Constant Curvature Graph Convolutional Networks

Gregor Bachmann[*1] **Gary Bécigneul**[*1] **Octavian-Eugen Ganea**[2]

Abstract

Interest has been rising lately towards methods representing data in non-Euclidean spaces, e.g. hyperbolic or spherical, that provide specific inductive biases useful for certain real-world data properties, e.g. scale-free, hierarchical or cyclical. However, the popular *graph neural networks* are currently limited in modeling data only via Euclidean geometry and associated vector space operations. Here, we bridge this gap by proposing mathematically grounded generalizations of *graph convolutional networks* (GCN) to (products of) constant curvature spaces. We do this by i) introducing a unified formalism permitting a differentiable interpolation between all geometries of constant curvature, ii) leveraging gyrobarycentric coordinates that generalize the classic Euclidean concept of the *center of mass*. Our class of models smoothly recover their Euclidean counterparts when the curvature goes to zero from either side. Empirically, we outperform Euclidean GCNs in the tasks of node classification and distortion minimization for symbolic data exhibiting non-Euclidean behavior, according to their discrete curvature.

1. Introduction

Graph Convolutional Networks. The success of convolutional networks and deep learning for image data has inspired generalizations for graphs for which sharing parameters is consistent with the graph geometry. Bruna et al. (2014); Henaff et al. (2015) are the pioneers of spectral graph convolutional neural networks in the graph Fourier space using localized spectral filters on graphs. However, in order to reduce the graph-dependency on the Laplacian eigenmodes, Defferrard et al. (2016) approximate the convolutional filters using Chebyshev polynomials leveraging a result of Hammond et al. (2011). The resulting method (discussed in Appendix A) is computationally efficient and superior in terms of accuracy and complexity. Further, Kipf & Welling (2017) simplify this approach by considering first-order approximations obtaining high scalability. The proposed *graph convolutional networks* (GCN) is interpolating node embeddings via a symmetrically normalized adjacency matrix, while this weight sharing can be understood as an efficient diffusion-like regularizer. Recent works extend GCNs to achieve state of the art results for link prediction (Zhang & Chen, 2018), graph classification (Hamilton et al., 2017; Xu et al., 2018) and node classification (Klicpera et al., 2019; Veličković et al., 2018).

Euclidean geometry in ML. In machine learning (ML), data is most often represented in a Euclidean space for various reasons. First, some data is *intrinsically* Euclidean, such as positions in 3D space in classical mechanics. Second, intuition is easier in such spaces, as they possess an appealing vectorial structure allowing basic arithmetic and a rich theory of linear algebra. Finally, a lot of quantities of interest such as distances and inner-products are known in closed-form formulae and can be computed very efficiently on the existing hardware. These operations are the basic building blocks for most of today's popular machine learning models. Thus, the powerful simplicity and efficiency of Euclidean geometry has led to numerous methods achieving state-of-the-art on tasks as diverse as machine translation (Bahdanau et al., 2015; Vaswani et al., 2017), speech recognition (Graves et al., 2013), image classification (He et al., 2016) or recommender systems (He et al., 2017).

Riemannian ML. In spite of this success, certain types of data (e.g. hierarchical, scale-free or spherical data) have been shown to be better represented by non-Euclidean geometries (Defferrard et al., 2019; Bronstein et al., 2017; Nickel & Kiela, 2017; Gu et al., 2019), leading in particular to the rich theories of manifold learning (Roweis & Saul, 2000; Tenenbaum et al., 2000) and information geometry (Amari & Nagaoka, 2007). The mathematical framework in vigor to manipulate non-Euclidean geometries is known as *Riemannian geometry* (Spivak, 1979). Although its theory leads to many strong and elegant results, some of its basic

[*]Equal contribution [1]Department of Computer Science, ETH Zürich [2]Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology. Correspondence to: Gregor Bachmann <gregor.bachmann@inf.ethz.ch>, Gary Bécigneul <garyb@mit.edu>, Octavian-Eugen Ganea <oct@mit.edu>.

Proceedings of the 37^{th} International Conference on Machine Learning, Online, PMLR 119, 2020. Copyright 2020 by the author(s).

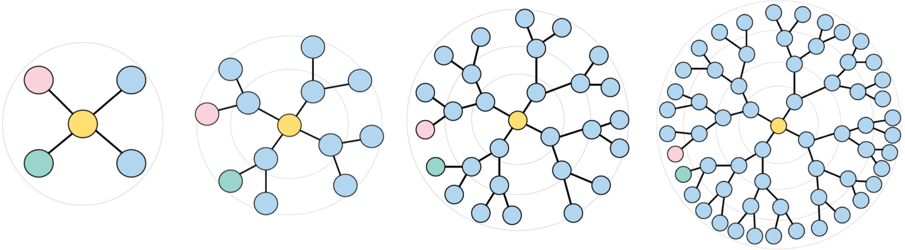

Figure 1. Euclidean embeddings of trees of different depths. All the four most inner circles are identical. Ideal node embeddings should match in distance the graph metric, e.g. the distance between the pink and green nodes should be the same as their shortest path length. Notice how we quickly run out of space, e.g. the pink and green nodes get closer as opposed to farther. This issue is resolved when embedding trees in hyperbolic spaces.

quantities such as the distance function $d(\cdot, \cdot)$ are in general not available in closed-form, which can be prohibitive to many computational methods.

Representational Advantages of Geometries of Constant Curvature. An interesting trade-off between general Riemannian manifolds and the Euclidean space is given by manifolds of *constant sectional curvature*. They define together what are called *hyperbolic* (negative curvature), *elliptic* (positive curvature) and Euclidean (zero curvature) geometries. As discussed below and in Appendix B, Euclidean spaces have limitations and suffer from large distortion when embedding certain types of data such as trees. In these cases, the hyperbolic and spherical spaces have representational advantages providing a better inductive bias for the respective data.

The **hyperbolic space** can be intuitively understood as a continuous tree: the volume of a ball grows exponentially with its radius, similarly as how the number of nodes in a binary tree grows exponentially with its depth (see fig. 1). Its tree-likeness properties have long been studied mathematically (Gromov, 1987; Hamann, 2017; Ungar, 2008) and it was proven to better embed *complex networks* (Krioukov et al., 2010), *scale-free graphs* and *hierarchical data* compared to the Euclidean geometry (Cho et al., 2019; Sala et al., 2018; Ganea et al., 2018b; Gu et al., 2019; Nickel & Kiela, 2018; 2017; Tifrea et al., 2019). Several important tools or methods found their hyperbolic counterparts, such as variational autoencoders (Mathieu et al., 2019; Ovinnikov, 2019), attention mechanisms (Gulcehre et al., 2018), matrix multiplications, recurrent units and multinomial logistic regression (Ganea et al., 2018a).

Similarly, **spherical geometry** provides benefits for modeling spherical or cyclical data (Defferrard et al., 2019; Matousek, 2013; Davidson et al., 2018; Xu & Durrett, 2018; Gu et al., 2019; Grattarola et al., 2018; Wilson et al., 2014).

Computational Efficiency of Constant Curvature Spaces (CCS). CCS are some of the few Riemannian manifolds to possess closed-form formulae for geometric quantities of interest in computational methods, i.e. distance, geodesics, exponential map, parallel transport and their gradients. We also leverage here the closed expressions for weighted centroids.

"Linear Algebra" of CCS: Gyrovector Spaces. In order to study the geometry of constant negative curvature in analogy with the Euclidean geometry, Ungar (1999; 2005; 2008; 2016) proposed the elegant non-associative algebraic formalism of **gyrovector spaces**. Recently, Ganea et al. (2018a) have linked this framework to the Riemannian geometry of the space, also generalizing the building blocks for non-Euclidean deep learning models operating with hyperbolic data representations.

