Hyperbolic Deep Neural Networks: A Survey

Wei Peng, Tuomas Varanka, Abdelrahman Mostafa, Henglin Shi, and Guoying Zhao, *Fellow, IEEE*

Abstract—Recently, hyperbolic deep neural networks (HDNNs) have been gaining momentum as the deep representations in the hyperbolic space provide high fidelity embeddings with few dimensions, especially for data possessing hierarchical structure. Such a hyperbolic neural architecture is quickly extended to different scientific fields, including natural language processing, single-cell RNA-sequence analysis, graph embedding, financial analysis, and computer vision. The promising results demonstrate its superior capability, significant compactness of the model, and a substantially better physical interpretability than its counterpart in the euclidean space. To stimulate future research, this paper presents a comprehensive review of the literature around the neural components in the construction of HDNN, as well as the generalization of the leading deep approaches to the hyperbolic space. It also presents current applications of various tasks, together with insightful observations and identifying open questions and promising future directions.

Index Terms—Neural networks on Riemannian manifold, hyperbolic neural networks, Poincaré model, Lorentz model

1 Introduction

Data with tree structure and hierarchy are ubiquitous in natural scenarios and the real-world [1], [2], [3], [4]. Lots of such data, e.g., Internet [5], brain networks [6], the world trade network [7], and financial networks [8], exhibit a highly non-euclidean latent anatomy and negatively curved properties [9], [10], [11]. Similar properties can be observed from tasks such as recommendation system [12], social media analysis [13], knowledge graph embedding [14], single-cell RNA-sequence analysis [15] and image retrieval [16]. In parallel, current machine learning, especially deep learning [17] has provided a powerful data-driven way to analyze and understand data. At its core lies the expectation of learning low-dimensional and semantically rich representations of data. Deep neural networks, constructed with a multi-layered structure, parameterized with millions of parameters, and boosted with comprehensive and highly-optimized deep learning libraries [18], [19], [20], theoretically have the potential to fit any complex functions, leading to the domination of many research fields, such as image classification [21], [22], machine translation tasks [23], and even video games playing [24].

Nevertheless, in such successful applications, regular grid data in the euclidean space, e.g., text and images, is the main focus. Besides, the learning process is conducted in the intuition-friendly euclidean space, which is a flat space with zero curvature. However, neural architectures operating in the euclidean space rely heavily on the locality and are originally designed for grid data, thus not necessarily providing the most powerful or meaningful geometrical representations for structured data in a non-euclidean space. Typically, the non-euclidean spaces including the elliptic space (a sphere) with a constant positive curvature [25] and the hyperbolic space with constant negative (sectional) curvature [26] should be considered. Neural networks in the elliptic space [27], [28], [29], [30], using spherical harmonic transform or Laplacian-based graph convolution, have been successfully applied to spherical signals. Analogous to this, there is a strong expectation to construct neural networks in the hyperbolic space for data possessing hierarchies, as hierarchies can be represented in such space with low distortion [31].

Apart from the structured data, the underlying relationships between regular samples and important developmental progresses can be also modeled by hierarchy in the hyperbolic space. From the perspective of cognitive science, it is widely accepted that human beings use hierarchy to organize actions [32] and object categories [33], [34], [35]. An interesting study [36] finds that both natural odors and human perceptual descriptions of smells can be described using a three-dimensional hyperbolic space. In single-cell analysis, the cell developmental processes show strong hierarchical relationships and can be described by a hyperbolic space [15]. Zhou *et al.* [37] also found that genetic variation and their expression can be modeled by a low-dimensional hyperbolic geometry. Interestingly, current study [16] even finds that the features of mini-ImageNet [38] and CIFAR [39], which are learned in the euclidean space using neural architectures like VGG [40], Inception [41], and Resnet [21], have apparent hyperbolic properties. Therefore, it is necessary and advantageous to construct hyperbolic deep neural networks (HDNNs) to efficiently deal with far more complex irregular data, and interpret and reason more complex relations beyond euclidean space, thus producing success in the hyperbolic space [31], [42].

Recently, numerous HDNN architectures [4], [43], [44], [45], [46], [47], [48], [49], [50] are developed to solve a variety of different machine learning tasks. Hyperbolic spaces have been proposed as an alternative continuous approach to learn hierarchical representations for data from textual [51],

• *The authors are with the Center for Machine Vision and Signal Analysis, University of Oulu, 90570 Oulu, Finland. E-mail: {wei.peng, Tuomas.Varanka, Abdelrahman.Mostafa, henglin.shi, guoying.zhao}@oulu.fi.*

Manuscript received 10 Feb. 2021; revised 15 Dec. 2021; accepted 16 Dec. 2021. Date of publication 21 Dec. 2021; date of current version 3 Nov. 2022.
This work was supported in part by the Academy of Finland for ICT 2023 Project under Grant 328115, in part by Academy Professor Project EmotionAI under Grants 336116 and 345122, in part by Project MiGA under Grant 316765, and in part by Infotech Oulu.
(Corresponding author: Guoying Zhao.)
Recommended for acceptance by J. M. Solomon.
Digital Object Identifier no. 10.1109/TPAMI.2021.3136921

[52], [53] and graph-structured data [46], [48], to biology [15], [37] and images [16], [54]. The reason for this interest is that the hyperbolic metric approximates the exponential expansion of possible states of the system described by a hierarchical tree-like process. The negative-curvature of the hyperbolic space results in drastically different geometric properties, which makes the circle circumference ($2\sinh r$) and disc area ($2\pi(\cosh r-1)$) grow exponentially with radius r, as opposed to the euclidean spaces where they only grow linearly and quadratically. Therefore, hyperbolic spaces have recently gained momentum to model data in the space that exhibits certain desirable geometric hierarchical characteristics. To summarize, there are several potential advantages of utilizing hyperbolic deep neural networks to represent data:

- A better generalization capability of the model, with less overfitting, computational complexity, and requirement of training data.
- Reduction in the number of model parameters and embedding dimensions.
- A low distortion embedding, which preserves the local and global geometric information.
- A better model understanding and interpretation, which can provide a conformal mapping to euclidean space such that it is friendly to down-stream tasks.

Although constructing neural networks in hyperbolic spaces has gained considerable attention recently, this is an extremely challenging task as euclidean neural operators will not consistently work in the space. Generalizing euclidean operations, from basic arithmetic operations, e.g., additions and multiplications, to neural operators like convolutions and poolings, to the hyperbolic space is also arduous. To the best of our knowledge, a survey of hyperbolic deep neural networks does not exist in this field. This article makes the first attempt and aims to provide a comprehensive review of the literature around hyperbolic deep neural networks for machine learning tasks. Our goals are to 1) provide a concise context and explanation to enable the reader to become familiar with the basics of hyperbolic geometry, 2) review the current literature related to algorithms and applications about hyperbolic deep learning, and 3) identify open questions and promising future directions.

The article is organized as follows. In Section 2, we introduce the fundamental concepts about hyperbolic geometry, making the paper self-contained. Section 3 introduces the generalization of important euclidean neural components to the hyperbolic space. We then review the constructions for hyperbolic deep neural networks, including building networks on two commonly used hyperbolic models, Lorentz Model and Poincaré Model in Section 4. In Section 5, we describe applications for testing hyperbolic deep neural networks and discuss the performance of different approaches under different settings. Finally, in Section 6 we identify open problems and possible future research directions.

2 Hyperbolic Geometry

2.1 Mathematical Preliminaries

Manifold. A manifold $\mathcal{M}$ of dimension n is a topological space of which each point's neighborhood can be locally approximated by the euclidean space $\mathbb{R}^n$.

Tangent Space. For each point $x \in \mathcal{M}$, the tangent space $\mathcal{T}_x\mathcal{M}$ of $\mathcal{M}$ at x is defined as an n-dimensional vector-space approximating $\mathcal{M}$ around x at a first order.

Riemannian Metric. The metric tensor gives a local notion of angle, length of curves, surface area, and volume. For a manifold $\mathcal{M}$, a Riemannian metric $\mathfrak{g}$ is a smooth family of inner products on the associated tangent space: $\langle\cdot,\cdot\rangle_x : \mathcal{T}_x\mathcal{M}\times \mathcal{T}_x\mathcal{M} \rightarrow \mathbb{R}$. A given smooth manifold can be equipped with many different Riemannian metrics.

Riemannian Manifold. A Riemannian manifold [55] is then defined as a manifold equipped with a group of Riemannian metrics $\mathfrak{g}$, which is formulated as a tuple $(\mathcal{M}, \mathfrak{g})$ [56].

Geodesics. Geodesics is the the generalization of a straight line in the euclidean space. It is the constant speed curve giving the shortest (straightest) path between pairs of points.

Exponential Map. The exponential map takes a vector $v \in \mathcal{T}_x\mathcal{M}$ of a point $x \in \mathcal{M}$ to a point on the manifold $\mathcal{M}$, i.e., $\mathrm{Exp}_x : \mathcal{T}_x\mathcal{M} \rightarrow \mathcal{M}$ by moving a unit length along the geodesic uniquely defined by $\gamma(0) = x$ with direction $\gamma^{'}(0) = v$. Different manifolds have their own way to define the exponential maps. Generally, this is very useful when computing the gradient, which provides update that the parameter moves along the geodesic emanating from the current parameter position.

Logarithmic Map. As the inverse of the aforementioned exponential map, the logarithmic map projects a point $z \in \mathcal{M}$ on the manifold to the tangent space of another point $x \in \mathcal{M}$, which is $\mathrm{Log}_x : \mathcal{M} \rightarrow \mathcal{T}_x\mathcal{M}$. Like the exponential map, there are different logarithmic maps for different manifolds.

Parallel Transport. Parallel Transport $\mathcal{PT}_{u\rightarrow v}$ from vector $u \in \mathcal{M}$ to $v \in \mathcal{M}$ defines the transporting of the local geometry along smooth curves that preserves the metric tensors. It is a map from tangent space $\mathcal{T}_u\mathcal{M}$ to $\mathcal{T}_v\mathcal{M}$ that carries a vector in $\mathcal{T}_u\mathcal{M}$ along the geodesic from u to v.

Gromov δ-Hyperbolicity. Gromov δ-hyperbolicity [57], [58] is used to evaluate the hyperbolicity of a dataset/space. Normally, it is defined under four-point condition, say points a, b, c, v. A metric space (X, d) is δ-hyperbolic if there exists a $\delta > 0$ such that these four points in X: $\langle a, b\rangle_v \geq \min\{\langle a, c\rangle_v, \langle b, c\rangle_v\} - \delta$, where the $\langle , \rangle_v$ with respect to a third point v is the Gromov product [31] of two points and it is defined as $\langle a, b\rangle_v = \frac{1}{2}(d(a, v) + d(b, v) - d(a, b))$ with $d(,)$ as the distance function. For instance, euclidean space $\mathbb{R}^n$ is not δ-hyperbolic, Poincaré disc ($\mathbb{B}^2$) is (log3)-hyperbolic.

2.2 Isometric Models in the Hyperbolic Space

The five isometric models [59], [60], i.e., the Lorentz (Hyperboloid) model, the Poincaré ball model, the Poincaré half space model, the Klein model, and the Hemishpere model, are the well-known models of hyperbolic space.[1] They are embedded sub-manifolds of ambient real vector spaces. We detail the most commonly used three models, i.e., the Lorentz, Klein, and Poincaré models, as illustrated in Fig. 1, for constructing hyperbolic deep neural networks. Please refer to the Appendix, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPAMI.2021.3136921, for more details about the other models.

1. We introduce models with constant sectional curvature of -1. This can be easily generalized to other (negative) curvatures.

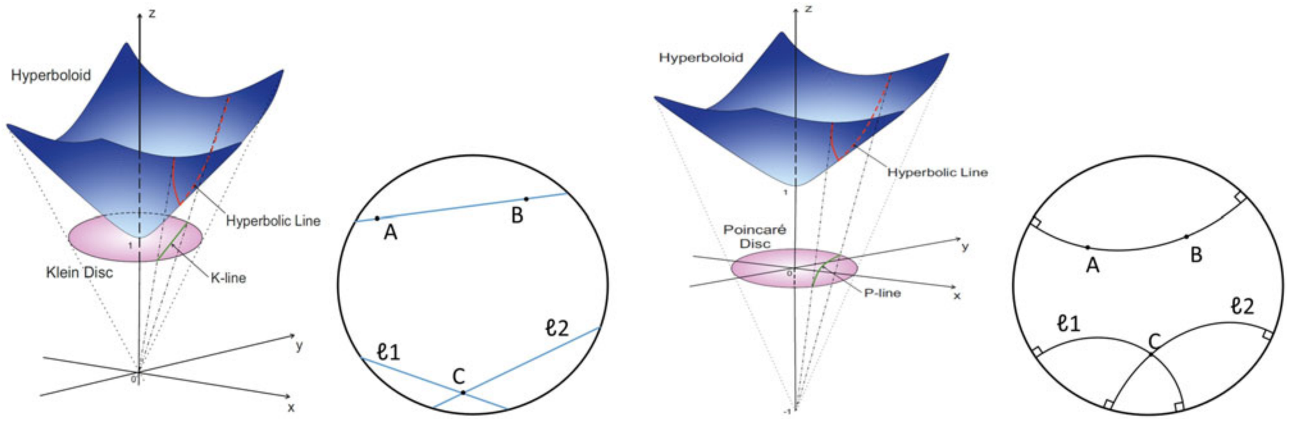


Fig. 1. Illustration of Klein model (left two) and Poincaré model (Right two) in the hyperbolic space. Leftmost: the relationships between Lorentz model and Klein model. We provide the examples of 'straight line' in Klein model (Second from the left). Rightmost: the Poincaré model and the examples of 'straight line' in it. Its relationship with Lorentz model is provided in the second from the right.

2.2.1 Lorentz Model

The Lorentz model $\mathbb{L}^n$ of an n dimensional hyperbolic space is a manifold embedded in the $n+1$ dimensional Minkowski space. The Lorentz model is defined as the upper sheet of a two-sheeted n-dimensional hyperbola with the metric $\mathfrak{g}^{\mathbb{L}}$, which is

$$\mathbb{L}^n = \{x = (x^0, \ldots, x^n) \in \mathbb{R}^{n+1} : \langle x, x\rangle_{\mathbb{L}} = -1, x^0 > 0\}, \quad (1)$$

in which the $\langle , \rangle_{\mathbb{L}}$ represents the Lorentzian inner product

$$\langle x, y\rangle_{\mathbb{L}} = x^T \mathfrak{g}^{\mathbb{L}} y = -x^0 y^0 + \sum_{i=1}^{n} x^i y^i, x \text{ and } y \in \mathbb{R}^{n+1}, \quad (2)$$

where $\mathfrak{g}^{\mathbb{L}}$ is a diagonal matrix with entries of 1s, except for the first element being -1. For any $x \in \mathbb{L}^n$, we can get that $x^0 = \sqrt{1 + \sum_{i=1}^{n} (x^i)^2}$. The distance in the Lorentz Model is defined as

$$d(x, y) = \operatorname{arcosh}(-\langle x, y\rangle_{\mathbb{L}}). \quad (3)$$

The main advantage of this parameterization model is that it provides an efficient space for Riemannian optimization. An additional advantage is that its distance function avoids numerical instability, when compared to Poincaré model, in which the instability arises from the fraction.

