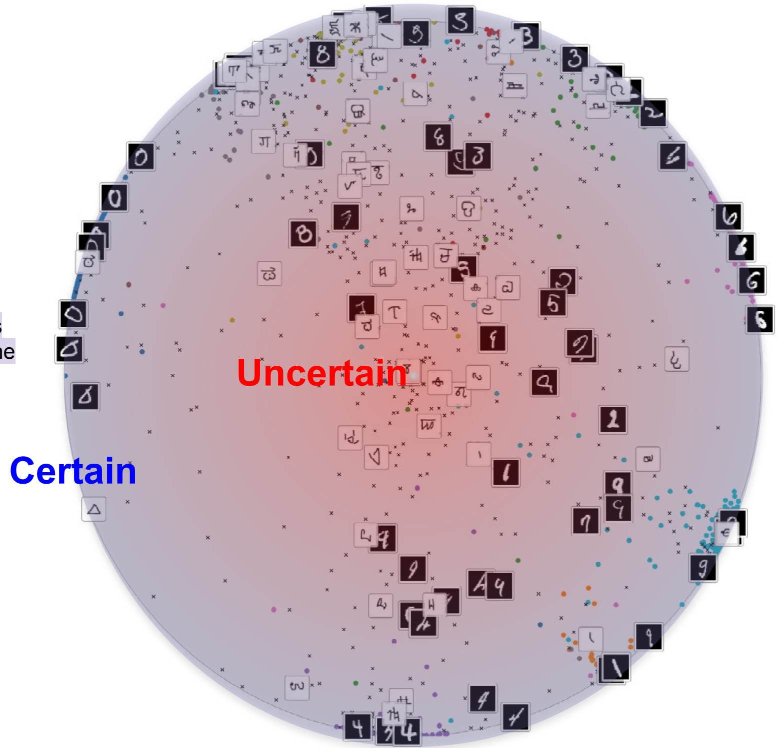


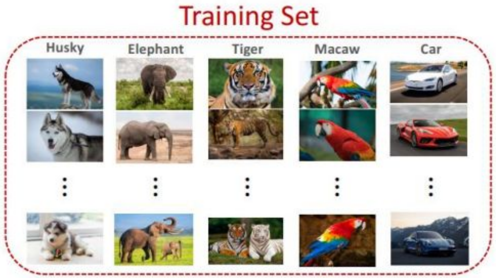
Figure 1: An example of two–dimensional Poincaré embeddings computed by a hyperbolic neural network trained on MNIST, and evaluated additionally on Omniglot. Ambiguous and unclear images from MNIST, as well as most of the images from Omniglot, are embedded near the center, while samples with clear class labels (or characters from Omniglot similar to one of the digits) lie near the boundary. *For inference, Omniglot was normalized to have the same background color as MNIST. Omniglot images are marked with black crosses, MNIST images with colored dots.

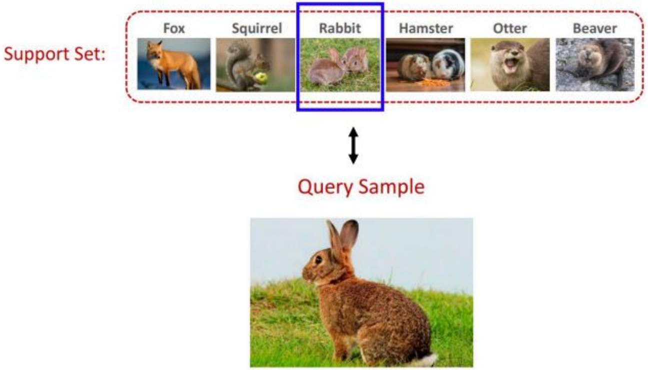
Few-shot classification

Validate Hypothesis

(=The few-shot classification task can benefit from hyperbolic embeddings, due to the ability of hyperbolic space to accurately reflect even very complex hierarchical relations between data points)