However, *it remains unclear how to extend in a principled manner the connection between Riemannian geometry and gyrovector space operations for spaces of constant positive curvature (spherical)*. By leveraging Euler's formula and complex analysis, we present to our knowledge the first unified gyro framework that allows for a differentiable interpolation between geometries of constant curvatures irrespective of their signs. This is possible when working with the Poincaré ball and stereographic spherical projection models of respectively hyperbolic and spherical spaces.

GCNs in Constant Curvature Spaces. In this work, we introduce an extension of graph convolutional networks that allows to learn representations residing in (products of) **constant curvature spaces with any curvature sign**. We achieve this by combining the derived unified gyro framework together with the effectiveness of GCNs (Kipf & Welling, 2017). Concurrent to our work, Chami et al. (2019); Liu et al. (2019) consider graph neural networks that learn embeddings in hyperbolic space via tangent space aggregation. Their approach will be analyzed more closely in section 3.4. Our model is more general as it produces representations in a strict super-set containing the hyperbolic space.

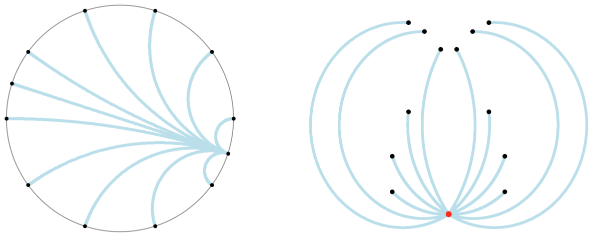

Figure 2. Geodesics in the Poincaré disk (left) and the stereographic projection of the sphere (right).

2. The Geometry of Constant Curvature Spaces

Riemannian Geometry. A manifold $\mathcal{M}$ of dimension d is a generalization to higher dimensions of the notion of surface, and is a space that locally *looks* like $\mathbb{R}^d$. At each point $\mathbf{x} \in \mathcal{M}$, $\mathcal{M}$ can be associated a *tangent space* $T_\mathbf{x}\mathcal{M}$, which is a vector space of dimension d that can be understood as a first order approximation of $\mathcal{M}$ around $\mathbf{x}$. A *Riemannian metric* g is given by an inner-product $g_\mathbf{x}(\cdot, \cdot)$ at each tangent space $T_\mathbf{x}\mathcal{M}$, $g_\mathbf{x}$ varying smoothly with $\mathbf{x}$. A given g defines the *geometry* of $\mathcal{M}$, because it can be used to define the distance between $\mathbf{x}$ and $\mathbf{y}$ as the infimum of the lengths of smooth paths $\gamma : [0, 1] \to \mathcal{M}$ from $\mathbf{x}$ to $\mathbf{y}$, where the length is defined as $\ell(\gamma) := \int_0^1 \sqrt{g_{\gamma(t)}(\dot{\gamma}(t), \dot{\gamma}(t))}\mathrm{d}t$. Under certain assumptions, a given g also defines a *curvature* at each point.

Unifying all curvatures κ. There exist several models of respectively constant positive and negative curvatures. For positive curvature, we choose the stereographic projection of the sphere, while for negative curvature we choose the Poincaré model which is the stereographic projection of the Lorentz model. As explained below, this choice allows us to generalize the gyrovector space framework and unify spaces of both positive and negative curvature κ into a single model which we call the κ-stereographic model.

The κ-stereographic model. For a curvature $\kappa \in \mathbb{R}$ and a dimension $d \geq 2$, we study the model $\mathfrak{st}^d_\kappa$ defined as $\mathfrak{st}^d_\kappa = \{\mathbf{x} \in \mathbb{R}^d \mid -\kappa||\mathbf{x}||_2^2 < 1\}$ equipped with its *Riemannian metric* $g^\kappa_\mathbf{x} = \frac{4}{(1+\kappa||\mathbf{x}||^2)^2}\mathbf{I} =: (\lambda^\kappa_\mathbf{x})^2\mathbf{I}$. Note in particular that when $\kappa \geq 0$, $\mathfrak{st}^d_\kappa$ is $\mathbb{R}^d$, while when $\kappa < 0$ it is the open ball of radius $1/\sqrt{-\kappa}$.

Gyrovector spaces & Riemannian geometry. As discussed in section 1, the gyrovector space formalism is used to generalize vector spaces to the Poincaré model of hyperbolic geometry (Ungar, 2005; 2008). In addition, important quantities from Riemannian geometry can be rewritten in terms of the Möbius vector addition and scalar-vector multiplication (Ganea et al., 2018a). We here extend gyrovector spaces to the κ-stereographic model, *i.e.* allowing positive curvature.

For $\kappa > 0$ and any point $\mathbf{x} \in \mathfrak{st}^d_\kappa$, we will denote by $\tilde{\mathbf{x}}$ the unique point of the sphere of radius $\kappa^{-\frac{1}{2}}$ in $\mathbb{R}^{d+1}$ whose stereographic projection is $\mathbf{x}$. As detailed in Appendix C, it is given by

$$\tilde{\mathbf{x}} := (\lambda^\kappa_\mathbf{x}\mathbf{x}, \kappa^{-\frac{1}{2}}(\lambda^\kappa_\mathbf{x} - 1)). \tag{1}$$

For $\mathbf{x}, \mathbf{y} \in \mathfrak{st}^d_\kappa$, we define the κ-**addition**, in the κ-stereographic model by:

$$\mathbf{x} \oplus_\kappa \mathbf{y} = \frac{(1 - 2\kappa\mathbf{x}^T\mathbf{y} - \kappa||\mathbf{y}||^2)\mathbf{x} + (1 + \kappa||\mathbf{x}||^2)\mathbf{y}}{1 - 2\kappa\mathbf{x}^T\mathbf{y} + \kappa^2||\mathbf{x}||^2||\mathbf{y}||^2} \in \mathfrak{st}^d_\kappa. \tag{2}$$

The κ-addition is defined in all the cases except for spherical geometry and $\mathbf{x} = \mathbf{y}/(\kappa\|\mathbf{y}\|^2)$ as stated by the following theorem proved in Appendix C.2.1.

Theorem 1 (Definiteness of κ-addition)**.** *We have* $1 - 2\kappa\mathbf{x}^T\mathbf{y} + \kappa^2||\mathbf{x}||^2||\mathbf{y}||^2 = 0$ *if and only if* $\kappa > 0$ *and* $\mathbf{x} = \mathbf{y}/(\kappa\|\mathbf{y}\|^2)$.

For $s \in \mathbb{R}$ and $\mathbf{x} \in \mathfrak{st}^d_\kappa$ (and $|s \tan_\kappa^{-1} \|\mathbf{x}\|| < \kappa^{\frac{1}{2}}\pi/2$ if $\kappa > 0$), the κ-**scaling** in the κ-stereographic model is given by:

$$s \otimes_\kappa \mathbf{x} = \tan_\kappa\left(s \cdot \tan_\kappa^{-1} ||\mathbf{x}||\right)\frac{\mathbf{x}}{||\mathbf{x}||} \in \mathfrak{st}^d_\kappa, \tag{3}$$

where $\tan_\kappa$ equals $\kappa^{-1/2}\tan$ if $\kappa > 0$ and $(-\kappa)^{-1/2}\tanh$ if $\kappa < 0$. This formalism yields simple closed-forms for various quantities including the distance function (see fig. 3) inherited from the Riemannian manifold $(\mathfrak{st}^d_\kappa, g^\kappa)$, the exp and log maps, and geodesics (see fig. 2), as shown by the following theorem.