2.2.2 Klein Model

Klein model is also known as the Beltrami-Klein model, named after the Italian mathematician Eugenio Beltrami and German mathematician Felix Klein. The Klein model of hyperbolic space is a subset of $\mathbb{R}^n$. As illustrated in Fig. 1. It is the isometric image of the Lorentz model under the stereographic projection [60]. The Klein model is obtained by mapping $x \in \mathbb{L}^{n+1}$ to the hyperplane $x^0 = 1$, using rays emanating from the origin. Formally, the Klein model is defined as

$$\mathbb{K}^n = \{x \in \mathbb{R}^n : ||x|| < 1\}. \quad (4)$$

The distance is

$$d(x, y) = \operatorname{arcosh}\left(1 + \frac{1 - \langle x, y\rangle}{\sqrt{(1 - ||x||^2)(1 - ||y||^2)}}\right). \quad (5)$$

A straight line, e.g., line $\overline{AB}$ in the second figure from the left of Fig. 1, in Klein model is an intersection of a plane with the disk, thus it is still straight like in euclidean space. Therefore, Klein model is commonly used to compute the middle point. This model is not conformal to the euclidean model, which means that angles and circles are distorted,

2.2.3 Poincaré Model

The Poincaré model, as shown in Fig. 1, is given by projecting each point of $\mathbb{L}^n$ onto the hyperplane $x^0 = 0$, using the rays emanating from (-1, 0,..., 0). The Poincaré model $\mathbb{B}$ is a manifold equipped with a Riemannian metric $\mathfrak{g}^{\mathbb{B}}$. This metric is conformal to the euclidean metric $\mathfrak{g}^E = I^n$ with the conformal factor $\lambda_x = \frac{2}{1-||x||^2}$, and $\mathfrak{g}^{\mathbb{B}} = \lambda_x^2 \mathfrak{g}^E$. Formally, an n dimensional Poincaré unit ball (manifold) is defined as

$$\mathbb{B}^n = \{x \in \mathbb{R}^n : ||x|| < 1\}, \quad (6)$$

where $|| \cdot ||$ denotes the euclidean norm. The distance between $x, y \in \mathbb{B}^n$ is defined as

$$d(x, y) = \operatorname{arcosh}\left(1 + 2\frac{||x - y||^2}{(1 - ||x||^2)(1 - ||y||^2)}\right). \quad (7)$$

As illustrated in Fig. 2, the left one compares the hyperbolic distance between two points (curves in blue) with that in the euclidean space (lines in black). Like in the left line group, given a constant hyperbolic distance 0.7, the corresponding euclidean distances decrease dramatically when the points approach the unit boundary. On the contrary, as shown in the right line group, when fix the euclidean distance to 0.36, the hyperbolic distances grow significantly when the points are closed to the border. The hyperbolic distance $d(x, y) \approx 2||x - y||$ when both x and y are closed to the origin with zero norms, which means the model resembles euclidean geometry near the center. As the points move away from the origin and approach the border, the

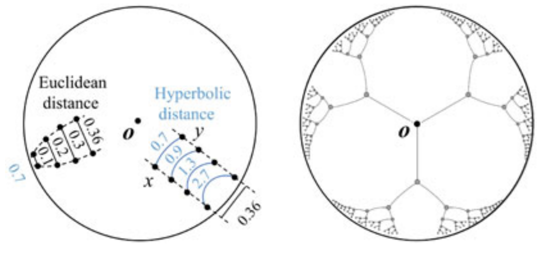


Fig. 2. Illustration of Poincaré Disk (2D Poincaré model), Left: the distances comparison between euclidean space (in black) and hyperbolic space (in blue). Right: an example of modeling a tree using hyperbolic model.

norms are close to one and the distance grows exponentially [61]. As a comparison, for a regular tree with branching factor b, there are $(b+1)b^{l+1}$ nodes at level l. The number of nodes in this tree grows exponentially when the tree goes deeper. Therefore, this exponential growing property of hyperbolic space fits very well with the depth increasing in the tree. Besides, when both x and y are approaching the boundary, the distance between x and y can be $d(x,y) \approx d(O,x) + d(O,y)$, which means the shortest path between x and y approaches the path through the origin O. From the definition of Gromov-product [61], we know that such data has a very small δ value of the Gromov δ-hyperbolicity. This is analogous to the property of trees in which the shortest path between two sibling nodes is the path through their parent. Therefore, the Poincaré disk is very suitable for modeling a tree [62], as shown in the rightmost of Fig. 2.

Since they are isometric models in the hyperbolic space, they can be transferred between each other by mapping functions. Fig. 1 provides the visualizations of their relationships. For Lorentz and Poincaré models, the map can be described as

$$x = (x^0, \ldots, x^n) \in \mathbb{L}^n \Leftrightarrow \left(\frac{x^1}{1+x^0}, \ldots, \frac{x^n}{1+x^0}\right) \in \mathbb{B}^n. \quad (8)$$

Likewise, the Klein model can be transferred from Lorentz model by the projection

$$x = (x^0, \ldots, x^n) \in \mathbb{L}^n \Leftrightarrow \left(\frac{x^1}{x^0}, \ldots, \frac{x^n}{x^0}\right) \in \mathbb{K}^n. \quad (9)$$

3 Generalizing Euclidean Operations to the Hyperbolic Space

Although the use of hyperbolic embeddings (first proposed by Kleinberg *et al.* [63]) in machine learning was already introduced early in 2007, only recently there are methods being extended to deep neural networks. Traditional euclidean neural networks heavily depend on locality, thus cannot directly be applied to hyperbolic space. Fundamental neural operations like linear projection, average pooling, and feature concatenation, will not work, as the outputs would not necessarily lie in the manifold. Therefore, generalizing euclidean operations to the hyperbolic space plays a key role in constructing hyperbolic neural networks.

Constructing deep neural networks in the hyperbolic space is not as easy as it is on the euclidean space. One of the most crucial reasons is that it is the non-trivial or impossible principled generalizations of basic operations, e.g., vector addition, matrix-vector multiplication. Work [43] provides a pioneer study of how classical euclidean deep learning tools can be generalized to the hyperbolic space. Fueled by this, many current works generalize various deep learning operations as it is the key step towards to hyperbolic deep neural networks. In this section, we review the research literature which is trying to generalize operations, e.g., basic addition, mean and neural network layers, to the hyperbolic space.

3.1 Basic Arithmetic Operations

Basic mathematical operations, like addition and multiplication, are fundamental components of neural networks. They are everywhere in the neural network components, like in convolutional filters, fully connected layers, and activation functions.

One simple way to perform these computations is to approximate them by employing the tangent space. However, as observed in some works [47], [64], the approximation in the tangent space can have a negative impact on the learning process. Specifically, the procedures follow a manifold-tangent manifold scheme, basically by transforming features between hyperbolic spaces and tangent spaces via the logarithmic and exponential maps, in which the logarithmic and exponential maps require a series of hyperbolic and inverse hyperbolic functions. However, the mapping between the manifold and the tangent space is only locally diffeomorphic, which may distort the global structure of the hyperbolic manifold [65], [66]. Besides, the compositions of these functions are complicated and usually range to infinity, significantly weakening the stability of models.

Another good choice is the Gyrovector space [67], which is a generalization of euclidean vector spaces to models of hyperbolic space based on Möbius transformations. Specifically, for a model $\mathbb{B}$, the Gyrovector space provides a non-associatitative algebraic formulation for studying hyperbolic geometry, in analogy to the way vector spaces are used in euclidean geometry. In the Gyrovector space, the *Möbius addition* $\oplus$ for x and y in model $\mathbb{B}$ is defined as

$$x \oplus y = \frac{(1 + 2\langle x, y\rangle + ||y||^2)x + (1 - ||x||^2)y}{1 + 2\langle x, y\rangle + ||x||^2||y||^2}. \quad (10)$$

This is a generalization of the addition in euclidean space. $x \oplus y$ will recover to $x + y$ when the curvature goes to zero. In addition, the Möbius subtraction $\ominus$ is simply defined as: $x \ominus y = x \oplus (-y)$.

Then the *Möbius scalar multiplication* $\otimes$ is defined as

$$r \otimes x = \begin{cases} \tanh(r \operatorname{artanh}(||x||) \frac{x}{||x||}, & x \in \mathbb{B}^n \\ 0, & x = 0, \end{cases} \quad (11)$$

where r is a scalar factor. In fact, all above-mentioned operations can also be conducted in the tangent space by using the exponential and logarithmic maps. As provided by [44], the *Möbius scalar multiplication* can be obtained by projecting x in the tangent space at 0, multiplying this projection by the scalar r in the tangent space. Then projecting it back on the manifold with the exponential map, which means

$$r \otimes x = \mathrm{Exp}_0(r \mathrm{Log}_0(x)). \quad (12)$$

With a similar strategy, the authors derived the *Möbius vector multiplication* $M^{\otimes}(x)$ between the matrix M and input x, which is defined as

$$M^{\otimes}(x) = \tanh\left(\frac{||Mx||}{||x||}\operatorname{actanh}(||x||)\right)\frac{Mx}{||Mx||}. \tag{13}$$

Based on the Möbius tranformations, the authors [43] also derived a closed-form expression of Möbius exponential and logarithmic maps for the Poincaré model. For a vector $v \in \mathcal{T}_x\mathcal{M}$ in the tangent space, the exponential map is defined as

$$\operatorname{Exp}_x(v) = x \oplus \left(\tanh\left(\frac{\lambda_x||v||}{2}\right)\frac{v}{||v||}\right), \tag{14}$$

and as the inverse operation of the exponential map, for a point $y \in \mathbb{B}^n$ on the manifold, the logarithmic map is defined as

$$\operatorname{Log}_x(y) = \frac{2}{\lambda_x}\operatorname{artanh}(||-x\oplus y||)\frac{-x\oplus y}{||-x\oplus y||}, \tag{15}$$

where λ_x is the conformal factor, as mentioned in Section 2.2.3.

3.2 Mean in the Hyperbolic Space

The simple but valuable mean computation is one of the most fundamental operations for machine learning approaches. For example, it can be used to build the batch normalization [68] and average pooling [41] in deep neural networks, as well it can be employed to learn the latent distribution, e.g., in variational Auto-Encoder [69], [70], of data (feature). The weighted average counts much for information aggregation in graph convolutional networks (GCNs) [71]. However, unlike in the euclidean space, the mean computation cannot be conducted simply by averaging the inputs, which may lead to a result out of the manifold. Basically, the primary approaches to generalize the mean to hyperbolic space are tangent space aggregation [47], Einstain midpoint method [44], and the Fréchet mean method [64].

Tangential aggregations is one of the most straightforward ways to compute the mean in the hyperbolic space. Generally, the mean aggregation μ in euclidean is defined as a weighted average on the involved neighbors, $\mathcal{N}(i)$, of a center i, which is

$$\mu = \sum_{j\in\mathcal{N}(i)} w_j x_j. \tag{16}$$

However, directly utilizing Eq. (16) in the hyperbolic space is not reasonable since the resulting average may be out of the manifold. Work [47] turns to the tangent space and an attention based aggregation is proposed to compute the aggregated information. Specifically, given the corresponding hyperbolic feature representation, one can compute the attention weights w_j first, then the mean (aggregated information) μ is

$$\mu = \operatorname{Exp}_x\left(\sum_{j\in\mathcal{N}(i)} w_j \operatorname{Log}_x(x_j)\right). \tag{17}$$

Work [44] proposes to compute it with *Einstein midpoint*. Einstein midpoint is an extension of the mean operation to hyperbolic spaces, which has the most concise form in the Klein coordinates. The Einstein midpoint of N samples is defined as

$$\mu = \frac{\sum_{i=1}^N \gamma_i x_i}{\sum_{i=1}^N \gamma_i}, \tag{18}$$

in which the x_i is the i-th sample represented using coordinates in Klein model. The $\gamma_i = \frac{1}{\sqrt{1-||x_i||^2}}$ are the Lorentz factors. One can easily execute midpoint computations by simply projecting to the Klein model from various models of hyperbolic space since all of them are isomorphic. For instance, from Eqs. (8) and (9), the transition between the Poincaré ($x_{\mathbb{B}}$) and Klein ($x_{\mathbb{K}}$) models can be derived as $x_{\mathbb{K}} = \frac{2x_{\mathbb{B}}}{1+||x_{\mathbb{B}}||^2}$. Therefore, based on the Einstein midpoint in Eq. (18), we can get the mean in Poincaré model as

$$\mu_{\mathbb{B}} = \frac{\mu}{1+\sqrt{1-||\mu||^2}}. \tag{19}$$

There is also a closed-form expression for Poincaré model to compute the average (midpoint) in the Gyrovector spaces. Work [72] defines a gyromidpoint, which is

$$m(x_{(1)},\ldots,x_{(N)};\alpha) = \frac{1}{2}\oplus\left(\sum_{i=1}^N \frac{\alpha_i\gamma_i}{\sum_{j=1}^N \alpha_j(\gamma_j-1)}x_{(i)}\right), \tag{20}$$

with $\alpha = (\alpha_1,\ldots,\alpha_N)$ as the weights for each sample $x_{(i)}$ and $\gamma_i = \frac{2}{||x_{(i)}||^2}$.

Recently, using the *Fréchet mean* [73], Luo *et al.* derives a closed-form gradient expression for the mean on Riemannian manifolds [64]. There are considerable years for generalizing euclidean mean in non-euclidean geometries [73], using Fréchet mean. However, the Fréchet mean does not have a closed-form solution, and its computation involves the argmin operation that cannot be easily differentiated. Besides, as mentioned by work [64], computing the Fréchet mean relies on some iterative solvers, which is computationally inefficient, numerical unstable thus not friendly to deep neural networks. Therefore, the authors derived an optimization objective for mean (and variance) computation in the hyperbolic space, which is

$$\mu_{fr} = \arg\min_{\mu\in\mathcal{M}} \frac{1}{N}\sum_{i=1}^N d(x_i,\mu)^2, \tag{21}$$

$$\delta_{fr}^2 = \min_{\mu\in\mathcal{M}} \frac{1}{N}\sum_{i=1}^N d(x_i,\mu)^2. \tag{22}$$

However, the general formulation of Fréchet mean requires an argmin operation and offers no closed-form solution, thus both computation and differentiation are problematic. Inspired by work [74], the authors provided its generalization which allows to differentiate the argmin operation on the manifold. Therefore, they provided their closed-form of the Fréchet mean.

Based on the aforementioned various ways to compute the mean, fundamental neural components, e.g., average pooling layer, batch normalization layer, can be constructed.

3.3 Concatenation and Split Operations

Concatenation and split are commonly used in current deep neural networks, e.g., feature fusion in multimodal learning, operations like GCN filters, and attention mechanism [23]. Simple as they are, they cannot directly be applied in the hyperbolic space as the operations are not manifold preserving ones. For example, points $(0.99, 0)$ and $(0, 0.99)$ are both on the Poincaré disk. However, the resultant point $(0.99, 0, 0, 0.99)$ from concatenation are surly not on a Poincaré model as its norm is larger than one. Therefore, new approaches should be provided for concatenation and split in the hyperbolic space.

As previously described operations, the hyperbolic concatenation and split can be obtained by using the tangent space. Specifically, for an n-dimensional feature embedding $x \in \mathbb{B}^n$ (Note this can be easily generalized to other hyperbolic models) in the hyperbolic space, it can be split into N feature representations V

$$V = \{v_1 \in \mathbb{R}^{n_1}, \ldots, v_N \in \mathbb{R}^{n_N}\} = \mathrm{Log}_0(x), \tag{23}$$

subject to $\sum_{i=1}^{N} n_i = n$. Then, the tangent vector can be mapped to the hyperbolic space using the exponential map. Likewise, for N parts feature representation V in the hyperbolic space, the tangent space can also be used to perform concatenation, which is

$$x = \mathrm{Exp}_0(\mathrm{Log}_0(v_1)|\mathrm{Log}_0(v_2), \ldots, |\mathrm{Log}_0(v_N)), \tag{24}$$

where $|$ denotes the concatenation operation in the tangent space and v_i represents one feature in the hyperbolic space.

However, as pointed out by work [4], merely splitting the coordinates will lower the norm of the output Gyrovectors, which will limit the representational power. Therefore, work [4] proposes a β-split and β-concatenation, as an analogy to the generalization criterion in euclidean neural networks [75].

The β-split and β-concatenation provided by [4] introduce a scalar coefficient $\beta_n = B(\frac{n}{2}, \frac{1}{n})$, where B is the Beta distribution. With this scalar coefficient, the tangent vectors are scaled before being projected back to the hyperbolic space. Therefore β-split is

$$V = \{\mathrm{Exp}_0(\beta_{n_1}\beta_n^{-1}v_1), \ldots, \mathrm{Exp}_0(\beta_{n_N}\beta_n^{-1}v_N)\}. \tag{25}$$

For the β-concatenation

$$x = \mathrm{Exp}_0(\beta_n\beta_{n_1}^{-1}v_1|\beta_n\beta_{n_2}^{-1}v_1, \ldots, |\beta_n\beta_{n_N}^{-1}v_N). \tag{26}$$

Work [43] presents another way to perform vector concatenation in the hyperbolic space, by introducing linear projection functions based on the Möbius transformations. Specifically, for a set of hyperbolic representations $\{x_1 \in \mathbb{B}^{n_1}, \ldots, x_N \in \mathbb{B}^{n_N}\}$, a group of projection functions $\{M_1 \in \mathbb{B}^{n,n_1}, \ldots, M_N \in \mathbb{B}^{n,n_N}\}$ is introduced. Then the concatenated result is

$$x = M_1 \otimes x_1 \oplus, \ldots, \oplus M_N \otimes x_N. \tag{27}$$

However, compared to the previous β methods, this one applies the Möbius transformations (addition and multiplication) many times, as mentioned by [4], which incurs a heavy computational cost and an unbalanced priority in each input sub-Gyrovector.

3.4 Fully-Connected Layers

Fully-connected layer (FC), or linear transform layer, defined as $y = Ax + b$, is also one important component of deep neural networks, in which all inputs from one layer are connected to every activation unit of the next layer. With analogy to euclidean FC layers, works [4], [43] generalized it to the hyperbolic space. In work [43], fully-connected layers are constructed by Matrix-vector multiplication

$$y = A \otimes x \oplus b = \mathrm{Exp}_0(A\mathrm{Log}_0(x)) \oplus b. \tag{28}$$

Here, the bias translations can be further conducted by Möbius translation, which first maps the bias to the tangent space of origin and then parallel transports it to the tangent space of the addend, finally maps back the result to the manifold, which means

$$x \oplus b = \mathrm{Exp}_0(\mathcal{PT}_{0\to x}(\mathrm{Log}_0(b))) = \mathrm{Exp}_x\left(\frac{\lambda_0}{\lambda_x}\mathrm{Log}_0(b)\right), \tag{29}$$

in which λ_x is the conformal factors defined in Section 2.2.3. However, work [4] points out that with the Möbius translation, such a surface is no longer a hyperbolic hyperplane. Besides, the shape of the contour surfaces is determined since the norm of each row vector a_k and bias b contribute to the total scale and shift. To deal with this problem, work [4] provides a Poincaré FC layer based on a linear transformation. They argued that the circular reference between a_k and q_{a_k,r_k} can be unraveled by considering the tangent vector at the origin, $z_k \in \mathcal{T}_0\mathbb{B}^n$, from which a_k is parallel transported. In this way, a new linear transformation is formulated as

$$v_k(x) = 2||z_k||\mathrm{arsinh}\left(\lambda_x\langle x, \frac{z_k}{||z_k||}\rangle\cosh(2r_k) - (\lambda_x - 1)\sinh(2r_k)\right), \tag{30}$$

where z_k is the generalization of the parameter a_k in A. Based on this, they provided their FC layer, which is

$$Y = \frac{w}{1 + \sqrt{1 + ||w||^2}}, \tag{31}$$

where $w = (\sinh(v_k(x)))$. In this way, we avoid using the tangent space to approximate the hyperplane such that can more properly make use of the hyperbolic nature.