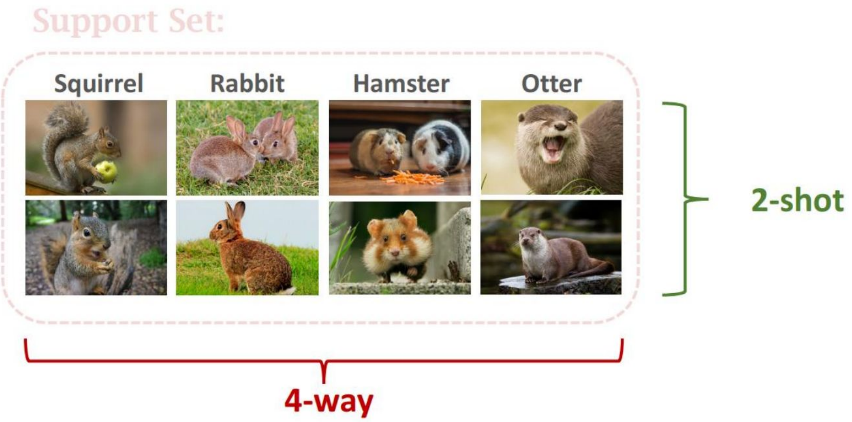
Few-shot learning





Few-shot classification

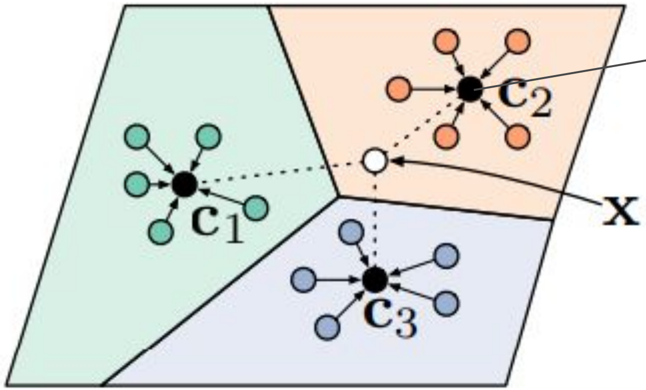
Validate Hypothesis
(=The few-shot classification task can benefit from hyperbolic embeddings, due to the ability of hyperbolic space to accurately reflect even very complex hierarchical relations between data points)



Few-shot classification

Validate Hypothesis
(=The few-shot classification task can benefit from hyperbolic embeddings, due to the ability of hyperbolic space to accurately reflect even very complex hierarchical relations between data points)

Prototypical networks (ProtoNets, NIPS 2017)



(3 way 5 shot)

Average ●s

Euclidean mean operation

replace

$$\mathrm{HypAve}(\mathbf{x}_1, \ldots, \mathbf{x}_N) = \sum_{i=1}^{N} \gamma_i \mathbf{x}_i / \sum_{i=1}^{N} \gamma_i$$

Few-shot classification

Validate Hypothesis

(=The few-shot classification task can benefit from hyperbolic embeddings, due to the ability of hyperbolic space to accurately reflect even very complex hierarchical relations between data points)

Prototypical networks (ProtoNets, NIPS 2017)

Table 4: Few-shot classification accuracy results on *Mini*ImageNet on 1-shot 5-way and 5-shot 5-way tasks. All accuracy results are reported with 95% confidence intervals.

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

Few-shot classification

Validate Hypothesis

(=The few-shot classification task can benefit from hyperbolic embeddings, due to the ability of hyperbolic space to accurately reflect even very complex hierarchical relations between data points)