Theorem 2 (Extending gyrovector spaces to positive curvature)**.** *For* $\mathbf{x}, \mathbf{y} \in \mathfrak{st}^d_\kappa$, $\mathbf{x} \neq \mathbf{y}$, $\mathbf{v} \neq \mathbf{0}$, *(and* $\mathbf{x} \neq -\mathbf{y}/(\kappa\|\mathbf{y}\|^2)$ *if* $\kappa > 0$*), the distance function is given by*[a]*:*

$$d_\kappa(\mathbf{x}, \mathbf{y}) = 2|\kappa|^{-1/2}\tan_\kappa^{-1} \| -\mathbf{x} \oplus_\kappa \mathbf{y}\|, \tag{4}$$

the unit-speed geodesic from $\mathbf{x}$ *to* $\mathbf{y}$ *is unique and given by*

$$\gamma_{\mathbf{x}\to\mathbf{y}}(t) = \mathbf{x} \oplus_\kappa \left(t \otimes_\kappa (-\mathbf{x} \oplus_\kappa \mathbf{y})\right), \tag{5}$$

and finally the exponential and logarithmic maps are described as:

$$\exp^\kappa_\mathbf{x}(\mathbf{v}) = \mathbf{x} \oplus_\kappa \left(\tan_\kappa\left(|\kappa|^{\frac{1}{2}}\frac{\lambda^\kappa_\mathbf{x}||\mathbf{v}||}{2}\right)\frac{\mathbf{v}}{||\mathbf{v}||}\right) \tag{6}$$

$$\log^\kappa_\mathbf{x}(\mathbf{y}) = \frac{2|\kappa|^{-\frac{1}{2}}}{\lambda^\kappa_\mathbf{x}}\tan_\kappa^{-1} || -\mathbf{x} \oplus_\kappa \mathbf{y}||\frac{-\mathbf{x} \oplus_\kappa \mathbf{y}}{|| -\mathbf{x} \oplus_\kappa \mathbf{y}||} \tag{7}$$

[a] We write $-\mathbf{x} \oplus \mathbf{y}$ for $(-\mathbf{x}) \oplus \mathbf{y}$ and not $-(\mathbf{x} \oplus \mathbf{y})$.

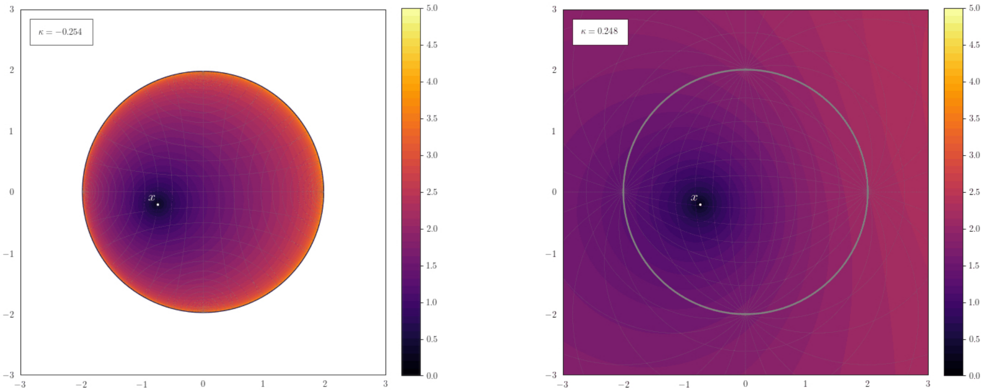

Figure 3. Heatmap of the distance function $d_\kappa(\mathbf{x}, \cdot)$ in $\mathfrak{st}_\kappa^2$ for $\kappa = -0.254$ (left) and $\kappa = 0.248$ (right).

Proof sketch:
The case $\kappa \leq 0$ was already taken care of by Ganea et al. (2018a). For $\kappa > 0$, we provide a detailed proof in Appendix C.2.2. The exponential map and unit-speed geodesics are obtained using the Egregium theorem and the known formulas in the standard spherical model. The distance then follows from the formula $d_\kappa(\mathbf{x}, \mathbf{y}) = \| \log_{\mathbf{x}}^{\kappa}(\mathbf{y}) \|_{\mathbf{x}}$ which holds in any Riemannian manifold. □

Around $\kappa = 0$. One notably observes that choosing $\kappa = 0$ yields all corresponding Euclidean quantities, which guarantees a *continuous* interpolation between κ-stereographic models of different curvatures, via Euler's formula $\tan(x) = -i \tanh(ix)$ where $i := \sqrt{-1}$. But is this interpolation *differentiable* with respect to κ? It is, as shown by the following theorem, proved in Appendix C.2.3.

Theorem 3 (Differentiability of $\mathfrak{st}_\kappa^d$ w.r.t. κ around 0)**.** *Let $\mathbf{v} \neq \mathbf{0}$ and $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$, such that $\mathbf{x} \neq \mathbf{y}$ (and $\mathbf{x} \neq -\mathbf{y}/(\kappa\|\mathbf{y}\|^2)$ if $\kappa > 0$). Quantities in Eqs. (4,5,6, 7) are well-defined for $|\kappa| < 1/\min(\|\mathbf{x}\|^2, \|\mathbf{y}\|^2)$, i.e. for κ small enough. Their first order derivatives at 0^- and 0^+ exist and are equal. Moreover, for the distance we have up to quadratic terms in κ:*

$$
\begin{aligned}
d_\kappa(\mathbf{x}, \mathbf{y}) &\approx 2\|\mathbf{x} - \mathbf{y}\| \\
&\quad - 2\kappa\Big(\|\mathbf{x} - \mathbf{y}\|^3/3 + (\mathbf{x}^T\mathbf{y})\|\mathbf{x} - \mathbf{y}\|^2\Big)
\end{aligned}
\tag{8}
$$

Note that for $\mathbf{x}^T\mathbf{y} \geq 0$, this tells us that an infinitesimal change of curvature from zero to small negative, *i.e.* towards 0^-, while keeping $\mathbf{x}, \mathbf{y}$ fixed, has the effect of increasing their distance.

As a consequence, we have a unified formalism that allows for a differentiable interpolation between all three geometries of constant curvature.

3. κ-GCNs

We start by introducing the methods upon which we build. We present our models for spaces of constant sectional curvature, in the κ-stereographic model. However, the generalization to cartesian products of such spaces (Gu et al., 2019) follows naturally from these tools.

3.1. Graph Convolutional Networks

The problem of node classification on a graph has long been tackled with explicit regularization using the graph Laplacian (Weston et al., 2012). Namely, for a directed graph with adjacency matrix $\mathbf{A}$, by adding the following term to the loss: $\sum_{i,j} \mathbf{A}_{ij} \|f(\mathbf{x}_i) - f(\mathbf{x}_j)\|^2 = f(\mathbf{X})^T \mathbf{L} f(\mathbf{X})$, where $\mathbf{L} = \mathbf{D} - \mathbf{A}$ is the (unnormalized) graph Laplacian, $D_{ii} := \sum_k A_{ik}$ defines the (diagonal) degree matrix, f contains the trainable parameters of the model and $\mathbf{X} = (x_i^j)_{ij}$ the node features of the model. Such a regularization is expected to improve generalization if connected nodes in the graph tend to share labels; node i with feature vector $\mathbf{x}_i$ is represented as $f(\mathbf{x}_i)$ in a Euclidean space.

