Hyperbolic Neural Networks

Octavian-Eugen Ganea, Gary Bécigneul, Thomas Hofmann

NIPS 2018

Matei Bejan

UNIVERSITATEA DIN BUCUREȘTI

VIRTUTE ET SAPIENTIA

Table of contents

Introduction

- Many types of complex data (e.g. graph data) from a multitude of fields (biology, computer vision) exhibit a highly non-Euclidean latent structure.
- More complex geometries, the lack of closed form expressions for basic objects (e.g. distances, geodesics, parallel transport), thus the lack of deep learning progress in non-Euclidean settings.
- The authors aim to bridge the gap between hyperbolic and Euclidean geometry in the context of deep learning by generalizing in a principled manner both the basic operations as well as multinomial regression and neural networks to the Poincaré model of the hyperbolic geometry.
- Authors showcase how Euclidean and hyperbolic spaces can be continuously deformed into each other through a series of empirical tests.

Geometric Setting

Geometric Setting

- Use the Poincaré ball model $(\mathbb{D}^n, g^{\mathbb{D}})$, where $\mathbb{D} = \{x \in \mathbb{R}^n \mid ||x|| < 1\}$, the Rimmanian metric: $g_x^{\mathbb{D}} = \lambda_x^2 g^E$, with $\lambda_x = \frac{2}{1-||x||^2}$ and $g^E = I_n$ the Euclidean metric tensor.
- Introduce the Möbius addition of $x, y \in \mathbb{D}_c^n : \oplus_c$ and the Möbius scalar multiplication for $x \in \mathbb{D}_c^n$ and $r \in R : \otimes_c$. When c = 0, we retrieve the Euclidean addition and multiplication, respectively, of two vectors in $\mathbb{R}^n$
- Use the tensor g^c to induce the distance function on $\mathbb{D}_c^n$.

Gyrovector Spaces and the Poincaré Ball

- Use the Möbius addition to determine the exponential and logarithmic maps.
- Use exponential and logarithmic maps to obtain the Möbius scalar multiplication.

  This enables the generalization of scalar multiplication to matrix-vector multiplication on the Poincaré balls, which will form the building blocks of HNNs.
- Connect parallel transport along the unique geodesic from 0 to x to.

  This is crucial for defining biases and performing hyperparameter optimization between different tangent spaces.

Hyperbolic Neural Networks

Hyperbolic multinomial logistic regression

=-1000pt

- Very math-heavy section of the paper.
- Authors use MLR as a case study in order to explain how to construct a softmax layer for logits in the Poincaré ball.
- Main idea is to adapt the Euclidean definition to the hyperbolic setting by replacing the usual additive operator with the Möbius addition $\oplus_c$.
- The resulting Poincaré hyperplane can be described as the union of images of all geodesics in that Poincaré ball orthogonal to some gyrovector and containing another.

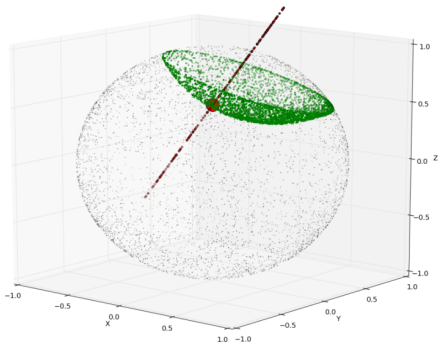

Figure 1: *An example of a hyperbolic hyperplane in $\mathbb{D}_1^3$ plotted using sampling. The red point is p. The shown normal axis to the hyperplane through p is parallel to a.*

Hyperbolic feed-forward layers

- Introduce ball-to-ball mapping functions that will play the role of linear mappings in normal Euclidean neural networks.
- Introduce Möbius matrix-vector multiplication by identifying the linear map with its matrix representation.
- We can use either the Möbius sum with a hyperbolic bias or the exponential map with an Euclidean bias.

  This is due to the fact that the composition of Möbius maps from one layer to another of the HNN can be re-written as an exponential map of Euclidean layer-to-layer maps.

Hyperbolic RNN

A **naive RNN** has the following form:

$$h_{t+1} = \varphi(W \cdot h_t + U \cdot x_t + b),$$

where h denote the hidden layers, W, the weights, x, the inputs, U, the cell state, and b represents the bias. φ is a pointwise non-linearity: tanh, sigmoid, ReLU etc.