3.5 Convolutional Neural Network Operations

There is limited research about convolutional layers in this space. Work [16] proposes to address the common computer vision tasks, e.g., image classification and person reidentification, using hyperbolic geometry. However, only the decision hyperplanes are established in the hyperbolic space. Thus, the authors did not generalize CNN to hyperbolic space.

Basically, the generalization of a CNN can also be simply conducted by using the tangent space. However, as pointed out by [46], stacking multiple CNNs in the tangent space may collapse to a vanilla euclidean CNN. This is because the exponential map at kth layer would have been cancelled by the logarithmic map at next layer. This can be avoided by either applying activation function after the exponential map or adding bias b using the parallel transport.

However, the advantages of hyperbolic geometry may not be well adapted if only using the tangent space. Work [4] provides a novel method to bridge this gap. By using the β-concatenation and the Poincaré fully-connected (FC) layer, the authors presented a method to build the convolutional layer. In particular, given a C-channel input tensor $x \in \mathbb{B}^{C\times W\times H}$ on the Poincaré ball, for each of the $W \times H$ feature pixels, the representations in the reception field of a convolutional filter with size K are concatenated into a single vector $x \in \mathbb{B}^{nK}$, using the β-concatenation. Then naturally, a Poincaré FC layer, which will be detailed in Section 3.4, can be employed to transfer the feature on the manifold. Let $C^{'}$ be the output channels of the CNN layer, then there will be $C^{'}$ groups of such transformations.

3.6 Recurrent Neural Network Operations

Recurrent neural networks (RNNs) [76], [77] are commonly used in sequence learning tasks. Formally, a RNN can be defined by

$$h_{t+1} = \delta(Wh_t + Ux_t + b), \tag{32}$$

where h_{t+1} is the hidden state of the next step, which is updated using the current hidden state h_t and input x_t. δ is a non-linear function. W and U are learnable parameters, and b is the corresponding bias. Work [43] generalizes the RNN to the hyperbolic space, leveraging the Möbius operations in Gyrovector space. The RNN in the hyperbolic space can be defined by

$$h_{t+1} = \delta^{\otimes}(W \otimes h_t + U \otimes x_t \oplus b), \tag{33}$$

where $\oplus$ and $\otimes$ are the generalization of original $+$ and $\times$ in Gyrovector space, as defined in Section 2. The authors also extended the same idea into the gated recurrent unit (GRU) architecture [78], with the same strategy.

Existing works are limited to the Poincaré model with the corresponding operations defined in the Gyrovector space. However, these kinds of operations are always costly when compared to the euclidean counterparts. Future works can explore more efficient ways and extend to other hyperbolic models.

3.7 Activation Function

Activation function provides a non-linear projection of the feed-in features such that much richer semantic representation can be learned. The expected activation function for hyperbolic model should be the non-linear function which as well preserve the manifold. Fortunately, for some hyperbolic models, e.g., Poincaré and Klein models, the manifold is defined only by the norm constraints. Therefore, the activation function is manifold preserving once it is a norm decreasing operation. Hence, any pointwise euclidean activation functions which do not increase the norm can be directly applied to these models. For example, Liu *et al.* [46] directly applied the norm decreasing ReLU [79] and leaky ReLU [80] activation functions in the Poincaré model. However, manifold preserving activation functions are different for different manifolds. For Lorentz model, as the origin in Poincaré model is the pole vector in Lorentz model, these activation functions cannot be applied as they will depart the point away from the manifold. As Lorentz model is isometric to Poincaré model, work [46] maps the point from Lorentz model to Poincaré model, conducts the above mentioned euclidean activation functions, and then maps feature back to the original model.

Ganea *et al.* proposed a Möbius version of projection function [43], which can also be employed to activation functions. In particular, for a function $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$, its Möbius version $f^{\otimes}$ is

$$f^{\otimes}(x) = \mathrm{Exp}_0(f(\mathrm{Log}_0(x))). \tag{34}$$

The authors utilized the tangent space of the origin to perform the function f. A hyperbolic activation function can be also realized by this way, in which case, the input dimension n is equal to the output dimension m. Following the same idea, work [47] provides a similar activation function for graph convolutional networks. The only difference is that they considered the curvatures of different layers. Thus, the logarithmic map and exponential map are defined at the point origins in the manifold with different curvatures.

Interestingly, work [4] removes the activation functions, since the authors thought that the operation on the manifold itself is non-linear, which obviates the need for activation functions.

3.8 Batch Normalization

Batch Normalization (BN) [68] limits the internal covariate shift by normalizing the activation of each layer. It is commonly used to speed up the training procedure of neural networks, as well as to make the training process more stable. The basic idea behind is normalizing of the feature representations by re-centering and re-scaling. Specifically, given a batch of m data-points, The BN algorithm will first compute the mean μ of this batch. Based on μ, the mini-batch variance σ is also computed. Then, two learnable parameters are introduced, which are the scale parameter γ and the shift parameter β. The input activation x is then re-centered and re-scaled, which is $y = \gamma x + \beta$. Theoretically, this BN operation can be easily generalized to the manifold via transferring to the tangent space. Work [64] provides an alternative based on Fréchet mean [73]. In particular, the authors formulated the Riemannian extension of the standard euclidean Batch Normalization by a differentiable Fréchet mean, as described in Section 3.2.

3.9 Classifiers and Multiclass Logistic Regression

lassifier is an essential component for classification tasks. In the context of deep learning, multiclass logistic regression (MLR) or softmax regression is commonly used to perform multi-class classification in euclidean space. Formally, given K classes, MLR is introduced to predict the probabilities of each class $k \in \{1, 2, 3, \ldots, K\}$ based on the input representation x

$$p(y = k, x) \propto \exp \ (\langle a_k, x\rangle - b), \tag{35}$$

where a_k denotes the normal vector and $b \in \mathbb{R}$ is the scalar shift. Then, the decision hyperplane determined by $a \in \mathbb{R}^n \backslash \{\mathbf{0}\}$ and b is defined by $H_{a,b} = \{x \in \mathbb{R}^n : \langle a, x\rangle - b = 0\}$. Note that $\exp$ is the exponential function, not the manifold map function Exp. According to [81], the MLR can be reformulated as

$$p(y = k, x) \propto \exp \ (sign(\langle a_k, x\rangle - b)||a_k||d(x, H_{a_k,b_k})), \tag{36}$$

where $d(x, H_{a_k,b_k})$ is the euclidean distance between x and the hyperplane H_{a_k,b_k}. To further generalize it to the hyperbolic space, work [43] re-parameterizes the scalar term $b \in \mathbb{R}$ with a new set of parameters $p_k \in \mathbb{R}^n$, by which they reformulated the hyperplane: $\hat{H}_{a,p} = \{x \in \mathbb{R}^n : \langle a, x - p\rangle = 0\}$, and $\hat{H}_{a,p} = \hat{H}_{a,\langle a,p\rangle}$. In this way, the MLR is rewritten as

$$p(y = k, x) \propto \exp \ (sign(\langle a_k, x - p_k\rangle)||a_k||d(x, \hat{H}_{a_k,p_k})). \tag{37}$$

Then, the definition of the hyperbolic setting is simply achieved by replacing the addition $+$ with Möbius addition $\oplus$.

However, this causes an undesirable increase in the parameters from $n + 1$ to $2n$ in each class k. As pointed out by [4], there is no need to introduce countless choices of p_k to determine the same discriminative hyperplane. Instead, they introduced another scalar parameter $r_k \in \mathbb{R}$ such that the bias vector $q_{a_k,r_k} = r_k \frac{a_k}{||a_k||}$ parallels to the normal vector a_k. That is

$$\hat{H}_{a_k,r_k} = \{x \in \mathbb{R}^n | \langle a_k, -q_{a_k,r_k} + x\rangle = 0\} = H_{a_k,r_k||a_k||}. \tag{38}$$

Based on this, the MLR is reformulated as

$$p(y = k, x) \propto \exp(sign(\langle a_k, -q_{a_k,r_k} + x\rangle)||a_k||d(x, \hat{H}_{a_k,r_k})). \tag{39}$$

In addition to the MLRs in the hyperbolic space, there are also works constructing classifiers in this space. Work [82] introduces a hyperbolic formulation of support vector machine classifier (H-SVM). Work [83] provides theoretical guarantees for learning a SVM in the hyperbolic space. Besides, an efficient strategy is introduced to learn a large-margin hyperplane, by the injection of adversarial examples. Please refer to Appendix, available in the online supplemental material, for more details.

3.10 Optimization

Optimizer plays a crucial role in training deep neural networks. It largely influences the convergence of the training process, the training speed, and the final predictive performance. Therefore, generalizing optimizers to hyperbolic space is as important as constructing hyperbolic neural architectures [84]. In terms of stochastic optimizers on Riemannian manifolds, one pioneer work should be the Riemannian stochastic gradient descent (RSGD), provided by Bonnabel *et al.* [85]. They pointed out that for the standard stochastic gradient descent in $\mathbb{R}^n$, seeking the matrix with certain rank, which best approximates the updated matrix, can be numerically costly, especially for very large parameter matrix. To enforce the rank constraint, a more natural way is to endow the parameter space with a Riemannian metric and perform a gradient step within the manifold of fixed-rank matrices. To address this issue, they proposed to replace the usual update in SGD using an exponential map (Exp) with the following update:

$$W_{t+1} = \mathrm{Exp}_{W_t}(-\alpha g_t), \tag{40}$$

where $g_t \in \mathcal{T}_{W_t}\mathcal{M}$ denotes the Riemannian gradient of the objective at point W_t. The provided algorithm is completely intrinsic, which does not depend on a specific embedding of the manifold or on the choice of local coordinates. For manifolds where Exp function is not known in a closed form, it is common to replace it with a first order approximation [86].

Current deep learning optimization approaches prefer adaptive strategies, e.g., Adagrad [87] and Adam [88], which dynamically incorporates knowledge of the geometry of the data observed in earlier iterations to perform more informative gradient-based learning. Although the input data is with very high dimension, useful features have very different frequencies. When the gradient vectors are sparse, the update of the neural network can often be performed in time proportional to the support of the gradient. One of the current challenges of generalizing the adaptivity of these optimization methods to hyperbolic space is that the manifold does not provide an intrinsic coordinate system, while Riemannian manifold only allows to work in a certain local coordinate systems. Therefore, it is non-trivial to extend the optimizers to hyperbolic space in an intrinsic manner (coordinate-free). Suggested by [89], one solution can be fixing a canonical coordinate system in the tangent space and then parallelly transporting it along the optimization trajectory. However, general Riemannian manifold depends on both the chosen path and the curvature, which will give the parallel transport a rotational component (holonomy). This will break the gradient sparsity and thus harm the benefit of adaptivity. For instance, imagining the vector x is initially sparse, e.g., $x = (2.5, 0, 0)$, with the parallel transport, the vector may be rotated and thus has other components $x = (1.2, 0.4, 0.1)$, which definitely brakes the coordinate-wise updates and sparsity.

To avoid such problem, work [89] represent a n-dimensional manifold $\mathcal{M}$ by a Cartesian product of n manifolds, which means $\mathcal{M} = \mathcal{M}_1 \times \cdots \times \mathcal{M}_n$. Based on this, the authors provided the Riemannian Adagrad, which is defined by

$$W_{t+1} = \mathrm{Exp}_{W_t}(-\alpha g_t) \tag{41}$$

$$W^i_{t+1} = \mathrm{Exp}_{W^i_t}\left(-\frac{\alpha g^i_t}{\sqrt{\sum_{k=1}^t ||g^i_k||^2_{x^i_k}}}\right), \tag{42}$$

where the $||g^i_k||^2_{x^i_k} = \mathfrak{g}(g^i_k, g^i_k)$ is a Riemannian norm. They further extended it to Riemannian Adam by introducing the momentum term and an adaptive term.

4 Hyperbolic Deep Neural Networks

Overwhelming number of studies using deep neural networks [17] ranging from convolutional neural networks

TABLE 1
Summary of the Advanced Machine Learning Methods in the Hyperbolic Space

Method	Year	Architecture	Tasks	G.	Institution	Source
PEmbedding [103]	2017	Embedding	NLP and Graph	$\mathbb{B}$	Facebook	NeurIPS
Coalescent [102]	2017	Embedding	Graph	$\mathbb{B}$	TU-Dresden	Nature
HyperGraph [108]	2017	Embedding	Network Vertex Classification	$\mathbb{B}$	ICL	MLG W
TextHyper [52]	2018	AE(GRUs)	Text Embedding	$\mathbb{B}$	Google	NAACL W
h-MDS [109]	2018	Embedding	Tree and tree-like data modeling	$\mathbb{B}$	Stanford	PMLR
HyperQA [110]	2018	Encoder-Decoder	Neural Question Answering	$\mathbb{B}$	NTU	WSDM
HyperCone [111]	2018	Embedding	NLP and Graph	$\mathbb{B}$	ETH Zürich	ICML
HNN [43]	2018	RNN	NLP(textual entailment and noisy prefixes)	$\mathbb{B}$	ETH Zürich	NeurIPS
Lorentz [45]	2018	Embedding	Taxonomies Embedding, Graph and Historical Linguistics	$\mathbb{L}$	Facebook	ICML
HyperBPR [112]	2018	BPR [113]	Recommender Systems	$\mathbb{B}$	NTU	AAAI
ProductM [114]	2018	Embedding	Tree(with Cycle)	Mixed	Stanford	ICLR
HAN [44]	2019	Attention Module	Graph, Machine translation and Relational Modeling	$\mathbb{L}(\mathbb{K})$	Deepmind	ICLR
HGCN [47]	2019	GNN	Graph	$\mathbb{L}$	Stanford	NeurIPS
PGlove [51]	2019	Glove [105]	Word Embedding	Mixed	ETH Zürich	ICLR
HGNN [46]	2019	GNN	Graph	$\mathbb{B}$,$\mathbb{L}$	Facebook	NeurIPS
Tiling [106]	2019	Tiling	NLP and Compressing	$\mathbb{L}$	Cornell	NeurIPS
PTaxo [115]	2019	Embedding	Taxonomy Induction	$\mathbb{B}$	U. Hamburg	ACL
H-SVM [83]	2019	SVM	NLP(Word E) Graph (Node C)	$\mathbb{B}$	MIT	AISTATS
H-Recom [12]	2019	BPR	Recommender Systems	$\mathbb{L}$	ASOS	CoRR
MuRP [116]	2019	Bilinear	knowledge Graph	$\mathbb{B}$	U. Edinburgh	NeurIPS
RAO [90]	2019	Optimizor	NLP	$\mathbb{B}$	ETH Zürich	ICLR
WrapN [117]	2019	VAE	NLP,MNIST and Atari trajectories	$\mathbb{B}$	U.Tokyo	ICML
LDistance [118]	2019	Embedding	NLP	$\mathbb{B}$	U. Toronto	ICML
PVAE [119]	2019	VAE	NLP, and MNIST	$\mathbb{B}$	Oxford	NeurIPS
CCM-AAE [120]	2019	AE	MNIST C, Graph	Mixed	U.Lugano	ASOC
HWAE [54]	2019	VAE	G MNIST, Graph and Tree	$\mathbb{B}$	ETH Zürich	-
gHHC [121]	2019	Clustering	Clustering ImagNet, Multi-Task	$\mathbb{B}$	U. Mass	KDD
HGAT [122]	2020	GNN	Graph	$\mathbb{B}$	BUPT	AAAI
H-STGCN [48]	2020	GNN	Skeleton Action	$\mathbb{B}$	U. Oulu	ACM MM
HyperKG [14]	2020	Translational	Knowledge Graph	$\mathbb{B}$	EPFL	ESWC
HyperML [123]	2020	Metric Learning	Recommender Systems	$\mathbb{B}$	NTU	WSDM
LorentzFM [124]	2020	Triangle Inequality	Recommender Systems	$\mathbb{B}$	eBay	AAAI
APo-VAE [53]	2020	VAE	NLP (dialog-response generation)	$\mathbb{B}$	Duke	-
H-Image [16]	2020	Embedding	Image c, few-shot	$\mathbb{B}$	SIST	CVPR
HyperText [125]	2020	RNN	NLP(text classification)	$\mathbb{B}$	Huawei	EMNLP
κ-GCN [49]	2020	GNN	Graph	Mixed	ETH Zürich	ICML
FMean [64]	2020	GNN	Graph	$\mathbb{B}$	Cornell	ICML
H-NormF [126]	2020	Norm. Flow	Graph	$\mathbb{B}$	McGill	ICML
κ-Stereographic [127]	2020	GNN	Graph	Mixed	SIST	ICML
L-Group [128]	2020	Group	jet physics	$\mathbb{L}$	U. Chicago	ICML
MVAE [129]	2020	VAE	Image reconstruction and Tree systhsis	Mixed	ETH Zürich	ICLR
HypHC [130]	2020	Clustering	Clustering(*e.g.*, CIFAR-100)	$\mathbb{B}$	Stanford	NeurIPS
R-NormF [131]	2020	Norm. Flow	Earth sciences	$\mathbb{B}$	Oxford	NeurIPS
RH-SVM [84]	2020	SVM	ImageNet(Pick 2classes)	$\mathbb{L}$	Princeton	NeurIPS
TREEREP [132]	2020	Embedding	Tree	$\mathbb{B}$	U. Michigan	NeurIPS
UltraH [133]	2020	Embedding	Graph	Mixed	NVIDIA	NeurIPS
GIL [134]	2020	GNN	Graph	$\mathbb{B}$	CAS	NeurIPS
P-Maps [15]	2020	Embedding	Biology	$\mathbb{B}$	Facebook	Nature
HNN++ [4]	2021	CNN,Transformer	NLP, Clustering, Machine Translation	$\mathbb{B}$	U. Tokyo	ICLR
Hyper-gene [37]	2021	Embedding	Biology	$\mathbb{B}$	Salk	Cell
scPhere [135]	2021	VAE	Biology	$\mathbb{L}$	MIT and Harvard	Nature