With the aim to obtain more scalable models, Defferrard et al. (2016); Kipf & Welling (2017) propose to make this regularization implicit by incorporating it into what they call *graph convolutional networks* (GCN), which they motivate as a first order approximation of spectral graph convolutions, yielding the following scalable layer architecture (detailed in Appendix A):

$$
\mathbf{H}^{(t+1)} = \sigma\left(\tilde{\mathbf{D}}^{-\frac{1}{2}} \tilde{\mathbf{A}} \tilde{\mathbf{D}}^{-\frac{1}{2}} \mathbf{H}^{(t)} \mathbf{W}^{(t)}\right) \tag{9}
$$

where $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$ has added self-connections, $\tilde{D}_{ii} = \sum_k \tilde{A}_{ik}$ defines its diagonal degree matrix, σ is a non-linearity such as sigmoid, $\tanh$ or $\mathrm{ReLU} = \max(0, \cdot)$, and $\mathbf{W}^{(t)}$ and $\mathbf{H}^{(t)}$ are the parameter and activation matrices of layer t respectively, with $\mathbf{H}^{(0)} = \mathbf{X}$ the input feature matrix.

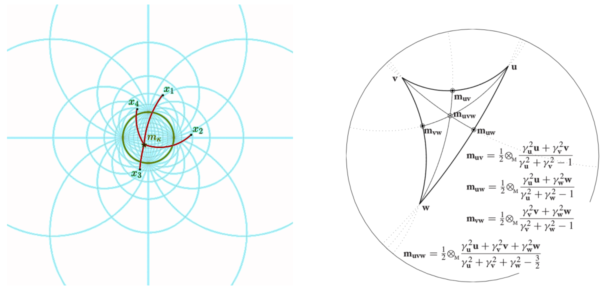
x_1
x_2
x_3
x_4
m_κ
u
v
w
$\mathbf{m_{uv}}$
$\mathbf{m_{uvw}}$
$\mathbf{m_{vw}}$
$\mathbf{m_{uw}}$
$\mathbf{m_{uv}} = \frac{1}{2} \otimes_M \frac{\gamma_\mathbf{u}^2 \mathbf{u} + \gamma_\mathbf{v}^2 \mathbf{v}}{\gamma_\mathbf{u}^2 + \gamma_\mathbf{v}^2 - 1}$
$\mathbf{m_{uw}} = \frac{1}{2} \otimes_M \frac{\gamma_\mathbf{u}^2 \mathbf{u} + \gamma_\mathbf{w}^2 \mathbf{w}}{\gamma_\mathbf{u}^2 + \gamma_\mathbf{w}^2 - 1}$
$\mathbf{m_{vw}} = \frac{1}{2} \otimes_M \frac{\gamma_\mathbf{v}^2 \mathbf{v} + \gamma_\mathbf{w}^2 \mathbf{w}}{\gamma_\mathbf{v}^2 + \gamma_\mathbf{w}^2 - 1}$
$\mathbf{m_{uvw}} = \frac{1}{2} \otimes_M \frac{\gamma_\mathbf{u}^2 \mathbf{u} + \gamma_\mathbf{v}^2 \mathbf{v} + \gamma_\mathbf{w}^2 \mathbf{w}}{\gamma_\mathbf{u}^2 + \gamma_\mathbf{v}^2 + \gamma_\mathbf{w}^2 - \frac{3}{2}}$

Figure 4. Left: Spherical gyromidpoint of four points. Right: Möbius gyromidpoint in the Poincaré model defined by (Ungar, 2008) and alternatively, here in eq. (12).

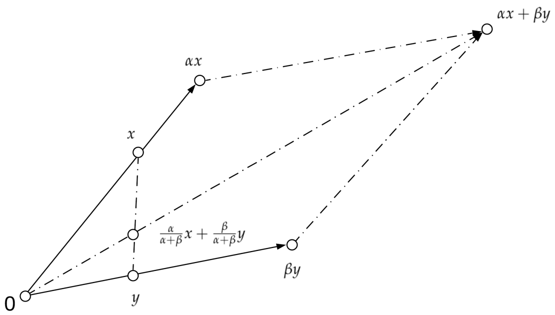
$\alpha x + \beta y$
αx
x
$\frac{\alpha}{\alpha+\beta} x + \frac{\beta}{\alpha+\beta} y$
βy
y
0

Figure 5. Weighted Euclidean midpoint $\alpha\mathbf{x} + \beta\mathbf{y}$

3.2. Tools for a κ-GCN

Learning a parametrized function f_θ that respects hyperbolic geometry has been studied in Ganea et al. (2018a): neural layers and hyperbolic softmax. We generalize their definitions into the κ-stereographic model, unifying operations in positive and negative curvature. We explain how curvature introduces a fundamental difference between *left* and *right* matrix multiplications, depicting the *Möbius* matrix multiplication of Ganea et al. (2018a) as a *right* multiplication, independent for each embedding. We then introduce a *left* multiplication by extension of gyromidpoints which ties the embeddings, which is essential for graph neural networks.

3.3. κ-Right-Matrix-Multiplication

Let $\mathbf{X} \in \mathbb{R}^{n \times d}$ denote a matrix whose n rows are d-dimensional embeddings in $\mathfrak{st}_\kappa^d$, and let $\mathbf{W} \in \mathbb{R}^{d \times e}$ denote a weight matrix. Let us first understand what a right matrix multiplication is in Euclidean space: the Euclidean right multiplication can be written row-wise as $(\mathbf{XW})_{i\bullet} = \mathbf{X}_{i\bullet}\mathbf{W}$. Hence each d-dimensional Euclidean embedding is modified *independently* by a right matrix multiplication. A natural adaptation of this operation to the κ-stereographic model yields the following definition.

Definition 1. *Given a matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ holding κ-stereographic embeddings in its rows and weights $\mathbf{W} \in \mathbb{R}^{d \times e}$, the κ-**right-matrix-multiplication** is defined row-wise as*

$$\begin{aligned}(\mathbf{X} \otimes_\kappa \mathbf{W})_{i\bullet} &= \exp_0^\kappa\left((\log_0^\kappa(\mathbf{X})\mathbf{W})_{i\bullet}\right) \\ &= \tan_\kappa\left(\alpha_i \tan_\kappa^{-1}(||\mathbf{X}_{\bullet i}||)\right) \frac{(\mathbf{XW})_{i\bullet}}{||(\mathbf{XW})_{i\bullet}||}\end{aligned} \tag{10}$$

where $\alpha_i = \frac{||(\mathbf{XW})_{i\bullet}||}{||\mathbf{X}_{i\bullet}||}$ and $\exp_0^\kappa$ and $\log_0^\kappa$ denote the exponential and logarithmic map in the κ-stereographic model.

This definition is in perfect agreement with the hyperbolic scalar multiplication for $\kappa < 0$, which can also be written as $r \otimes_\kappa \mathbf{x} = \exp_0^\kappa(r \log_0^\kappa(\mathbf{x}))$. This operation is known to have desirable properties such as associativity (Ganea et al., 2018a).