This formula can be naturally generalized to the hyperbolic space as follows

Definition

For parameters $W \in \mathcal{M}_{m,n}(\mathbb{R}), U \in \mathcal{M}_{m,d}(\mathbb{R}), b \in \mathbb{D}_c^m$ we have:

$$h_{t+1} = \varphi^{\otimes_c}(W \otimes_c h_t \oplus_c U \otimes_c x_t \oplus_c b), h_t \in \mathbb{D}_c^n, x_t \in \mathbb{D}_c^d.$$

Hyperbolic RNN

The same generalization can be applied to **GRU units**:

$$r_t = \sigma(W^r \cdot h_{t-1} + U^r \cdot x_t + b^r), \qquad z_t = \sigma(W^z \cdot h_{t-1} + U^z \cdot x_t + b^z)$$
$$\widetilde{h}_t = \varphi(W(r_t \odot h_{t-1}) + U \cdot x_t + b), \qquad h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \widetilde{h}_t$$

We end up with

Definition

- $r_t = \sigma log_0^c(W_c^r h_{t-1} \oplus_c U^r \otimes_c x_t \oplus_c b^r)$ (similar for z^t)
- $\widetilde{h}_t^{\otimes_c} = \varphi(Wdiag(r_t)) \otimes_c h_{t-1})_c U \otimes_c x_t \oplus_c b)$
- $h_t = h_{t-1} \oplus_c diag(z_t) \otimes_c (-h_{t-1} \oplus_c \widetilde{h}_t)$

Note that when c goes to zero, one recovers the usual GRU. Moreover, if $z_t = 0$ or 1, then h_t becomes h_{t1} or $\widetilde{h}_t$ respectively,in a similar as fashion as the usual GRU.

Model Architecture

- Hyperbolic neural network layers stacked manner exactly like standard Euclidean layers. Hyperbolic and Euvlidean layers can be interlocked.
- The authors embed the two sentences using two distinct hyperbolic RNNs or GRUs.

  The sentence embeddings are then fed together with their squared distance (hyperbolic or Euclidean) to a Feed-Forward NN (hyperbolic or Euclidean) which is further fed to an MLR (hyperbolic or Euclidean) that gives probabilities of the classes.
- Authors use cross-entropy loss.
- Optimization w.r.t. hyperbolic parameters based on Riemannian gradients, which are rescaled Euclidean gradients when working in the conformal Poincaré model.
- The highly non-convex spectrum of hyperbolic neural networks sometimes results in convergence to poor local minima, suggesting that initialization is very important.

Datasets & Experiments

Datasets: SNLI

- Largest real dataset for the task of textual entailment.
- Consists of 570K training, 10K validation and 10K test sentence pairs.
- The "contradiction" and "neutral" classes are merged into a single class of negative sentence pairs, while the "entailment" class gives the positive pairs.

	SNLI	PREFIX-10%	PREFIX-30%	PREFIX-50%
FULLY EUCLIDEAN RNN	**79.34** %	89.62 %	81.71 %	72.10 %
HYP RNN+FFNN, EUCL MLR	**79.18** %	96.36 %	**87.83** %	**76.50** %
FULLY HYPERBOLIC RNN	78.21 %	**96.91** %	87.25 %	62.94 %
FULLY EUCLIDEAN GRU	**81.52** %	95.96 %	86.47 %	75.04 %
HYP GRU+FFNN, EUCL MLR	79.76 %	**97.36** %	**88.47** %	**76.87** %
FULLY HYPERBOLIC GRU	**81.19** %	**97.14** %	**88.26** %	**76.44** %

Table 1: Test accuracies for various models and four datasets. “Eucl” denotes Euclidean, “Hyp” denotes hyperbolic. All word and sentence embeddings have dimension 5. We highlight in **bold** the best baseline (or baselines, if the difference is less than 0.5%).

Datasets: PREFIX

- Proof-of-concept task of *detection of noisy prefixes*.
- Synthetic family of datasets PREFIX-Z (for Z being 10%, 30% or 50%).
- For each random first sentence of random length at most 20 and one random prefix of it, a second positive sentence is generated by randomly replacing Z% of the words of the prefix, and a second negative sentence of same length is randomly generated.

	SNLI	PREFIX-10%	PREFIX-30%	PREFIX-50%
FULLY EUCLIDEAN RNN	**79.34** %	89.62 %	81.71 %	72.10 %
HYP RNN+FFNN, EUCL MLR	**79.18** %	96.36 %	**87.83** %	**76.50** %
FULLY HYPERBOLIC RNN	78.21 %	**96.91** %	87.25 %	62.94 %
FULLY EUCLIDEAN GRU	**81.52** %	95.96 %	86.47 %	75.04 %
HYP GRU+FFNN, EUCL MLR	79.76 %	**97.36** %	**88.47** %	**76.87** %
FULLY HYPERBOLIC GRU	**81.19** %	**97.14** %	**88.26** %	**76.44** %

Table 1: Test accuracies for various models and four datasets. "Eucl" denotes Euclidean, "Hyp" denotes hyperbolic. All word and sentence embeddings have dimension 5. We highlight in **bold** the best baseline (or baselines, if the difference is less than 0.5%).