Here, G. means the type of ***G****eometry. Its value 'Mixed' means the methods combine more than one different geometries, i.e., euclidean, elliptic, and hyperbolic geometries, to build the model.*

(CNN) [90], recurrent neural networks (RNN) [77], [91], graph neural networks (GNN) [71], [92] to generative models like variational auto-encoder (VAE) [69] have showcased the superiority of various deep neural networks in different research tasks. With the success of all these neural architectures in euclidean space, we have the reason to expect their generalization and performance in the hyperbolic space. Fortunately, we do observe a considerable number of research extending such euclidean architectures to the hyperbolic space, mainly using Poincaré and Lorentz models. This section describes various hyperbolic deep neural architectures. We try our best to collect and summarize all advanced hyperbolic machine learning methods, as illustrated in Table 1. One can find that most of the works are from the prominent conferences/journals, e.g., Nature, Cell, NeurIPS, ICML, and ICLR. Besides, increasing institutions are getting into this potential research field. From the method perspective, the Poincaré model and Lorentz model are prominent hyperbolic models for generalizing neural networks. At the same time, the former one (Poincaré model) is dominated in the hyperbolic deep neural networks. In the following part, we detail different kinds of hyperbolic architectures, including hyperbolic embedding, hyperbolic clustering, hyperbolic attention networks, hyperbolic graph neural networks, hyperbolic normalizing flow, hyperbolic variational auto-encoder, and hyperbolic neural networks with mixed geometries.

4.1 Hyperbolic Embeddings

Embedding [93] data into a standard space makes it possible to use properties of the target space as additional structure in the original dataset, and brings to front information that is hard to detect in the raw input. Embedding has been found relevant to a variety of subjects such as data visualization, network analysis, routing, localization, machine learning, statistics, biology and many others.

There have been many works [63], [94], [95], [96], [97], [98], [99], [100] considering hyperbolic space as an alternative for various embedding tasks. Walter [94] provided a construction of a distance preserving embedding of high-dimensional data into the hyperbolic space for interactive visualization. In Internet routing, Kleinberg [63] presented a constructive proof that every finite, connected, undirected graph has a greedy embedding in two-dimensional hyperbolic space, thus introducing hyperbolic geometry for greedy routing in geographic communication networks. Later, Cvetkovski *et al.* [99] extended it to dynamic graphs, i.e., communication networks whose topology changes over time. Similarly, Shavitt *et al.* [95] embedded the Internet distance metric in a hyperbolic space and by carefully selecting the curvature, they improved the accuracy of Internet distance embedding. Boguñá *et al.* [5] resolved the serious scaling limitations of Internet by the embeddings of the AS (autonomous system) Internet topology in the hyperbolic space to perform greedy shortest path routing. For complex networks, Krioukov [98] showed that heterogeneous degree distributions and strong clustering can emerge by assuming an underlying hyperbolic geometry, thus developed a geometric framework to model complex networks using hyperbolic space. Bläsius *et al.* [100] constructed and implemented a new maximum likelihood estimation algorithm that embeds scale-free graphs in the hyperbolic space. Coalescent embedding [101] was the first model-free unsupervised kernel learning based solution for inferring the graph embedding in the Poincaré disk.

Currently, Nickel *et al.* [102] proposed to learn a Poincaré embedding for symbolic data, while considering the latent hierarchical structures. This work proves that Poincaré embeddings can outperform euclidean embeddings significantly on data with latent hierarchies, in terms of both representation capacity and generalization ability. Following this study, many new embedding methods in the hyperbolic space are proposed. They can be roughly summarized into four categories, i.e., tangent optimized, fully Riemannian optimization, numerical stable embedding, and combinatorially tree embedding (in Appendix, available in the online supplemental material).

Specifically, the Poincaré embedding [102] is trying to find embeddings $\Theta = \{\theta_i\}_{i=1}^{n}$, where $\theta \in \mathbb{B}^d, ||\theta_i|| < 1$ in the unit d-dimensional Poincaré ball for a set of symbols with size of n. Therefore, the optimization problem can be framed as $\Theta^* = argmin_\Theta \mathcal{L}(\Theta)$, where $\mathcal{L}(\cdot)$ is a task-related loss function. For instance, in the hierarchy embedding task, the loss function over the entire dataset $\mathcal{D}$ can be represented as

$$\mathcal{L}(\Theta) = \sum_{(u,v)\in\mathcal{D}} log \frac{e^{-d_{\mathbb{B}}(u,v)}}{\sum_{v'\in\mathcal{N}(u)} e^{-d_{\mathbb{B}}(u,v')}}, \tag{43}$$

where $\mathcal{N}(u)$ denotes a set of negative examples for u. This loss function encourages related objects to be closer to each other than objects without an obvious relationship. The embedding is further optimized utilizing the RSGD with the exponential map, which is a scaled version of the euclidean gradient. Specifically, the retraction operation $\mathcal{R}_w(v) = w + v$ is utilized as the approximation of the exponential map, Exp. Then, a projection, which normalizes the embeddings with a norm bigger than one, is utilized to ensure the embeddings remain within the Poincaré model. Thus

$$W_{t+1} = \text{proj}\left(W_t - \eta \frac{(1 - ||W_t||^2)^2}{4} \triangledown_E\right), \tag{44}$$

where $\triangledown_E$ is the euclidean gradient, and η is the learning rate. The normalization function is $\text{proj}(x) = x/||x||$ if $||x|| \geq 1$, otherwise $\text{proj}(x) = x$. Similarly, work [52] proposes text and sentence embedding with Poincaré model. The authors proposed to re-parametrize the Poincaré embeddings such that the projection step is not required. On top of the existing encoder architectures, e.g., LSTMs, a reparameterization technique is introduced to map the output of the encoder to the Poincaré ball, which is defined as

$$\Theta = \delta(\phi_{norm}(h)) \frac{\phi_{dir}(h)}{||\phi_{dir}(h)||}, \tag{45}$$

where h represents the hidden representation of encoder, and δ denotes the sigmoid function. The function $\phi_{dir} : \mathbb{R}^d \to \mathbb{R}^d$ is used to compute a direction vector. Function $\phi_{norm} : \mathbb{R}^d \to \mathbb{R}$ is a norm function. In this way, the authors mapped the encoder embeddings to the Poincaré ball and the Adam is introduced to optimize the parameters of the encoder.

Work [43] presents hyperbolic entailment cones to especially deal with more complicated connections like asymmetric relations in directed acyclic graphs, thus further improving the performance of hyperbolic embeddings. The authors pointed out that most of the embedding points of the method [102] collapse on the boundary of Poincaré ball. To address this issue, they generalized the idea of order embeddings [103], which views hierarchical relations as partial orders defined on a family of nested geodesically convex cones, into the hyperbolic space.

Specifically, in a vector space, a convex cone S is a set that is closed under non-negative linear combinations, which means for vectors $v_1, v_2 \in S$, then $\alpha v_1 + \beta v_2 \in S, \forall \alpha, \beta > 0$. Since the cones are defined in the vector space, the authors proposed to build the hyperbolic cones using the exponential map, which leads to the definition of S-cone at point x, which is

$$\mathfrak{S}_x = \text{Exp}_x(S), S \in \mathcal{T}_x\mathcal{M}. \tag{46}$$

To avoid heavy cone intersections and scale exponentially with the space dimension, the authors further constructed the angular cones in the Poincaré ball. To achieve this, they introduced the so-called cone aperture functions $\phi(x)$ such that the angular cones $\mathfrak{S}_x^{\phi(x)}$ follow four intuitive properties, including axial symmetry, rotation invariant, continuous of cone aperture function, and the transitivity of nested angular cones. With all of this the authors provided a closed-form of the angular cones, which are

$$\mathfrak{S}_x^{\phi(x)} = \left\{ (\pi - \angle Oxy) \leq \text{arsin} \left(K \frac{1 - ||x||^2}{||x||} \right) \right\}, \tag{47}$$

where O in angle $\angle Oxy$ is the origin, K is a constant. They optimize the objective using *fully Riemannian optimization*, instead of using the first-order approximation as work [102] did.

In addition, Tifrea *et al.* [51] extends an unsupervised word embedding algorithm, Glove [104], to the Riemannian manifolds. They proposed to embed words in a Cartesian product of hyperbolic spaces which they theoretically connected to the Gaussian word embeddings and their Fisher geometry.

Instead of providing new embedding methods in the hyperbolic space, work [105] addresses the *numerical instability* issue of the networks. Like mentioned in work [105], the difficulty is caused by floating-point computation and amplified by the ill-conditioned Riemannian metrics. As points move far away from the origin, the error caused by using floating-point numbers to represent them will be unbounded. For the Poincaré model, the distance changes rapidly when the points are close to the ball boundary such that it is not well conditioned. While for Lorentz model, it is not bounded such that it will experience large numerical error when the points are far away from the origin. Therefore, for the representation in the hyperbolic space, it is desirable to find a method that can represent any point with small fixed bounded error in an efficient way. To this end, work [105] presents a tiling-based model to utilize the integer-lattice square tiling (or tessellation) [106] in the hyperbolic space to construct a constant-error representation. They proved that the representation error, the error of computing distances, and the error of computing the gradient are bounded by a fixed value that is independent of distance to the origin.

4.2 Hyperbolic Cluster Learning

Hierarchical Clustering (HC) [135], which generally constructs a hierarchy over clusters with the form of a multi-layered tree whose leaves correspond to samples and internal nodes correspond to clusters, is a fundamental problem in data analysis, visualization, and mining the underlying relationships. Mainstream methods including bottom-up linkage methods [136], and recently, cost function based methods [137]. However, these methods are either not amenable for stochastic gradient methods or computationally expensive. Gradient-based hyperbolic hierarchical clustering, gHHC [120], a geometric heuristic to provide an approximate distribution over lowest common ancestor (LCA), is proposed over continuous representations of tree in the hyperbolic space (Poincaré model), based on the observation that child-parent relationships can be modelled by the *distances* and *norms* of the embedded node representations. The authors used the norm of vectors to model depth in the tree, requiring child nodes to have a larger norm than their parents. The root is near the origin of the space and leaves near the edge of the ball. Formally, let $Z = \{z_1, z_2, \dots, z_k\}, z_i \in \mathbb{B}^d$ represent the node representation in the d-dimensional Poincaré ball. Then a child-parent dissimilarity function is used to encourage the children have a smaller norm than the parent, which is

$$d_{cp}(\mathcal{T}_c, \mathcal{T}_p) = d_{\mathbb{B}}(z_c, z_p)(1 + max(||z_p|| - ||z_c||, 0)), \tag{48}$$

where z_c, z_p are the hyperbolic embedding of nodes $\mathcal{T}_c, \mathcal{T}_p$. If the norm of the parent node is smaller than the child, then the dissimilarity will just be the distance in the hyperbolic space. Otherwise, this dissimilarity will be bigger than the distance. Then this dissimilarity function is used to model a distribution over the tree structure to encode the uncertainty, which is $P_{par}(\mathcal{T}_p|\mathcal{T}_c, Z) \propto \exp(-d_{cp}(\mathcal{T}_c, \mathcal{T}_p))$, thus the tree distribution over embedding will be

$$P_{par}(\mathcal{T}|Z) \propto \prod_{\mathcal{T}_p} \prod_{\mathcal{T}_c \in children(\mathcal{T}_p)} P_{par}(\mathcal{T}_p|\mathcal{T}_c, Z). \tag{49}$$

In contrast with previous gradient-based approaches, this approach has theoretical guarantees in terms of clustering quality and empirically outperforms agglomerative heuristics.

4.3 Hyperbolic Attention Network

Currently, attention mechanism [23] for various neural networks becomes one of the most attractive research topics. Outstanding architectures include the neural Transformer [23], BERT [138], and even the graph attention networks [121], [139], [140]. While attention mechanisms have become the de-facto standard for NLP tasks, their momentum has continuously been extended to computer vision applications [141]. At its core lies the strategy of focusing on the most relevant parts of the input to make decisions. Different from euclidean space, the distances defined in the hyperbolic space highly depend on their locations (e.g., close to the origin or to the boundary) thus they have their own ways to measure the similarity/dissimilarity, when computing the attention. Therefore, it is important to design hyperbolic attention network such that the correlations can be captured reasonably in terms of the hyperbolic topology and semantic representations.

Work [44] extends it into the hyperbolic space (Lorentz model), utilizing the Lorentzian distance and the Einstein midpoint method to conduct the score matching and aggregation. First, the data representation is organized by a pseudo-polar coordinate, in which an activation $x \in \mathbb{R}^{n+1} = (\mathbf{d}, r)$ is constructed by a n dimensional normalized angle $\mathbf{d}$,$||\mathbf{d}|| = 1$ and a scalar radius r. Then, a well-developed map function is introduced to project the activation to the hyperbolic space, which is $\pi((\mathbf{d}, r)) = (\sinh(r)\mathbf{d}, \cosh(r))$. It is easy to see that the projected point lies in the Lorentz model. Then for hyperbolic matching, the authors took $\alpha(q_i, k_j) = f(-\beta d_{\mathcal{L}(q_i,k_j)} - c)$, in which the negative Lorentzian distance (scaled by $-\beta$ and shifted by c) is utilized to measure the correlation (matching score α). Since there is no natural definition of mean on the manifold, they turn to Einstein midpoint to conduct hyperbolic aggregation. Specifically, the agregated message m_i can be represented as

$$m_i(\{\alpha_{ij}\}_j, \{v_{ij}\}_j) = \sum_j \left[\frac{\alpha_{ij}\gamma(v_{ij})}{\sum_l \alpha_{il}\gamma(v_{il})}\right] v_{ij}, \tag{50}$$

where $\gamma(v_{ij})$ is the Lorentz factor at point v_{ij}, and v_{ij} is defined on the Klein model. On the top of the proposed hyperbolic attention network, the authors further formulated the Hyperbolic Transformer model, which is proved to have the superiority when compared to the euclidean Transformer.

Based on the euclidean graph attention network (GAT) [139], work [121] generalizes it to a hyperbolic GAT. The idea is very simple, just replacing the euclidean operation with Möbius operations, which means the matching score

$$\alpha_{ij} = f(W \otimes h_i, W \otimes h_j). \tag{51}$$

They further defined the f function based on the hyperbolic distance. Just as work [44], the negative of the distance between nodes is utilized as the matching score. The scores are finally normalized using softmax function, otherwise all the scores are negative. The hyperbolic aggregation is simply conducted on the tangent space, as it is done in euclidean space.

Shimizu *et al.* [4] proves that three different kinds of hyperbolic centroids, including the Möbius gyromidpoint [72], Einstein midpoint [72] and the centroid of the squared Lorentzian distance [117], are the same midpoint operations projected on each manifold and exactly matches each other. Based on this observation, they explored on Möbius gyromidpoint and generalized it by extending to the entire real value weights (previously, it is defined under the condition of non-negative weights) by regarding a negative weight as an additive inverse operation. So the centroid with real weights $\{v_i \in \mathbb{R}\}_{i=1}^N$ is

$$\mu = [x_i, v_i] = \frac{1}{2} \oplus \left(\frac{\sum_{i=1}^N v_i \lambda x_i}{\sum_{i=1}^N ||v_i|| \lambda x_i} \right). \tag{52}$$

With the above weights, the authors computed the attention in the hyperbolic space. Given the source and target as sequences of Gyrovectors, first, the proposed Poincaré FC layers (in Section 3.4) are utilized to construct the queries, keys, and values. Then, in order to build the multi-head attention, they are broken down into several parts. Like previous methods, the negative distances are also employed to measure the matching scores. Finally, the message from multi-head is aggregated using the proposed Poincaré weighted centroid. The authors built a hyperbolic set transformer model and compared to its euclidean counterpart [142]. The result shows that the hyperbolic one can at least get equivalent performance, at the same time showing a remarkable stability and consistently converges.