3.4. κ-Left-Matrix-Multiplication as a Midpoint Extension

For graph neural networks we also need the notion of message passing among neighboring nodes, *i.e.* an operation that *combines / aggregates* the respective embeddings together. In Euclidean space such an operation is given by the left multiplication of the embeddings matrix with the (preprocessed) adjacency $\hat{\mathbf{A}}$: $\mathbf{H}^{(l+1)} = \sigma(\hat{\mathbf{A}}\mathbf{Z}^{(l)})$ where $\mathbf{Z}^{(l)} = \mathbf{H}^{(l)}\mathbf{W}^{(l)}$. Let us consider this left multiplication. For $\mathbf{A} \in \mathbb{R}^{n \times n}$, the matrix product is given row-wise by:

$$(\mathbf{AX})_{i\bullet} = A_{i1}\mathbf{X}_{1\bullet} + \cdots + A_{in}\mathbf{X}_{n\bullet}$$

This means that the new representation of node i is obtained by calculating the linear combination of all the other node embeddings, weighted by the i-th row of $\mathbf{A}$. An adaptation to the κ-stereographic model hence requires a notion of *weighted linear combination.*

We propose such an operation in $\mathfrak{st}_\kappa^d$ by performing a κ-scaling of a *gyromidpoint* $-$ whose definition is reminded below. Indeed, in Euclidean space, the weighted linear combination $\alpha\mathbf{x}+\beta\mathbf{y}$ can be re-written as $(\alpha+\beta)m_\mathbb{E}(\mathbf{x}, \mathbf{y}; \alpha, \beta)$ with Euclidean midpoint $m_\mathbb{E}(\mathbf{x}, \mathbf{y}; \alpha, \beta) := \frac{\alpha}{\alpha+\beta}\mathbf{x} + \frac{\beta}{\alpha+\beta}\mathbf{y}$. See fig. 5 for a geometric illustration. This motivates generalizing the above operation to $\mathfrak{st}_\kappa^d$ as follows.

Definition 2. *Given a matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ holding κ-stereographic embeddings in its rows and weights $\mathbf{A} \in \mathbb{R}^{n \times n}$, the κ-**left-matrix-multiplication** is defined row-wise as*

$$(\mathbf{A} \boxtimes_\kappa \mathbf{X})_{i\bullet} := (\sum_j A_{ij}) \otimes_\kappa m_\kappa(\mathbf{X}_{1\bullet}, \cdots, \mathbf{X}_{n\bullet}; \mathbf{A}_{i\bullet}). \tag{11}$$

The κ-scaling is motivated by the fact that $d_\kappa(\mathbf{0}, r \otimes_\kappa \mathbf{x}) =$

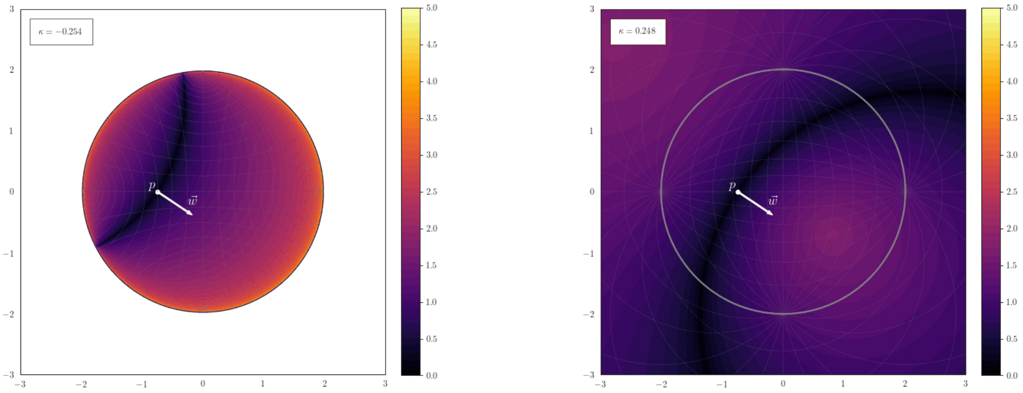

Figure 6. Heatmap of the distance from a $\mathfrak{st}_\kappa^2$-hyperplane to $x \in \mathfrak{st}_\kappa^2$ for $\kappa = -0.254$ (left) and $\kappa = 0.248$ (right)

$|r| d_\kappa(\mathbf{0}, \mathbf{x})$ for all $r \in \mathbb{R}$, $\mathbf{x} \in \mathfrak{st}_\kappa^d$. We remind that the gyromidpoint is defined when $\kappa \leq 0$ in the κ-stereographic model as (Ungar, 2010):

$$m_\kappa(\mathbf{x}_1, \cdots, \mathbf{x}_n; \boldsymbol{\alpha}) = \frac{1}{2} \otimes_\kappa \left(\sum_{i=1}^{n} \frac{\alpha_i \lambda_{\mathbf{x}_i}^\kappa}{\sum_{j=1}^{n} \alpha_j (\lambda_{\mathbf{x}_j}^\kappa - 1)} \mathbf{x}_i \right), \tag{12}$$

with $\lambda_{\mathbf{x}}^\kappa = 2/(1 + \kappa \|\mathbf{x}\|^2)$. Whenever $\kappa > 0$, we have to further require the following condition:

$$\sum_j \alpha_j (\lambda_{\mathbf{x}_j}^\kappa - 1) \neq 0. \tag{13}$$

For two points, one can calculate that $(\lambda_{\mathbf{x}}^\kappa - 1) + (\lambda_{\mathbf{y}}^\kappa - 1) = 0$ is equivalent to $\kappa \|\mathbf{x}\| \|\mathbf{y}\| = 1$, which holds in particular whenever $\mathbf{x} = -\mathbf{y}/(\kappa \|\mathbf{y}\|^2)$. See fig. 4 for illustrations of gyromidpoints.

Our operation $\boxtimes_\kappa$ satisfies interesting properties, proved in Appendix C.2.4:

Theorem 4 (Neuter element & κ-scalar-associativity)**.** *We have* $\mathbf{I}_n \boxtimes_\kappa \mathbf{X} = \mathbf{X}$*, and for* $r \in \mathbb{R}$*,*

$$r \otimes_\kappa (\mathbf{A} \boxtimes_\kappa \mathbf{X}) = (r\mathbf{A}) \boxtimes_\kappa \mathbf{X}.$$

The matrix A. In most GNNs, the matrix $\mathbf{A}$ is intended to be a preprocessed adjacency matrix, *i.e.* renormalized by the diagonal degree matrix $\mathbf{D}_{ii} = \sum_k A_{ik}$. This normalization is often taken either *(i)* to the left: $\mathbf{D}^{-1}\mathbf{A}$, *(ii)* symmetric: $\mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$ or *(iii)* to the right: $\mathbf{A}\mathbf{D}^{-1}$. Note that the latter case makes the matrix *right-stochastic*[1], which is a property that is preserved by matrix product and exponentiation. For this case, we prove the following result in Appendix C.2.5:

Theorem 5 (κ-left-multiplication by right-stochastic matrices is intrinsic)**.** *If* $\mathbf{A}, \mathbf{B}$ *are right-stochastic,* ϕ *is a isometry of* $\mathfrak{st}_\kappa^d$ *and* $\mathbf{X}, \mathbf{Y}$ *are two matrices holding* κ*-stereographic embeddings:*

$$\begin{aligned} \forall i, \quad d_\phi &= d_\kappa\left((\mathbf{A} \boxtimes_\kappa \phi(\mathbf{X}))_{i\bullet}, (\mathbf{B} \boxtimes_\kappa \phi(\mathbf{Y}))_{i\bullet}\right) \\ &= d_\kappa((\mathbf{A} \boxtimes_\kappa \mathbf{X})_{i\bullet}, (\mathbf{B} \boxtimes_\kappa \mathbf{Y})_{i\bullet}). \end{aligned} \tag{14}$$

The above result means that $\mathbf{A}$ can easily be preprocessed as to make its κ-left-multiplication intrinsic to the metric space $(\mathfrak{st}_\kappa^d, d_\kappa)$. At this point, one could wonder: does there exist other ways to take weighted centroids on a Riemannian manifold? We comment on two plausible alternatives.

Fréchet/Karcher means. They are obtained as $\arg\min_{\mathbf{x}} \sum_i \alpha_i d_\kappa(\mathbf{x}, \mathbf{x}_i)^2$; note that although they are also intrinsic, they usually require solving an optimization problem which can be prohibitively expensive, especially when one requires gradients to flow through the solution $-$ moreover, for the space $\mathfrak{st}_\kappa^d$, it is known that the minimizer is unique if and only if $\kappa \geq 0$.