Multinomial Logistic Regression Experiment

MLR tested on the task of subtree classification using the WordNet noun tree.

Pre-trained Poincaré embeddings of the WordNet noun hierarchy.

Three variants of MLR are then trained on top of pre-trained Poincaré embeddings:

- Hyperbolic MLR.
- Euclidean MLR applied on hyperbolic embeddings.
- Euclidean MLR applied on embeddings in the tangent space at 0 using the log_0 map.

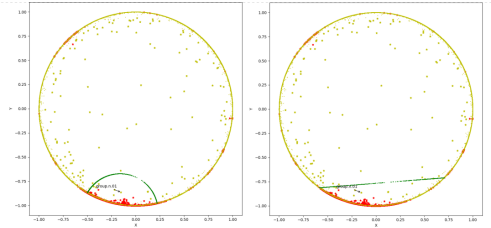

Figure 2: *Hyperbolic (left) vs Direct Euclidean (right) binary MLR used to classify nodes as being part in the* GROUP.N.01 *subtree of the WordNet noun hierarchy solely based on their Poincaré embeddings. The positive points (from the subtree) are in red, the negative points (the rest) are in yellow and the trained positive separation hyperplane is depicted in green.*

Multinomial Logistic Regression Experiment

WordNet subtree	Model	D = 2	D = 3	D = 5	D = 10
animal.n.01 3218 / 798	Hyp	$\mathbf{47.43 \pm 1.07}$	$\mathbf{91.92 \pm 0.61}$	$\mathbf{98.07 \pm 0.55}$	$\mathbf{99.26 \pm 0.59}$
	Eucl	41.69 ± 0.19	68.43 ± 3.90	95.59 ± 1.18	$\mathbf{99.36 \pm 0.18}$
	$\log_{\mathbf{0}}$	38.89 ± 0.01	62.57 ± 0.61	89.21 ± 1.34	98.27 ± 0.70
group.n.01 6649 / 1727	Hyp	$\mathbf{81.72 \pm 0.17}$	$\mathbf{89.87 \pm 2.73}$	$\mathbf{87.89 \pm 0.80}$	$\mathbf{91.91 \pm 3.07}$
	Eucl	61.13 ± 0.42	63.56 ± 1.22	67.82 ± 0.81	$\mathbf{91.38 \pm 1.19}$
	$\log_{\mathbf{0}}$	60.75 ± 0.24	61.98 ± 0.57	67.92 ± 0.74	$\mathbf{91.41 \pm 0.18}$
worker.n.01 861 / 254	Hyp	$\mathbf{12.68 \pm 0.82}$	$\mathbf{24.09 \pm 1.49}$	$\mathbf{55.46 \pm 5.49}$	$\mathbf{66.83 \pm 11.38}$
	Eucl	10.86 ± 0.01	22.39 ± 0.04	35.23 ± 3.16	47.29 ± 3.93
	$\log_{\mathbf{0}}$	9.04 ± 0.06	22.57 ± 0.20	26.47 ± 0.78	36.66 ± 2.74
mammal.n.01 953 / 228	Hyp	$\mathbf{32.01 \pm 17.14}$	$\mathbf{87.54 \pm 4.55}$	$\mathbf{88.73 \pm 3.22}$	$\mathbf{91.37 \pm 6.09}$
	Eucl	$\mathbf{15.58 \pm 0.04}$	44.68 ± 1.87	59.35 ± 1.31	77.76 ± 5.08
	$\log_{\mathbf{0}}$	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 $\log_{\mathbf{0}}$ denotes applying Euclidean MLR in the tangent space at $\mathbf{0}$, after projecting all hyperbolic embeddings there with $\log_{\mathbf{0}}$.

Results

- On SNLI, Hyperbolic NNs perform similarly as with their Euclidean counterparts.
- Hyperbolic and Euclidean MLR are on par when used in conjunction with hyperbolic sentence embeddings.
- Hyperbolic RNNs and GRUs have the most significant improvement over their Euclidean variants when the underlying data structure is more tree-like, e.g. for PREFIX-10%. As soon as the underlying structure diverges more and more from a tree, the accuracy gap decreases.

Nota Bene: The fully Euclidean baseline models might have an advantage over hyperbolic baselines because more sophisticated optimization algorithms such as Adam do not have a hyperbolic analogue at the moment.

Conlusions

1. MLR, FFNNs, RNNs or GRUs can be generalized in a principled manner to all spaces of constant negative curvature combining Riemannian geometry with the theory of gyrovector spaces.
2. Hyperbolic models outperform or are on par with their corresponding Euclidean architectures on sequential data with implicit hierarchical structure.