4.4 Hyperbolic Graph Neural Network

Recently, there has been a growing passion of modeling graphs in the hyperbolic space [46], [48], [49]. A core reason for that is learning hierarchical representations of graphs is easier in the hyperbolic space due to the curvature and the geometrical properties of the hyperbolic space. Such spaces were shown by Gromov to be well suited to represent tree-like structures [31] with low distortion as objects requiring an exponential number of dimensions in euclidean space can be represented in a polynomial number of dimensions in the hyperbolic space. As an alternative, graph neural networks (GNNs) [71], [92], [143] are powerful tools for data with non-euclidean graph structures. However, the operations are still in euclidean space, which does not make full use of the geometric property. Current studies of GNN in the hyperbolic space [46], [47], [48] show a superiority, in terms of both model compactness and the predictive performance, when compared to their counterparts in the euclidean space. This suggests the essential of generalizing graph neural networks to hyperbolic space.

GNN can be interpreted as performing message passing between nodes, which can be formulated as

$$h_i^{k+1} = \sigma\left(\sum_{j \in N(i)} A_{ij} W^k h_i^k \right), \tag{53}$$

where h_i^{k+1} represents the hidden representation of the ith node at the $(k+1)$-layer, W^k denotes the weight of the network at k layer. The A_{ij} is the entry of the normalized adjacency matrix A. Eq. (53) performs the information aggregation around the neighbor nodes $N(i)$ of node i to update the representation of this node.

Works [46], [48] provide a straightforward way to extend the graph neural network to hyperbolic space, using tangent space. Work [46] utilizes the logarithmic map $\mathrm{Log}_{x'}$ at a chosen point x', such that the functions with trainable parameters are executed there. Thus, the graph neural operation in a hyperbolic space is

$$h_i^{k+1} = \sigma\left(\mathrm{Exp}_{x'}\left(\sum_{j \in N(i)} A_{ij} W^k \mathrm{Log}_{x'}(h_i^k) \right) \right), \tag{54}$$

where an exponential map $\mathrm{Exp}_{x'}$ is applied afterwards to map the learned feature back to the manifold. The authors moved the activation function σ to the tangent space, since they suggested that otherwise the hyperbolic operation will collapse to the vanilla euclidean GCN as the exponential map will be canceled by the logarithmic map at next layer.

Work [48] shares the same idea to construct spatial temporal graph convolutional networks [144] in the hyperbolic space and applied this for dynamic graph sequences. They further explored the projection dimension in the tangent space, using neural architecture search (NAS) [145]. Chami *et al.* [47] decoupled the message passing procedure of GCN before generalization. The operation of GCN is divided into three parts, including feature transform, neighborhood aggregation, and activating by the activation function. Then, the authors provided operations for corresponding parts in the hyperbolic space.

Bachmann *et al.* [49] presented a novel κ-GCN in the hyperbolic space, which is a mathematically grounded generalization of GCN to constant curvature spaces. Specifically, they extended the operations in Gyrovector space to the space with constant positive curvature, which is the stereographic spherical projection models in their study. By this way, they provided a uniform GCN for spaces with different kinds of curvature (0, negative, and positive), which is called the κ-GCN. Work [126] further improves this method by providing a more reasonable definition of the gradient of curvature at zero, since the original one is incomplete.

4.5 Hyperbolic Normalizing Flows

Normalizing flows [146] involve learning a series of invertible transformations, which are used to transform a sample from a simple base distribution to a sample from a richer distribution. The models produce tractable distributions where both sampling and density evaluation can be efficient and exact. However, for the current euclidean normalizing flows, data with hierarchies embedded in the euclidean space will suffer high embedding distortion [93]. Besides, sampling from densities defined on euclidean space cannot guarantee the generated points still lie on the underlying hyperbolic space. Therefore, it is fundamental to construct normalizing flows in the hyperbolic space.

Literally, normalizing flows have already been extended to Riemannian manifolds (spherical spaces) [50], [147], [148]. Work [125] is the pioneer one to present a new normalizing flow in the hyperbolic space. They proposed first elevated normalizing flows to hyperbolic space (leveraging Lorentz model) using coupling transforms defined on the tangent bundles. Then, they explicitly utilized the geometric structure of hyperbolic spaces and further introduced Wrapped Hyperboloid Coupling (WHC), which is a fully invertible and learnable transformation.

Based on the tangent space, work [125] provides a method called Tangent Coupling, which builds upon the real-valued non-volume preserving transformations (RealNVP flow) [149] and introduces the efficient affine coupling layers. Specifically, this work follows normalizing flows designed with partially-ordered dependencies [149]. They defined a class of invertible parametric hyperbolic functions $f_i^{\mathcal{TC}} : \mathbb{L}^k \to \mathbb{L}^k$. The coupling layer is implemented using a binary mask, and partitions the input x into two sets $x_1 = x^{1:d}, x_2 = x^{d+1:n}$, For the first set x_1, its elements are transformed elementwise independently of other dimensions, while the transform of second set x_2 is based on the first one. Thus, the overall transformation of one layer is

$$\hat{f}^{\mathcal{TC}}(\hat{x}) = \begin{cases} \hat{z}_1 = \hat{x}_1 \\ \hat{z}_2 = \hat{x}_2 \odot \delta(s(\hat{x}_1)) + t(\hat{x}_1) \end{cases}$$

$$f^{\mathcal{TC}}(x) = \mathrm{Exp}_0(\hat{f}^{\mathcal{TC}}(\mathrm{Log}_0(x)), \tag{55}$$

where $\odot$ and δ are pointwise multiplication and pointwise non-linearity, respectively. s and t are map functions, which are implemented as linear neural layers and conduct the projection from $\mathcal{T}_o\mathbb{L}^d \to \mathcal{T}_o\mathbb{L}^{n-d}$. So the transformed result will be $\hat{z}$, which is the concatenation of resulted and mapped $\hat{z}_1$ and $\hat{z}_2$ on the manifold. They also provided an efficient expression for the Jacobian determinant by using the chain rule and identity.

To make full use of the expression power of the manifold, the authors conducted operations using parallel transport, instead of only operating in the tangent space of origin. Likewise, they provided an efficient expression for the Jocobian determinant. In this way, they built two different kinds of normalizing flows in the Lorentz model and improved the performance.

4.6 Hyperbolic Variational Auto-Encoders

Variational Auto-Encoder (VAE) [69], [70] is a popular probabilistic generative model, composed of an auxiliary encoder that draws samples of latent code from the approximate posterior (conditional density), and a decoder generating observations $x \in \mathcal{X}$ from latent variables $z \in \mathcal{Z}$. However, the vanilla VAE posterior is parameterized as a unimodal distribution such that it is not able to allow the structure assumption for data distributed in the hyperbolic space. Unfortunately, such a normal prior for the low-dimensional latent variables will encourage the low-dimensional representations of different samples to the center of the latent space, even for data consisting of distinct structures. Besides, embedding non-euclidean data to a euclidean space introduces significant distortion for commonly used dimensionality reduction tools, which is not good for visualization. Thus, it is meaningful to construct hyperbolic VAEs.

One of the main challenges of generalizing VAE to the hyperbolic space is the generalization of the latent distribution learning. As mentioned in [50], [118], there are mainly three ways to model the normal distributions in the hyperbolic space.

First, the Riemannian Normal [150] with Fréchet mean [73] μ and dispersion parameter σ. Sometimes, it is also referred as maximum entropy normal. In particular, the Riemannian normal distribution is defined as

$$\mathcal{N}_{\mathcal{M}}(z|\mu, \sigma^2) = \frac{1}{Z(\sigma)} \exp\left(\frac{-d_{\mathcal{M}}(\mu, z)}{2\sigma^2}\right), \tag{56}$$

where $d_{\mathcal{M}}$ is the induced distance [118], [151] and $Z(\sigma)$ is a normalization constant.

Second, the Restricted Normal. Restricting the sampled points from the normal distribution to sub-manifolds. It has also been treated as the maximum entropy distribution with respect to the ambient euclidean metric [152]. For instance, in the work Hyperspherical variational auto-encoders [153], the authors presented a novel VAE model, called $\mathcal{S}$-VAE, via von Mises-Fisher (*vMF*) distribution, in which the encoder is a homeomorphism and can provide an invertible and globally continuous mapping.

Third, the Wrapped Normal [116], [118], [154]. This distribution is constructed by utilizing the exponential map of a Gaussian distribution on the tangent space centered at the mean value. Specifically, there are four steps to get a wrapped normal distribution. First, sample one point from the euclidean normal distribution $\mathcal{N}(0, \sigma)$. Second, concatenate 0 as the zeroth coordinate of this point and transfer it to the tangent space of the origin. Third, parallel transport the sample from the current tangent space to the tangent space at the point μ. Finally, map the point from the tangent space to the manifold. In this way, a latent sample on the manifold is obtained. As mentioned in [53], the Wrapped Normal has the following reparametrizable form:

$$z = \mathrm{Exp}_\mu(\mathcal{PT}_{0\to\mu}(v))), v \in \mathcal{N}(0, \sigma). \tag{57}$$

However, the Riemannian Normal distribution could be computationally expensive for sampling if it is based on rejection sampling. The Restricted Normal like vMF has only a single scalar covariance parameter, while other approaches can parametrize the covariance in different dimensions separately. On the contrary, sampling with Wrapped Normal distributions are very computationally efficient. Based on the aforementioned generalization of the normal distribution in the non-euclidean space, there are numerous works [53], [54], [116], [118], [119] constructing VAE in the hyperbolic space, aiming at imposing structure information on the latent space.

Ovinnikov *et al.* [54] provided a closed-form definition of Gaussian distribution in the hyperbolic space, as well as the sampling rules for the prior and posterior distribution, to endow a VAE latent space with the ability to model underlying structure via the Poincaré ball model. Specifically, based on the maximum entropy generalization of Guassian distribution [150], they derived the normalization constant in Eq. (56) by decomposing it into radial and angular components. Based on the Wasserstein Autoencoder [155] framework, which is introduced to circumvent the high variance associated with the Monte-Carlo approximation, they built each layer using the hyperbolic feedforward layer provided by [43]. They also provided a generalization of the reparameterization trick by using the Möbius transformations. They further relaxed the constraint to the posterior by using the maximum mean discrepancy [156] and the network is optimized by RSGD [85]. However, as pointed out by [118], the authors had to choose a Wasserstein Auto-Encoder framework since they could not derive a closed-form solution of the ELBO's entropy term. Besides, work [116] mentions that the approximation of the likelihood and its gradient can be avoided.

Nagano *et al.* [116] provided a new normal distribution function in the hyperbolic space (Lorentz model), which is called Pseudo-Hyperbolic Gaussian, and it can be utilized to construct and learn a probabilistic model like VAE in this non-euclidean space. The authors emphasized that this distribution is computed analytically and could be sampled efficiently. Pseudo-Hyperbolic Gaussian can be constructed with four steps, as mentioned above in the wrapped normal. The author further highlighted their contributions by deriving the density of Pseudo-Hyperbolic Gaussian distribution $\mathcal{G}(\mu, \Sigma)$ due to the exponential map and the parallel transport in the wrapped normal are differentiable. Since they provided a closed-form of the density function, they could evaluate the ELBO exactly and no need to introduce the reparameterization trick in this hyperbolic VAE.

However, as pointed out by [118], the neural layers are still the euclidean ones, which do not take into account the hyperbolic geometry. Therefore, work [118] introduces a VAE that respects the geometry of the hyperbolic latent space. This is done by adding a generalization of the decision hyperplane in euclidean space. Normally, the euclidean linear affine transformation is $f(z) = sign(\langle a, z - p\rangle)||a||d(z, H_{a,p})$, where a is the coefficient, p is the intercept (offset). $H_{a,p}$ denotes a hyperplane passing through p with a as the normal direction, thus $d(z, H_{a,b})$ means the euclidean distance of z to the hyperplane. Analogue to the euclidean linear function $f(z)$, they generalized it like

$$f_{a,p}(z) = sign(\langle a, \mathrm{Log}_p(z)\rangle)||a||_p d^{\mathbb{B}}(z, H^{\mathbb{B}}_{a,p}). \tag{58}$$

Inspired by the MLR in work [43], the first layer of the decoder, which is called the gyroplane layer and is chosen to be a concatenation with a Poincaré operator f. Then it is then composed with a standard feed-forward neural network.

The gyroplane layer is then composed with a standard feed-forward neural network. For the encoder part, the author also changed the last layer by adding an exponential map for the Fréchet mean, and a softplus function for the positive defined Σ.

Then, the ELBO of VAE is optimized via an unbiased Monte Carlo (MC) estimator with two main Gaussian generalisations, which are wrapped normal and Riemannian normal generalization. Through a hyperbolic polar change of coordinates, they provided efficient and reparameterizable sampling schemes to calculate the probability density functions.

Compared to the above-mentioned methods, the work [53] highlights an implicit posterior and data-driven prior. They proposed an adversarial Poincaré variational autoencoder (APo-VAE), using a wrapped normal distribution as the prior and the variational posterior for a more expressive generalization. However, they replaced the tangent space sampling step in Eq. (57) with a more flexible implicit distribution from $\mathcal{N}(0, \mathbf{I})$, inspired by work [157]. Then, a geometry-aware Poincaré decoder is constructed, which shares the same idea as it for the decoder in work [118].

ApoVAE further optimized the variational bound by adversarially training this model by exploiting the primal-dual formulation of Kullback-Leibler (KL) divergence based on the Fenchel duality [158]. The training procedure is following the training scheme of coupled variational Bayes (CVB) from work [159] and implicit VAE [157]. Meanwhile, inspired by [160], they replaced the prior with a data-driven alternative to reduce the induced bias.

5 Applications and Performance

Various applications from different research fields can benefit from hyperbolic deep neural networks, since the latent hierarchical structure is a generic property of real-world data. Thus, we will introduce the applications for processing data which contains hierarchical structures, such as graph embedding learning, natural language processing (NLP), and the analysis of data with tree-like properties. Moreover, we also notice there exists an increasing potential for the application of hyperbolic neural networks on data that has no obvious hierarchical structures, such as images. As a result, how hyperbolic networks can be adapted to computer vision tasks is also introduced in this section. Besides, another interesting application is about hyperbolic models for biology. Current studies demonstrate attractive results for measuring cells and their activities. Thus, we also introduce advanced approaches for computational biology with hyperbolic geometry.

5.1 Hyperbolic Models for Graph Applications

Hierarchies are ubiquitous in graph data. There are numerous works leveraging deep hyperbolic neural networks dealing with the graph tasks, including node classification [47], graph classification [46], [48], link prediction [47], and graph embedding [49]. Here, we only concentrate on the hyperbolic models on top of the GNN architecture, although there are many other existing hyperbolic embedding methods, like PVAE [118], HAT [44], which also consider the modeling of graph (mainly for natural language or networks).

Works [101], [107] are pioneering works to introduce the concept of graph embeddings in the hyperbolic space. Recently, HGNN [46] and HGCN [47] are almost proposed at the same time to build GNN in the hyperbolic space. The HGCN is built on the Lorentz model. The authors applied

HGCN to both node classification and link prediction tasks. Experiments show that for a dataset with low δ-hyperbolicity, the HGCN can get a performance much better than models built in the euclidean space. The HGNN provides models on both the Poincaré model and Lorentz model. The authors dealt with graph classification tasks, in which the graphs are synthetically generated with three distinct graph generation algorithms. The results show that when the embedding dimension is smaller than 20, the HGNN has a significant superiority. We can also see that for most cases, HGNN on the Lorentz model performs better than that on the Poincaré model. While increasing the dimensions, this advantages disappeared and when the dimension is larger than 256, the HGNN even performs worse than its euclidean counterpart. Currently, work [49] derives a general version of GCN with constant curvature, κ-GCN, which significantly minimizes the graph embedding distortion and gets a superior performance on the node classification tasks.

Most of previous mentioned tasks are focused on static graphs, while work [48] considered the graph classification task with a dynamic graph input. They constructed and searched for the optimal ST-GCN in the Poincaré model. Another interesting work [126] conducted extensive experiments on four different kinds of graph tasks, including node classification, link prediction, graph classification, and graph embedding, to provide a profound analysis of when the hyperbolic space can provide a superior performance. The experimental results suggest that the non-euclidean space are not always a better choice than the euclidean counterpart. As summarized by work [126], in the task that labels depend only on the local neighborhood of the node, hyperbolic models may be inferior to their euclidean counterparts. However, many other factors, like the the way to build the manifold and the corresponding optimization strategy, may also lead to this result. Therefore, more explorations are needed to draw this conclusion.

5.2 Knowledge Graph Completion

Knowledge graph is a multi-relational graph representation of a collection of facts $\mathcal{F}$, formed in a set of triplet (e_s, r, e_o), where e_s is the subject entity, e_o is the object entity and r is a binary relation (typed directed edges) between them. $(e_s, r, e_o) \in \mathcal{F}$ denotes subject entity e_s is related to object entity e_o by the relation r. As mentioned by work [115], knowledge graphs often exhibit multiple hierarchies simultaneously. For instance, nodes near the root of the tree under one relation may be leaf nodes under another. The challenge for representing multi-relational data lies in the difficulty to represent entities shared across relations, such that different hierarchies are formed under different relations.