Tangential aggregations. The linear combination is here lifted to the tangent space by means of the exponential and logarithmic map and were in particular used in the recent works of Chami et al. (2019) and Liu et al. (2019).

Definition 3. *The tangential aggregation of* $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathfrak{st}_\kappa^d$ *w.r.t. weights* $\{\alpha_i\}_{1 \leq i \leq n}$*, at point* $\mathbf{x} \in \mathfrak{st}_\kappa^d$ *(for* $\mathbf{x}_i \neq -\mathbf{x}/(\kappa \|\mathbf{x}\|^2)$ *if* $\kappa > 0$*) is defined by:*

$$\mathfrak{tg}_{\mathbf{x}}^\kappa(\mathbf{x}_1, ..., \mathbf{x}_n; \alpha_1, ..., \alpha_n) := \exp_{\mathbf{x}}^\kappa \left(\sum_{i=1}^{n} \alpha_i \log_{\mathbf{x}}^\kappa(\mathbf{x}_i) \right). \tag{15}$$

The below theorem describes that for the κ-stereographic model, this operation is also intrinsic. We prove it in Appendix C.2.6.

[1] $\mathbf{M}$ is *right-stochastic* if for all i, $\sum_j M_{ij} = 1$.

Theorem 6 (Tangential aggregation is intrinsic)**.** *For any isometry ϕ of $\mathfrak{st}_\kappa^d$, we have*

$$\mathfrak{tg}_{\phi(\mathbf{x})}(\{\phi(\mathbf{x}_i)\};\{\alpha_i\}) = \phi(\mathfrak{tg}_{\mathbf{x}}(\{\mathbf{x}_i\};\{\alpha_i\})). \qquad (16)$$

3.5. Logits

Finally, we need the logit and softmax layer, a neccessity for any classification task. We here use the model of Ganea et al. (2018a), which was obtained in a principled manner for the case of negative curvature. Their derivation rests upon the closed-form formula for distance to a hyperbolic hyperplane. We naturally extend this formula to $\mathfrak{st}_\kappa^d$, hence also allowing for $\kappa > 0$ but leave for future work the adaptation of their theoretical analysis.

$$p(y = k|\mathbf{x}) = \mathrm{S}\left(\frac{||\mathbf{a}_k||_{\mathbf{p}_k}}{\sqrt{|\kappa|}}\sin_\kappa^{-1}\left(\frac{2\sqrt{|\kappa|}\langle\mathbf{z}_k,\mathbf{a}_k\rangle}{(1+\kappa||\mathbf{z}_k||^2)||\mathbf{a}_k||}\right)\right), \qquad (17)$$

where $\mathbf{a}_k \in \mathcal{T}_0\mathfrak{st}_\kappa^d \cong \mathbb{R}^d$ and $\mathbf{p}_k \in \mathfrak{st}_\kappa^d$ are trainable parameters, $\mathbf{x} \in \mathfrak{st}_\kappa^d$, is the input,
$z_k = -\mathbf{p}_k \oplus \mathbf{x}$ and $\mathrm{S}(\cdot)$ is the softmax function.
We reference the reader to Appendix D for further details and to fig. 6 for an illustration of eq. 17.

3.6. κ-GCN

We are now ready to introduce our κ-stereographic GCN (Kipf & Welling, 2017), denoted by κ-GCN[2]. Assume we are given a graph with node level features $G = (V, \mathbf{A}, \mathbf{X})$ where $\mathbf{X} \in \mathbb{R}^{n\times d}$ with each row $\mathbf{X}_{i\bullet} \in \mathfrak{st}_\kappa^d$ and adjacency $\mathbf{A} \in \mathbb{R}^{n\times n}$. We first perform a preprocessing step by mapping the Euclidean features to $\mathfrak{st}_\kappa^d$ via the projection $\mathbf{X} \mapsto \mathbf{X}/(2\sqrt{|\kappa|}||\mathbf{X}||_{\max})$, where $||\mathbf{X}||_{\max}$ denotes the maximal Euclidean norm among all stereographic embeddings in $\mathbf{X}$. For $l \in \{0, \dots, L-2\}$, the $(l+1)$-th layer of κ-GCN is given by:

$$\mathbf{H}^{(l+1)} = \sigma^{\otimes_\kappa}\left(\hat{\mathbf{A}} \boxtimes_\kappa \left(\mathbf{H}^{(l)} \otimes_\kappa \mathbf{W}^{(l)}\right)\right), \qquad (18)$$

where $\mathbf{H}^{(0)} = \mathbf{X}$, $\sigma^{\otimes_\kappa}(\mathbf{x}) := \exp_{\mathbf{0}}^\kappa(\sigma(\log_{\mathbf{0}}^\kappa(\mathbf{x})))$ is the Möbius version (Ganea et al., 2018a) of a pointwise non-linearity σ and $\hat{\mathbf{A}} = \tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}$. The final layer is a κ-logit layer (Appendix D):

$$\mathbf{H}^{(L)} = \mathrm{softmax}\left(\hat{\mathbf{A}}\ \mathrm{logit}_\kappa\left(\mathbf{H}^{(L-1)}, \mathbf{W}^{(L-1)}\right)\right), \qquad (19)$$

where $\mathbf{W}^{(L-1)}$ contains the parameters $\mathbf{a}_k$ and $\mathbf{p}_k$ of the κ-logits layer. A very important property of κ-GCN is that its architecture recovers the Euclidean GCN when we let curvature go to zero:

$$\kappa\text{-GCN} \xrightarrow{\kappa \to 0} \text{GCN}.$$

Table 1. Minimum achieved average distortion of the different models. $\mathbb{H}$ and $\mathbb{S}$ denote hyperbolic and spherical models respectively.

MODEL	TREE	TOROIDAL	SPHERICAL
$\mathbb{E}^{10}$ (LINEAR)	0.045	0.0607	0.0415
$\mathbb{E}^{10}$ (RELU)	0.0502	0.0603	0.0409
$\mathbb{H}^{10}$ (κ-GCN)	**0.0029**	0.272	0.267
$\mathbb{S}^{10}$ (κ-GCN)	0.473	0.0485	**0.0337**
$\mathbb{H}^5 \times \mathbb{H}^5$ (κ-GCN)	0.0048	0.112	0.152
$\mathbb{S}^5 \times \mathbb{S}^5$ (κ-GCN)	0.51	**0.0464**	0.0359
$\left(\mathbb{H}^2\right)^4$ (κ-GCN)	0.025	0.084	0.062
$\left(\mathbb{S}^2\right)^4$ (κ-GCN)	0.312	0.0481	0.0378

4. Experiments

We evaluate the architectures introduced in the previous sections on the tasks of node classification and minimizing embedding distortion for several synthetic as well as real datasets. We detail the training setup and model architecture choices to Appendix F.

Minimizing Distortion Our first goal is to evaluate the graph embeddings learned by our GCN models on the representation task of fitting the graph metric in the embedding space. We desire to minimize the average distortion, i.e. defined similarly as in Gu et al. (2019): $\frac{1}{n^2}\sum_{i,j}\left(\left(\frac{d(\mathbf{x}_i,\mathbf{x}_j)}{d_G(i,j)}\right)^2 - 1\right)^2$, where $d(\mathbf{x}_i, \mathbf{x}_j)$ is the distance between the embeddings of nodes i and j, while $d_G(i, j)$ is their graph distance (shortest path length).