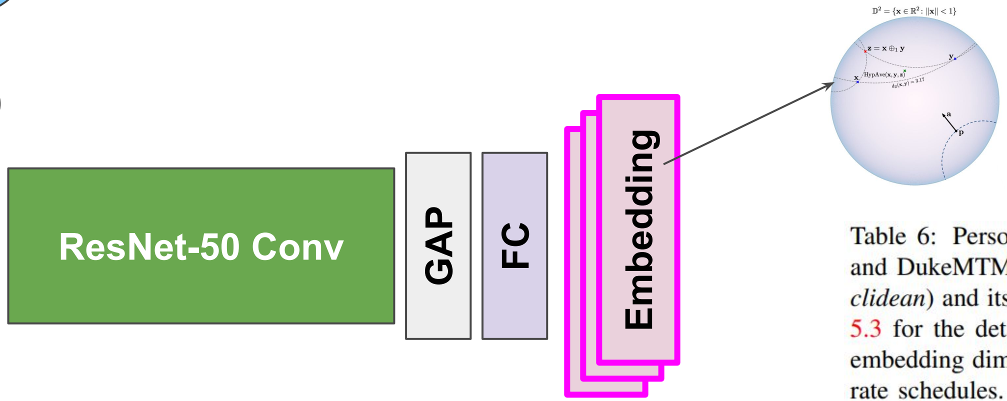
Prototypical networks (ProtoNets, NIPS 2017)

[redacted]

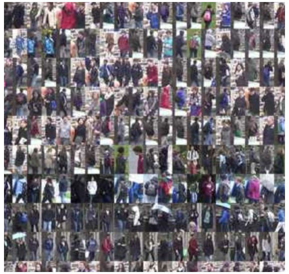
Table 5: Few-shot classification accuracy results on CUB dataset [55] on 1-shot 5-way task, 5-shot 5-way task. All accuracy results are reported with 95% confidence intervals. For each task, the best-performing method is highlighted.

	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	$55.92 \pm 0.95\%$	$72.09 \pm 0.76\%$
Standard	ProtoNet [43]	4 Conv	$51.31 \pm 0.91\%$	$70.77 \pm 0.69\%$
	MACO [15]	4 Conv	60.76%	74.96%
	RelationNet [48]	4 Conv	$62.45 \pm 0.98\%$	$76.11 \pm 0.69\%$
	Baseline++ [4]	4 Conv	$60.53 \pm 0.83\%$	$79.34 \pm 0.61\%$
	DN4-DA [23]	4 Conv	$53.15 \pm 0.84\%$	$81.90 \pm 0.60\%$
ProtoNet	**Hyperbolic ProtoNet**	4 Conv	$\mathbf{64.02 \pm 0.24\%}$	$\mathbf{82.53 \pm 0.14\%}$

Person re-identification



Market-1501



DukeMTMC-reID

Table 6: Person re-identification results for Market-1501 and DukeMTMC-reID for the classification baseline (*Euclidean*) and its hyperbolic counterpart (*Hyperbolic*). (See 5.3 for the details). The results are shown for the three embedding dimensionalities and for two different learning rate schedules. For each dataset and each embedding dimensionality value, the best results are bold, they are all given by the hyperbolic version of classification (either by the schedule *sch#1* or *sch#2*). The second-best results are underlined.

	Market-1501				DukeMTMC-reID			
	Euclidean		Hyperbolic		Euclidean		Hyperbolic	
dim, lr schedule	r1	mAP	r1	mAP	r1	mAP	r1	mAP
32, sch#1	71.4	49.7	69.8	45.9	56.1	35.6	56.5	34.9
32, sch#2	68.0	43.4	**75.9**	**51.9**	57.2	35.7	**62.2**	**39.1**
64, sch#1	80.3	60.3	83.1	60.1	69.9	48.5	**70.8**	**48.6**
64, sch#2	80.5	57.8	**84.4**	**62.7**	68.3	45.5	70.7	48.6
128, sch#1	86.0	67.3	**87.8**	**68.4**	74.1	53.3	**76.5**	**55.4**
128, sch#2	86.5	68.5	86.4	66.2	71.5	51.5	74.0	52.2

The hyperbolic version generally performs better than the Euclidean baseline

04

Conclusion

Discussion and Conclusion

- Shown that across a number of tasks, in particular in **the few-shot image classification**, learning hyperbolic embeddings can result in **a substantial boost in accuracy**.
- Speculate that the negative curvature of the hyperbolic spaces allows for embeddings that are **better conforming** to the intrinsic geometry of at least some image manifolds with their hierarchical structure.
- **A better understanding of when and why the use of hyperbolic geometry** is warranted is therefore **needed**.

Thank you!