Most of the previous works [161], [162] model in the euclidean space, relying on the inner product or euclidean distance as a similarity measure, which can be categorised as translational models and bilinear models, respectively. Work [163] proposes to embed the entities in a Riemannian manifold, where each relation is modeled as a move to a point and they also defined specific novel distance dissimilarities for the relations. However, as pointed out by [115], this model defined in the hyperbolic space does not outperform euclidean models. Work [115] presents a new bilinear model called MuRP to embed hierarchical multi-relational data in the Poincaré ball model of the hyperbolic space. MuRP defines a basis score function for multi-relational graph embedding and generalizes it to the hyperbolic space. Experiments show that MuRP can get superior performances on the link prediction task. Besides, it requires far fewer dimensions than euclidean embeddings to achieve comparable performance. Work [164] points out that MuRP is not able to encode some logical properties of relationships. Therefore, the authors leveraged the hyperbolic isometry to simultaneously exhibit logical patterns and hierarchies, and achieved the current best performance.

On the contrary, work [14] proposes a new translational model in Poincaré model, where both the entities and the relations are embedded. Compared to the translational models in euclidean space, this method almost doubles the performance in terms of the mean reciprocal rank (MRR) metric. However, as mentioned by the authors, the HyperKG excludes from the comparison with many recent works that explores advanced techniques, thus this method is not comparable to the state-of-the-art methods, as listed in results in [115].

5.3 Natural Language Processing (NLP)

As summarized in Table 1, there are more than one third hyperbolic methods being presented to deal with NLP tasks, of which specific tasks include text classification [124], taxonomy induction [114], taxonomies embedding [45], word embeddings [51], [52], Lexical Entailment [102] and text generation [53]. Natural language often conceives a latent hierarchical structure, e.g., linguistic ontology. It is natural to turn to the hyperbolic space. Another advantage of modeling in the hyperbolic space, as mentioned by work [53], is that the latent representation allows more control of the sentences we want to generate. In the following part, we will detail different NLP tasks using hyperbolic geometry, from tasks of embedding, generation, to classification.

In the task of *word embeddings*, work [116] proposes a Gaussian-like distribution in the hyperbolic space, which is called pseudo-hyperbolic Gaussian. Based on this, a hyperbolic VAE is presented to deal with the word embeddings. Work [51] adapts the Glove [104] algorithm to learn unsupervised word embeddings in this type of Riemannian manifolds. To this end, they proposed to embed words in a Cartesian product of hyperbolic spaces which they theoretically connected to the Gaussian word embedding and the Fisher geometry. Some notable founds are, based on their method, the fully unsupervised model can almost outperform all supervised euclidean counterparts. Once trained with a small amount of weakly supervision for the hypernymy [165] score, they can obtain significant improvements and this result is much better that the models in euclidean space.

In the *taxonomy embedding* task, work [102] is one pioneer research piece that can learn embeddings in the hyperbolic space. To evaluate its ability to infer hierarchical relationships *without supervision*, they trained on data where the hierarchy of objects is not explicitly encoded. A significant improvement was witnessed in the taxonomy embedding task. Dhingra *et al.* [52] proposed a re-parametrization of Poincaré embeddings that removes the need for the projection step and allows the use of any of the popular optimization techniques in deep learning, such as Adam [88]. After this several works are presented to deal with the stability of the hyperbolic embedding. Nickel *et al.* [45] pointed out that the Lorentz model is substantially more efficient thus proposed to learn a continuous representation using the model and further improves the performance. Marc *et al.* [117] mentioned that Poincaré distance is numerically unstable and suggested the squared Lorentzian distance as a better choice. Based on this, they learned a closed-form squared Lorentzian distance and thus improved the performance on the this task. Besides, Yu *et al.* [105] constructed the hyperbolic model for dealing with the numerical instability of the previous hyperbolic networks. Based on the Lorentz model, they provide a very efficient model to learn the embedding. For instance, they can even compress the embeddings down to 2% of a Poincaré embedding on the WordNet Nouns. Very recently, Shimizu *et al.* [4] shows the superior parameter efficiency of their methods compared to conventional hyperbolic components, and the stability and outperform over their euclidean counterparts.

Then, we introduce the hyperbolic models for generation tasks. In the *text generation* task, HyperQA [109] is the first to model QA pairs in the hyperbolic space. HyperQA is an extremely fast and parameter efficient model that achieves very competitive results on multiple QA benchmarks, especially when compared to euclidean methods at that time. ApoVAE [53] also deals with the dialog-response generation problem. It optimized the variational bound by adversarially training and exploited the primal-dual formulation of KL divergence based on the Fenchel duality [158]. In *neural machine translation*, based on the hyperbolic attention mechanism, work [44] provides a hyperbolic Transformer, which shows the improvement over the original one, especially when the model capacity is restricted.

Finally, hyperbolic models are also commonly used for text-related classification tasks. In the task of *sentence entailment classification*, works [43] and [4] proposed the hyperbolic MLRs. The results confirm the tendency of the hyperbolic MLRs to outperform the euclidean version in all settings. At the same time, the hyperbolic MLR from work [4] shows more stable training, relatively narrower confidence intervals, and at least comparable performance with only half of the parameters compared to the MLR in [43]. In the task of *text classification*, HyperText [124] performs significantly better than the state-of-the-art text classifier, FastText [166], in euclidean space. Besides, HyperText with 50-dimension achieves better performance to FastText with 300-dimension, which proves the hyperbolic space is a better choice for this task. Also, works [124] and [167] benefits from the hyperbolic methods in the Chinese text analysis tasks. *Textual entailment*, which is also called natural language inference, is a binary classification task to predict whether the second sentence (hypothesis) can be inferred from the first one (premise). HNN [43] embedded two sentences using two distinct hyperbolic RNNs. With the corresponding distances, the sentence embeddings are then fed into a feedforward network and predicted with an MLR. Interestingly, the results shows that the fully euclidean baselines might even have an advantage over hyperbolic models. On top of pre-trained Poincaré embeddings [102], they conducted experiments on the WordNet noun hierarchy to evaluate the hyperbolic MLR. On this subtree classification task, hyperbolic MLR displays a clear advantage to the euclidean counterpart. Nevertheless, in what case hyperbolic is more suitable is not clear enough.

5.4 Hyperbolic Space for Recommender Systems

One of the most important factors to the success of a recommender system is the accurate representation of user preferences and item characteristics, modelled by complex networks. As mentioned in [98], hyperbolic geometry naturally emerges from network heterogeneity in the same way that network heterogeneity emerges from hyperbolic geometry. Therefore, given the complex nature of these networks, the hyperbolic space is more suitable to embed them than its euclidean counterpart. Based on the above observations, work [12] embeds bipartite user-item graphs in the hyperbolic space. This recommendation algorithm learns to rank loss that represents user-item correlations, using of hyperbolic representations through an analogy with complex networks. This algorithm shows a clear advantage when compared with the system in the euclidean space. The recommender system based on this algorithm is scaled to millions of users.

Although this system shows obvious superiority, work [122] points out that it does not learn the embeddings in a metric learning manner. Therefore, the authors explored the connections between metric learning and collaborative filtering, thus proposed a highly effective model for recommender systems. They constructed an input triplet tuple with the user, an item liked by the user, and the item unliked by the user. Then they proposed to learn the user-item joint metric in the hyperbolic space. At the same time, they introduced so-called local and global factor to better embed user-item pairs to the hyperbolic space and preserve good structure quality for metric learning.

As pointed out by work [123], in the factorization machine model [168], the naive inner product is not expressive enough for spurious or implicit feature interactions. Therefore, higher-order factorization machine [169] is proposed to learn higher-order feature interactions efficiently. As suggested by collaborative metric learning [170], learning distance instead of inner product has advantages to provide a fine-grained embedding space which could capture the representation for item-user interactions, item-item and user-user distances at the same time. Thus, the triangle inequality is more preferred than the inner product. Inspired by this, work [123] proposes a model named Lorentzian Factorization Machine (LorentzFM), which learns feature interactions with a score function measuring the validity of triangle inequalities. The authors argued that the feature interaction between two points can be learned by the *sign* of the triangle inequality for Lorentz distance, rather than using the distance itself. Based on this, they presented the model in the hyperbolic space.

5.5 Hyperbolic Models for Computer Vision

The passion of solving computer vision tasks using hyperbolic models is inspired from the observation that similar hierarchical relations between images are also common in computer vision tasks [16]. Besides, hierarchies investigated in NLP can be also transcended to the visual domain, like the knowledge graph.

Work [16] is one of the pioneering methods to model images in the hyperbolic space. To prove it is reasonable to utilize the hyperbolic space for image-based tasks, the δ-Hyperbolicity is used to measure the property of "negatively curved" of the features extracted from the embedding network. The authors concluded that the feature embeddings from current famous architectures like ResNet [21], VGG19 [40], and InceptionV3 [41] are with a small δ thus suggested the learned features process strong hierarchical relationships. Based on these observations, the authors constructed the analogues of layers in the hyperbolic spaces. They evaluated their models in computer vision tasks, including person re-identification [171], and few-shot classification [172], and results proved its superiority.

Another study in in this field is trying to generalize generative models like VAE to the hyperbolic space and deal with image reconstruction or generation tasks. Work [54] provides a Wasserstein Autoencoder for the Poincaré model and applied it to the task of generating binarized MNIST digits in order to get an intuition for the properties of the latent hyperbolic geometry. However, they did not provide a better reconstruction results when compared to the euclidean counterpart. As mentioned by the authors, both models meet with a dimension mismatch problem [173] such that reconstructed samples present a deteriorating quality as the dimensionality increases despite the lower reconstruction error. Compared to work [118], the authors derived a closed-form solution of the ELBO with two different kinds of normal distributions in the hyperbolic space. Their VAE model outperforms its euclidean counterpart, especially for low latent dimensions. However, as the latent dimension increases, the embeddings quality decreases, hence the gain from the hyperbolic geometry is reduced, just as also observed by work [102]. The same situation is also found in the work [128], where a mixed geometry space is introduced. On the MINIST reconstruction task, they displayed clear advantage when setting the latent dimension to six. However, this advantage to euclidean spaces immediately disappears when they double the dimension of latent space. This can be also caused by the property of the datasets. More studies are needed to answer whether the hyperbolic space has its advantages to address such computer vision tasks. Besides, it needs to extend to more complicated settings in larger-scale cases.

5.6 Computational Biology With Hyperbolic Geometry

Computational Biology [174] is an interdisciplinary area using biological data-driven computational models to understand biological systems and relationships. Meanwhile, hierarchical representations, such as phylogenetic trees and clustering clades have long been applied to characterize differences between cells, proteins, the activity within cells [37]. It is natural to consider hyperbolic metric as an alternative when modeling biological data in computational biology. In line with this, multiple advances have been made in computational biology towards the goal of discovering and analyzing hierarchical structures from single-cell measurements. This section presents advanced results [15], [37], [134] in single-cell RNA-seq analysis, cells developmental processes, and gene expression analysis in the hyperbolic space.

For single-cell analysis, one major difficulty stems from how to reveal the progression of cells along continuous trajectories with multiple tree-like branches. Especially for complex hierarchies, modeling with efficient low-dimensional embeddings in euclidean space will substantially distort distances between measurements, which definitely is an unwanted issue for modeling the progression. Based on the hyperbolic embedding [102], Klimovskaia *et al.* [15] provided Poincaré maps for the discovery and analyzing of complex hierarchies in single-cell data. In addition, the approach deals with all these different tasks such as clustering, lineage detection, and pseudotime inference using a single embedding in an unsupervised manner, which is impossible for previous works like t-SNE [175], PCA [176], and UMAP [177].

There are three steps in this method, of which the first two are used to approximate an unknown manifold and the last is to learn the hyperbolic embedding. First, a connected k-nearest-neighbor graph (kNNG) [178] is constructed to embed each cell and measure their euclidean distances. This step codes the local geometries of an underlying manifold. Second, based on kNNG built before, to estimate the intrinsic geometry of the underlying manifold, the global geodesic distances are computed. Then, the third step learns a two-dimensional Poincaré embedding for each cell, which preserves the topology. This can place nodes with small distances (like cells in early developmental stage) close to the center of the Poincaré disk and nodes with large distances close to the boundary. Poincaré maps produce state-of-the-art two-dimensional representations of cell trajectories on multiple scRNAseq datasets.

Following this work, Ding *et al.* [134] focused on eliminating the batch-correction and addressing visual crowding issues of conventional generative modeling approaches for Single-cell RNA-seq (scRNA-seq). The motivations are: First, normally, scRNA-seq is very high-dimensional data with typically low intrinsic dimensionality due to the co-expressed property of genes. Second, the crowding issues caused by multidimensional normal prior assumption in euclidean space lead to unreasonable gathering at the center of the latent space, even for cells from distinct cell types. Third, datasets typically have multiple technical and biological factors which cannot be handled by current VAE or batch-correction method. To address above issues, Ding *et al.* provided to represent and infer of branched developmental trajectories in the hyperbolic spaces via deep generative embedding model, of which a wrapped normal [116], [118] distribution is used as the prior for the latent variable. In this way, latent representation is learned in the hyperbolic space and the resulted latent structure accounts for the multiple batch effects. Visualization on large datasets with multiple cell types and hierarchical structures shows a much better results without cell crowding problems, which proves the effectiveness of the learned representations.

Another interesting and valuable work is applying hyperbolic geometry to gene expressions [37] analysis, which forms an important part to understand how the genotype of an organism impacts its phenotype, like disease. The challenge of this problem lies at the complexity of their relationship as thousands of genes can affect a single phenotype of interest [179]. Fortunately, the widespread correlations between genes suggests that a low-dimensional geometry can be applied to model the genetic variation and their expressions [180]. Therefore, the authors in work [37] developed a quantitative test for distinguishing the curvature of the underlying low-dimensional geometry. Besides, by incorporating hyperbolic metric into the t-SNE method, they provided visualization tools for data that exhibit a low-dimensional hyperbolic geometry. Results on several gene expression datasets from mouse and human prove that gene expression can be effectively described using low-dimensional hyperbolic metric.

6 Discussion and Open Problems

6.1 When to Expect Benefits From Hyperbolic Networks

One observation is that hyperbolic embeddings or hyperbolic neural networks cannot consistently work better than the euclidean counterparts [43]. Many times the euclidean counterpart can recover its superiority when adding the embedding dimensions. It is not clear whether the hyperbolic model can only provide a more compact model or it can provide a more efficient model with significant performance improvement. A better understanding of when and why the use of hyperbolic geometry is justified is crucial. Based on current study results, we summarize when to expect such potential benefits.

First, when there is hierarchical data. This is quite clear as hyperbolic space is naturally a better place for such data and could provide a low distortion embedding. Therefore, for data like realistic complex networks [181], data with tree-structure, graph with power-law distribution (this can be organized with latent hierarchy), generally we can expect a better performance using the hyperbolic space. Second, when the data or feature has a low Gromov δ-hyperbolicity [57]. Basically, for hierarchical data, its δ will be very small. For data without a clear observed hierarchy, Gromov δ-hyperbolicity is a better way to measure this underlying structure. From work [16], we know for Poincaré disk, they get experimental value $\delta = 0.18$, and for the feature representations for miniImageNet learned from VGG, $\delta = 0.17$. Thus, data or feature with similar value can be treated as data possessing such hierarchy thus can expect an improvement by applying the hyperbolic space. Third, when the process is expected to have a hierarchical development. Although the data itself does not possess hierarchy, the evolutionary process of samples may have strong hierarchical relationships, e.g., the developmental progress of single cell [15]. Therefore, we can expect success in such relationship modeling task or other reason induction cases. Fourth, when dealing with huge dataset with extremely limited resources. Many high-dimensional data exhibit an underlying low-dimensional hyperbolic geometry [37]. As well, hyperbolic embeddings could provide an extremely low-dimensional coding with low distortion. In such case, the hyperbolic space can provide richer information and a more compact model. Fifth, when model interpretation is much more important than the performance, the hyperbolic space is also a better choice. One of the biggest issues of current euclidean neural network is the lack of interpretability. The black-box property of deep learning keeps causing concern in real-life applications. For example, a great deep model can perform very well on the seen-dataset, while it could provide extremely wrong predicts with very high confidence on the unseen dataset. On the contrary, hyperbolic model has much better interpretation as it would give an uncertain prediction with low confidence [15]. Last, hyperbolic neural models can also be introduced as a supplemental information branch for helping traditional euclidean neural model. For instance, work [133] develops more advanced architectures by the interaction of euclidean and hyperbolic spaces. The learned representations show a superiority.

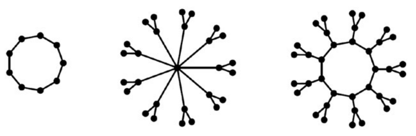


Fig. 3. Illustration of data with different structures. Leftmost shows a data with cycle. Middle shows a tree-structured data and the rightmost is a combination.

6.2 Which Hyperbolic Model?

As shown in Table 1, majority of the research in the literature is based on the Poincaré model. However, on one hand, current research has found that the Lorentz model has better numerical stability properties [45], due to its large variance when close to the boundary. On the top of the Lorentz model, Yu *et al.* provided a more stable model [105] by introducing tiling method. On the other hand, Lorentz model is un-bounded from the definition, which is not friendly to modern neural networks. Therefore, more studies are needed to choose the right hyperbolic model.