We create three synthetic datasets that best reflect the different geometries of interest: i) “Tree‘”: a balanced tree of depth 5 and branching factor 4 consisting of 1365 nodes and 1364 edges. ii) “Torus”: We sample points (nodes) from the (planar) torus, i.e. from the unit connected square; two nodes are connected by an edge if their toroidal distance (the warped distance) is smaller than a fixed $R = 0.01$; this gives 1000 nodes and 30626 edges. iii) “Spherical Graph”: we sample points (nodes) from $\mathbb{S}^2$, connecting nodes if their distance is smaller than 0.2, leading to 1000 nodes and 17640 edges.

For the GCN models, we use 1-hot initial node features. We use two GCN layers with dimensions 16 and 10. The non-Euclidean models do not use additional non-linearities between layers. All Euclidean parameters are updated using the ADAM optimizer with learning rate 0.01. Curvatures are learned using gradient descent and learning rate of 0.0001. All models are trained for 10000 epochs and we report the

[2] To be pronounced “kappa” GCN; the greek letter κ being commonly used to denote sectional curvature

Table 2. Node classification: Average accuracy across 5 splits with estimated uncertainties at 95 percent confidence level via bootstrapping on our datasplits. $\mathbb{H}$ and $\mathbb{S}$ denote hyperbolic and spherical models respectively.

MODEL	CITESEER	CORA	PUBMED	AIRPORT
$\mathbb{E}^{16}$ (KIPF & WELLING, 2017)	0.729 ± 0.0054	0.814 ± 0.004	0.792 ± 0.0039	0.814 ± 0.0029
$\mathbb{H}^{16}$ (CHAMI ET AL., 2019)	0.71 ± 0.0049	0.803 ± 0.0046	$\mathbf{0.798 \pm 0.0043}$	$\mathbf{0.844 \pm 0.0041}$
$\mathbb{H}^{16}$ (κ-GCN)	$\mathbf{0.732 \pm 0.0051}$	0.812 ± 0.005	0.785 ± 0.0036	0.819 ± 0.0033
$\mathbb{S}^{16}$ (κ-GCN)	0.721 ± 0.0045	$\mathbf{0.819 \pm 0.0045}$	0.788 ± 0.0049	0.809 ± 0.0058
PROD-GCN (κ-GCN)	0.711 ± 0.0059	0.808 ± 0.0041	0.781 ± 0.006	0.817 ± 0.0044

minimal achieved distortion.

Distortion results. The obtained distortion scores shown in table 1 reveal the benefit of our models. The best performing architecture is the one that matches the underlying geometry of the graph.

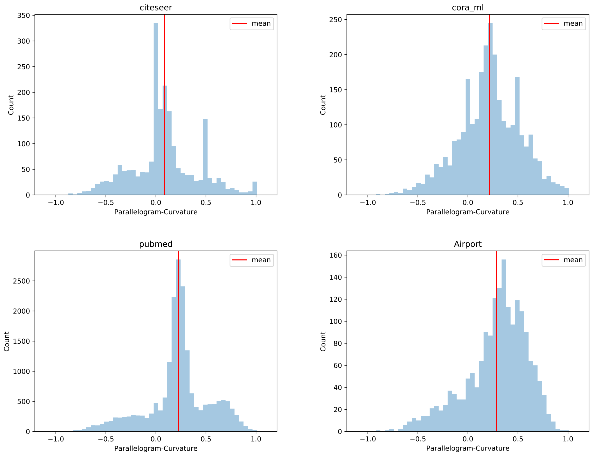

Figure 7. Histogram of Curvatures from ”Deviation of Parallogram Law”

4.1. Node Classification

We consider the popular node classification datasets Citeseer (Sen et al., 2008), Cora-ML (McCallum et al., 2000) and Pubmed (Namata et al., 2012). Node labels correspond to the particular subfield the published document is associated with. Dataset statistics and splitting details are deferred to the Appendix F due to the lack of space. We compare against the Euclidean model (Kipf & Welling, 2017) and the recently proposed hyperbolic variant (Chami et al., 2019).

Curvature Estimations of Datasets To understand how far are the real graphs of the above datasets from the Euclidean geometry, we first estimate the graph curvature of the four studied datasets using the **deviation from the Parallelogram Law** (Gu et al., 2019) as detailed in Appendix G. Curvature histograms are shown in fig. 7. It can be noticed that the datasets are mostly non-Euclidean, thus offering a good motivation to apply our constant-curvature GCN architectures.

Training Details We trained the baseline models in the same setting as done in Chami et al. (2019). Namely, for GCN we use one hidden layer of size 16, dropout on the embeddings and the adjacency of rate 0.5 as well as L^2-regularization for the weights of the first layer. We used ReLU as the non-linear activation function.

For the non-Euclidean architectures, we used a combination of dropout and dropconnect for the non-Euclidean models as reported in Chami et al. (2019), as well as L^2-regularization for the first layer. All models have the same number of parameters and for fairness are compared in the same setting, without attention. We use one hidden layer of dimension 16. For the product models we consider two-component spaces (e.g $\mathbb{H}^8 \times \mathbb{S}^8$) and we split the embedding space into equal dimensions of size 8. We also distribute the input features equally among the components. Non-Euclidean models use the Möbius version of ReLU. Euclidean parameters use a learning rate of 0.01 for all models using ADAM. The curvatures are learned using gradient descent with a learning rate of 0.01. We show the learnt values in Appendix F. We use early stopping: we first train for 2000 epochs, then we check every 200 epochs for improvement in the validation cross entropy loss; if that is not observed, we stop.

Node classification results. These are shown in table 2. It can be seen that our models are competitive with the Euclidean GCN considered and outperforms Chami et al. (2019) on Citeseer and Cora, showcasing the benefit of our proposed architecture.

5. Conclusion

In this paper, we introduced a natural extension of GCNs to the stereographic models of both positive and negative curvatures in a unified manner. We show how this choice of models permits a differentiable interpolation between positive and negative curvature, allowing the curvature to be trained independent of an initial sign choice. We hope that our models will open new exciting directions into non-Euclidean graph neural networks.

6. Acknowledgements

We thank prof. Thomas Hofmann, Andreas Bloch, Calin Cruceru and Ondrej Skopek for useful discussions and anonymous reviewers for suggestions.
Gary Bécigneul is funded by the Max Planck ETH Center for Learning Systems.

References

Abu-El-Haija, S., Kapoor, A., Perozzi, B., and Lee, J. *N-GCN: Multi-scale Graph Convolution for Semi-supervised Node Classification*. International Workshop on Mining and Learning with Graphs (MLG), 2018.

Amari, S.-i. and Nagaoka, H. *Methods of information geometry*, volume 191. American Mathematical Soc., 2007.

Bahdanau, D., Cho, K., and Bengio, Y. Neural machine translation by jointly learning to align and translate. *Proceedings of the International Conference on Learning Representations*, 2015.

Bronstein, M. M., Bruna, J., LeCun, Y., Szlam, A., and Vandergheynst, P. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4): 18–42, 2017.

Bruna, J., Zaremba, W., Szlam, A., and Lecun, Y. Spectral networks and locally connected networks on graphs. In *International Conference on Learning Representations (ICLR2014), CBLS, April 2014*, pp. http–openreview, 2014.

Chami, I., Ying, R., Ré, C., and Leskovec, J. Hyperbolic graph convolutional neural networks. *Advances in Neural Information processing systems*, 2019.

Chen, J., Ma, T., and Xiao, C. *Fastgcn: fast learning with graph convolutional networks via importance sampling*. ICLR, 2018.

Cho, H., DeMeo, B., Peng, J., and Berger, B. Large-margin classification in hyperbolic space. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 1832–1840, 2019.

Davidson, T. R., Falorsi, L., De Cao, N., Kipf, T., and Tomczak, J. M. *Hyperspherical Variational Auto-Encoders*. Uncertainty in Artificial Intelligence (UAI), 856- 865, 2018.

Defferrard, M., Bresson, X., and Vandergheynst, P. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*, pp. 3844–3852, 2016.

Defferrard, M., Perraudin, N., Kacprzak, T., and Sgier, R. Deepsphere: towards an equivariant graph-based spherical cnn. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019. URL https://arxiv.org/abs/1904.05146.

Deza, M. and Laurent, M. *Geometry of Cuts and Metrics*. Springer, Vol. 15, 1996.

Ganea, O., Bécigneul, G., and Hofmann, T. Hyperbolic neural networks. In *Advances in Neural Information Processing Systems*, pp. 5345–5355, 2018a.

Ganea, O.-E., Becigneul, G., and Hofmann, T. Hyperbolic entailment cones for learning hierarchical embeddings. In *International Conference on Machine Learning*, pp. 1632–1641, 2018b.

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. *Neural Message Passing for Quantum Chemistry*. Proceedings of the International Conference on Machine Learning, 2017.

Grattarola, D., Zambon, D., Alippi, C., and Livi, L. Learning graph embeddings on constant-curvature manifolds for change detection in graph streams. *stat*, 1050:16, 2018.

Graves, A., Mohamed, A.-r., and Hinton, G. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pp. 6645–6649. IEEE, 2013.

Gromov, M. Hyperbolic groups. In *Essays in group theory*, pp. 75–263. Springer, 1987.

Gu, A., Sala, F., Gunel, B., and Ré, C. Learning mixed-curvature representations in product spaces. Proceedings of the International Conference on Learning Representations, 2019.

Gulcehre, C., Denil, M., Malinowski, M., Razavi, A., Pascanu, R., Hermann, K. M., Battaglia, P., Bapst, V., Raposo, D., Santoro, A., et al. Hyperbolic attention networks. *Proceedings of the International Conference on Learning Representations*, 2018.

Hamann, M. On the tree-likeness of hyperbolic spaces. *Mathematical Proceedings of the Cambridge Philosophical Society*, pp. 1–17, 2017. doi: 10.1017/S0305004117000238.

Hamann, M. *On the tree-likeness of hyperbolic spaces*. Mathematical Proceedings of the Cambridge Philo- sophical Society, pp. 117, 2017.

Hamilton, W. L., Ying, R., and Leskovec, J. *Inductive Representation Learning on Large Graphs*. In Advances in Neural Information Processing Systems, 2017.

Hammond, D. K., Vandergheynst, P., and Gribonval, R. Wavelets on graphs via spectral graph theory. *Applied and Computational Harmonic Analysis*, 30(2):129–150, 2011.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

He, X., Liao, L., Zhang, H., Nie, L., Hu, X., and Chua, T.-S. Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web*, pp. 173–182. International World Wide Web Conferences Steering Committee, 2017.

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

Kingma, D. P. and Ba, J. *ADAM: A method for stochastic optimization*. ICLR, 2015.

Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks. *International Conference on Learning Representations*, 2017.

Klicpera, J., Bojchevski, A., and Günnemann, S. Predict then propagate: graph neural networks meet personalized pagerank. *International Conference on Learning Representations*, 2019.

Krioukov, D., Papadopoulos, F., Kitsak, M., Vahdat, A., and Boguná, M. Hyperbolic geometry of complex networks. *Physical Review E*, 82(3):036106, 2010.

Liu, Q., Nickel, M., and Kiela, D. Hyperbolic graph neural networks. *Advances in Neural Information processing systems*, 2019.

Mathieu, E., Lan, C. L., Maddison, C. J., Tomioka, R., and Teh, Y. W. Continuous hierarchical representations with Poincaré variational auto-encoders. *Advances in Neural Information Processing Systems*, 2019.

Matousek, J. *Lecture notes on metric embeddings*. 2013.

McCallum, A., Nigam, K., Rennie, J., and Seymore, K. *Automating the construction of internet portals with machine learning*. Information Retrieval, 3(2):127–163, 2000.

Namata, G., London, B., Getoor, L., and Huang, B. *Query-driven Active Surveying for Collective Classification*. International Workshop on Mining and Learning with Graphs (MLG), 2012.

Nickel, M. and Kiela, D. Poincaré embeddings for learning hierarchical representations. In *Advances in Neural Information Processing Systems*, pp. 6341–6350, 2017.

Nickel, M. and Kiela, D. Learning continuous hierarchies in the lorentz model of hyperbolic geometry. In *International Conference on Machine Learning*, 2018.

Ovinnikov, I. Poincaré Wasserstein autoencoder. *arXiv preprint arXiv:1901.01427*, 2019.

Roweis, S. T. and Saul, L. K. Nonlinear dimensionality reduction by locally linear embedding. *science*, 290(5500):2323–2326, 2000.

Sala, F., De Sa, C., Gu, A., and Re, C. Representation tradeoffs for hyperbolic embeddings. In *International Conference on Machine Learning*, pp. 4457–4466, 2018.

Sarkar, R. *Low distortion delaunay embedding of trees in hyperbolic plane*. International Symposium on Graph Drawing, pp. 355–366. Springer,, 2011.

Sen, P., Namata, G., Bilgic, M., Getoor, L., Gallagher, B., and Eliassi-Rad, T. *Collective Classification in Network Data*. AI Magazine, 29(3):93–106, 2008.

Spivak, M. A comprehensive introduction to differential geometry. volume four. 1979.

Tenenbaum, J. B., De Silva, V., and Langford, J. C. A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323, 2000.

Tifrea, A., Bécigneul, G., and Ganea, O.-E. Poincaré glove: Hyperbolic word embeddings. Proceedings of the International Conference on Learning Representations, 2019.

Ungar, A. *Barycentric Calculus in Euclidean and Hyperbolic Geometry*. World Scientific, ISBN 9789814304931, 2010.

Ungar, A. A. The hyperbolic pythagorean theorem in the Poincaré disc model of hyperbolic geometry. *The American mathematical monthly*, 106(8):759–763, 1999.

Ungar, A. A. *Analytic hyperbolic geometry: Mathematical foundations and applications*. World Scientific, 2005.

Ungar, A. A. A gyrovector space approach to hyperbolic geometry. *Synthesis Lectures on Mathematics and Statistics*, 1(1):1–194, 2008.

Ungar, A. A. *Analytic Hyperbolic Geometry in N Dimensions: An Introduction*. CRC Press, 2014.

Ungar, A. A. Novel tools to determine hyperbolic triangle centers. In *Essays in Mathematics and its Applications*, pp. 563–663. Springer, 2016.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems*, pp. 5998–6008, 2017.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. Graph attention networks. *International Conference on Learning Representations*, 2018.

Weston, J., Ratle, F., Mobahi, H., and Collobert, R. Deep learning via semi-supervised embedding. In *Neural Networks: Tricks of the Trade*, pp. 639–655. Springer, 2012.

Wilson, R. C., Hancock, E. R., Pekalska, E., and Duin, R. P. Spherical and hyperbolic embeddings of data. *IEEE transactions on pattern analysis and machine intelligence*, 36(11):2255–2269, 2014.

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., and Yu, P. S. A comprehensive survey on graph neural networks. *arXiv preprint arXiv:1901.00596*, 2019.

Xu, J. and Durrett, G. Spherical latent spaces for stable variational autoencoders. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 4503–4513, 2018.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. How powerful are graph neural networks? *International Conference on Learning Representations*, 2018.

Zhang, M. and Chen, Y. *Link prediction based on graph neural networks*. In Advances in Neural Information Processing Systems, 2018.