6.3 Neural Networks With Mixed Geometries

Currently, we also find a trend to construct neural networks utilizing mixed geometries.[2] As mentioned by work [113], the quality of the learned representations is determined by how well the geometry of the embedding space matches the underlying structure of the data. Therefore, for the real-world data possessing multiple complicated structures, like in the rightmost of Fig. 3, learning representations via hybrid geometry may be a better choice. Nonetheless, one major issue is how to construct the mixed manifold and at the same time determine its signature. Besides, current optimizations require a costly grid search to tune hyper-parameters. Efficient Optimization is also highly expected for the neural network with mixed geometries.

Fortunately, there are some representative attempts for mixed geometries, i.e., First, product spaces [113], which utilizes a Riemannian product manifold of model spaces [182], including hyperbolic spaces , spherical space, and the flat euclidean space. Second, mixed-curvature method [49], [128], which partitions the space into multiple component spaces and learn a curvature for each part. Third, pseudo-Riemannian manifolds of constant nonzero curvature [132], [183], which include the hyperbolic and spherical geometries and whose non-degenerate metric tensor is not constrained to be positive definite.

2. Note, the 'Mixed' methods listed in Table 1 combine geometries with different types of curvature, not different type of hyperbolic models.

6.4 Advanced Hyperbolic Networks

One potential direction can be the combination of Riemannian neural network with advanced deep learning technology. For instance, exploring new Riemannian neural architectures with advanced automatic machine learning methods, like NAS [145]. Work [48] provides to search the best projection dimension in the Poincaré model, utilizing the NAS method. However, there is much room to improve by automatically designing the neural modules, instead of only searching for optimal projection dimensions. Another important research topic can be the generalization of more sophisticated euclidean optimization algorithms. In many cases, as mentioned in [43], fully euclidean baseline models might have an advantage over hyperbolic baselines. One possible reason is that euclidean space is equipped with much more advanced optimization tools. Once the hyperbolic neural networks are also equipped with such tools, we can expect more from the powerful hyperbolic networks.

Acknowledgments

The authors would like to thank Octavian-Eugen Ganea, from MIT for the useful discussion, Emile Mathieu, from University of Oxford, for the explanation regarding the Gyroplane layer in their Poincaré Variational Auto-Encoder, and professor Stan Z. Li from Westlake University for the detailed comments of the Riemannian geometry.

References

[1] M. E. Newman, "Power laws, Pareto distributions and Zipf's law," *Contemporary Phys.*, vol. 46, no. 5, pp. 323–351, 2005.

[2] H. W. Lin and M. Tegmark, "Critical behavior in physics and probabilistic formal languages," *Entropy*, vol. 19, no. 7, 2017, Art. no. 299.

[3] K. Katayama and E. W. Maina, "Indexing method for hierarchical graphs based on relation among interlacing sequences of eigenvalues," *J. Inf. Process.*, vol. 22, no. 2, pp. 210–220, 2015.

[4] R. Shimizu, Y. Mukuta, and T. Harada, "Hyperbolic neural networks++," in *Proc. Int. Conf. Learn. Representations*, 2021. [Online]. Available: https://openreview.net/forum?id=Ec85b0tUwbA

[5] M. Boguná, F. Papadopoulos, and D. Krioukov, "Sustaining the internet with hyperbolic mapping," *Nature Commun.*, vol. 1, 2010, Art. no. 62.

[6] B. Tadić, M. Andjelković, and M. Šuvakov, "Origin of hyperbolicity in brain-to-brain coordination networks," *Front. Phys.*, vol. 6, pp. 1–12, 2018, Art. no. 7.

[7] G. García-Pérez, M. Boguñá, A. Allard, and M. Á. Serrano, "The hidden hyperbolic geometry of international trade: World trade atlas 1870–2013," *Sci. Rep.*, vol. 6, 2016, Art. no. 33441.

[8] M. Keller-Ressel and S. Nargang, "The hyperbolic geometry of financial networks," *Sci. Rep.*, vol. 11, no. 1, pp. 1–12, 2021.

[9] R. Krauthgamer and J. R. Lee, "Algorithms on negatively curved spaces," in *Proc. 47th Annu. IEEE Symp. Found. Comput. Sci.*, 2006, pp. 119–132.

[10] E. Begelfor and M. Werman, "The world is not always flat or learning curved manifolds," Sch. Eng. Comput. Sci., Hebrew Univ. Jerusalem., Jerusalem, Tech. Rep., vol. 3, no. 7, p. 8, 2005.

[11] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, "Geometric deep learning: going beyond euclidean data," *IEEE Signal Process. Mag.*, vol. 34, no. 4, pp. 18–42, Jul. 2017.

[12] B. P. Chamberlain, S. R. Hardwick, D. R. Wardrope, F. Dzogang, F. Daolio, and S. Vargas, "Scalable hyperbolic recommender systems," 2019, *arXiv:1902.08648*.

[13] K. Verbeek and S. Suri, "Metric embedding, hyperbolic space, and social networks," in *Proc. 13th Annu. Symp. Comput. Geometry*, 2014, pp. 501–510.

[14] P. Kolyvakis, A. Kalousis, and D. Kiritsis, "HyperKG: Hyperbolic knowledge graph embeddings for knowledge base completion," 2019, *arXiv:1908.04895*.

[15] A. Klimovskaia, D. Lopez-Paz, L. Bottou, and M. Nickel, "Poincaré maps for analyzing complex hierarchies in single-cell data," *Nature Commun.*, vol. 11, 2020, Art. no. 2966.

[16] V. Khrulkov, L. Mirvakhabova, E. Ustinova, I. Oseledets, and V. Lempitsky, "Hyperbolic image embeddings," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 6417–6427.

[17] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, May 28, vol. 521, no. 7553, pp. 436–444, 2015, doi: 10.1038/nature14539.

[18] M. Abadi *et al.*, "Tensorflow: A system for large-scale machine learning," in *Proc. 12th USENIX Symp. Oper. Syst. Des. Implementation*, 2016, pp. 265–283.

[19] A. Paszke *et al.*, "Automatic differentiation in PyTorch," 2017.

[20] A. Paszke *et al.*, "PyTorch: An imperative style, high-performance deep learning library," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 8026–8037.

[21] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

[22] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei , "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 248–255.

[23] A. Vaswani *et al.*, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 6000–6010.

[24] V. Mnih *et al.*, "Playing atari with deep reinforcement learning," 2013, *arXiv:1312.5602*.

[25] J. L. Coolidge, *The Elements of Non-Euclidean Geometry*. Oxford, U.K.: At the Clarendon Press, 1909.

[26] M. R. Bridson and A. Haefliger, *Metric Spaces of Non-Positive Curvature*. Berlin, Germany: Springer, 2013.

[27] C. Esteves, C. Allen-Blanchette, A. Makadia, and K. Daniilidis, "Learning SO (3) equivariant representations with spherical CNNs," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 52–68.

[28] T. S. Cohen, M. Geiger, J. Köhler, and M. Welling, "Spherical CNNs," in *Proc. Int. Conf. Learn. Representations*, 2018.

[29] M. Defferrard, M. Milani, F. Gusset, and N. Perraudin, "DeepSphere: A graph-based spherical CNN," in *Proc. Int. Conf. Learn. Representations*, 2020.

[30] N. Perraudin, M. Defferrard, T. Kacprzak, and R. Sgier, "DeepSphere: Efficient spherical convolutional neural network with healpix sampling for cosmological applications," *Astron. Comput.*, vol. 27, pp. 130–146, 2019.

[31] M. Gromov, "Hyperbolic groups," in *Essays in Group Theory*. Berlin, Germany: Springer, 1987, pp. 75–263.

[32] R. G. Barker and H. F. Wright, "Midwest and its children: The psychological ecology of an american town," Row, Peterson, 1955.

[33] A. M. Collins and M. R. Quillian, "Retrieval time from semantic memory," *J. Verbal Learn. Verbal Behav.*, Elsevier, vol. 8, no. 2, pp. 240–247, 1969.

[34] F. C. Keil, *Semantic and Conceptual Development: An Ontological Perspective*. Cambridge, MA, USA: Harvard University Press, 2013.

[35] D. M. Roy, C. Kemp, V. K. Mansinghka, and J. B. Tenenbaum, "Learning annotated hierarchies from relational data," in *Proc. Adv. Neural Inf. Process. Syst.*, 2007, pp. 1185–1192.

[36] Y. Zhou, B. H. Smith, and T. O. Sharpee, "Hyperbolic geometry of the olfactory space," *Sci. Adv.*, vol. 4, no. 8, 2018, Art. no. eaaq1458.

[37] Y. Zhou and T. O. Sharpee, "Hyperbolic geometry of gene expression," *Iscience*, vol. 24, no. 3, 2021, Art. no. 102225.

[38] O. Vinyals, C. Blundell, T. Lillicrap, and D. Wierstra, "Matching networks for one shot learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 3637–3645.

[39] A. Krizhevsky *et al.*, "Learning multiple layers of features from tiny images," Citeseer, 2009.

[40] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.

[41] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 1–9.

[42] Wiki. Non-euclidean space. Accessed: Oct. 5, 2021. [Online]. Available: https://en.wikipedia.org/wiki/Non-Euclidean_geometry

[43] O. Ganea, G. Bécigneul, and T. Hofmann, "Hyperbolic neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 5345–5355.

[44] C. Gulcehre *et al.*, "Hyperbolic attention networks," 2018, *arXiv:1805.09786*.

[45] M. Nickel and D. Kiela, "Learning continuous hierarchies in the lorentz model of hyperbolic geometry," in *Proc. 35th Int. Conf. Mach. Learn.*, 2018, pp. 3779–3788.

[46] Q. Liu, M. Nickel, and D. Kiela, "Hyperbolic graph neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 8230–8241.

[47] I. Chami, Z. Ying, C. Ré, and J. Leskovec, "Hyperbolic graph convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 4868–4879.

[48] W. Peng, J. Shi, Z. Xia, and G. Zhao, "Mix dimension in poincaré geometry for 3D skeleton-based action recognition," in *Proc. 28th ACM Int. Conf. Multimedia*, 2020, pp. 1432–1440.

[49] G. Bachmann, G. Bécigneul, and O. Ganea, "Constant curvature graph convolutional networks," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 486–496.

[50] D. J. Rezende *et al.*, "Normalizing flows on tori and spheres," in *Proc. 37th Int. Conf. Mach. Learn.*, 2020, pp. 8083–8092.

[51] A. Tifrea, G. Bécigneul, and O.-E. Ganea, "Poincar'e glove: Hyperbolic word embeddings," in *Proc. Int. Conf. Learn. Representations*, 2019.

[52] B. Dhingra, C. J. Shallue, M. Norouzi, A. M. Dai, and G. E. Dahl, "Embedding text in hyperbolic spaces," 2018, *arXiv:1806.04313*.

[53] S. Dai, Z. Gan, Y. Cheng, C. Tao, L. Carin, and J. Liu, "Apo-vae: Text generation in hyperbolic space," 2020, *arXiv:2005.00054*.

[54] I. Ovinnikov, "Poincar'e wasserstein autoencoder," 2019, *arXiv:1901.01427*.

[55] M. P. d. Carmo, *Riemannian Geometry*. Basel, Switzerland: Birkhäuser, 1992.

[56] S. Gallot, D. Hulin, and J. Lafontaine, *Riemannian Geometry*. Berlin, Germany: Springer, 1990.

[57] A. B. Adcock, B. D. Sullivan, and M. W. Mahoney, "Tree-like structure in large social and information networks," in *Proc. IEEE 13th Int. Conf. Data Mining.*, 2013, pp. 1–10.

[58] M. Bonk and O. Schramm, "Embeddings of Gromov hyperbolic spaces," in *Selected Works of Oded Schramm*. Berlin, Germany: Springer, 2011.

[59] E. Beltrami, "Teoria fondamentale degli spazii di curvatura costante," *Annali di Matematica Pura ed Applicata (1867–1897)*, vol. 2, pp. 232–255, 1868.

[60] J. W. Cannon, W. J. Floyd, R. Kenyon, W. R. Parry, "Hyperbolic geometry," in *Flavors of Geometry*. Cambridge, U.K.: Cambridge Univ. Press, pp. 59–110, 1997.

[61] M. Hamann, "On the tree-likeness of hyperbolic spaces," in *Mathematical Proceedings of the Cambridge Philosophical Society*. Cambridge, UK: Cambridge Univ. Press, 2018.

[62] A. Dyubina and I. Polterovich, "Explicit constructions of universal -trees and asymptotic geometry of hyperbolic spaces," *Bull. London Math. Soc.*, 2001.

[63] R. Kleinberg, "Geographic routing using hyperbolic space," in *Proc. IEEE INFOCOM 26th IEEE Int. Conf. Comput. Commun.*, 2007, pp. 1902–1909.

[64] A. Lou, I. Katsman, Q. Jiang, S. Belongie, S.-N. Lim, and C. De Sa, "Differentiating through the Fr'echet mean," in *Proc. 37 Int. Conf. Mach. Learn.*, pp. 6393–6403, 2020.

[65] Z. Huang, R. Wang, S. Shan, L. Van Gool , and X. Chen, "Cross euclidean-to-riemannian metric learning with application to face recognition from video," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 12, pp. 2827–2840, Dec. 2018.

[66] O. Tuzel, F. Porikli, and P. Meer, "Pedestrian detection via classification on riemannian manifolds," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 10, pp. 1713–1727, Oct. 2008.

[67] A. A. Ungar, "Hyperbolic trigonometry and its application in the poincaré ball model of hyperbolic geometry," *Comput. Math. Appl.*, vol. 41, pp. 135–147, 2001.

[68] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015, *arXiv:1502.03167*.

[69] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," 2014, *arXiv:1312.6114v10*.

[70] D. J. Rezende, S. Mohamed, and D. Wierstra, "Stochastic backpropagation and approximate inference in deep generative models," in *Proc. Int. Conf. Mach. Learn.*, 2014, pp. 1278–1286.

[71] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2016, *arXiv:1609.02907* .

[72] A. A. Ungar, "A gyrovector space approach to hyperbolic geometry," *Synth. Lectures Math. Statist.*, 2008.

[73] M. R. Fréchet, "Les éléments aléatoires de nature quelconque dans un espace distancié," *Annales de l'institut Henri Poincaré*, vol. 10, no. 4, pp. 215–310, 1948.

[74] S. Gould, B. Fernando, A. Cherian, P. Anderson, R. S. Cruz, and E. Guo, "On differentiating parameterized Argmin and Argmax problems with application to bi-level optimization," 2016, *arXiv:1607.05447*.

[75] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 1026–1034.

[76] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[77] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2013, pp. 6645–6649.

[78] K. Cho, B. v. M. C. Gulcehre, D. Bahdanau, F. B. H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder–decoder for statistical machine translation," 2014, *arXiv:1406.1078v3*.

[79] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proc. 27th Int. Conf. Mach. Learn.*, 2010, pp. 807–814.

[80] B. Xu, N. Wang, T. Chen, and M. Li, "Empirical evaluation of rectified activations in convolutional network," 2015, *arXiv:1505.00853*.

[81] G. Lebanon and J. Lafferty, "Hyperplane margin classifiers on the multinomial manifold," in *Proc. 21st Int. Conf. Mach. Learn.*, 2004.

[82] H. Cho, B. DeMeo, J. Peng, and B. Berger, "Large-margin classification in hyperbolic space," in *Proc. 22nd Int. Conf. Artif. Intell. Stat.*, 2019, pp. 1832–1840.

[83] M. Weber, M. Zaheer, A. S. Rawat, A. Menon, and S. Kumar, "Robust large-margin learning in hyperbolic space," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020.

[84] M. Kochurov, R. Karimov, and S. Kozlukov, "Geoopt: Riemannian optimization in pytorch," 2020, *arXiv:2005.02819v5*.

[85] S. Bonnabel, "Stochastic gradient descent on riemannian manifolds," *IEEE Trans. Autom. Control*, vol. 58, no. 9, pp. 2217–2229, Sep. 2013.

[86] H. Zhang and S. Sra,"First-order methods for geodesically convex optimization," in *Proc. Conf. Learn. Theory*, 2016, pp. 1617–1638.

[87] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *J. Mach. Learn. Res.*, vol. 12, pp. 2121–2159, 2011.

[88] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Representations*, 2015.

[89] G. Bécigneul and O.-E. Ganea, "Riemannian adaptive optimization methods," 2019, *arXiv:1810.00760v2*.

[90] Y. LeCun, Y. Bengio *et al.*, "Convolutional networks for images, speech, and time series," in *The Handbook of Brain Theory and Neural Networks*. Cambridge, MA, USA: MIT Press, 1995, pp. 255–258.

[91] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Trans. Signal Process.*, vol. 45, no. 11, pp. 2673–2681, Nov. 1997.

[92] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun , "Spectral networks and locally connected networks on graphs," 2013, *arXiv:1312.6203*.

[93] R. Sarkar, "Low distortion Delaunay embedding of trees in hyperbolic plane," in *Proc. Int. Symp. Graph Drawing.*, 2011, pp. 355–366.

[94] J. A. Walter, "H-MDS: A new approach for interactive visualization with multidimensional scaling in the hyperbolic space," *Inf. Syst.*, vol. 29, no. 4, pp. 273–292, 2004.

[95] Y. Shavitt and T. Tankel, "Hyperbolic embedding of internet graph for distance estimation and overlay construction," *IEEE/ACM Trans. Netw.*, vol. 16, no. 1, pp. 25–36, Feb. 2008.

[96] D. Krioukov, F. Papadopoulos, A. Vahdat, and M. Boguná, "Curvature and temperature of complex networks," *Phys. Rev. E*, vol. 80, 2009, Art. no. 035101.

[97] A. Cvetkovski and M. Crovella, "Hyperbolic embedding and routing for dynamic graphs," in *Proc. IEEE INFOCOM.*, 2009, pp. 1647–1655.

[98] D. Krioukov, F. Papadopoulos, M. Kitsak, A. Vahdat, and M. Boguná, "Hyperbolic geometry of complex networks," *Phys. Rev. E*, vol. 82, 2010, Art. no. 036106.

[99] A. Cvetkovski and M. Crovella, "Multidimensional scaling in the poincaré disk," 2011, *arXiv:1105.5332*.

[100] T. Bläsius, T. Friedrich, A. Krohmer, and S. Laue, "Efficient embedding of scale-free graphs in the hyperbolic plane," *IEEE/ACM Trans. Netw.*, vol. 26, no. 2, pp. 920–933, Apr. 2018.

[101] A. Muscoloni, J. M. Thomas, S. Ciucci, G. Bianconi, and C. V. Cannistraci, "Machine learning meets complex networks via coalescent embedding in the hyperbolic space," *Nature Commun.*, vol. 8, 2017, Art. no. 1615.

[102] M. Nickel and D. Kiela, "Poincaré embeddings for learning hierarchical representations," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 6341–6350.

[103] I. Vendrov, R. Kiros, S. Fidler, and R. Urtasun, "Order-embeddings of images and language," in *Proc. Int. Conf. Learn. Representations*, 2016.

[104] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2014, pp. 1532–1543.

[105] T. Yu and C. M. De Sa, "Numerically accurate hyperbolic embeddings using tiling-based models," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019.

[106] J. H. Conway, H. Burgiel, and C. Goodman-Strauss, *The Symmetries of Things*. Boca Raton, FL, USA: CRC Press, 2016.

[107] B. P. Chamberlain, J. Clough, and M. P. Deisenroth, "Neural embeddings of graphs in hyperbolic space," 2017, *arXiv:1705.10359*.

[108] C. De Sa , A. Gu, C. Ré, and F. Sala, "Representation tradeoffs for hyperbolic embeddings," in *Proc. Int. Conf. Mach. Learn. Res.*, 2018, pp. 4460–4469.

[109] Y. Tay, L. A. Tuan, and S. C. Hui, "Hyperbolic representation learning for fast and efficient neural question answering," in *Proc. 11th ACM Int. Conf. Web Search Data Mining*, 2018, pp. 583–591.

[110] O.-E. Ganea, G. Bécigneul, and T. Hofmann, "Hyperbolic entailment cones for learning hierarchical embeddings," in *Proc. Int. Conf. Mach. Learn.*, 2018.

[111] T. D. Q. Vinh, Y. Tay, S. Zhang, G. Cong, and X.-L. Li, "Hyperbolic recommender systems," Tech. Rep., 2018, *arXiv:1809.01703*.

[112] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme , "BPR: Bayesian personalized ranking from implicit feedback," in *Proc. 25th Conf. Uncertainty Artif. Intell.*, 2009, pp. 452–461.

[113] A. Gu, F. Sala, B. Gunel, and C. Ré, "Learning mixed-curvature representations in product spaces," in *Proc. Int. Conf. Learn. Representations*, 2018.

[114] R. Aly, S. Acharya, A. Ossa, A. Köhn, C. Biemann, and A. Panchenko, "Every child should have parents: A taxonomy refinement algorithm based on hyperbolic term embeddings," in *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics*, 2019, pp. 4811–4817.

[115] I. Balazevic, C. Allen, and T. Hospedales, "Multi-relational poincaré graph embeddings," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 4463–4473.

[116] Y. Nagano, S. Yamaguchi, Y. Fujita, and M. Koyama, "A wrapped normal distribution on hyperbolic space for gradient-based learning," in *Proc. 36th Int. Conf. Mach. Learn.*, 2019, pp. 4693–4702.

[117] M. Law, R. Liao, J. Snell, and R. Zemel, "Lorentzian distance learning for hyperbolic representations," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 3672–3681.

[118] E. Mathieu, C. Le Lan , C. J. Maddison, R. Tomioka, and Y. W. Teh, "Continuous hierarchical representations with poincaré variational auto-encoders," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 12565–12576.

[119] D. Grattarola, L. Livi, and C. Alippi, "Adversarial autoencoders with constant-curvature latent manifolds," *Appl. Soft Comput.*, vol. 81, p. 105511, 2019, Art. no. 105511.

[120] N. Monath, M. Zaheer, D. Silva, A. McCallum, and A. Ahmed, "Gradient-based hierarchical clustering using continuous representations of trees in hyperbolic space," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2019, pp. 714–722.

[121] Y. Zhang, X. Wang, X. Jiang, C. Shi, and Y. Ye, "Hyperbolic graph attention network," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 4375–4383.

[122] L. V. Tran, Y. Tay, S. Zhang, G. Cong, and X. Li, "Hyperml: A boosting metric learning approach in hyperbolic space for recommender systems," in *Proc. 13th Int. Conf. Web Search Data Mining*, 2020, pp. 609–617.

[123] C. Xu and M. Wu, "Learning feature interactions with lorentzian factorization machine," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 6470–6477.

[124] Y. Zhu, D. Zhou, J. Xiao, X. Jiang, X. Chen, and Q. Liu, "Hypertext: Endowing FastText with hyperbolic geometry," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2020, pp. 1166–1171.

[125] A. J. Bose, A. Smofsky, R. Liao, P. Panangaden, and W. L. Hamilton, "Latent variable modelling with hyperbolic normalizing flows," 2020, *arXiv:2002.06336*.

[126] M. Kochurov, S. Ivanov, and E. Burnaev, "Are hyperbolic representations in graphs created equal?," 2020, *arXiv:2007.07698v1*.

[127] A. Bogatskiy, B. Anderson, J. T. Offermann, M. Roussi, D. W. Miller, and R. Kondor, "Lorentz group equivariant neural network for particle physics," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 992–1002.

[128] O. Skopek, O.-E. Ganea, and G. Bécigneul, "Mixed-curvature variational autoencoders," in *Proc. Int. Conf. Learn. Representations*, 2020.

[129] I. Chami, A. Gu, V. Chatziafratis, and C. Ré, "From trees to continuous embeddings and back: Hyperbolic hierarchical clustering," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 714–722.

[130] E. Mathieu and M. Nickel, "Riemannian continuous normalizing flows," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 2503–2515.

[131] R. Sonthalia and A. C. Gilbert, "Tree! I am no tree! I am a low dimensional hyperbolic embedding," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 845–856.

[132] M. T. Law and J. Stam, "Ultrahyperbolic representation learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 1668–1678.

[133] S. Zhu, S. Pan, C. Zhou, J. Wu, Y. Cao, and B. Wang, "Graph geometry interaction learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 7548–7558.

[134] J. Ding and A. Regev, "Deep generative model embedding of single-cell RNA-Seq profiles on hyperspheres and hyperbolic spaces," *Nature Commun.*, vol. 12, 2021, Art. no. 2554.

[135] V. Cohen-Addad , V. Kanade, F. Mallmann-Trenn, and C. Mathieu, "Hierarchical clustering: Objective functions and algorithms," *J. ACM J. Altern. Complement. Med.*, vol. 66, no. 4, 2019, Art. no. 26.

[136] O. Yim and K. T. Ramdeen, "Hierarchical cluster analysis: Comparison of three linkage measures and application to psychological data," *Quantitative Methods Psychol.*, vol. 11, pp. 8–21, 2015.

[137] S. Dasgupta, "A cost function for similarity-based hierarchical clustering," in *Proc. 48th Annu. ACM Symp. Theory Comput.*, 2016, pp. 118–127.

[138] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Hum. Lang. Technol.*, 2019, pp. 4171–4186.

[139] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," 2017, *arXiv:1710.10903*.

[140] W. Peng, X. Hong, H. Chen, and G. Zhao, "Learning graph convolutional network for skeleton-based human action recognition by neural searching," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 2669–2676.

[141] A. Dosovitskiy *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," in *Proc. Int. Conf. Learn. Representations*, 2020.

[142] J. Lee, Y. Lee, J. Kim, A. Kosiorek, S. Choi, and Y. W. Teh, "Set transformer: A framework for attention-based permutation-invariant neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 3744–3753.

[143] M. Henaff, J. Bruna, and Y. LeCun, "Deep convolutional networks on graph-structured data," 2015, *arXiv:1506.05163*.

[144] S. Yan, Y. Xiong, and D. Lin, "Spatial temporal graph convolutional networks for skeleton-based action recognition," in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 7444–7452.

[145] T. Elsken, J. H. Metzen, and F. Hutter, "Neural architecture search: A survey," 2019, *arXiv:1808.05377v3*.

[146] D. J. Rezende and S. Mohamed, "Variational inference with normalizing flows," in *Proc. 32nd Int. Conf. Mach. Learn.*, 2015, pp. 1530–1538.

[147] M. C. Gemici, D. Rezende, and S. Mohamed, "Normalizing flows on Riemannian manifolds," 2016, *arXiv:1611.02304*.

[148] J. Brehmer and K. Cranmer, "Flows for simultaneous manifold learning and density estimation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020.

[149] L. Dinh, J. Sohl-Dickstein, and S. Bengio, "Density estimation using real NVP," 2016, *arXiv:1605.08803*.

[150] X. Pennec, "Intrinsic statistics on riemannian manifolds: Basic tools for geometric measurements," *J. Math. Imag. Vis.*, vol. 25, no. 1, pp. 127–154, 2006.

[151] S. Said, L. Bombrun, and Y. Berthoumieu, "New Riemannian priors on the univariate normal model," *Entropy*, vol. 16, no. 7, pp. 4015–4031, 2014.

[152] K. V. Mardia, *Statistics of Directional Data*. Cambridge, MA, USA: Academic Press, 2014.

[153] T. R. Davidson, L. Falorsi, N. De Cao , T. Kipf, and J. M. Tomczak, "Hyperspherical variational auto-encoders," in *Proc. 34th Conf. Uncertainty Artif. Intell.*, 2018, pp. 856–865.

[154] L. Falorsi, P. de Haan, T. R. Davidson, and P. Forré, "Reparameterizing distributions on lie groups," in *Proc. 22nd Int. Conf. Artif. Intell. Statist.*, 2019, pp. 3244–3253.

[155] I. Tolstikhin, O. Bousquet, S. Gelly, and B. Schoelkopf, "Wasserstein auto-encoders," in *Proc. Int. Conf. Learn. Representations*, 2018.

[156] K. M. Borgwardt, A. Gretton, M. J. Rasch, H.-P. Kriegel, B. Schölkopf, and A. J. Smola, "Integrating structured biological data by kernel maximum mean discrepancy," *Bioinformatics*, vol. 22, no. 14, pp. e49–e57, 2006.

[157] L. Fang, C. Li, J. Gao, W. Dong, and C. Chen, "Implicit deep latent variable models for text generation," in *Proc. Conf. Empirical Methods Nature Lang. Process.*, 2019, pp. 3946–3956.

[158] R. T. Rockafellar, "Extension of fenchel'duality theorem for convex functions," *Duke Math. J.*, vol. 33, no. 1, pp. 81–89, 1966.

[159] B. Dai *et al.*, "Coupled variational bayes via optimization embedding," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 9690–9700.

[160] J. Tomczak and M. Welling, "VAE with a VampPrior," in *Proc. Int. Conf. Artif. Intell. Stat.*, 2018, pp. 1214–1223.

[161] A. Bordes, N. Usunier, A. Garcia-Duran , J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 2787–2795.

[162] B. Yang, W.-t. Yih, X. He, J. Gao, and L. Deng, "Embedding entities and relations for learning and inference in knowledge bases," in *Proc. Int. Conf. Learn. Representations*, 2015.

[163] A. Suzuki, Y. Enokida, and K. Yamanishi, "Riemannian transe: Multi-relational graph embedding in non-euclidean space," 2018.

[164] I. Chami, A. Wolf, D.-C. Juan, F. Sala, S. Ravi, and C. Ré, "Low-dimensional hyperbolic knowledge graph embeddings," in *Proc. 58th Annu. Meeting Assoc. Comput. Linguistics*, 2020, pp. 6901–6914.

[165] I. Vulić, D. Gerz, D. Kiela, F. Hill, and A. Korhonen, "Hyperlex: A large-scale evaluation of graded lexical entailment," *Comput. Linguistics*, vol. 43, no. 4, pp. 781–835, 2017.

[166] A. Joulin, É. Grave, P. Bojanowski, and T. Mikolov, "Bag of tricks for efficient text classification," in *Proc. 15th Conf. Eur. Chapter ACL*, 2017, pp. 427–431.

[167] M. V. Micic and H. Chu, "Hyperbolic deep learning for chinese natural language understanding," 2018, *arXiv:1812.10408*.

[168] S. Rendle, "Factorization machines," in *Proc. IEEE Int. Conf. Data Mining.*, 2010, pp. 995–1000.

[169] M. Blondel, A. Fujino, N. Ueda, and M. Ishihata, "Higher-order factorization machines," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 3359–3367.

[170] C.-K. Hsieh, L. Yang, Y. Cui, T.-Y. Lin, S. Belongie, and D. Estrin, "Collaborative metric learning," in *Proc. 26th Int. Conf. World Wide Web*, 2017, pp. 193–201.

[171] L. Zheng, Y. Yang, and A. G. Hauptmann, "Person re-identification: Past, present and future," 2016, *arXiv:1610.02984*.

[172] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 4080–4090.

[173] P. K. Rubenstein, B. Schoelkopf, and I. Tolstikhin, "On the latent space of wasserstein auto-encoders," 2018, *arXiv:1802.03761*.

[174] M. Huerta, G. Downing, F. Haseltine, B. Seto, and Y. Liu, "Nih working definition of bioinformatics and computational biology," U.S. Nat. Inst. Health, p. 1, 2000.

[175] L. Van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, no. 86, pp. 2579–2605, 2008.

[176] M. Ringnér, "What is principal component analysis?," *Nature Biotechnol.*, pp. 303–304, 2008.

[177] L. McInnes, J. Healy, and J. Melville, "Umap: Uniform manifold approximation and projection for dimension reduction," 2018, *arXiv:1802.03426*.

[178] U. Von Luxburg, "A tutorial on spectral clustering," *Stat. Comput.*, vol. 17, pp. 395–416, 2007.

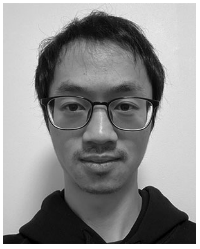
[179] T. A. Manolio *et al.*, "Finding the missing heritability of complex diseases," *Nature*, vol. 461, pp. 747–753, 2009.

[180] J. Novembre *et al.*, "Genes mirror geography within Europe," *Nature*, vol. 456, pp. 98–101, 2008.

[181] G. Bianconi and C. Rahmede, "Emergent hyperbolic network geometry," *Sci. Rep.*, vol. 7, 2017, Art. no. 41974.

[182] J. M. Lee, *Riemannian Manifolds: An Introduction to Curvature*. Berlin, Germany: Springer, 2006.

[183] B. O'neill, *Semi-Riemannian Geometry with Applications to Relativity*. Cambridge, MA, USA: Academic Press, 1983.



Wei Peng received the MS degree in computer science from the Xiamen University, Xiamen, China, in 2016. He is currently working toward the PhD degree with the Center for Machine Vision and Signal Analysis, University of Oulu, Oulu, Finland. He has authored or coauthored articles in mainstream conferences and journals, including AAAI, ICCV, *ACM Multimedia,* and the *IEEE Transactions on Image Processing*. His research interests include machine learning, affective computing, medical imaging, and human action analysis.

Tuomas Varanka received the BS and MS degrees in computer science and engineering from the University of Oulu in 2019 and 2020, respectively. He is currently working toward the PhD degree with the Center for Machine Vision and Signal Analysis, University of Oulu, Oulu, Finland. His work has focused on micro-expression recognition. His research interests include machine learning and affective computing.

Abdelrahman Mostafa received the MS degree in artificial intelligence from the University of Oulu, Finland, in 2020. He is currently working toward the PhD degree with the Center for Machine Vision and Signal Analysis, University of Oulu, Oulu, Finland. His research interests include machine learning, deep learning, and computer vision.

Henglin Shi received the MS degree in artificial intelligence from University of Oulu, Finland, where he is currently working toward the PhD degree. He has authored or coauthored articles in mainstream conferences and journals, such as BMVC, *IEEE Transactions on Multimedia*, and the *IEEE Transactions on Image Processing*. His research interests include machine learning, deep learning and computer vision.

Guoying Zhao (Fellow, IEEE) received the PhD degree in computer science from the Chinese Academy of Sciences, China. She is currently an academy professor with the Center for Machine Vision and Signal Analysis, University of Oulu, Finland. His research interests include affective computing and machine learning. She is IAPR fellow, ELLIS member, and an associate editor for *Pattern Recognition*, the *IEEE Transactions on Circuits and Systems for Video Technology*, and the *Image and Vision Computing Journals*.